

AD-A286 940



THE SECOND ANNUAL

European Software Engineering
Process Group Conference 1997



EUROPEAN SEPG

Delegate Material
TUTORIALS

98-00025



16-17th JUNE 1997

GRAND HOTEL KRASNAPOLSKY
AMSTERDAM

DISTRIBUTION STATEMENT A

Approved for public release;
Distribution Unlimited



TUTORIALS - Monday 16th June

Morning

- T101a** Process and Year 2000: Complacency or Hype?
Watts Humphrey, Virginia Soper & Peter Blundell
- T101b** Appraisals Using the CMM as a Reference Model
Steve Masters & Donna Dunaway
- T101c** IDEAL Approach to Implementing Software Process Improvement
Chuck Myers, Paul Goodman & Magnus Ahlgren
- T101d** Peer Reviews: The Key to Cost-Effective Quality
Fran O'Hara

Afternoon

- T102a** The People Capability Maturity Model (P-CMM) - A Brief Introduction
Bill Curtis
- T102b** Dealing with the Underworld - Accelerating SPI
Mike Morrell, Wilko van Asseldonk & Jeroen Brinkman
- T102c** Software Process Improvement: Business Impacts and Value
David Zubrow
- T102d** Effective Implementation of CMM Levels 2 & 3
Magnus Ahlgren

TUTORIALS - Tuesday 17th June

Morning

- T201a** Measurement Symposium (ALL DAY)
- KEYNOTE: Controlling Outsourced Software Contracts
Charles Symons
- Metrics in Small Companies - Coupling a Metrics Programme to Your Business Model
Tom Raaijmakers
- Quantitative Management of Software Process Improvement
Christof Ebert
- Process Improvement by Software Measurement - Current and Future Directions for GQM Method
Rini van Solingen & Egon Berghout
- Software Metrics - Real World Experiences
John Holt
- The MIAMI Experience
Chris Herbert
- Trends in Software Process Maturity
David Zubrow

- T201b** Dependence to Influence: Developing and Nurturing Effective Sponsorship
Chuck Myers
- T201c** Risk Management in Practice: Effective and Ineffective
Audrey Dorofee & Ray Williams
- T201d** Personal Software Process
Watts Humphrey
- T201e** A Method for Defining and Improving Software Processes
James Hart
- Afternoon**
- T202b** Blind Faith is Not the Answer - Establishing a Customer Focused Requirements
Process that Consistently Delivers
Mac Craigmyle
- T202c** Process Improvement Action Planning
John Vu
- T202d** Project Planning, Tracking and Configuration Management for Project Leaders
Tim Kasse & Peter Leeson

The Year 2000 and Process Improvement

Watts S. Humphrey
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Sponsored by the U.S. Department of Defense

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 1

Solving the year 2000 problem
(Y2K) will not improve your
business.

But you are more likely to have
a business.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 2

Agenda

The year 2000 problem (Y2K)

Attacking the problem

How a mature process helps

A Y2K strategy

References

Conclusions

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 3

The Problem - 1

Without correction, most software systems will not work, starting with dates of 1/1/2000.

This is true of

- hardware
- software
- support systems

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 4

The Problem - 2

The principal causes of the Y2K problem are

- date abbreviations: 99 instead of 1999
- use of incorrect dates in random number generators, keys, etc.
- use of 00 or 99 as reserved numbers
- incorrect date calculations

Date calculations

- every 4th year is leap year
- except every 100th year
- except every 400th year
- except ...

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 5

Leap Year Calculations

The year has 365.242199 mean solar days.

- Every 4 years, there is an extra 0.968796 days.
- So we add a day on leap year

In every 100 years

- we have added 0.7801 days too much
- so we skip a leap year

In every 400 years

- we are 0.8796 days short
- so we add a leap year day, etc.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 6

The Problem - 3

The principal consequences of Y2K are

- application programs may fail
- systems programs may fail
- the hardware may fail
- vendor products may fail
- customer data may be incorrect
- anyone's data could corrupt your system

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 7

Is the Problem Real?

If your organization does not believe Y2K is real

- and if you do not have an aggressive program
- you are in danger of a business disaster

The evidence is overwhelming.

- Every study agrees.
- Repeat studies find the costs are higher the closer they look.
- The cost of recovery is increasing by 20% to 40% with each year of delay.
- There will not be a magic tool or fix.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 8

Some Examples - 1

U.S. Social Security Administration

- A major system failed in 1989.
- They have 30MLOC and 20MLOC in development.
- Their total estimate is 300 programmer years.

Yellow Technology

- 165 systems, 13,000 modules, 15MLOC
- Analysis took 10 people 3 months
- The total estimate is 150 to 200 programmer years

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 9

Some Examples - 2

A Cap Gemini study of 3 organizations

- financial services
 - 40 systems
 - 6.5 MLOC
 - 89,535 dates to change
- Insurance
 - 231 systems
 - 14 MLOC
 - 243,313 dates to change
- Manufacturer
 - 329 systems
 - 59 MLOC
 - 346,412 dates to change

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 10

The Key Issue is Schedule

The end date is fixed.

It is before 1/1/2000.

There is no technology fix on the way.

If Y2K costs more than planned, that could be painful.

If you are late, that could be fatal.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 11

Process Needs

To reliably meet the date, you need an effective process.

The critical process areas are

- Requirements Management**
- Software Subcontract Management**
- Software Project Planning**
- Software Project Tracking and Oversight**
- Software Quality Assurance**
- Software Configuration Management**
- Peer Reviews**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 12

There is Good and Bad News

With CMMSM

- bad news: process improvement takes time
- good news: the first benefit is better schedule management

With PSPSM

- bad news: learning PSP is a lot of work
- good news: you can do it in a few weeks
- it will
 - accelerate CMM improvement
 - show engineers how to make schedules
 - produce high-quality software

SMCapability Maturity Model, CMM, Personal Software Process, and PSP are service marks of Carnegie Mellon University.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 13

The Topics of this Talk

The principal steps for attacking the Y2K problem

How process improvement can help

A process strategy for Y2K

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 14

The Principal Y2K Steps

The principal steps for attacking Y2K are

- build awareness
- inventory your systems
- assess these systems
- plan the repair work
- do the repair work
- test the fixes
- install and use the repaired systems

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 15

Creating Awareness

If your organization is not aware now, there is not much more I can say.

Keep trying to wake them up.

But if they do not act soon, update your resume.

Remember, this is a software quality problem.

If there is a disaster, you will be blamed.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 16

Inventory Your Systems

Analyze the systems you currently use.

Follow a triage strategy.

In medical terms triage is where you

- ignore the patients who will die with treatment**
- ignore the patients who will survive without treatment**
- only work on patients who will survive with treatment**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 17

Triage for Y2K

For Y2K, triage means

- Ignore the systems that do not have Y2K problems.**
- Ignore systems with Y2K problems that will not cause business failure.**
 - their failure may cause inconvenience**
 - but the business will survive**
- Only work on those systems that will cause business failure.**
 - Do these first.**
 - After they are fixed, think about the others.**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 18

The Inventory Results

A list of all products

- those to ignore
- those that are critical

The deadline dates for each

- planning and inventory systems look ahead
- billing and payroll systems look back

Remember to include

- internally and purchased systems
- support hardware and software
- tools and utilities

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 19

Other Systems

Also have someone look at

- the elevator system
- heating systems
- telephone and communications systems
- your bank's systems
- your suppliers' systems
- your customers' systems

Almost any of these could impact operations

- you need to know which could fail
- either fix them, replace them, or have a disaster plan

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 20

Process Impact on Inventory - 1

The inventory work is a form of requirements

You need an orderly process for

- eliciting requirements
- reviewing requirements
- approving requirements
- making sure the developers understand the requirements

And these processes need to be tailored to the Y2K problem.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 21

Process Impact on Inventory - 2

Requirements management is crucial.

The organization's needs will change.

- So should the Y2K plan.
- These changes must be managed.
- If not, they will destroy the project.

You cannot afford small uncontrolled requirements changes.

They will nibble you to death.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 22

Assessment and Planning

First, determine the strategy for each product.

- Should you ignore it?
- Should you fix it?
- Should you develop a replacement?
- Or should you buy a replacement?

These decisions must be based on

- a technical understanding of the work
- an assessment of costs and development time
- an estimate of the consequences of doing nothing

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 23

Deciding to Fix or Replace

You can learn little by looking at the product.

You need to know

- what it costs to maintain the product
- the number of defects found and remaining
- likely future maintenance costs

You also need to know

- if the program will meet future application needs
- what a replacement would cost
- if commercial replacements are available

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 24

A Y2K Rule

Unless the work is nearly done, do not develop new or replacement systems.

New development is far too risky, unless

- **your development group consistently meets all its dates**
- **development will be done by early 1999**

Planning and conversion will take time.

- **You can't afford to spend this time testing defective new products.**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 25

Process Impact on Assessment and Planning - 1

The critical KPAs for the assessment and planning phase are

- **Software Project Planning**
- **Software Subcontract Management**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 26

Process Impact on Assessment and Planning - 2

A mature process provides a wealth of data for the make, fix, or buy decisions.

The engineers will make better plans.

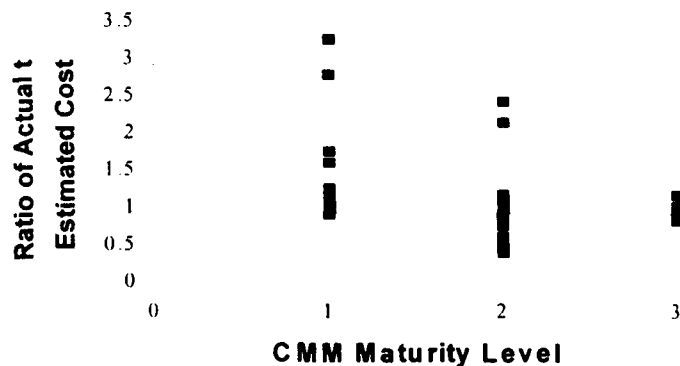
You will get Y2K data as the work is done.

- an early indication of estimate accuracy
- a better basis for projecting completion
- a sound basis for updating plans

Copyright © 1997 Carnegie Mellon University

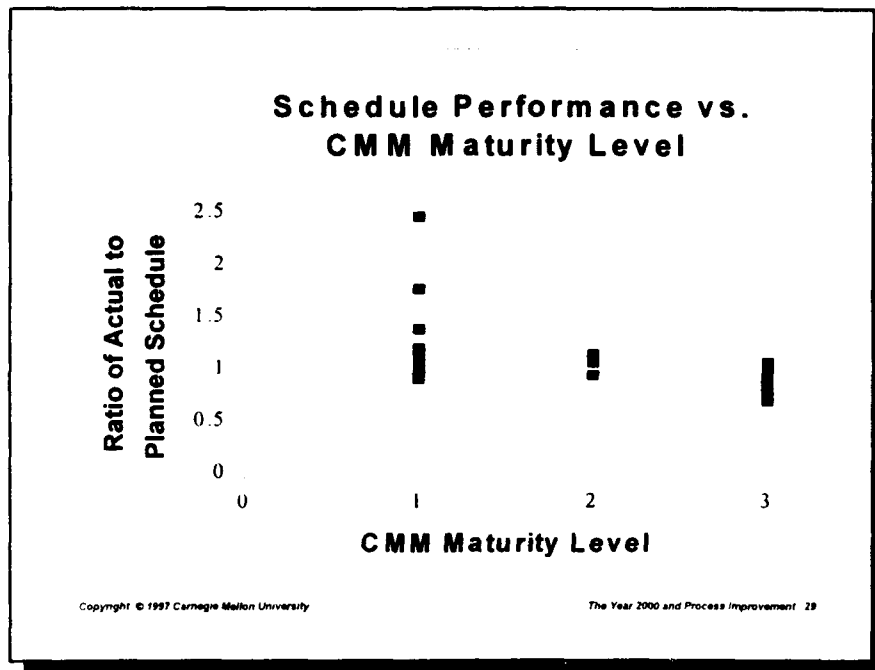
The Year 2000 and Process Improvement 27

Cost Performance vs. CMM Maturity Level



Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 28



Development

The development steps are

- Analyze the critical products.
- Design and make the needed fixes.
- Ensure that the fixes are correct and do not cause regressions.

To do this, the engineers need to

- understand the product
- design the fix
- make it consistent with all interfacing products
- make it consistent with Y2K design standards

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 30

The Importance of Quality

When engineers make fixes, they inject defects.

- these can cause serious problems**
- not only at the Y2K dates**
- but both before and long afterwards**

Poor quality work also impacts

- estimate accuracy**
- the schedule**
- the ability to predict completion**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 31

Defects in Modified Code - 1

O/S 360 MVS - 15 % of fixes injected errors

Hughes study - 17 % of error fixes incorrect

AT&T - 33 % of fixes injected defects

Ohba - 19 % fix error rate

Levendel - 1 in 3 fixes in error

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 32

Defects in Modified Code - 2

Gibson - 58 % of 8 line fixes in error

DoD - fix errors versus change size

- 1 line - 50 %**
- 5 line - 75 %**
- 20 line - 35 %**

U.S. Air Force Missile system - 75 % of system test fixes injected defects

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 33

Process Impact on Development - 1

The critical KPAs for the development phase are

- Software Project Tracking and Oversight**
- Peer Reviews**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 34

Process Impact on Development - 2

**The CMM introduces peer reviews at Level 2.
These result in**

- **higher-quality code**
- **reduced testing time**
- **accelerated schedules**

**The PSP shows engineers how to remove most
defects before compile and test. This**

- **saves development time**
- **produces essentially defect-free programs**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 35

AIS System Test Reduction

Non-PSP	Size	System Test Time
A	30 req.	2 months
B	19 req.	3 cycles
C	30 req.	2 months
D	15.8 KLOC	1.5 months
PSP		
E	11.7 KLOC	1.5 months
F	24 req	5 days
G	2.3 KLOC	2 days
H	6.2 KLOC	4 days
I	1.4 KLOC	4 days
J	13.3 KLOC	2 days

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 36

Testing

For Y2K, testing will be a challenge.

Testing will take time.

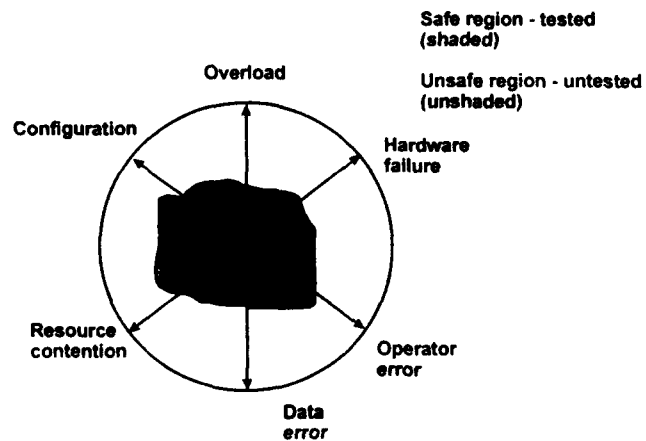
Testing cannot be exhaustive.

To get a quality product, you must put a quality product into test.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 37

Stress Operating Regions



Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 38

Unique Problems of Y2K Test - 1

It will not always be possible to advance dates.

You cannot test a system unless either

- **all the changes are made or**
- **special interface code is developed and tested**

You cannot generally test

- **suppliers' systems**
- **customers' actions**
- **some commercial hardware and software**

Unique Problems of Y2K Test - 2

Some systems will have early critical dates.

Thus you must test with a mix of

- **retrofitted and unretrofitted applications**
- **old and new databases**
- **non-conforming customers and suppliers**
-

Also remember to merge in ongoing maintenance at every test cycle.

The Y2K Testing Strategy

The testing must be designed to find poor-quality components, not to find and fix defects.

The defective components are then returned to development for inspection and repair.

To do this, you must

- keep test data**
- use these data to find the poor components**
- train the developers to inspect for defects**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 41

Process Impact on Testing

The critical process capabilities are

- Software Quality Assurance**
- Software Configuration Management**

These capabilities are needed during test.

Peer reviews should also be used as a filter at test entry.

The objective is to find and clean up the defective work before test.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 42

Installation and Use

The steps in installation and use are

- train the system users
- install new procedures and systems
- track and fix defects
- use defect data to identify poor quality products to return for repair

The Operating Environment

Starting well before 1/1/2000, you will face a mixed operational environment.

Some systems must be fixed early.

- forecasting
- inventory management
- financial planning
- scheduling

Also, some systems will have dependencies on others that are not retrofitted.

Expect a Dynamic Environment

The operational systems will change weekly and sometimes daily.

You must know the status

- of every system**
- of the system dependencies**

You need these data to

- operate the systems**
- and to find and fix defects in the systems**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 45

Process Impact on Installation and Use

To help, these process improvements must be put in place early.

They will pay off with

- on-time system installations**
- few defects**
- known system status and procedures**

Training will be important here and throughout the Y2K project.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 46

A Process Improvement Strategy

A mature software process would be a big help in addressing the Y2K problem.

But what if you do not have a mature process?

You need a pragmatic strategy.

- **Build on current process improvement work.**
- **Augment this work for critical Y2K needs.**

If you have done an assessment

- **review the results**
- **accelerate the critical improvements**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 47

Process Assessment

If you have not done a process assessment, and do not have one planned

- **you need an immediate process evaluation**
- **this will identify critical process weaknesses**

Get an assessment or capability evaluation as soon as possible.

- **Use the results to guide your improvement work.**
- **Follow the Y2K improvement strategy.**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 48

PSP Training

Identify the Y2K development team.

Get them all trained on the PSP as soon as possible.

Get one of your engineers trained as a PSP instructor so you can train new team members.

The instructor will also help and guide the Y2K team in using the PSP in their work.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 49

Y2K Improvement Priorities - 1

You need the following CMM KPAs to guide the inventory, assessment, and planning work.

- Requirements Management
- Software Subcontract Management
- Software Project Planning

Do these first.

- they will help you plan the work
- and they will identify the time and resources needed

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 50

Y2K Improvement Priorities - 2

As soon as possible, and before starting development, address the following KPAs.

- **Software Project Tracking and Oversight**
- **Peer Reviews**
- **Software Quality Assurance**

These will help you

- **track the development work**
- **produce higher-quality fixes**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 51

Y2K Improvement Priorities - 3

The next critical need is software configuration management.

SCM is important during development.

It is critical during test and operations.

It will help you

- **control the environment**
- **manage installation evolution**
- **recover from regressions**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 52

Y2K Improvement Priorities - 4

While all the other KPAs would help, you don't have time for everything else.

You need to use process improvement triage.

- Focus on the critical KPAs.**
- Do them in step with the Y2K work.**
- Get them in place as fast as possible.**

This will accelerate all your other work.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 53

Be Pragmatic

The objective is to guide the management and development work.

Your objective is not to pass some Level 2 or Level 3 checkpoint.

Get the critical KPAs in place

- get them used**
- and move on to installing the next KPAs**

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 54

Make Line Management Responsible

The line managers must be responsible for the process improvement work.

- **This work is for them.**
- **It will help them do their jobs better.**

If the line managers are not responsible

- **nothing will happen**
- **certainly not in time**

Measure Line Management

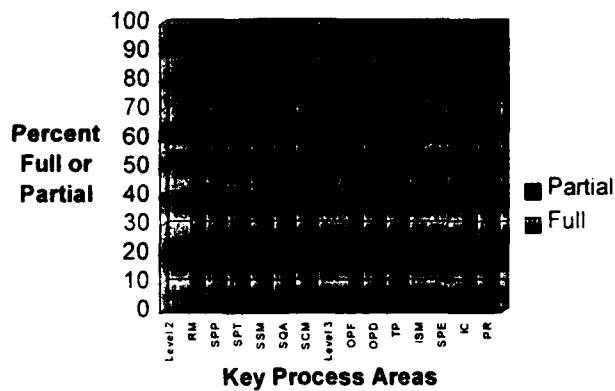
Measure progress every month.

- **Post the status of each department.**
- **Have senior managers review progress every month.**

When managers' evaluations depend on process improvement results

- **they will get it done**
- **and faster than you thought possible**

An Example Progress Report



Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 57

The Impact of Y2K

Think of Y2K as a hurricane.

•dangerous and damaging while it lasts

If you are not prepared

•and if you use a poor-quality process

•and there is anything worth saving

•the cleanup will take years

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 58

Y2K References

Some useful references on Y2K are

- "Year 2000 Computing Crisis: An Assessment Guide," GAO/AMD-10.1.14 <http://www.gao.gov/>
- R.A. Martin, "Dealing With Dates: Solutions for the Year 2000," IEEE Computer, May 1997.
- T. Backman, "Summary of the Mitre Assessment of the Effects of Two-Digit Years for the Year 2000," Jan. 1996, <http://www.mitre.org/research/y2k>.
- "The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation," IBM Report GC28-1251-00, 1996, <http://www.s390.ibm.com/stories/tran2000.html>

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 59

PSP References

For more information on the PSP, see

- www.sei.cmu.edu

The SEI offers PSP instructor training.

- Contact SEI industry relations.

The SEI also maintains a list of qualified PSP instructors.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 60

More References

Also see the following general Y2K references:

SEI: <http://www.sei.cmu.edu/-reengineering>

STSC: <http://www.stsc.hill.af.mil/RENG/index.html#2000>

MITRE: <http://www.mitre.org/research/y2k>

ISA:

http://www.mitre.org/research/cots/COMPLIANCE_CAT.html

IBM: <http://www.software.ibm.com/year2000/>

BCS: <http://www.bcs.org.uk/millen.htm>

GTE:

http://www.mitre.org/research/cots/GTE_CRITERIA/html

Peter de Jager: <http://www.year2000.com>

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 61

In Conclusion, Remember

The Y2K problem is serious and real.

There will be no magic technical fix.

**If you do not have an active program underway,
you are late.**

**You should build on the process work you have
underway.**

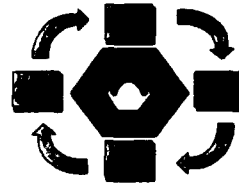
Accelerate work on the critical KPAs.

Copyright © 1997 Carnegie Mellon University

The Year 2000 and Process Improvement 62

Year 2000 and Software Process Improvement

European SEPG - Amsterdam



Virginia Soper
virginia_soper@vnet.ibm.com
June 16, 1997

The Year 2000 Challenge

- > How complex is it?
- > What makes it unique?
- > How can it be addressed most effectively?
- > Who will be the winners?



A key challenge for software development is the increased speed of delivery of high quality solutions to address the year 2000 problem.

- Although the problem is a technical one, the solution is potentially one of the biggest logistical and management headaches you may have ever encountered. (*John Phelps, Gartner Group*)
- If customers are to be successful in tackling the Year 2000 issue, they need to focus on specific date change methodologies, processes -- and over all project management. (*John Phelps, Gartner Group*)
- Check everything.....: system development practices; system procurement practices; vendor applications..... (*CCTA*)
- Testing the whole of a mainframe application is no mean undertaking. It is rarely possible to take an application out of commission in order to change the system date and run Year 2000 tests. Hot backup and disaster recovery service providers should now be looking to extend their facilities to offer a test environment for mainframe Year 2000 projects. (*CCTA*)



Why is Year 2000 a complex and different problem?

- Amount of corrections and amount of test (50% of the resources)
- Unknown variety of possible defects and of secondary defects generated by changes
- Very large project size and manageability challenge encompassing the entire organization with a great need for coordination required by the dependencies between applications and the amount of external and internal interfaces
- Magnitude extends beyond in-house applications spanning system software, middleware, standard applications, hardware, network and non-IT issues - making testing very complicated
- Firm immovable deadline
- Limited resources and shortage of skills
- Need for a structured approach and well-defined standards used all over the organization to ensure quality and working interrelationships between the applications



Comparing year 2000 to another large complex issue, EURO single currency, highlights more of its unique characteristics.

Year 2000 and EURO

• Similarities

- ▼ Multiple application changes
- ▼ Use of tools (e.g. similar supp. toolset for assessment and detailed analysis)
- ▼ Use of same resources
- ▼ Complex testing
- ▼ Methodology
- ▼ Industry and country awareness at different levels:

• Differences

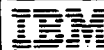
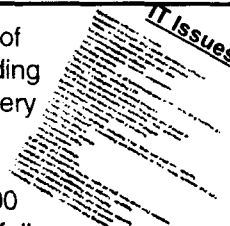
- ▼ Movable vs. immovable
- ▼ EURO not sudden death
- ▼ EURO => Business benefits
- ▼ EURO could collapse (low risk)
- ▼ Y2000 is everybody
- ▼ Testing activities:
 - Y2K = automated regression testing
 - EURO = business functional testing



© Copyright IBM Corp.

IBM (IDC) Market Survey: Some of the main observations...

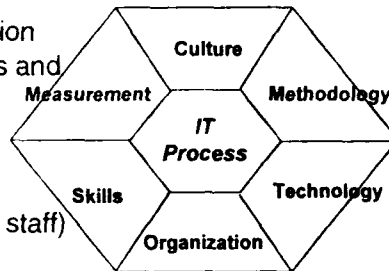
- IT people today have a good level of awareness of the Year 2000 problem; however, the understanding of the depth and impacts of the issue is still not very high.
- Many CEO's are still not aware that the Year 2000 issue is critical to their business and needs their full support; many still consider it as a purely technical problem to be solved business as usual by their IT department.
- The difficulty of the Year 2000 issue is still underestimated. Companies who have completed readiness projects consistently rated activities as more difficult than the sample average.



© Copyright IBM Corp.

The current business alignment, service delivery and development processes need to be assessed and enhancements identified to ensure a level of maturity consistent with the complex logistical and management challenges of a year 2000 project.

- How effective are the processes today?
- Are there business initiatives in the pipeline that may be impacted by the year 2000 readiness work?
- What changes to existing application portfolio strategy, processes, skills and organization will be required?
- Are current standards and tools adequate?
- Is the organization (management, staff) and culture ready for the extent of change?



IBM
© Copyright IBM Corp.

Unmanaged technology risk leads to business exposure. The technological root cause needs to be managed proactively whereas business exposure requires contingency planning.

IT Process risks/root causes

- Business/IT Alignment
- Project Management
- Testing
- Configuration Management
- Change Management
- Quality Assurance
- Service Delivery

Business risks/consequences

- Survival
- Financial Loss
- Damaged Image
- Embarrassment



Dependent upon

- Approach, Assessment
- Master Plan
- Management Commitment

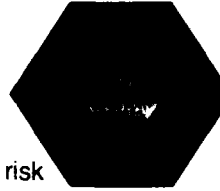


IBM
© Copyright IBM Corp.

Software Process Improvement is a very important part of the Assessment phase of the year 2000 comprehensive methodology.

OBJECTIVES of Year 2000 Software Process Assessment:

- Assess the development environment and key processes for year 2000 adaptation
- Assess the risk of year 2000 with focus on the key processes
- Investigate the change readiness of the key processes
- Give recommendations to minimize the company's risk for year 2000 adaptation



Replace



Reuse



Retire



Refurbish

The Assessment is specific to the scope of the problem which is sized by the following techniques:

- Scanning and pattern matching of applications with year 2000 scanning tool
- Samples from program are manually investigated for year 2000 problems and corrected to measure the time it takes to adapt and correct an application
- Evaluation of hardware and software vendors
- Investigation and revision of IT strategy and plans to fit the year 2000 resource demand



Copyright IBM Corp.

Collected facts, structured in a logic diagram, lead to conclusions and recommendations: a precise and objective description and prescription relevant to the specific year 2000 situation.

Hypothesis
covering
year 2000

Key and
Control
Questions

Individual interviews

Group interviews

Documentation

Questionnaire

AD Benchmark

Conclusions

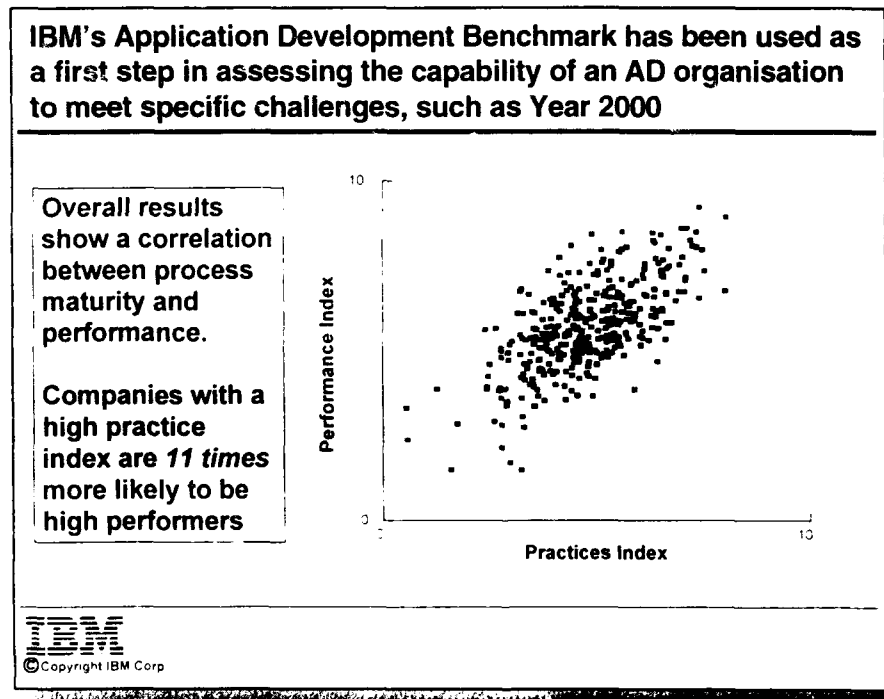
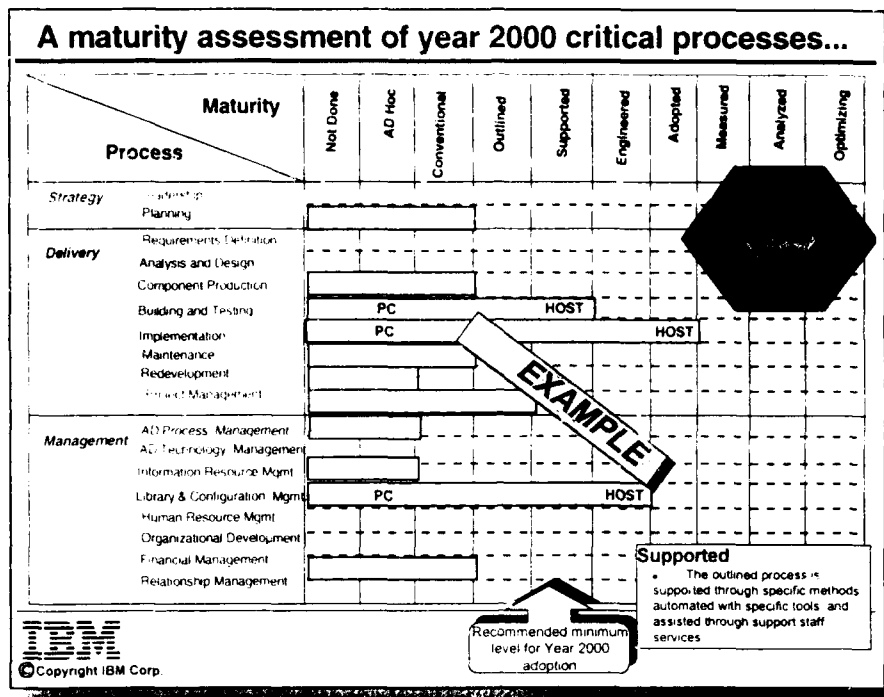
Findings

Risk and Maturity Evaluation
of critical processes

Recommendations



Copyright IBM Corp.



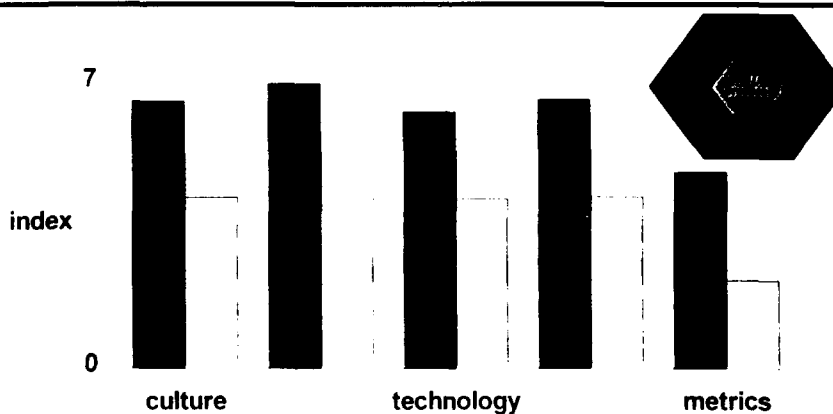
Questions look for objective responses over a range from 1 to 5. Analysis of the raw data creates practice and performance indices for positioning on an X/Y axis. Sample questions:

	1	2	3	4	5	Score
6. Inspections or walkthroughs	Code reviewed informally	Code formally inspected and resolution of defects verified		Formal inspection and correction of all deliverables		
9. Defect density of delivered software	More than 3 defects per delivered FP (30 defects per KLOC Cobol) in first year after delivery	Around 0.6 defects per delivered FP (6 defects per KLOC Cobol) in first year after delivery		Less than 0.1 defects per delivered FP (1 defect per KLOC Cobol) in first year after delivery		



Copyright IBM Corp.

There is a complex interrelationship of practices which must be handled in parallel. Leaders consistently show higher levels of performance in these key areas of IT Effectiveness.



Copyright IBM Corp.

■ Leaders ■ Laggards

The Risk assessment is a very important input to the planning process to initiate specific SPI activities where the risk is high and to predict the likelihood of meeting project plan deadlines.

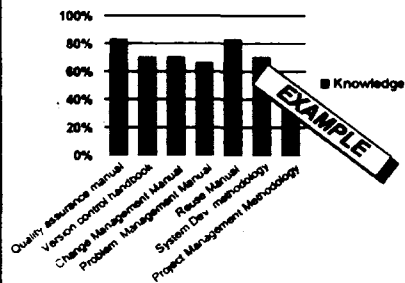
Risk assessment for Year 2000 projects	Very Low	Low	Average	High	Very High
Assessment of Source Code / Version Control Capability					
Assessment of QA / Testing Capability					
Assessment of Change Management Capability					
Requirements are the basis of Project Plans					
Deliverables, Work Effort, Costs, Schedule and critical Computer Resources are documented in Project Plans					
Changes to Deliverables and Commitments are tracked and reviewed					
The work of Contractors / Outsourcers is carefully planned and reviewed					
Many fixes required or many breakdowns					
Problem Management for identifying and tracking problems exists and is followed					
Overall Risk Assessment					



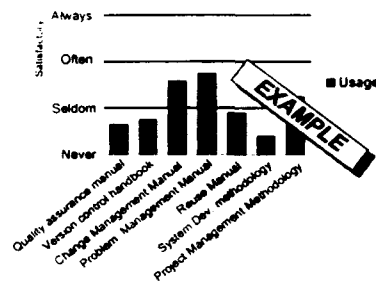
© Copyright IBM Corp.

The knowledge and use of methodologies in an organization forecast the degree to which standards and discipline will play a critical role in the year 2000 implementation.

Knowledge of methodologies



Actual use of methodologies



Despite the existence and knowledge of methodologies, they are seldom used.

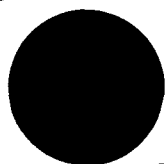


© Copyright IBM Corp.

Efficient communication and coordination of readiness activities will be required both internally throughout the company and externally with vendors and suppliers.

Do you exchange data electronically with outside organizations ?

71%



29%

Are you working with them to co-ordinate Year 2000 activities ?

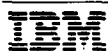
31%



69%

■ No ■ Yes

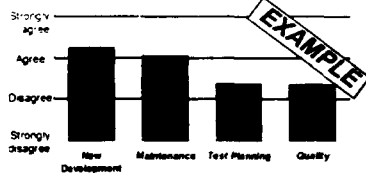
IBM (IDC) Market Survey



Copyright IBM Corp.

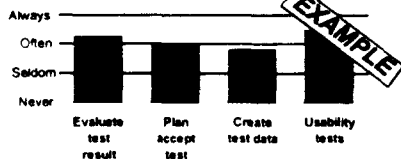
The critical role of the testing process in year 2000 will require more structure and rigorous standards. Unique project characteristics suggest shifts in approach and coverage.

Test Thoroughness



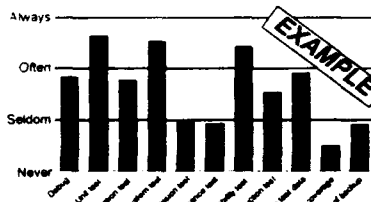
EXAMPLE

Degree of User Involvement



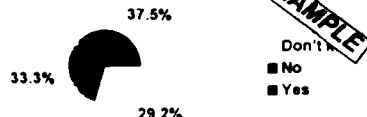
EXAMPLE

Testing Approach
To what degree do you use the following type of tests



EXAMPLE

Is testing of new dev. and maintenance performed differently?



Don't know

■ No ■ Yes



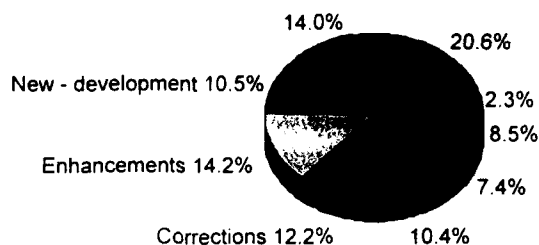
Copyright IBM Corp.

Normal business activities may be impacted by lower levels of service delivery from development and maintenance departments, the degree of which can be forecast by various indicators.

An assessment of how time is spent on activities in AD organizations is a time forecaster, quality predictor, and service delivery indicator for the year 2000 project.

Time distribution in AD departments

The average time spent on maintenance in 10 organizations is very low compared to the year 2000 task ahead.



IBM
Copyright IBM Corp.

There are dramatic differences in the performance of European software developers. How much better are the leaders when compared with the laggards?

Practice category: Leader achievement:

development productivity	>5x better
maintenance productivity	>10x better
quality	>30x better
delivery on-time and within budget	4x better
development vs maintenance effort	2x better
estimating accuracy	within 10% (vs >40%)



Application Development Practices and Performance in Europe

IBM
Copyright IBM Corp.

As a result of a year 2000 assessment, 6 top priority action areas are often identified. For each area, an owner is appointed to develop a plan and implement that process improvement. The positive side effects from SPI live on even after year 2000.

Priority Actions



BENEFITS

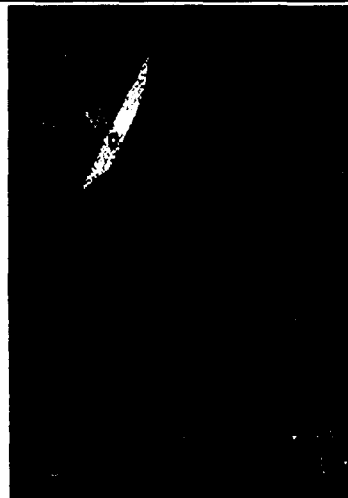
- Improved management and application development processes
- Strengthened Change Management and Configuration Management, including version control
- Better organization, management and utilization of data: a thorough clean-up of the entire application portfolio removing dead code and replacing applications that are not cost effective to maintain
- Improved testing environment; fewer defects
- A more structured, disciplined way of working using well-defined standards and processes to assure higher quality
- Productivity tools and testing tools for year 2000 can also support the future development and maintenance process



© Copyright IBM Corp.

Summary

- > Year 2000 is a high risk project demanding top management attention.
- > Software Process Improvement with a focus on year 2000 will reduce the risk and improve the chance of your company surviving year 2000.
- > IT Effectiveness performed in parallel with year 2000 transformation will enhance your competitive position.



© Copyright IBM Corp.

For more information, please contact....

Europe

- > Charlotte A. Pedersen, IBM Consulting Group, IBM Denmark A/S, Nymoellevej 91, DK - 2800 Lyngby, Denmark, Tel. +45 - 45 23 33 96, Fax. +45 - 45 87 44 38. Internet: chap@dk.ibm.com

Internet

- > <http://www.software.ibm.com/year2000>
- > <http://www.europe.ibm.com/software/benchmk>



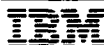
©Copyright IBM Corp.

Position your organization to be a leader...



*Examine your IT Effectiveness
now to enable your organization
to cope with the year 2000
challenge*

...you have only 927 days left!!!



©Copyright IBM Corp.

FACTS ABOUT

IBM AND THE YEAR 2000

Introduction

The "eve" of the new millennium -- when 1999 becomes the year 2000 -- presents a significant challenge to users of computer systems and applications using two digits to represent the year.

That simple change on the calendar may skew the accuracy of data created by computer applications from word processors to databases, and could affect calculations, comparisons and data sorting in systems ranging from the desktop to the largest server. Although the price tag for making systems Year 2000 ready (+) could be high, failure to act will adversely affect a broad array of commercial, industrial, educational and government operations.

This situation -- which can exist in any level of hardware or software, in microcode, in new and old applications, in files and databases, and on any computing platform -- must be addressed by all those who use and depend on information technology (IT).

The Challenge

Many computer operating systems and applications use a standard two-digit year field -- MM/DD/YY -- where YY represents the year. For example, a computer would write January 1, 1999, as 01/01/99 (in binary 0's and 1's, of course). When the year 1999 rolls over to 2000, many systems that use the two-digit year field will express the first day of the new year as 01/01/00, and assume the "00" means 1900 -- reading "00" as coming before "99" in numerical sequence.

Many existing programs calculate the length of time between dates by subtracting two-digit year fields from each other. In 1999, for example, an application computing the age of someone born in 1953 would provide an answer of 46 years ($99 - 53 = 46$). But a year later, when 1999 becomes 2000, if the two-digit year field is not addressed, the age of that person could be computed as -53 years ($00 - 53 = -53$).

Similar problems may occur with invoices, payrolls, credit card transactions, bill payments, inventory systems, auto loans, auto license renewals and automated elevator operations.

That reality prompts some questions.

First, why didn't programmers foresee from the beginning that two-digit year date fields would be a problem and use four digits to represent the year?

Until recently, computer memory and storage were costly and in short supply, and performance could be adversely affected by the manipulation of "unnecessary" data. It made sense to save several characters in every date entry in a database, especially those containing millions of records. And even programmers who considered the issue may reasonably have assumed that the applications they were writing would be replaced long before this calendar change could cause problems. Remarkably, many of those old programs are still in use, often as an important part of company information systems, and the data and processing are still in the same two-digit format.

Second, why wasn't this situation remedied before now?

In many cases, date fields cannot easily be expanded to include more than two numbers. Computers cannot be instructed to globally insert the digits "19" before the two-digit year field, because some dates may refer to a different century. The solution is to update or replace old programs and databases or painstakingly find every place in a system that uses the date to trigger a calculation or a routine and rewrite the code. All applications must be changed in a co-ordinated fashion and tested to ensure they can properly handle dates within and between the 20th and 21st centuries.

And third, why have companies waited until now to grapple with the date problem?

Not all companies have waited. A few industries started using four-digit year date fields when bank application and insurance programs first began to deal with amortization and interest table calculations into the year 2000 and beyond. Some computer users may have waited, though, because the task of fixing the problem is daunting. Some organizations may need to analyze, change and test millions of lines of code, many hundreds of applications and many thousands of routines and sub-routines, all on a co-ordinated basis. The task could be very difficult and expensive.

Misconceptions

There are some general misunderstandings and myths regarding the Year 2000 challenge. For example:

- *This is a problem that occurs only after December 31, 1999.*

In fact, difficulties caused by the year 2000 challenge are already occurring. For example, some applications that deal with future dates, in such areas as mortgages, insurance policies and driver's license renewals have already encountered problems.

- *This is just a hardware clock problem.*

In fact, both hardware and software are affected. The internal timers in hardware may need to be tested for Year 2000 readiness and reset. In software, application programs

and operating systems using two digits for year representation could experience difficulties even if the hardware they are running on has clock and/or system timer services that provide a four-digit format. And it is a problem that will appear in a variety of programs, including those purchased from solution developers, written in-house by a customer's own system and application programmers, licensed from various software vendors, or shared among the I/S community.

- *This is a problem that occurs only in mainframe systems and/or legacy applications.*

In fact, any system or program (large or small) can be affected if it uses only two digits for year representation.

IBM's View

IBM recognizes that the Year 2000 challenge is not limited to a given class of customers, vendors, industries or nations. It affects virtually everyone.

The transition is -- or should be -- a key concern for all of our customers, and is, therefore, of great importance to us.

IBM wants to ensure that our customers are aware of the challenge facing them. We are actively using various methods to raise awareness and inform our customers of technology, support and services available to help them in making a Year 2000 transition. For some time, we have been encouraging our customers to take actions necessary to assess and meet this challenge. Any delays in getting started on this effort are likely to compound the difficulty of the task and increase the expense.

The following pages will briefly describe some of the ways IBM is implementing a five-point strategy -- awareness, planning and support, Year 2000 ready products, services and tools -- to assist customers and work with others in the industry in addressing the Year 2000 challenge on a global, cross-platform basis.

IBM Activities

In October 1995, IBM announced its intention to provide customers with products, services, tools and support to assist in making Year 2000 transitions. With that announcement, IBM said it was "sharing what we have learned about the Year 2000 with our customers, and all computer users, to help them make date transitions as smooth as possible." Among the steps IBM has taken to inform and support our customers and others are:

Communications

IBM has participated in various public conferences, congressional hearings and consultant and press briefings to increase awareness regarding the Year 2000 challenge. Through such forums and groups as the AS/400 Influencers, AIX home page, IBM CIO conferences, Success '96, VISTA, COMMON, GUIDE and SHARE, IBM has urged customers and users to take action to meet the challenge. In addition, IBM has formed vendor and customer councils to facilitate the exchange of information and views on the Year 2000 challenge. IBM is making information regarding Year 2000 available through a Technical Support Center in Europe (+353 1 815 9600) and via the Internet at <http://www.software.ibm.com/year2000>. IBM plans to continue these communications initiatives on a worldwide basis in 1997.

Year 2000 Planning and Implementation Guide

IBM has made available to everyone at no charge via the Internet a comprehensive Year 2000 resource guide. The 200-page document -- entitled "The Year 2000 And 2-Digit Dates: A Guide For Planning And Implementation" -- is available on the World Wide Web through the IBM Software Home Page, at <http://www.software.ibm.com/year2000/index.html>.

This document includes a compilation of IBM's Year 2000 findings, recommended approaches and product listings. Also included in the guide is a bibliography of other Year 2000 publications available throughout the industry.

Following publication of the guide's first edition in 1995, IBM has updated the document five times, most recently in December 1996. There have been several thousand downloads of the guide since its initial offering. Hardcopies are available for a nominal fee. As IBM gathers information about new and emerging Year 2000 products and services, and as the Year 2000 readiness status of IBM's products is updated, IBM will incorporate that information in future editions of the Planning and Implementation Guide.

Technical Support Centers

IBM is establishing four Year 2000 Technical Support Centers (TSCs) around the world specifically dedicated to responding to inquiries and providing technical support to customers and IBM employees. Customers will be able to call in their questions toll-free and receive electronic support via the Internet (<http://www.software.ibm.com/year2000>). The first TSC has already opened in the United States, and IBM plans to have all four of the TSCs operating in early 1997.

Year 2000-Ready Software

The most current versions and releases of nearly all key IBM system software products and approximately 1,900 IBM application products are Year 2000 ready today.

IBM provides a table in the Planning and Implementation Guide which lists IBM products and spells out the versions or releases that are Year 2000-ready to assist customers in planning their own analyses, updating and testing of user and vendor applications and data. IBM continues to assess its other products to determine whether they can be added to the list. The Guide and product list will be periodically updated.

Year 2000 Hardware Support

The hardware timers on IBM System/390*, AS/400* and RS/6000* servers and Personal Systems* computers using PowerPC* technology (specifically listed in the Planning and Implementation Guide) are not affected by the Year 2000 date change.

IBM Personal Systems computers and IBM PC Servers introduced in 1996 (specifically listed in the Planning and Implementation Guide) will handle the century rollover automatically. Some current and earlier PCs will automatically update the century; others may need to receive a simple command or use a special utility. These systems need to be tested because there are different basic input/output systems (BIOS) handling the timing routine.

Customers are encouraged to consult the IBM Internet website (<http://www.software.ibm.com/year2000>) for current information on the Year 2000 readiness of IBM hardware products. IBM recommends that users contact other manufacturers and vendors for information regarding Year 2000 readiness of non-IBM products.

Consulting & Services

In addition to the Implementation and Planning Guide, IBM is making available to customers a set of fee-based services from its Global Services team to help companies develop Year 2000-ready solutions for their own applications, system software and hardware.

IBM is engaged worldwide to deliver consulting and services to clients operating in both centralized and distributed computing environments. These new services use a comprehensive methodology based on years of experience in legacy transformation consulting. They seek to optimize a customer's Year 2000 investment activities with their current and planned strategic business initiatives.

Consulting and services solutions make date-field transitions easier by bringing together proven techniques and state-of-the-art technologies to help reduce cost, redundancy and complexity for the customer.

IBM is aware that some customers may prefer to handle their own transitions. To help those customers, IBM is drawing on its Year 2000 experience to develop packaged offerings consisting of tools and know-how.

Tools & Solutions

IBM offers a set of software tools to assist customers with their Year 2000 transitions. These tools and compilers are platform-specific and target the host application development environment. They support MVS*, OS/390*, OS/400*, AIX*, OS/2*, VSE* and VM* customers.

Today IBM has capabilities available for the MVS COBOL environment delivered by the IBM VisualAge for COBOL, Professional* product. IBM intends to provide similar Year 2000 support for PL/I applications running on MVS, as well as applications running in the VSE environment.

A list of these tools, compilers and products -- from IBM as well as other vendors -- is included in the Planning and Implementation Guide.

Business Partners

In IBM's view, effectiveness in meeting the Year 2000 challenge depends on co-operation across the IT industry. IBM is actively working with its Business Partners to encourage them to make their products Year 2000 ready, to train their employees and to offer Year 2000 services and tools, as soon as possible. IBM will continue to provide information about the availability of Business Partners' Year 2000 products and services in the latest version of the Planning and Implementation Guide, and it will be working with its Partners In Development organizations by helping them communicate information about their Year 2000 ready applications through such forums as the Internet, business shows and conferences.

ITAA Certification

The Information Technology Association of America (ITAA) introduced on October 1, 1996, the ITAA*2000 certification program designed to evaluate the processes and methods that companies use to develop specific Year 2000 solutions. The program will help provide greater assurance that commercial products and services will support the transition to the next millennium. IBM participated in the pilot program to help establish ITAA's Year 2000 certification process and will continue working on other ongoing projects.

Education & Training

IBM is educating its customers, Business Partners and employees to prepare for the Year 2000. For example, more than 1,200 IBM employees received specific technical training in Year 2000 methodology and tools in 1996 and many more will follow during 1997. In addition, IBM will deliver a series of customer seminars in major cities in the United States and around the world in early 1997.

Consulting and services training will also be offered to customers and Business Partners as part of an integrated IBM services offering beginning in January 1997. With such education, IBM will help customers and independent services providers acquire the skills to implement the methodology and use the tools.

Summary

IBM recognizes the Year 2000 transition as a serious business issue affecting most users of computer systems and the information technology industry at large. For that reason, the company does not minimize the task facing those users and is committed to assisting our customers in dealing with the challenge.

IBM's goal is to ease the transition for its customers through effective communication; a range of offerings, tools and support; and working with others to expand the capabilities and resources available to meet the challenge.

IBM's customers and others should understand that information regarding the Year 2000 challenge is changing rapidly. We are all learning and at the same time searching for the most efficient ways to address this challenge. Today's answers may be found deficient tomorrow. For that reason, IBM encourages its customers to continually assess the latest information, products and technology available to assist in the Year 2000 transition. While IBM will continue to broaden its own efforts, IBM emphasizes to its customers that they should act today to address the Year 2000 challenge.

For Further Information

World Wide Web

<http://www.software.ibm.com/year2000>

Customers and Computer Users

Year 2000 Technical Support Center for Europe in Dublin, Ireland:

Telephone +353 1 815 9600 (or fax +353 1 815 9601)

+ A product is year 2000 ready if the product, when used in accordance with its associated documentation, is capable of correctly processing, providing and/or receiving date data within and between the 20th and 21st centuries, provided all other products (for example, software, hardware and firmware) used with the product properly exchange date data with it.

* Indicates trademark or registered trademark of International Business Machines Corporation.

Dec. 16, 1996

Driving and Focusing Year 2000 projects

Peter Blundell

Year 2000 Coordinator
British Airways

16/06/1997 Page 1

BRITISH AIRWAYS

Agenda

- ◆ Background & BA
- ◆ Gaining commitment
- ◆ Cooperation within aviation industry
- ◆ Planning
- ◆ Processes
- ◆ People
- ◆ Building the infrastructure



16/06/1997 Page 2

BRITISH AIRWAYS

BA business opportunities - size and scope

- ◆ World's most profitable airline
- ◆ World leader as airline and in IT
- ◆ Over 35 million passengers per year
- ◆ Most international passengers
- ◆ Over 250 aircraft
- ◆ Concorde to Boeing 737

16/06/1997 Page 3

BRITISH AIRWAYS

Business opportunities - alliances & partners

- ◆ Global Alliances with American 
- ◆ Subsidiaries in France, Germany & UK
- ◆ Franchisees in UK and elsewhere 
- ◆ Marketing agreements world-wide
– e.g. America West, Canadian

manx
AIRLINES *europa***RYMON**
AIRWAYS**CITYFLYEREXPRESS**
LOGANAIR

16/06/1997 Page 4

DEUTSCHE BA**TAT**
EUROPEAN AIRLINES

Today's Situation

Potential areas for change

- ◆ 450 applications
- ◆ Assumed many systems would die before 2000
- ◆ Commercial - Ticket & Fare Validity
- ◆ Operational - Flight dates
- ◆ Human Resources - Event dates
- ◆ Finance - Ledger postings



16/06/1997 Page 5

BRITISH AIRWAYS

Gaining commitment

- ◆ Steps to get the support of management
- ◆ Getting buy-in from:
 - CEO & Board
 - Managers in the business
 - CIO and IT Senior Management
 - IT Project managers
 - The whole organisation

16/06/1997 Page 6

BRITISH AIRWAYS

Large change

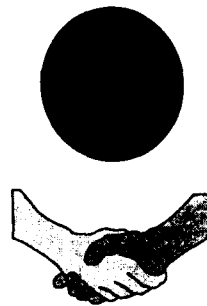
- ◆ Major communications programme
 - Communications plan
- ◆ Largest ever IT project
- ◆ Affects all the business
- ◆ Major project management challenges
- ◆ Many independent sub-projects
- ◆ No formal cost-justification

16/06/1997 Page 7

BRITISH AIRWAYS

Scope & Potential failures

- ◆ Risk management plan
- ◆ Many both major and minor
- ◆ Rest of world affected
- ◆ Dependencies
- ◆ Suppliers (IT & other)
- ◆ Business partners
- ◆ Non-IS maintained systems



16/06/1997 Page 8

BRITISH AIRWAYS

Business led transition

- ◆ Customer Business Managers
- ◆ Understand / oversee changes
- ◆ Support testing
- ◆ Prove processes & contingency
- ◆ Non-IS maintained systems
 - Embedded systems
 - Local PC systems
 - Links to other companies



BRITISH AIRWAYS

16/06/1997 Page 9

Cooperation with other airlines

- ◆ No competitive advantage
- ◆ Airline industry trade association
 - IATA
- ◆ Joint ventures
 - Computer Reservations Systems
 - Galileo International
- ◆ Business Partners
 - Qantas, American
- ◆ IT Customers through Speedwing
 - Air Canada, British Midland, ...



BRITISH AIRWAYS

16/06/1997 Page 10

Cooperation opportunities

- ◆ Electronic commerce
 - Reservations
 - Flight Operations / Traffic
- ◆ Shared presentations / awareness
- ◆ Common approach to systems
- ◆ Common approach to suppliers
 - non-IT
 - IT



16/06/1997 Page 11

BRITISH AIRWAYS

Cooperation in aviation industry

- ◆ Airport operators
 - Baggage Services
 - Ramp Handling
 - Check-in
- ◆ Aviation authorities
 - Air Traffic Control
- ◆ Aircraft manufacturers
 - Avionics
- ◆ Weather Services



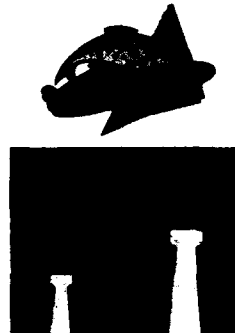
16/06/1997 Page 12

BRITISH AIRWAYS

Cooperation with others

Supply chain

- ◆ Catering
- ◆ Fuel
- ◆ Ground Transport
- ◆ Cargo
- ◆ Engineering
- ◆ Properties
- ◆ Security

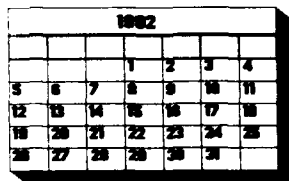


16/06/1997 Page 13

BRITISH AIRWAYS

Planning

- ◆ identifying the requirements for change
- ◆ securing a budget



16/06/1997 Page 14

BRITISH AIRWAYS

The plan



- ◆ You should have started by now
- ◆ Plan by deadline or failure date:
 - Earlier of 6 months ahead of need
 - end-1998
- ◆ Fix PC & minor problems in 1999
- ◆ BA detailed plans complete now
- ◆ Coordinated through Steering Group

16/06/1997 Page 15

BRITISH AIRWAYS

Planning factors

- ◆ Application portfolio size
- ◆ Amount of inhouse owned code
- ◆ Age profile of applications
- ◆ Code quality
- ◆ Language mix
- ◆ Existing IS commitments
- ◆ Skills available and needed
- ◆ Choice of conversion approach

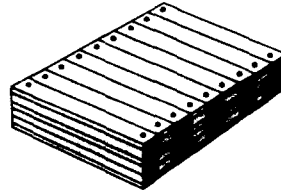
16/06/1997 Page 16

BRITISH AIRWAYS

Technical Impact

◆ Programs

- Date Horizon
- Date calculations
- Sorting by date
- Data entry of dates
- Display and reporting of dates



◆ Skills for legacy systems

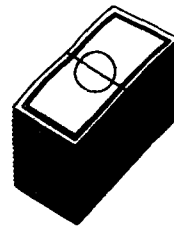
◆ Documentation

16/06/1997 Page 17

BRITISH AIRWAYS

Costs

- ◆ In 1996-9 plans and budgets
- ◆ Grown as project has evolved
- ◆ For BA, millions of pounds
 - but commercially sensitive
- ◆ More than just maintenance
- ◆ Include PCs & Embedded systems
- ◆ Include user costs for testing & PCs



16/06/1997 Page 18

BRITISH AIRWAYS

Processes - existing

- ◆ Existing where possible
- ◆ Project approval
- ◆ Many large maintenance projects
- ◆ Project planning
- ◆ Resource Reporting
- ◆ Local Risk Management

16/06/1997 Page 19

BRITISH AIRWAYS

Processes -New

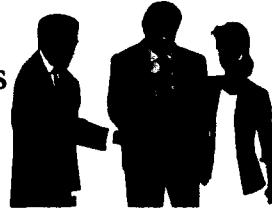
- ◆ Communications planning
- ◆ Inventory
- ◆ Programme responsibilities
- ◆ Milestone achievement
- ◆ Key Performance Indicators
- ◆ Global Risk Management
- ◆ CIO reporting

16/06/1997 Page 20

BRITISH AIRWAYS

People - The Project Team

- ◆ Clearly defined responsibilities
- ◆ Central coordinator
 - Project Office
- ◆ Local coordinators
- ◆ Management plan
- ◆ Strengths
 - Project Management
 - Communications, Technical



16/06/1997 Page 21

BRITISH AIRWAYS

Staffing choices

- ◆ Internal staff
- ◆ Inhouse managed contractors
- ◆ Offshore contractors
- ◆ Subcontract completely
- ◆ Third parties
- ◆ Time

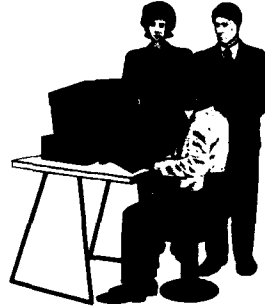


16/06/1997 Page 22

BRITISH AIRWAYS

Inhouse Staffing

- ◆ Existing skills base
- ◆ Recruitment
- ◆ IT backlog and current projects
- ◆ Motivation
 - New versus legacy systems
- ◆ Staff retention
 - Growing demand by service companies
 - Increased market rates
 - Financial inducements



16/06/1997 Page 23

BRITISH AIRWAYS

Third party software suppliers

- ◆ Compliance
 - Commitment
 - Plans
 - Availability
- ◆ Cost
- ◆ Implementation
- ◆ Contractual difficulties
- ◆ Technical challenges



16/06/1997 Page 24

BRITISH AIRWAYS

Building the infrastructure

What tools to use

- ◆ Dedicated Year 2000 test systems
- ◆ New test databases
- ◆ Software Change Management
- ◆ Systems date setting
- ◆ Code scanners & analysers

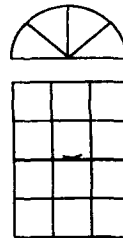


16/06/1997 Page 25

BRITISH AIRWAYS

Conversion process

- ◆ What to look out for
- ◆ Choose method of conversion
 - Date expansion
 - Sliding window
 - Fixed window

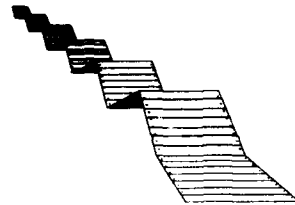


16/06/1997 Page 26

BRITISH AIRWAYS

Change options

- ◆ Database or file conversion
- ◆ Program conversion
- ◆ Build replacement system
- ◆ Purchase package



BRITISH AIRWAYS

16/06/1997 Page 27

BA Approach



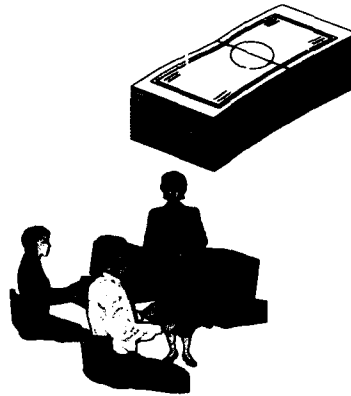
- ◆ Cross branch coordination group
- ◆ Each branch responsible
- ◆ Most applications built as compliant
 - Database dates are held correctly
- ◆ Minimise impact on user
- ◆ Unchanged date formats
 - Screens reports files

BRITISH AIRWAYS

16/06/1997 Page 28

What we suggest you do

- ◆ Plan
- ◆ Prioritise
- ◆ Invest
- ◆ Develop
- ◆ Test
- ◆ Implement



BRITISH AIRWAYS

16:06/1997 Page 29

Year 2000 Summary

- ◆ Management commitment
 - Senior management fully aware
- ◆ Planning
 - Initial evaluation complete
- ◆ Development
 - First projects well underway
- ◆ Implementation
 - Up to December 1998



BRITISH AIRWAYS

16:06/1997 Page 30

Any Questions

◆ Peter Blundell



16/06/1997 Page 11

BRITISH AIRWAYS



Carnegie Mellon University
Software Engineering Institute

Appraisals Using the CMMSM as a Reference Model

Donna K. Dunaway & Steve Masters

**Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213**

Sponsored by the U.S. Department of Defense

© 1997 Carnegie Mellon University

E-SEPG-1



Carnegie Mellon University
Software Engineering Institute

Agenda

9:00 - 10:30	Part I	Donna Dunaway
10:30 - 11:00	Break	
11:00 - 12:30	Part II	Steve Masters

© 1997 Carnegie Mellon University

E-SEPG-2



Tutorial - Part I

Background on process improvement

IDEALSM approach to process improvement

Maturity profile data

History of SEI appraisal methods

CMM appraisal framework (CAF)

Current SEI appraisal methods

Assessments versus evaluations

Assessment p

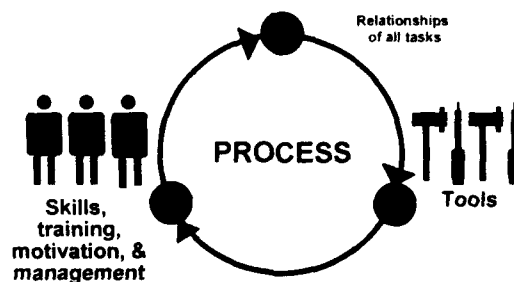
SEI Appraiser Program

SMIDEAL, Capability Maturity Model, and CMM are service marks of Carnegie Mellon University



A Definition of Software Process

The system of all tasks and the supporting tools, standards, methods, and practices involved in the production and evolution of a software product throughout the software life cycle

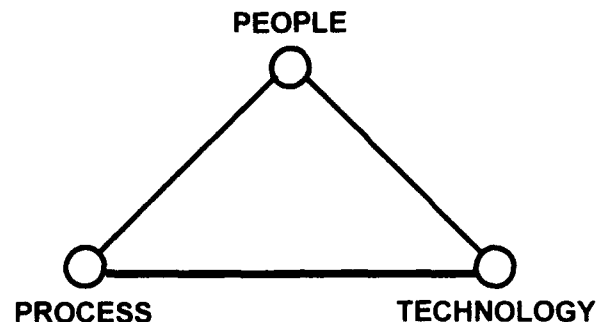




Carnegie Mellon University
Software Engineering Institute

Why is Process Important?

Process is one of 3 quality leverage points.



© 1997 Carnegie Mellon University

E-SEPG-5



Carnegie Mellon University
Software Engineering Institute

Common Points In The Quality Movement

Enabling improvement is a management responsibility.

Improvement focuses on fixing the process, not on blaming the people.

Improvement must be measured and periodically reinforced.

Improvement requires constancy of investments, rewards, and incentives.

Improvement is a continuous process.

© 1997 Carnegie Mellon University

E-SEPG-6



Carnegie Mellon University
Software Engineering Institute

The Process Management Premise

The quality of a (software) system is largely governed by the quality of the process used to develop and maintain it.

This premise implies focus on process as well as product.

The value of this premise is visible worldwide in the Total Quality Management (TQM) movements in the manufacturing and service industries.

[Source: Humphrey, "Managing the Software Process"]

© 1997 Carnegie Mellon University

E-SEPG-7



Carnegie Mellon University
Software Engineering Institute

Process Improvement Paradigm

Characterize the current state of software practice.

Set objectives and priorities for process improvement.

Establish a plan for process improvement and technology introduction.

Assign dedicated resources.

Assess progress against improvement goals.

© 1997 Carnegie Mellon University

E-SEPG-8



Carnegie Mellon University
Software Engineering Institute

Principles of Process Change

Major changes must start at the top.

Fix the process, not the people.

Understand the current process first.

Change is continuous.

Improvement requires investment.

Retaining improvement requires periodic reinforcement.

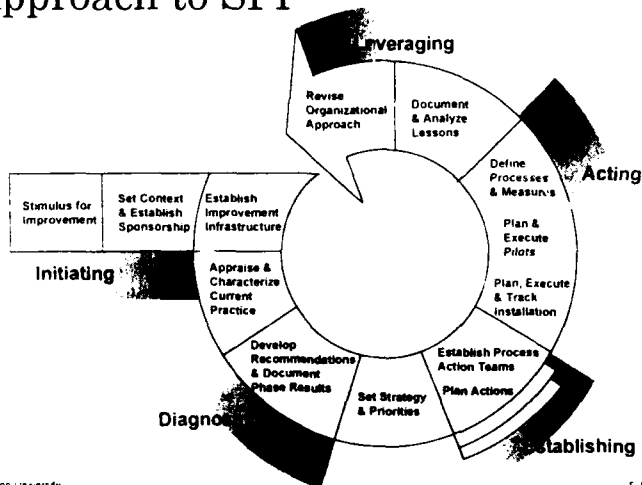
© 1997 Carnegie Mellon University

E SEPG-9



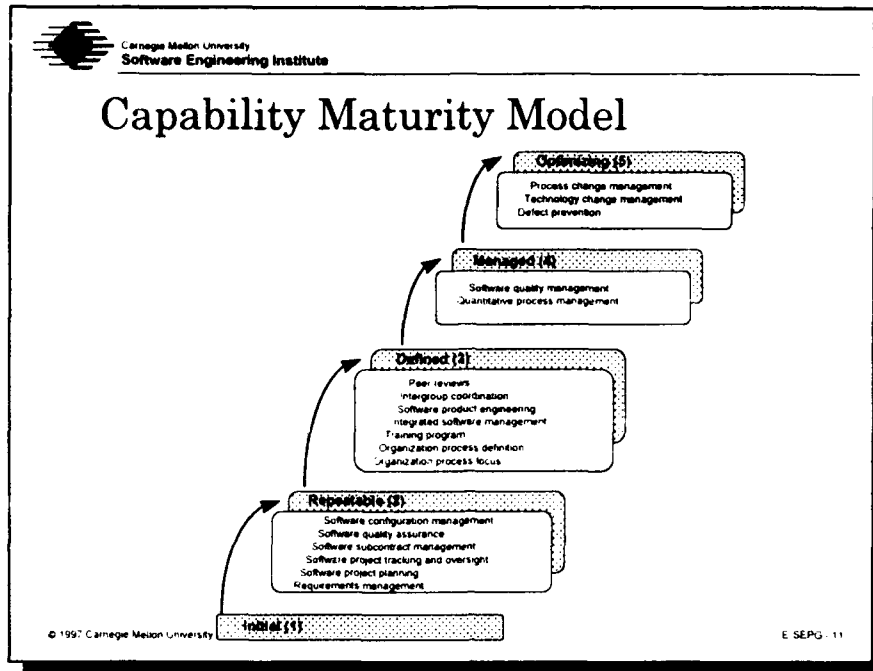
Carnegie Mellon University
Software Engineering Institute

IDEALSM is an Integrated Approach to SPI



© 1997 Carnegie Mellon University

E SEPG-10



 Carnegie Mellon University
Software Engineering Institute

Results of Process Improvement

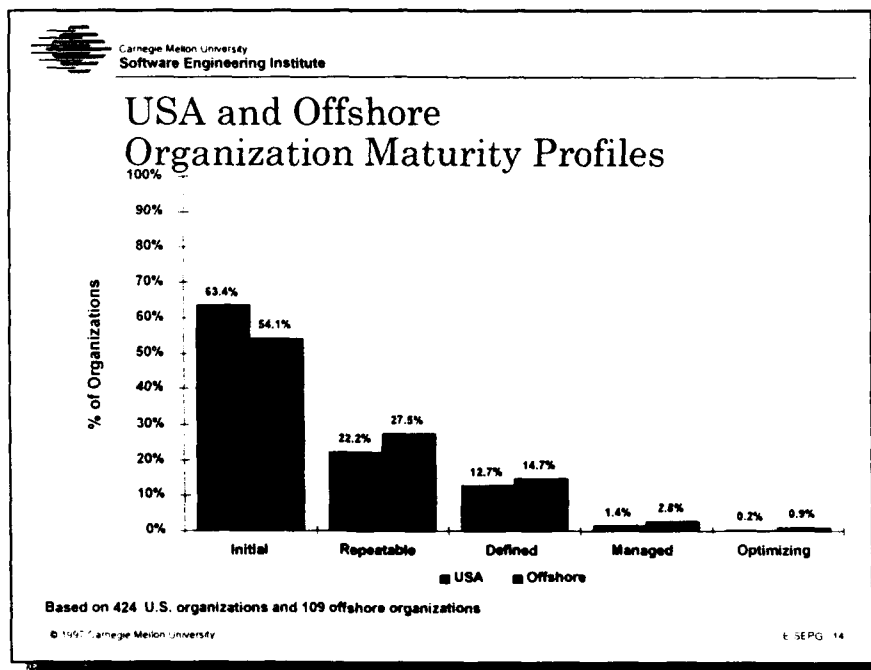
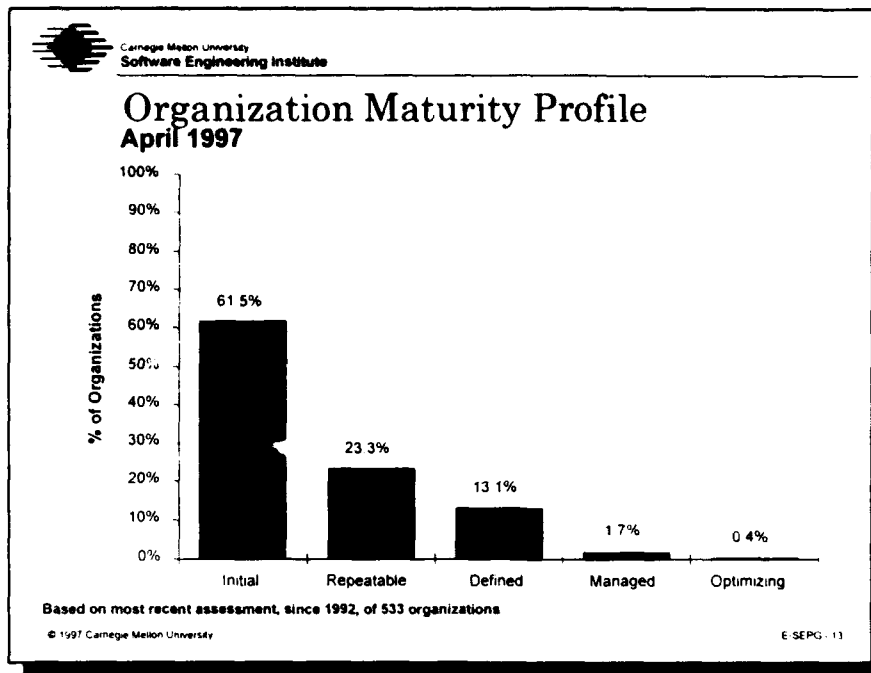
Benefits of CMM-Based Software Process Improvement: Initial Results
[CMU/SEI-94-TR-13]

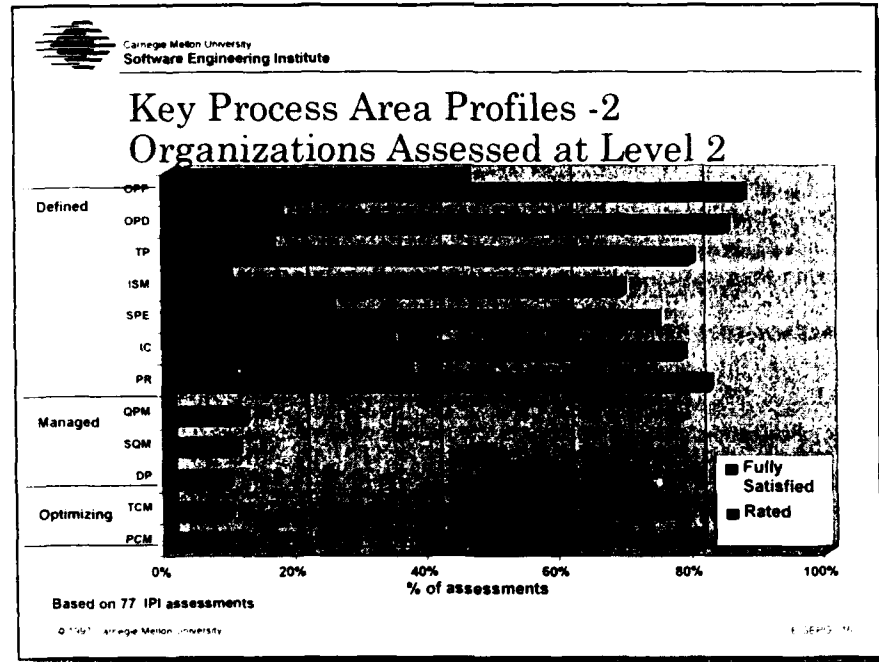
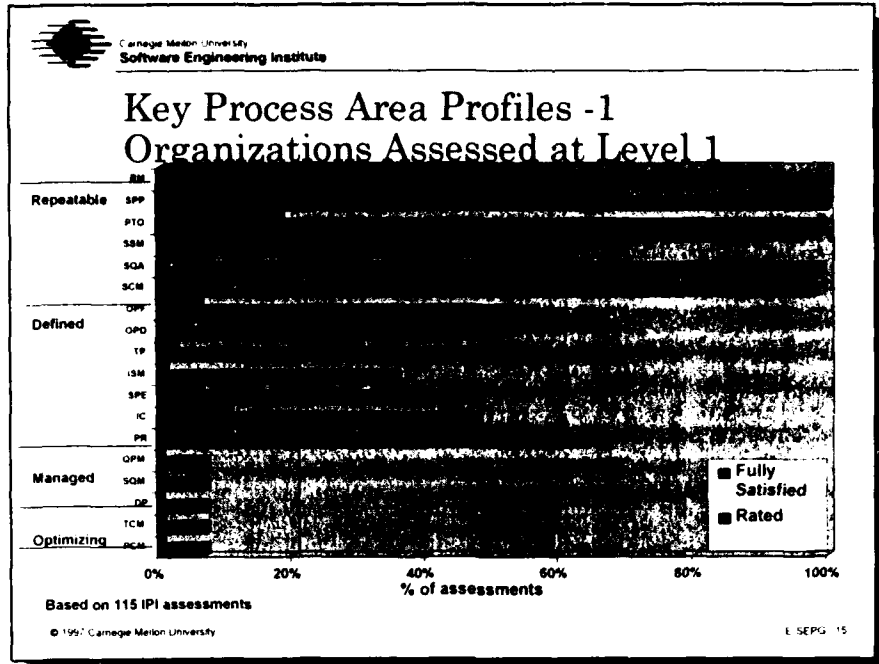
Moving on up: Data and Experience Doing CMM-Based Software Process Improvement
[CMU/SEI-95-TR-08]

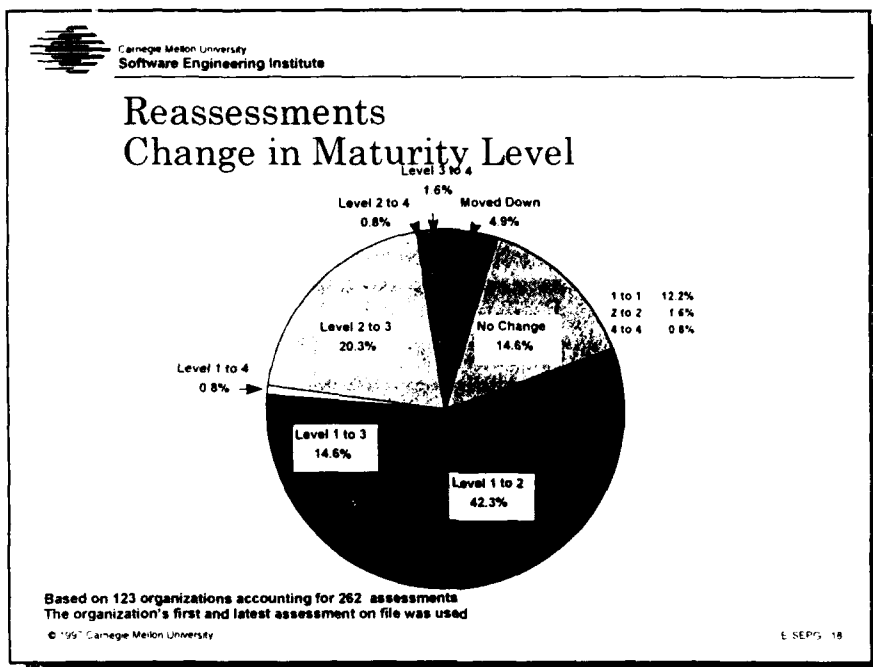
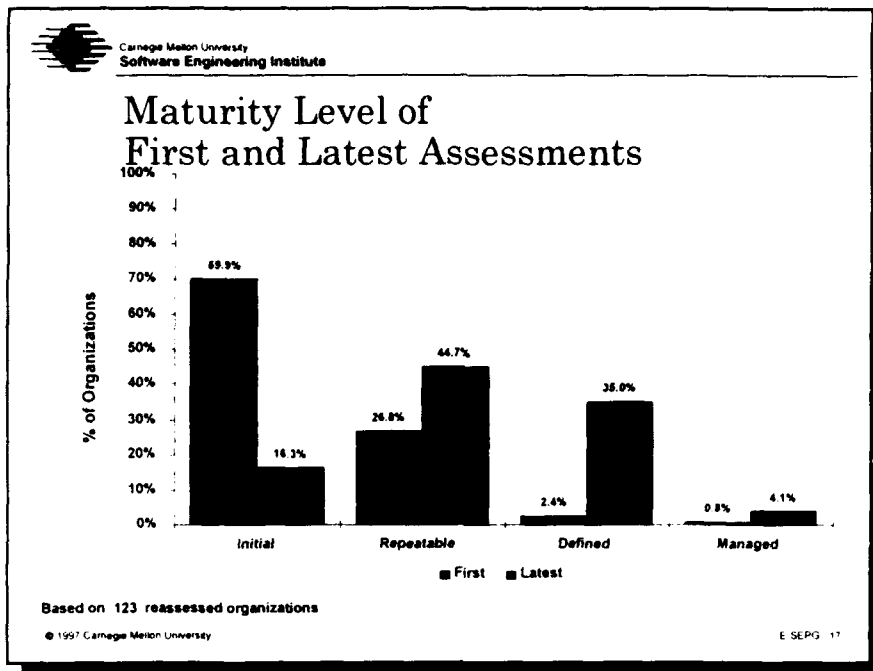
After the Appraisal: A Systematic Survey of Process Improvement, its Benefits, and Factors that Influence Success
[CMU/SEI-95-TR-009]

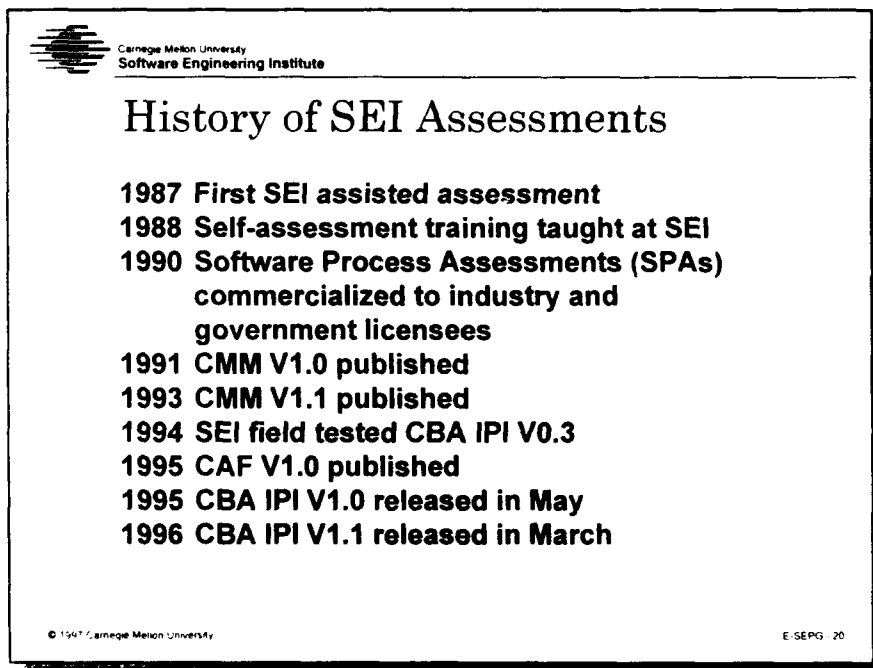
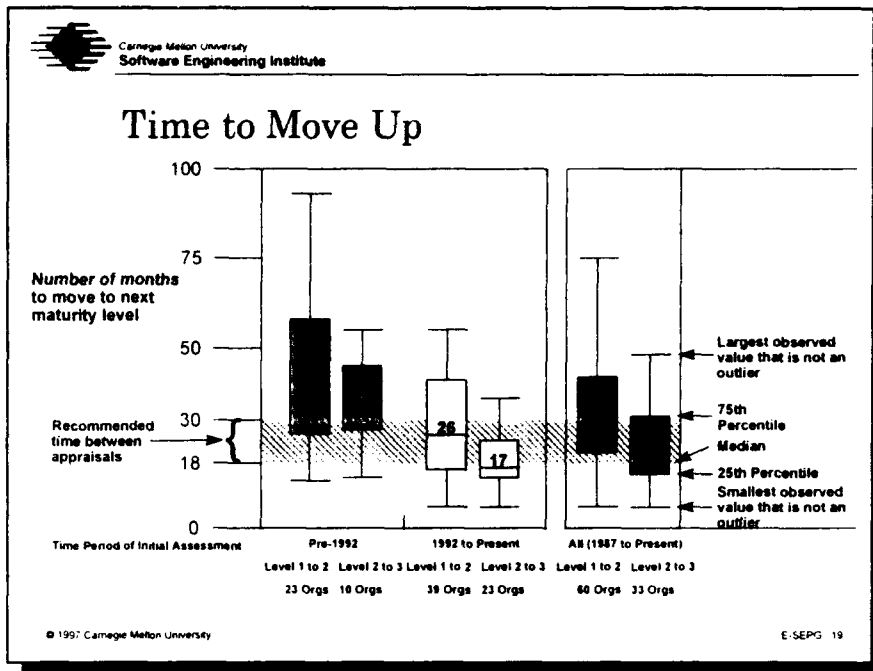
Process Maturity Profile of the Software Community published semi-annually by the SEI on website: www.sei.cmu.edu

© 1997 Carnegie Mellon University E SEPG - 12











Carnegie Mellon University
Software Engineering Institute

What is the CAF?

The CMMSM Appraisal Framework (CAF) identifies the requirements for a CMM-based appraisal method.

The CAF provides the framework for rating the process maturity of an organization against CMM V1.1.

CMM Appraisal Framework, Version 1.0 (CMU/SEI-95-TR-001) was published in February 1995.

CMM is a service mark of Carnegie Mellon University.

© 1997 Carnegie Mellon University

E-SEPG - 21



Carnegie Mellon University
Software Engineering Institute

CAF Requirements

The CAF lists 40 requirements that must be satisfied by CAF-compliant methods.

- 2 requirements concerning documenting CAF compliance and providing guidance for appraisal phases
- 16 requirements for the “plan and prepare for appraisal” phase
- 17 requirements for the “conduct appraisal” phase
- 5 requirements for the “report results” phase

© 1997 Carnegie Mellon University

E-SEPG - 22



Carnegie Mellon University
Software Engineering Institute

CAF Requirement Example

"Requirement R1. A CAF compliant appraisal method shall be documented, including at a minimum:

- *Identifying the versions of the CMM and the CAF on which it depends.*
- *Documenting the manner in which it has implemented appraisal method activities, artifacts and guidance required by the CAF."*

Source: CMU/SEI-95-TR-001

© 1997 Carnegie Mellon University

E-SEPG 23



Carnegie Mellon University
Software Engineering Institute

Plan and Prepare For Appraisal

Provide guidance for developing an appraisal plan that contains

- goals and constraints of appraisal
- scope (CMM, organization)
- sponsor commitment

Provide guidance for selecting and preparing

- team
- participants

© 1997 Carnegie Mellon University

E-SEPG 24



Carnegie Mellon University
Software Engineering Institute

Conduct Appraisal

Subphases

- collect and record data
- consolidate data
- make rating judgments

© 1997 Carnegie Mellon University

E SEPG-25



Carnegie Mellon University
Software Engineering Institute

Collect and Record Data

Provide guidance for collecting and recording data and classifying with respect to

- administering instruments
- conducting presentations
- conducting interviews
- reviewing documentation

© 1997 Carnegie Mellon University

E SEPG-26

Monday 16 June

(T101b) S-13



Carnegie Mellon University
Software Engineering Institute

Consolidate Data

Provide guidance for consolidating data in regard to

- **organizing and recording data**
- **validating data**
- **determining coverage**
- **maintaining traceability**

© 1997 Carnegie Mellon University

E-SEPG-27



Carnegie Mellon University
Software Engineering Institute

Make Rating Judgments -1

An appraisal team rates a key process area (KPA) goal when valid observations meet the method's coverage criteria.

An appraisal team rates a KPA after it has rated each of the associated goals.

© 1997 Carnegie Mellon University

E-SEPG-28



Carnegie Mellon University
Software Engineering Institute

Make Rating Judgments -2

A goal is rated

- **Satisfied** if the associated findings indicate that the goal is implemented and institutionalized as defined in the CMM or that adequate alternative practices exist.
- **Unsatisfied** if the associated findings indicate that there are significant weaknesses in the implementation and institutionalization of the goal and adequate alternative practices do not exist.

© 1997 Carnegie Mellon University

E SEPG - 29



Carnegie Mellon University
Software Engineering Institute

Report Results

Appraisal team must report the following to the sponsor:

- **scope (CMM and organization)**
- **KPA findings**
- **ratings**

Team leader must report results to the SEI.

Provide guidance for

- **protecting confidentiality of appraisal information**
- **retention of appraisal records**

© 1997 Carnegie Mellon University

E SEPG - 30



Carnegie Mellon University
Software Engineering Institute

Current SEI Appraisal Methods

Requirements established by:

- CMM Appraisal Framework (CAF)

CAF-compliant methods:

- CMM-Based Appraisal for Internal Process Improvement (CBA IPI)
- Software Capability Evaluation V3.0

Other:

- Interim Profile

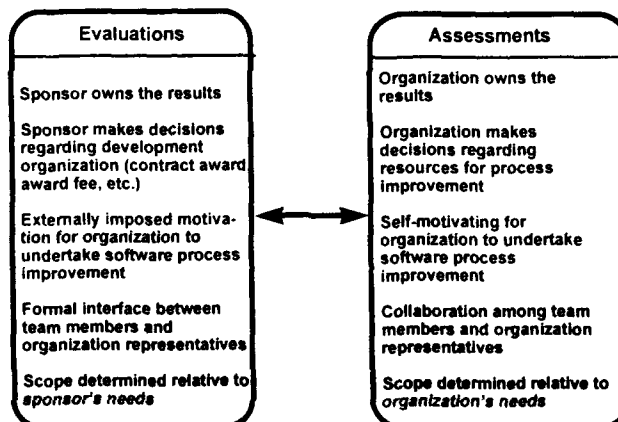
© 1997 Carnegie Mellon University

E-SEPG-31



Carnegie Mellon University
Software Engineering Institute

Comparison of Appraisal Methods



© 1997 Carnegie Mellon University

E-SEPG-32



Carnegie Mellon University
Software Engineering Institute

What is an Assessment?

An appraisal, by a trained team of experienced software professionals, of an organization's current software process, based on:

- review of 4 representative projects
- in-depth discussions with project leaders, practitioners, and middle managers
- collective assessment team knowledge and experience

Assessment findings includes:

- KPA strengths and weaknesses
- KPA profile
- assessed maturity level

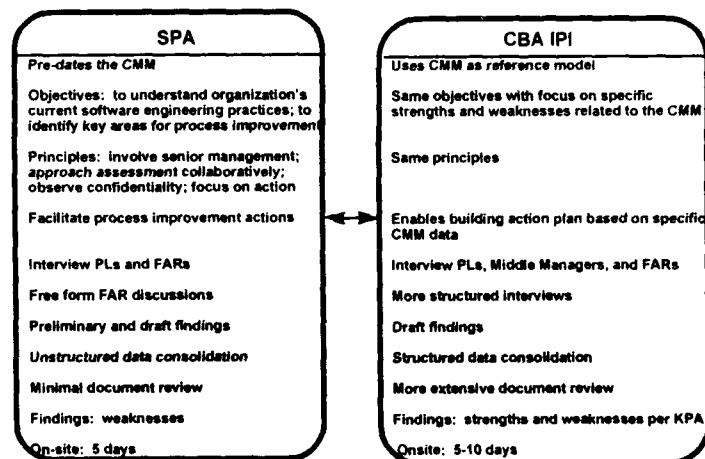
© 1997 Carnegie Mellon University

E-SEPG - 33



Carnegie Mellon University
Software Engineering Institute

Comparison of Assessment Methods



© 1997 Carnegie Mellon University

E-SEPG - 34



Carnegie Mellon University
Software Engineering Institute

Assessment Principles

Start with a process framework.

Observe strict confidentiality.

Involve senior management.

Approach assessment collaboratively.

Focus on action.

© 1997 Carnegie Mellon University

E-SEPG-35



Carnegie Mellon University
Software Engineering Institute

Start with a Process Framework

Provides a basis for orderly exploration.

Helps establish problem priorities.

Enables team to work together on key issues and recommendations.

Process framework includes:

- **process management concepts and principles**
- **the Capability Maturity Model**
- **CAF compliant appraisal method**

© 1997 Carnegie Mellon University

E-SEPG-36



Carnegie Mellon University
Software Engineering Institute

Observe Strict Confidentiality

Only composite results are given to management.

- no attribution to individuals
- no project identification

Assessment team and participants agree to keep all information confidential.

SEI will not disclose results.

Organization is free to disclose results.

© 1997 Carnegie Mellon University

E-SEPG-37



Carnegie Mellon University
Software Engineering Institute

Involve Senior Management

The software process is a management responsibility.

Management must provide visible sponsorship of assessment and improvement action.

Management must participate in periodic reviews of progress.

© 1997 Carnegie Mellon University

E-SEPG-38



Carnegie Mellon University
Software Engineering Institute

Approach Assessment Collaboratively

Assessment team brings:

- **process framework**
- **assessment facilitation skills**
- **judgment and experience**

Assessment participants bring:

- **knowledge about the current process**
- **desire to improve process**

A synergistic process.

© 1997 Carnegie Mellon University

E SEPG - 39



Carnegie Mellon University
Software Engineering Institute

Focus on Action

Be prepared to take action or don't assess.

© 1997 Carnegie Mellon University

E SEPG - 40



Carnegie Mellon University
Software Engineering Institute

SEI Appraiser Program

Goals of SEI Appraiser Program:

- maximize value and use of SEI appraisal methods facilitated by qualified, trained individuals
- transition appraisal technology in an effective manner, maintaining consistency and quality in the process

180 Authorized Lead Assessors

25 Authorized Lead Evaluators (pilot program)

© 1997 Carnegie Mellon University

E-SEPG-41



Carnegie Mellon University
Software Engineering Institute

Authorization of Lead Assessors - 1

Prerequisites:

- experience as team member on at least two CBA IPI assessments within prior 24 months
- software development and management experience
- master's degree in technical area or equivalent experience
- SEI Introduction to CMM course

© 1997 Carnegie Mellon University

E-SEPG-42



Carnegie Mellon University
Software Engineering Institute

Authorization of Lead Assessors - 2

Training: CBA Lead Assessor Training

Observation: Lead a CBA IPI being observed by authorized Lead Assessor within 24 months of training with report and recommendation returned to SEI

© 1997 Carnegie Mellon University

E-SEPG - 43



Carnegie Mellon University
Software Engineering Institute

Lead Assessor Responsibilities - 1

Conduct CBA IPI Team Training for their assessment teams after each team member has received CMM training

Lead or participate in CBA IPI assessments (at least two within a 24-month period)

File reports with the SEI for each assessment

Purchase and use SEI materials

© 1997 Carnegie Mellon University

E-SEPG - 44



Carnegie Mellon University
Software Engineering Institute

Lead Assessor Responsibilities - 2

Complete upgrade courses and use upgraded materials

Cooperate in random audits by the SEI; take any remedial action recommended

Request renewal of authorization and pay renewal fee

© 1997 Carnegie Mellon University

E-SEPG-45



Carnegie Mellon University
Software Engineering Institute

Associated Costs

SEI Appraiser Program

- renewal fee (every 24 months) \$1,000

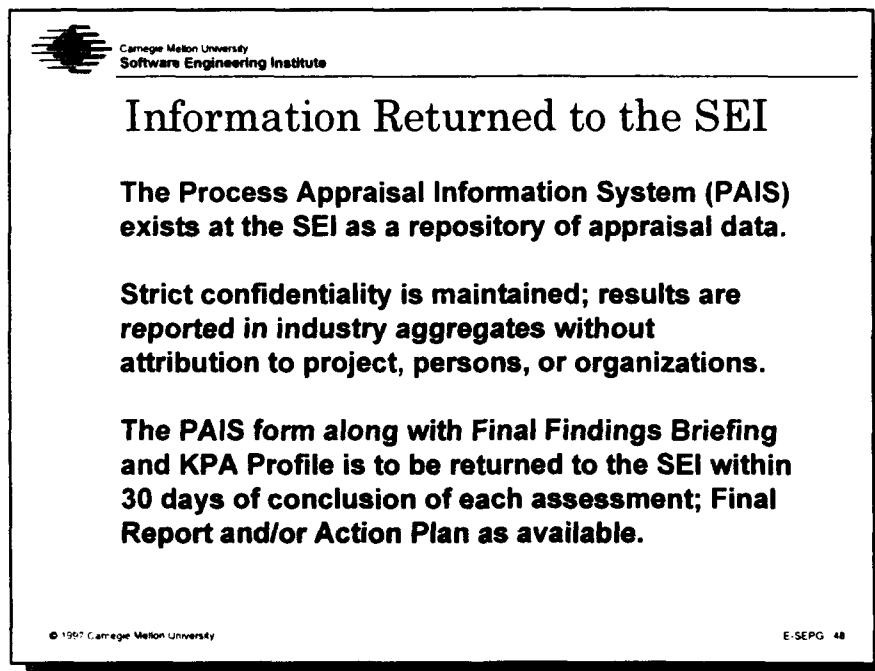
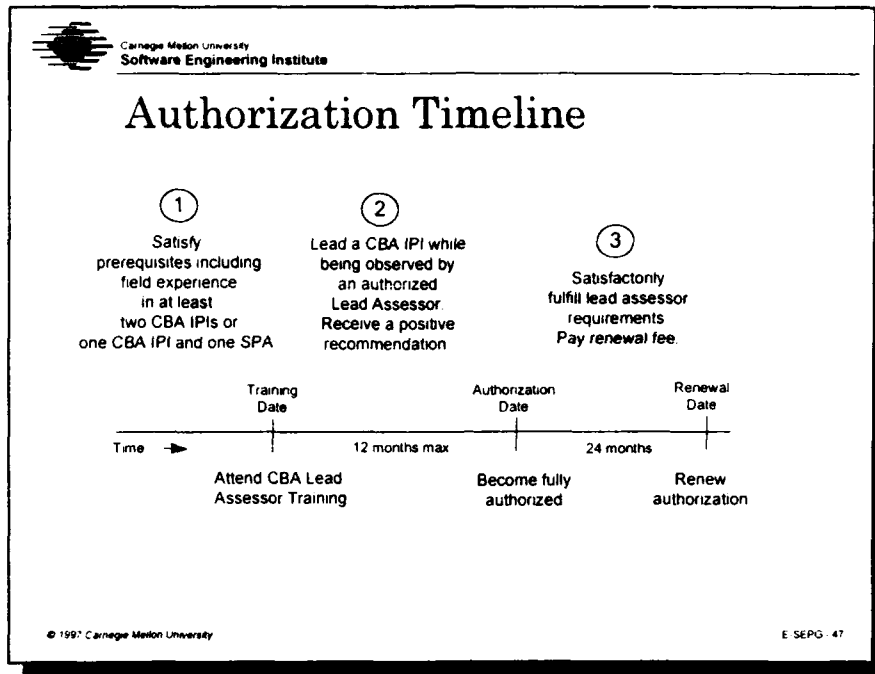
SEI materials

- single assessment kit \$1,000
- (international) \$2,000
- quantity assessment kit \$5,000
- (international) \$10,000

***international fee applies if an applicant offices outside of the US**

© 1997 Carnegie Mellon University

E-SEPG-46





Carnegie Mellon University
Software Engineering Institute

Renewal of Authorization

Authorization as a Lead Assessor is valid for a two-year period

Renewal considerations:

- **sponsor feedback forms**
- **team member feedback forms**
- **random audit results, if any**
- **substantiated reports of any misuse of SEI materials and methodology**
- **review of assessment data returned to SEI**

© 1997 Carnegie Mellon University

E-SEPG - 49



Carnegie Mellon University
Software Engineering Institute

Summary - Part I

We have covered:

- **background on process improvement**
- **IDEALSM approach to process improvement**
- **maturity profile data**
- **history of SEI appraisal methods**
- **CMM appraisal framework (CAF)**
- **current SEI appraisal methods**
- **assessments versus evaluations**
- **assessment principles**
- **SEI Appraiser Program**

© 1997 Carnegie Mellon University

E-SEPG - 50



Carnegie Mellon University
Software Engineering Institute

Agenda

9:00 - 10:30	Part I	Donna Dunaway
10:30 - 11:00	Break	
11:00 - 12:30	Part II	Steve Masters

© 1997 Carnegie Mellon University

E-SEPG-51



Carnegie Mellon University
Software Engineering Institute

Tutorial - Part II

What Can the CBA IPI Method Accomplish?
Pre-On-site Activities
On-site Activities
Minimum Requirements for a CBA IPI
Post-assessment Activities

© 1997 Carnegie Mellon University

E-SEPG-52



Carnegie Mellon University
Software Engineering Institute

The CBA IPI Process

A team of 4-10 experienced software professionals is formed and receive team training from an authorized Lead Assessor.

The team prepares for the onsite period, performing all necessary planning activities to ensure a successful assessment.

Data is gathered, analyzed, and presented to the organization's senior management during an intensive onsite period.

© 1997 Carnegie Mellon University

E-SEPG - 53



Carnegie Mellon University
Software Engineering Institute

CBA IPI Goals

Provide an accurate picture relative to the CMM:

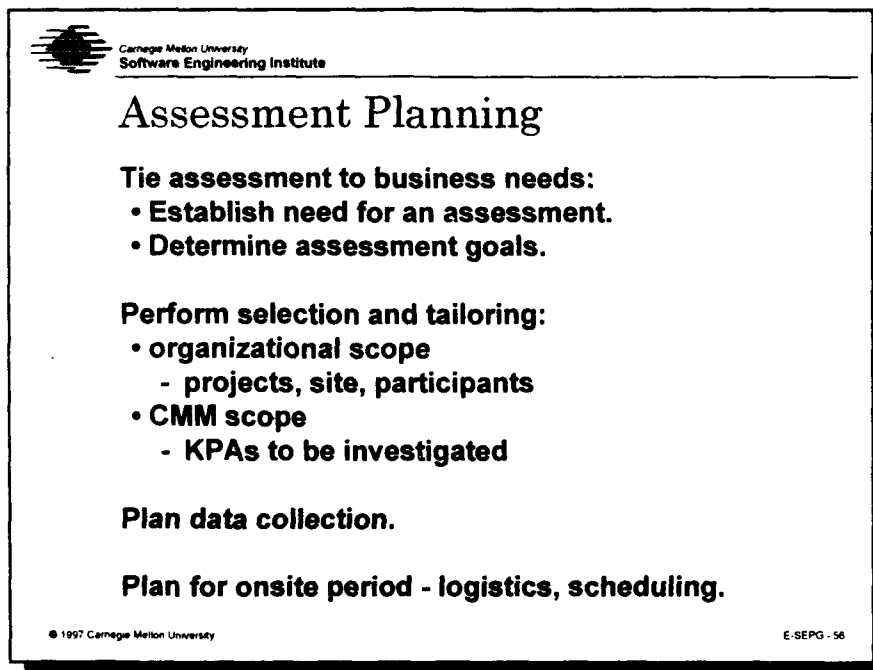
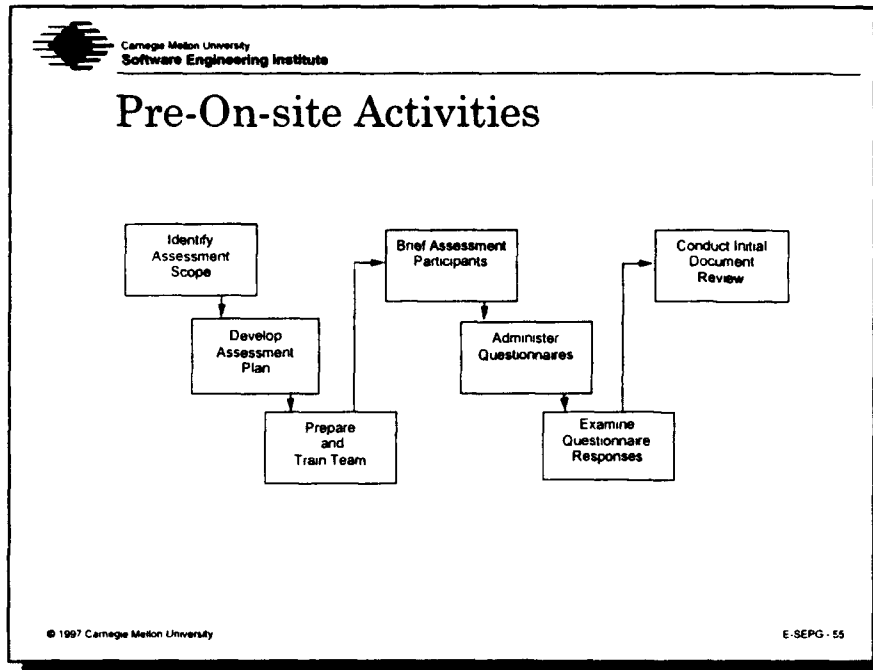
- **Collect process data to understand the current implemented process.**
- **Identify process strengths and weaknesses.**
- **Determine the satisfaction of the CMM KPAs investigated.**

Facilitate continued commitment to SPI:

- **Motivate — obtain the “buy-in.”**
- **Build ownership of results.**
- **Provide a framework and catalyst for action.**
- **Sustain sponsorship and establish commitment.**

© 1997 Carnegie Mellon University

E-SEPG - 54





Carnegie Mellon University
Software Engineering Institute

Sponsor Interaction

Determine that sponsor understands the assessment process; provide executive briefing as needed.

Set sponsor's expectations on participation in assessment activities, resources required, and assessment outputs.

Obtain sponsor's commitment.

Create initial assessment plan with sponsor's blessing (i.e., signature).

Provide sponsor with confidence report.

© 1997 Carnegie Mellon University

E-SEPG - 57



Carnegie Mellon University
Software Engineering Institute

Assessment Goal Setting

Determine sponsor's business goals.

Determine alignment of business goals and assessment goals.

Establish agreement with sponsor on assessment scope (both organizational and CMM).

© 1997 Carnegie Mellon University

E-SEPG - 58



Carnegie Mellon University
Software Engineering Institute

Team Selection Criteria

**Software engineering field experience (avg.
team member > 6 years; team total > 25 years)**

Management experience (team total > 10 years)

**Life-cycle phases experience (75% of team
must have experience in at least 1/3 of
organization's life-cycle phases)**

**At least one team member must be from the
site and have knowledge of the organizational
environment with no vested interest in results.**

© 1997 Carnegie Mellon University

E-SEPG - 59



Carnegie Mellon University
Software Engineering Institute

General Guidelines for Assessment Team Selection

Select experienced software professionals

- **the most respected, influential individuals
who balance the team**
- **persons from different parts of the
organization and from different disciplines**
- **persons with good people skills who are
motivated to improve the process**
- **persons who will not inhibit the free flow of
information during assessment discussions**

© 1997 Carnegie Mellon University

E-SEPG - 60



Carnegie Mellon University
Software Engineering Institute

Participant Selection Criteria

Questionnaire respondents

Project leader interviewees

Middle manager interviewees

Functional area representatives

© 1997 Carnegie Mellon University

E-SEPG - 81



Carnegie Mellon University
Software Engineering Institute

Logistics

Scheduling

Facilities

Tools and materials

Transportation and accommodations

Catering

Support

Coordination

© 1997 Carnegie Mellon University

E-SEPG - 82



Carnegie Mellon University
Software Engineering Institute

Develop Assessment Plan - 1

Use organization and project questionnaires to obtain site demographics.

Obtain site information packet.

Document assessment scope (organization and CMM).

Assist sponsor in selection of team members, projects, and participants.

Identify and document tailoring to the method.

© 1997 Carnegie Mellon University

E-SEPG - 63



Carnegie Mellon University
Software Engineering Institute

Develop Assessment Plan - 2

Identify and request documents for initial review.

Determine schedule

- participant selection
- team training
- assessment participants briefing
- questionnaire administration
- on-site week

Develop and document the plan; obtain sponsor's approval.

© 1997 Carnegie Mellon University

E-SEPG - 64



Carnegie Mellon University
Software Engineering Institute

Train the Team

Provide CMM training.

Provide team training for all team members not previously trained in CBA IPI.

- **Tailor team training if needed.**
- **Prepare exercises to use with team training.**
- **Provide team building.**
- **Establish team ground rules.**
- **Prepare team building exercise.**

© 1997 Carnegie Mellon University

E-SEPG - 05



Carnegie Mellon University
Software Engineering Institute

Brief Assessment Participants

Purpose: to ensure that there is a common understanding and a clear set of expectations regarding the upcoming assessment

Discuss:

- **role of the CMM in process improvement**
- **business value of moving up in maturity**
- **objectives of this particular assessment**
- **activity flow with objectives of each activity**
- **specific participant schedules**
- **any questions**

© 1997 Carnegie Mellon University

E-SEPG - 05

Carnegie Mellon University
Software Engineering Institute

Administer Questionnaire

Questionnaire respondents are carefully selected and scheduled.

Facilitator presents site terminology that differs from the CMM.

Questionnaires are administered in a group setting.

Questionnaire responses are used primarily to help focus the data gathering sessions during the on-site period.

© 1997 Carnegie Mellon University

E-SEPG - 67

Carnegie Mellon University
Software Engineering Institute

Examine Questionnaire Responses

Responses can be summarized in tabular form, i.e., the Response Summary Sheets (RSSs).

Team members individually review the aggregate maturity questionnaire (MQ) responses noting significant response patterns and relevant comments.

The entire team reviews MQ responses on a KPA by KPA basis and agrees to additional information needed.

KPA mini-teams create notes for their KPAs and take notes for use in scripting questions for the project managers.

© 1997 Carnegie Mellon University

E-SEPG - 68



Carnegie Mellon University
Software Engineering Institute

Conduct Initial Document Review

Objectives include understanding the life cycle and architecture in use, and mapping documents to the CMM.

Library management procedures should be established.

Strategies include following threads, using expertise in specific area, and utilizing the site members of the assessment team.

Team members should take notes and keep document reference sheets up to date, and keep information needed references current.

© 1997 Carnegie Mellon University

E SEPG - 69



Carnegie Mellon University
Software Engineering Institute

Assessment Participant Roles - 1

Senior site manager - sponsors assessment; publicly supports process improvement activities; recipient of final findings briefing

Middle managers - come from line or staff management positions; selected based on their affiliation with the software development process; will participate in group discussion

© 1997 Carnegie Mellon University

E SEPG - 70



Carnegie Mellon University
Software Engineering Institute

Assessment Participant Roles - 2

Project Leaders - participate in personal interviews; review draft findings

Functional Area Representatives (FARs) - participate in group discussions; review draft findings

© 1997 Carnegie Mellon University

E SEPG - 71



Carnegie Mellon University
Software Engineering Institute

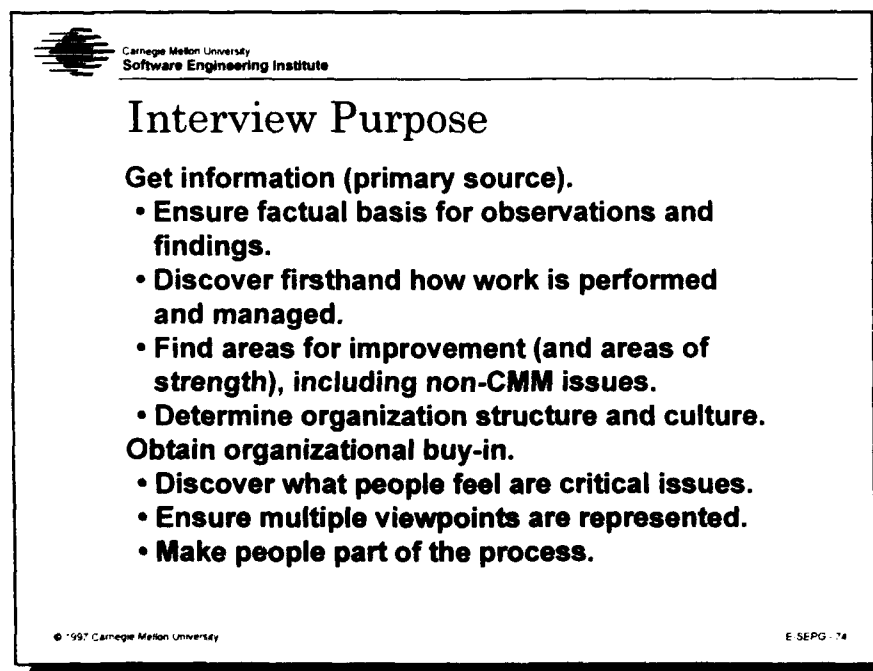
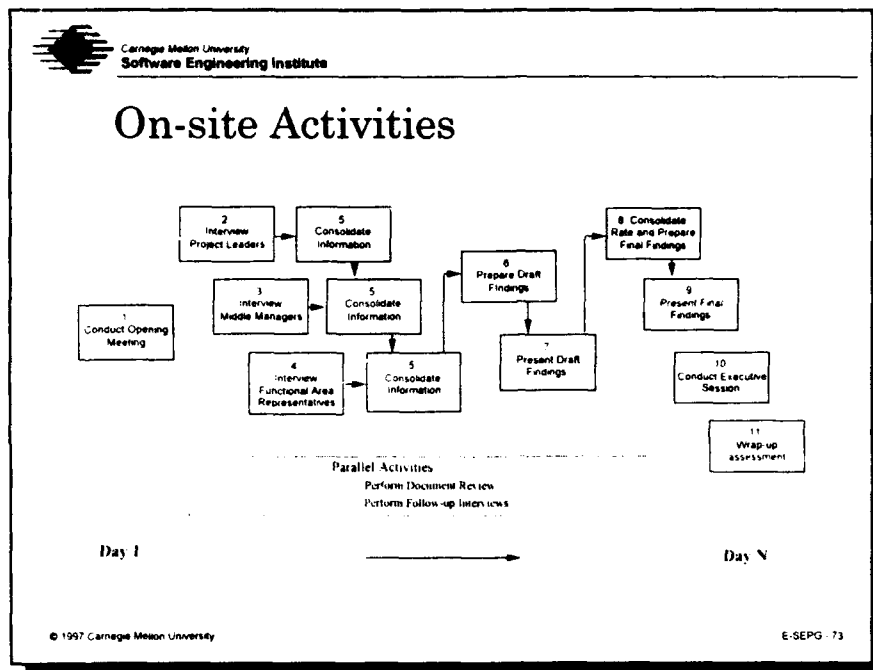
Assessment Team Members - Roles and Responsibilities

Key team roles on-site:

- **Lead Assessor**
- **team members**
- **librarian**
- **KPA mini-teams for consolidation and interviewing**
- **interview facilitators**
- **findings presenter(s)**
- **site coordinator**

© 1997 Carnegie Mellon University

E SEPG - 72





Carnegie Mellon University
Software Engineering Institute

Interviewing

Interviews:

- types: project leaders, middle managers, functional area representatives
- opening
- closing

Questions:

- types: direct, open-ended, context-free
- sequence: general to more specific
- scripting

© 1997 Carnegie Mellon University

E-SEPG-75



Carnegie Mellon University
Software Engineering Institute

Follow-Up Interviews

When do you hold a follow-up interview?

- when there are holes in coverage
- to resolve inconsistencies

Objectives

- Identify areas for improvement.
- Include critical players.
- Understand how work is performed (get specifics).
- Clarify previous information.
- Ensure coverage of KPAs.

© 1997 Carnegie Mellon University

E-SEPG-76



Carnegie Mellon University
Software Engineering Institute

Presentations

- Opening Briefing**
- Draft Findings Presentation**
- Final Findings Presentation**

© 1997 Carnegie Mellon University E-SEPG-77



Carnegie Mellon University
Software Engineering Institute

Sources of Data

- Instruments:**
 - maturity questionnaire
 - site information package
- Documents:**
 - organization, project, implementation level
- Interviews:**
 - project leaders
 - middle managers
 - functional area representatives
 - follow-on interviews
- Presentations:**
 - draft findings presentation
 - demonstration of tools

© 1997 Carnegie Mellon University E-SEPG-78



Carnegie Mellon University
Software Engineering Institute

Objectives of Consolidation

Organize the information collected:

- **Summarize and combine information.**
- **Classify information.**
- **Relate the information to a reference model.**

Determine whether the information is sufficient for rating judgments.

If not, decide what else is needed and plan the required data collection.

© 1997 Carnegie Mellon University

E-SEPG-79



Carnegie Mellon University
Software Engineering Institute

Consolidation Activity Tactics

Manage time.

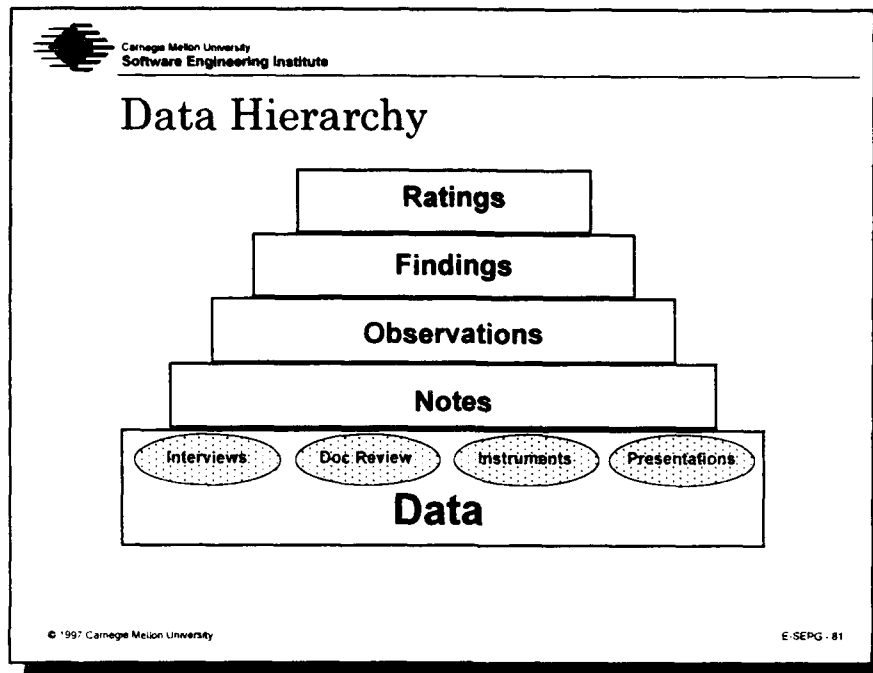
Work in parallel where possible.

Keep an open mind until sufficient data have been obtained.

Use team consensus to help eliminate individual biases.

© 1997 Carnegie Mellon University

E-SEPG-80



The slide is titled 'Observations' and defines the term. It includes a list of three bullet points describing what observations are. The slide is part of a presentation from Carnegie Mellon University's Software Engineering Institute.

Carnegie Mellon University
Software Engineering Institute

Observations

Observations are

- **statements by the team based on information heard or seen during data collection sessions**
- **typically mapped to a particular KPA activity or common feature**
- **possibly unrelated to the CMM, but identify an organizational issue that has an impact on software development capability**

© 1997 Carnegie Mellon University E-SEPG-82



Carnegie Mellon University
Software Engineering Institute

Judgments Relative to Observations

Each observation during data consolidation must be:

- accurate
- corroborated
- consistent

A set of observations must provide sufficient coverage of model components.

© 1997 Carnegie Mellon University

E-SEPG - 83



Carnegie Mellon University
Software Engineering Institute

Rules of Corroboration

Observations are based on data from at least two independent sources.

Observations are based on data obtained during at least two different data gathering sessions.

Observations are confirmed by at least one data source reflecting work actually being done.

A portion of the observations related to each KPA goal are supported by a review of related documentation.

© 1997 Carnegie Mellon University

E-SEPG - 84



Carnegie Mellon University
Software Engineering Institute

Coverage Criteria

A CMM component (activity or institutionalization common feature) is considered to be *covered* if the data gathered relevant to the component

- **represent the organizational units within the assessment scope**
- **represent the software life-cycle phases in use within the organization**
- **address**
 - **each of the key practices of the activities performed**
 - **each of the institutionalization common features**
- **in enough depth to determine the extent of their implementation and institutionalization**

© 1997 Carnegie Mellon University

E-SEPG - 85



Carnegie Mellon University
Software Engineering Institute

Purpose of Rating

To characterize an organization's software process maturity relative to the CMM

- **determined by the consensus judgment of the assessment team**
- **based on the assessment data**
- **using the rating process**

© 1997 Carnegie Mellon University

E-SEPG - 86



Carnegie Mellon University
Software Engineering Institute

Rating Scale

CMM component	Rating scale
goal KPA	satisfied, not satisfied, not applicable, not rated
maturity level*	Level 1 through Level 5

* Maturity level rating is optional.

© 1997 Carnegie Mellon University

E-SEPG - 87



Carnegie Mellon University
Software Engineering Institute

Rating Scale Definitions

Not applicable

- KPAs do not apply within the organization.

Not rated

- insufficient coverage of component
- outside assessment scope

Satisfied

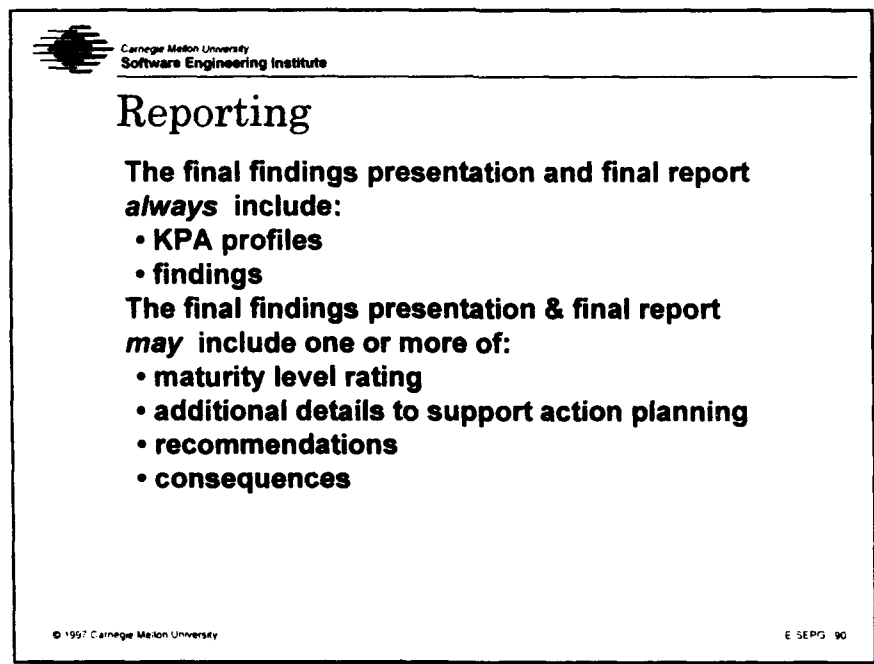
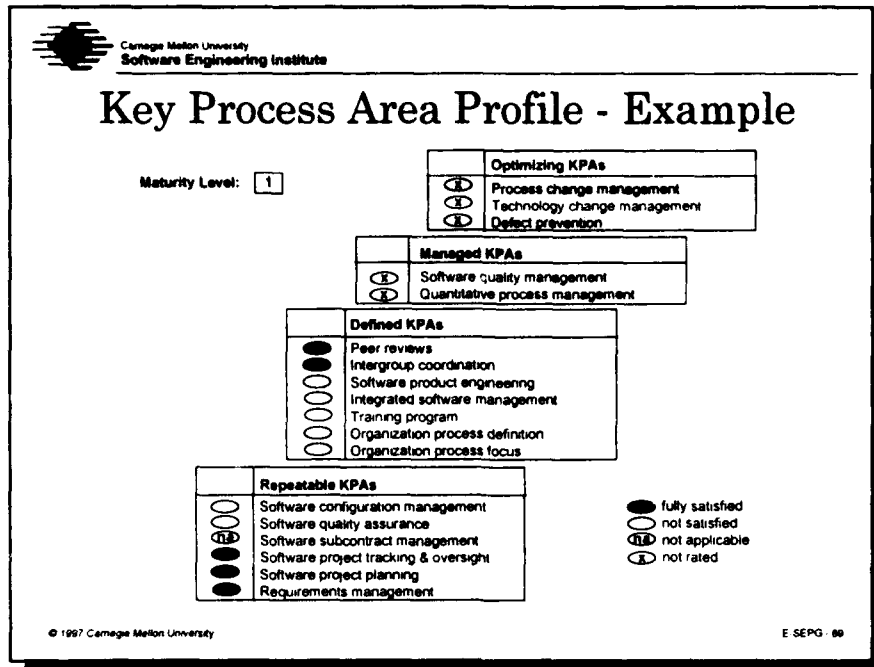
- CMM key practices are implemented and institutionalized.
- Alternate practices achieve KPA goals.

Not satisfied

- Significant weaknesses prevent implementation and institutionalization of practices defined in the CMM, and no adequate alternatives are in place.

© 1997 Carnegie Mellon University

E-SEPG - 88





Carnegie Mellon University
Software Engineering Institute

Minimum Requirements for a CBA IPI

For an assessment to be considered a CBA IPI, the assessment must meet minimum requirements concerning the following:

- the assessment team
- an assessment plan
- data collection
- data validation
- ratings
- reporting of assessment results

© 1997 Carnegie Mellon University

E-SEPG-91



Carnegie Mellon University
Software Engineering Institute

Requirements for the Assessment Team

The assessment team must be led by an SEI-authorized Lead Assessor.

The team shall consist of between 4 - 10 team members, with at least 1 team member being from the organization being assessed.

All team members must receive the SEI's Intro to the CMM course, or its equivalent, and the SEI's CBA IPI team training course.

Team members must meet the selection guidelines relative to software engineering and management experience.

© 1997 Carnegie Mellon University

E-SEPG-92



Carnegie Mellon University
Software Engineering Institute

Tailoring Options for the Assessment Team

The size of the assessment team, as long as the team consists of between 4 and 10 qualified individuals.

The composition of the team as to whether they are internal or external to the organization, as long as one team member is from the organization being assessed.

© 1997 Carnegie Mellon University

E-SEPG-93



Carnegie Mellon University
Software Engineering Institute

Requirements for the Assessment Plan

An assessment plan must be created that at a minimum contains the following:

- the goals for the assessment
- the CMM and organizational scope
- a schedule for assessment activities
- the assessment outputs and any anticipated follow-on activities
- planned tailoring of the assessment method
- risks and constraints associated with execution of the assessment
- the sponsor's authorization for the assessment

© 1997 Carnegie Mellon University

E-SEPG-94



Carnegie Mellon University
Software Engineering Institute

Tailoring Options for the Assessment Plan

The assessment plan may be tailored by:

- the weighting of assessment goals
- the specific organizational entities that comprise the organizational scope for the assessment
- the specific KPAs selected that comprise the CMM scope for the assessment
- the number of projects and their particular characteristics
- the length of time for the assessment

© 1997 Carnegie Mellon University

E-SEPG-95



Carnegie Mellon University
Software Engineering Institute

Requirements for Data Collection

Assessment data must be classified with respect to instruments, presentations, interviews, and documents and at a minimum contain:

- instrument data from at least the project leaders
- interview data from selected project via individual interviews
- interview data from functional area representatives (practitioners) and middle managers via group interviews
- document data for each of the KPA goals within the CMM scope of the assessment
- presentation data via a review of the draft findings with the assessment participants

Confidentiality of data sources must be protected.

© 1997 Carnegie Mellon University

E-SEPG-95



Carnegie Mellon University
Software Engineering Institute

Tailoring Options for Data Collection

Data collection may be tailored by:

- collecting instrument data from more respondents than the project leaders
- collecting the site information packet
- collecting a project leader interview with more than one project representative
- conducting part of a group interview "free form" where interviewees are asked to discuss anything they feel the team should know
- increasing the emphasis on collecting document data
- varying the number of draft finding sessions that are held

© 1997 Carnegie Mellon University

E-SEPG - 97



Carnegie Mellon University
Software Engineering Institute

Requirements for Data Validation

Data must be validated using the rules of corroboration and must sufficiently cover the CMM components within the assessment scope, the organization, and the software development life cycle.

The rules of corroboration are as follows:

- observations¹ are based on data from at least two independent sources
- observations are based on data obtained during at least two different data gathering sessions
- observations are confirmed by at least one data source reflecting work actually being done

¹observations are based on information extracted from data collection sessions

© 1997 Carnegie Mellon University

F-SEPG - 98



Carnegie Mellon University
Software Engineering Institute

Tailoring Options for Data Validation

Data validation may be tailored by:

- **using individuals or mini-teams for data gathering and consolidation tasks**
- **the extent of documentation that is collected**

© 1997 Carnegie Mellon University

E-SEPG-99



Carnegie Mellon University
Software Engineering Institute

Requirements for Rating

Ratings must be based on the CAF criteria for rating the process maturity of an organization against the CMM.

© 1997 Carnegie Mellon University

E-SEPG-100



Carnegie Mellon University
Software Engineering Institute

Tailoring Options for Rating

Rating may be tailored by:

- adding a "partially satisfied" rating for KPAs which would translate to "unsatisfied" for maturity level ratings
- extending ratings to common features and/or key practices
- rating the organization's maturity level

© 1997 Carnegie Mellon University

E-SEPG - 101



Carnegie Mellon University
Software Engineering Institute

Requirements for Reporting of Assessment Results

A final findings briefing must be given to the sponsor that presents the strengths and weaknesses of each KPA within the assessment scope, as well as a KPA profile that indicates whether KPAs are satisfied, unsatisfied, not rated, or not applicable.

These data must be reported back to the SEI.

© 1997 Carnegie Mellon University

E-SEPG - 102



Carnegie Mellon University
Software Engineering Institute

Tailoring Options for Reporting of Assessment Results

Reporting of assessment results may be tailored by:

- **including consequences and/or recommendations with the assessment findings**
- **generating a written final assessment report that details the assessment findings**
- **producing project-specific reports (this will require modifications of the confidentiality agreement)**

© 1997 Carnegie Mellon University

E-SEPG-103



Carnegie Mellon University
Software Engineering Institute

Summary of Tailoring Options

CMM Key Process Areas to examine.

Number of projects and their characteristics.

People who will participate.

Rating the organization's maturity level.

Producing project specific reports.

Amount of time spent onsite.

Assessment team size and composition.

Number, size, style, and duration of interviews.

Sufficiency of coverage of CMM components.

Adequacy of documentation for CMM components.

© 1997 Carnegie Mellon University

E-SEPG-104



Carnegie Mellon University
Software Engineering Institute

Post-assessment Activities

- **assessment wrap-up**
 - team members brainstorm lessons learned (document them)
 - one team member is assigned to shepherd assessment details
 - complete feedback forms to send to the SEI
- **develop Final Report and/or hand-off assessment details to action planning team**

© 1997 Carnegie Mellon University

E SEPG-105



Carnegie Mellon University
Software Engineering Institute

Feedback to the SEI

Process Assessment Information System (PAIS) form with, at a minimum:

- final findings
- KPA profile

Feedback forms:

- sponsor
- team leader
- team members

© 1997 Carnegie Mellon University

E SEPG-106

... as a Reference Model



Carnegie Mellon University
Software Engineering Institute

Importance of Final Report

Valuable tangible product of the assessment.

Contains detailed assessment findings.

Forms a basis for action planning.

Serves as a frame of reference for subsequent assessments.

© 1997 Carnegie Mellon University

E-SEPG-107



Carnegie Mellon University
Software Engineering Institute

Final Report Structure

Executive summary

Findings and consequences

Recommendations

Appendices:

- **scope of the assessment**
- **participants in the assessment**
- **other relevant attachments**
- **any activities that were "excluded from coverage" (EC) in KPAs within the assessment scope**

© 1997 Carnegie Mellon University

E-SEPS-108



Carnegie Mellon University
Software Engineering Institute

Importance of Recommendations

Serve as a link between the findings and action plan (and between Diagnosing and Establishing phases).

Capture the insight of the assessment team members and other interested, qualified individuals who have participated in the assessment.

© 1997 Carnegie Mellon University

E-SEPG - 109



Carnegie Mellon University
Software Engineering Institute

Recommendations Guidelines

Address each Key Process Area finding as well as each non-CMM based finding.

Assign suggested priorities to each recommendation, considering high leverage business items.

Should be achievable by the next planned assessment.

Describe "what" needs to be done, not "how."

Be specific and concise.

Consider organizational impact.

© 1997 Carnegie Mellon University

E-SEPG - 110



Carnegie Mellon University
Software Engineering Institute

Summary - Part II

We have covered:

- **what the CBA IPI method can accomplish**
- **the pre-on-site activities**
- **the on-site activities**
- **the minimum requirements for a CBA IPI**
- **post-assessment activities**

© 1997 Carnegie Mellon University

E-SEPG - 111

Appraisals Using the CMMSM as a Reference Model

**Donna K. Dunaway and Steve Masters
Software Engineering Institute**

European SEPG97

Amsterdam, June 16-19, 1997

Abstract: This document provides a high-level overview of CMMSM-Based Appraisals. It provides a brief history of SEI appraisal methods, as well as establishing appraisals in the context of the IDEALSM approach to software process improvement. Additional detail is provided for the SEI's most current assessment method, CMM-Based Appraisal for Internal Process Improvement (CBA IPI). CBA IPI is a diagnostic tool that supports, enables, and encourages an organization's commitment to process improvement. The method helps an organization gain insight into its software development capability by identifying strengths and weaknesses of its current processes related to the Capability Maturity ModelSM for Software V1.1. The method focuses on identifying software improvements that are most beneficial, given an organization's business goals and current maturity level.

1 History of SEI Appraisal Methods

In accordance with the SEI's mission to provide leadership in advancing the state of the practice of software engineering by improving the quality of systems that depend on software, there has been strong emphasis within the SEI on treating software development tasks as processes that can be defined, practiced, measured, and improved. In early software process publications, a software maturity framework and questionnaire were developed to help organizations characterize the current state of their software practices, set goals for process improvement, and set priorities.

Software process is defined to mean the system of all tasks and the supporting tools, standards, methods, and practices involved in the production and evolution of a software product throughout the software life cycle. It has become widely accepted that the quality of a (software) system is largely governed by the quality of the process used to develop and maintain it.

The SEI assisted a number of organizations in performing assessments based largely on the maturity questionnaire. This early questionnaire provided a scoring mechanism for determining a project's maturity level. In 1988-91, the SEI provided training to organizations who

SM - Capability Maturity Model, CMM, and IDEAL are service marks of Carnegie Mellon University.

wished to perform self-assessments of their software processes.

In 1990 the SEI commercialized the software process assessment (SPA) to more broadly disseminate the technology, since the SEI was not equipped to handle the demand for assessment services. Industry and government licensees were selected as vendors to market assessment services. During 1991-1993, SEI self-assessment training was gradually phased out and replaced by vendor training. Data from these assessments have been collected by the SEI, and periodic reports are delivered on the state of the practice. The initial state of the practice reported on assessment data largely from project-based questionnaire data from 10 sites. The follow-up report included the first site-based software process maturity profile and provided an analysis of assessment findings from 59 sites showing the frequency with which key process deficiencies were identified by assessment teams. A recent report presents an analysis of assessment results from 48 organizations that have performed 2 or more assessments and focuses on the time required to increase process maturity. Updates on the process maturity profile of the software community are published twice a year with the latest profile containing data on 751 assessments including 265 CBA IPIs [Zubrow 97].

Based on the success and widespread use of the maturity framework and software process assessments, Version 1.0 of the CMM for Software was published in 1991. In 1993 the CMM was revised, and Version 1.1 was published. Various organizations modified SEI appraisals to reflect the CMM; however, the CBA IPI method is the first CMM-based assessment method released by the SEI. CBA IPI uses an updated maturity questionnaire consistent with CMM V1.1. Unlike the maturity questionnaire released in 1987, the current maturity questionnaire does not include a mechanism for scoring maturity levels.

The SEI has published reports that show a relationship between CMM-based improvement and organizational performance. One report documents process improvement efforts in 13 organizations by showing improvements in cycle time, defect density, and productivity. Benefit-to-cost ratios presented range from 4.0:1 to 8.8:1. In a more comprehensive report on the impact of CMM-based appraisals on subsequent software process improvement and organizational performance, survey results from 138 appraisal participants representing 56 appraisals are presented. The results show that, in general, increased process maturity results in better product quality, ability to meet schedule commitments, and other indicators of organizational performance.

2 IDEALSM Approach to Process Improvement

The CBA IPI method is a diagnostic tool used to appraise and characterize an organization's software processes and is used in the larger context of software process improvement. The SEI's recommended framework for software process improvement is the IDEALSM model. The IDEAL approach consists of five phases: initiating, diagnosing, establishing, acting, and leveraging. Appraisals are an integral part of the diagnosing phase of the IDEAL approach.

The initiating phase of process improvement should be successfully undertaken before the ap-

praisal start-up. First, some outside stimulus for improvement needs to occur. Then sponsorship is established with the site's senior management, and efforts toward building an improvement infrastructure are committed. An organization could be appraised for the first time or could be reappraising their processes in preparation for the next process improvement cycle. The IDEAL approach assumes that the appraisal will be followed by a thorough documentation of the appraisal results and the development of recommendations. The software process baseline established by the appraisal and the recommendations and priorities that come from the appraisal form a basis for follow-on activities. These activities are typically documented in an action plan for furthering process improvement activities.

3 Current SEI Appraisal Methods

To provide a framework for rating an organization's process capability against the CMM and to provide a basis for comparing assessment and evaluation results, the SEI published the CMM Appraisal Framework (CAF). The CAF identifies the requirements and desired characteristics of a CMM-based appraisal method in order to improve consistency and reliability of methods and their results. An appraisal method is CAF compliant when it meets all of the CAF requirements. The term appraisal as used at the SEI includes multiple methods, such as assessments and evaluations, all of which focus on an organization's software development process.

Both CBA IPI and Software Capability Evaluation (SCE) V3.0 were designed to be CAF compliant. The following subsections briefly describe the primary SEI appraisal methods, CBA IPI and SCE, and the differences between them. Interim Profile is a method to rapidly measure the status of an organization's software engineering process improvements between organizational assessments and was not intended to comply with the CAF; therefore, it is not included in our discussion of current SEI appraisal methods.

3.1 CMM-Based Appraisal for Internal Process Improvement (CBA IPI)

Using the Capability Maturity ModelSM for Software V.1.1 (CMM) as a reference model, the Software Engineering Institute (SEI) developed the CBA IPI V1.0 in 1995 for assessing an organization's software process capability. CBA IPI V1.1 was released in 1996 containing *modifications for clarification and simplification*. In-depth documentation and guidance on the CBA IPI method is available through CBA Lead Assessor Training.

The CBA IPI method was created in response to user needs for a CMM-based assessment method. The SPA method, which has become so familiar in the software community, pre-dated the CMM. Although many organizations have modified the SPA to reflect the CMM, there was a wide range of approaches and results.

The CBA IPI method was developed and field tested in 1994. After factoring in lessons learned from the community feedback, the SEI released CBA IPI V1.0 in May 1995. The method and

documentation were upgraded to CBA IPI V1.1 in March 1996. The CBA IPI method explicitly uses CMM V1.1 as a reference model. The data collection is based on key process areas (KPA's) of the CMM as well as non-CMM issues. CBA IPI is intended to establish consistency among CMM-based assessments so that results from one assessment can be compared to those of another. The CBA IPI method complies with the CAF, so results from a CBA IPI are intended to be consistent with results from other CAF-compliant methods. The CBA IPI method is described in more detail in the following sections of this report. Descriptions of the method roles and responsibilities and a glossary of terms commonly used in a CBA IPI are found in [Dunaway 96].

3.2 Software Capability Evaluation (SCE)

SCEs are used for software acquisition as a discriminator to select suppliers, for contract monitoring, and for incentives. They can also be used for evaluation of internal processes. SCE V2.0 was updated to reflect CMM V1.1. SCE V3.0 is CAF compliant. Therefore, results from a SCE V3.0 should be consistent with a CBA IPI V1.1 if the areas of investigation are the same and in relatively the same time frame. SCE is used to gain insight into the software process capability of a supplier organization and is intended to help decision makers make better acquisition decisions, improve subcontractor performance, and provide insight to a purchasing organization.

3.3 Differences Between Assessments and Evaluations

The basic difference between an assessment and an evaluation is that an assessment is an appraisal that an organization does to and for itself, and an evaluation is an appraisal where an external group comes into an organization and looks at the organization's process capability in order to make a decision regarding future business. The scope of an assessment is determined relative to the organization's needs and the business goals of the sponsor, who is usually the senior manager of the assessed organization. In contrast, the scope of an evaluation is determined relative to the needs of the sponsor, who is the individual or individuals responsible for deciding to conduct the evaluation of the organization(s) with whom the sponsor is currently or considering doing business.

After an assessment, the senior manager of the assessed organization owns the assessment findings and results and generally uses the results to formulate an action plan for the process improvement program. After an evaluation, the sponsor owns the evaluation results and uses the results to make decisions regarding the recipient organization(s), such as source selection, incentive fee awards, performance monitoring, risk management, and measurement of internal improvement; these results may or may not be shared with the evaluated organization.

Assessments are intended to motivate organizations to initiate or continue software process improvement programs. Evaluations are externally imposed motivation for organizations to undertake process improvement. Assessment teams represent a collaboration among team members, many of whom are from the organization being assessed. Evaluation teams, on the

other hand, take more of an audit-oriented approach with a more formal interface between team members and organization representatives; evaluation teams rarely include representatives from the evaluated organization.

3.4 Interim Profile

The Interim Profile (IP) is a method to rapidly measure the status of an organization's software engineering process improvements between organizational software process assessments. It is based on the CMM V1.1 and is designed to be used only by organizations that have completed an SEI-style assessment, have support for software engineering process improvement in place, and intend to use the results to monitor status and adjust their process improvement plan. A primary motivation for designing the IP method was to provide quantifiable data in an expedient fashion to allow periodic "temperature-checks" of process improvement activities. The IP method is based on the SEI maturity questionnaire with minimal use of other data sources. The IP method is not intended to comply with the CAF since it focuses on questionnaire responses and does not require other data collection mechanisms as required by the CAF. It is a "quick look" at the organization, not a full assessment, based on the participants' responses to the questionnaire.

4 The SEI's Current CMM-Based Assessment Method: CBA IPI

The SEI's current CMM-based assessment method is CMM-Based Appraisal for Internal Process Improvement (CBA IPI). The CBA IPI method is a diagnostic tool that enables an organization to gain insight into its software development capability by identifying strengths and weaknesses of its current processes, to relate these strengths and weaknesses to the CMM, to prioritize software improvement plans, and to focus on software improvements that are most beneficial, given its current level of maturity and the business goals.

The method is an assessment of an organization's software process capability by a trained group of professionals who work as a team to generate findings and ratings relative to the CMM key process areas within the assessment scope. The findings are generated from data collected from questionnaires, document review, presentations, and in-depth interviews with middle managers, project leaders, and software practitioners.

The CBA IPI method has two primary goals:

- to support, enable, and encourage an organization's commitment to software process improvement
- to provide an accurate picture of the strengths and weaknesses of the organization's current software process, using the CMM as a reference model, and to identify key process areas for improvement.

The approach of the CBA IPI method is to assemble and train a competent assessment team under the leadership of a Lead Assessor and to conduct a structured series of activities with

key people in the organization to understand their problems, concerns, and ideas for improvement. The method is based on the following key assessment principles:

- Use the Capability Maturity Model for Software V1.1 as a process reference model.
- Use a formalized assessment process that complies with the CMM Appraisal Framework.
- Involve senior management as the assessment sponsor.
- Base the assessment on the sponsor's business goals and needs.
- Observe strict confidentiality by guaranteeing that no information will be attributed to an individual or project.
- Approach the assessment as a collaboration between the assessment team and the organizational participants.

4.1 What Can The CBA IPI Method Accomplish?

The business needs for process improvement drive the requirements for an assessment. The business needs for process improvement generally include one or more of three closely related factors: reducing costs, improving quality, and decreasing time to market. The fundamental assumption is that these factors are largely determined by the development processes.

Since the CBA IPI method is designed to support a variety of assessment needs, the results of a CBA IPI can support a variety of activities that require detailed knowledge about the strengths and weaknesses of the software process, such as

- establishing software process action plans
- measuring actual progress against action plans
- identifying best (most effective) practices within the organization for transition elsewhere in the organization.

The CBA IPI method must build on an organization's commitment established during previous phase(s) of the software process improvement cycle. To support the goal of enabling and supporting an organization's commitment to process improvement, the assessment team and assessment participants must understand the sponsor's business goals and allow for a collaborative shaping of the assessment scope. It is important that the organization owns and "buys in" to the assessment results, identifying the most critical issues (CMM and non-CMM) that need to be addressed in order to sustain momentum for the follow-on activities.

The other primary goal of the CBA IPI is to provide an accurate picture of existing software processes. To accomplish this, the assessment team will

- provide the organization with the data needed to baseline the organization's software capability
- identify major non-CMM issues that have an impact on process improvement efforts
- identify strengths and weaknesses using the CMM as a reference model

- provide findings to the sponsor and the organization that are sufficiently complete to guide the organization in planning and prioritizing future process improvement activities

4.2 Minimum Requirements for a CBA IPI

For an assessment to be considered a CBA IPI, the assessment must meet the following minimum requirements concerning the assessment team, assessment plan, data collection, data validation, rating, and reporting of assessment results. Permissible tailoring options are provided with the requirements.

4.2.1 Assessment Team

The assessment team must satisfy the following requirements:

- The assessment team must be led by an authorized SEI Lead Assessor.
- The team shall consist of a minimum of 4 and a maximum of 10 team members. At least one team member must be from the organization being assessed.
- All team members must receive the SEI's Introduction to the CMM course, or its equivalent, and the SEI's CBA IPI team training course.
- Team members must meet the selection guidelines relative to software engineering and management experience.

Tailoring options are as follows:

- the size of the assessment team, as long as the team consists of between 4 and 10 qualified individuals
- the composition of the team as to whether they are internal or external to the organization, as long as one team member is from the organization being assessed

4.2.2 Assessment Plan

An assessment plan must be created that at a minimum contains the following:

- the goals for the assessment
- the CMM scope (KPAs to be examined) and the organization scope for the assessment including selected projects and assessment participants
- a schedule for assessment activities and identification of the resources to perform the activities
- the assessment outputs and any anticipated follow-on activities
- planned tailoring of the assessment method
- risks and constraints associated with execution of the assessment
- the sponsor's authorization for the assessment to be conducted

Tailoring options are as follows:

- weighting of the assessment goals. Depending upon the organization's motivation for having an assessment, more emphasis may be placed on one goal than the other, e.g., more focus toward supporting an organization's software process improvement than precise conformance relative to the CMM.
- the specific organizational entities that comprise the organization's scope for the assessment
- the specific KPAs selected that comprise the CMM scope for the assessment
- the number of projects and their particular characteristics
- the length of time for the assessment

4.2.3 Data Collection

Assessment data must be classified with respect to four data collection categories (instruments, presentations, interviews, and documents) and at a minimum contain the following:

- instrument data (maturity questionnaire responses) from at least the project leaders from the selected projects
- interview data from project leaders from selected projects via individual interviews
- interview data from functional area representatives (practitioners) and middle managers via group interviews
- document data for each of the KPA goals within the CMM scope of the assessment
- presentation data via a review of the draft findings with the assessment participants

In addition, confidentiality of data sources must be protected.

Tailoring options are as follows:

- Collect instrument data from more respondents than the project leaders.
- Collect the site information packet.
- Conduct a project leader interview with more than one project representative.
- Conduct part of a group interview "free form" where interviewees are asked to discuss anything they feel the assessment team should know.
- Increase the emphasis on collecting document data.
- Vary the number of draft finding sessions that are held.

4.2.4 Data Validation

Data must be validated using the rules of corroboration and must sufficiently cover the CMM components within the assessment scope, the organization, and the software development life cycle.

The rules of corroboration are as follows:

- Observations are based on data from at least two independent sources, e.g., two separate people or a person and a document.
- Observations are based on data obtained during at least two different data gathering sessions.

- Observations are confirmed by at least one data source reflecting work actually being done, e.g., an implementation level document or an interview with a person who is performing the work.

Tailoring options are as follows:

- use of individuals or mini-teams for data gathering and consolidation tasks
- extent of documentation that is collected

4.2.5 Rating

Ratings must be based on the CAF criteria for rating the process maturity of an organization against the CMM.

Tailoring options are as follows:

- Add a "partially satisfied" rating which would translate to "unsatisfied" for maturity level rating.
- Extend ratings to common features and/or key practices.
- Rate the organization's maturity level.

4.2.6 Reporting of Assessment Results

A final findings briefing must be given to the sponsor that presents the strengths and weaknesses of each KPA within the assessment scope, as well as a KPA profile that indicates whether KPAs are satisfied, unsatisfied, not rated, or not applicable. These data must be reported back to the SEI.

Tailoring options are as follows:

- Include consequences and/or recommendations with the assessment findings.
- Generate a written final assessment report that details the assessment findings.
- Produce project-specific reports (this will require modification of the confidentiality agreement).

5 CBA IPI Method Activities

The CBA IPI method consists of three phases. The first phase includes the activities necessary to plan and prepare for the assessment. The second phase consists of on-site activities for conducting the assessment, including techniques for gathering, organizing, and consolidating data. The final phase is to report the results.

5.1 Plan and Prepare the Assessment

The assessment team leader must clearly understand the sponsor's goals for having the assessment as well as any constraints that exist. A plan is developed based on the sponsor's goals. Assessment team members, projects, and assessment participants are selected ac-

cording to defined criteria. Documents are identified for initial review, and the logistics for the on-site visit are identified and planned. The assessment team must be trained in the CMM as well as the assessment method. A briefing is held for assessment participants so that each person understands the assessment process and has a clear set of expectations regarding the assessment outcomes. Information about the organization's software processes is collected from selected members of the organization using the SEI maturity questionnaire. The assessment team members examine the pattern and nature of responses on the maturity questionnaires completed by site personnel. An initial set of documents about the organization's software processes are reviewed to find additional areas for probing, to understand the life cycle(s) in use by the organization, and to map organization data to the CMM.

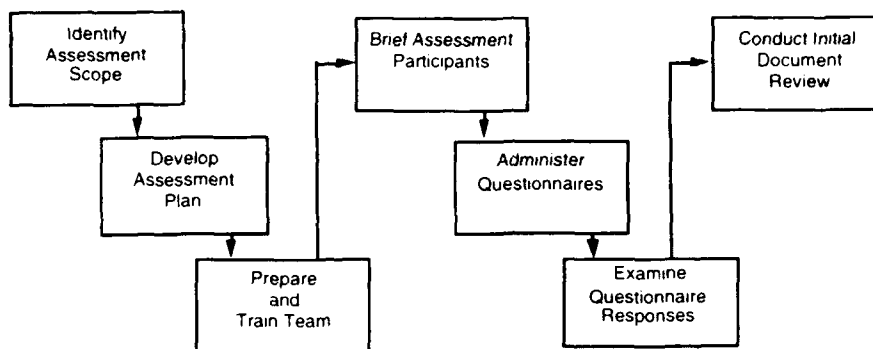


Figure 1: Plan and Prepare the Assessment

5.2 Conduct the Assessment

An opening meeting is held to kick off the on-site visit. The sponsor of the assessment opens the presentation to show visible support and urge participants to be forthcoming in interview sessions. Interviews are held to identify areas that people believe can and should be improved in the organization, to understand how the work is performed, to understand the processes in use, to understand the relationships among the organization-level processes and the project-level processes, and to ensure coverage of the CMM within the scope of the assessment

across the organization, as defined in the scope.

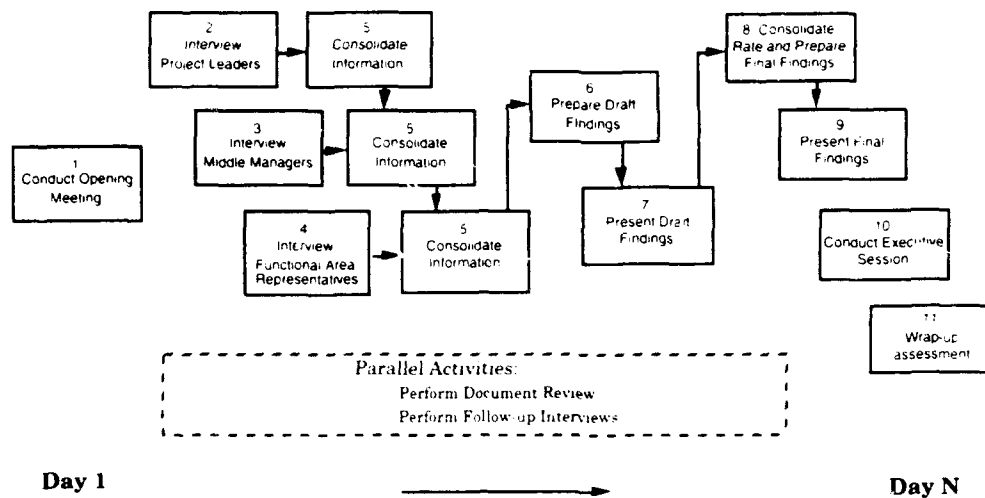


Figure 2: Conduct the Assessment

The assessment team consolidates information into a manageable set of observations which are categorized with reference to the KPAs of the CMM. All observations must be validated using the rules of corroboration. It is the team's responsibility to obtain sufficient information to cover the organization, the software development life cycle, and the CMM components within the assessment scope before any rating can be done.

Draft findings are generated and presented to the assessment participants to obtain validation from those who have provided information through interviews. When the team has achieved full coverage of the CMM, the organization, and the software life cycle, the rating process may begin by rating each goal for each key process area (KPA) within the assessment scope. Ratings are always established based on consensus of the entire assessment team. For each goal, the team reviews all weaknesses that relate to that goal and asks: "Is there a weakness significant enough to have a negative impact on the goal?" If so, the goal is rated unsatisfied. If the team decides that there are no significant weaknesses that have an impact on a goal, it is rated satisfied. For a KPA to be rated satisfied, all goals for the KPA must be rated satisfied. A KPA may be rated satisfied, unsatisfied, not applicable, or not rated. Assignment of a maturity level rating is optional at the discretion of the sponsor. For a particular maturity level rating to be achieved, all key process areas within and below a given maturity level must be satisfied. For example, for an organization to be rated at maturity level 3, all KPAs at level 3 and at level 2 must have been investigated during the assessment, and all KPAs must have been rated satisfied by the assessment team. The final findings presentation is developed by the team to present to the sponsor and the organization the strengths and weaknesses observed for each KPA within the assessment scope, the ratings of each KPA, and the maturity level rating if de-

sired by sponsor.

5.3 Report Results

The team leader, or designated presenter, presents the assessment results to the sponsor. The sponsor owns the assessment results and is free to use them as he or she sees fit. Organizational strengths are presented to validate what the organization is doing well. Strengths and weaknesses are presented for each key process area within the assessment scope as well as any non-CMM issues that affect process. A KPA profile is presented showing the individual KPA ratings.

An executive session usually follows the final findings briefing to allow the senior site manager to clarify any issues with the assessment team, to confirm his or her understanding of the software process issues, and to get guidance regarding the focus, timing, and priorities of the recommendations report and follow-on activities.

The team leader collects feedback from the assessment participants and the assessment team on the assessment process, collects information that needs to be reported to the SEI, and assigns responsibilities for follow-on activities.

6 Time Frame and Resource Requirements

In order to identify the resources that are needed to perform an assessment, a typical time frame is shown.

Months 1-2	Assessment planning and team training
Month 3	On-site assessment
Month 4	Final report delivery and recommendations briefing
Month 5	Action plan development
Months 6-24	Action plan implementation
Months 18-30	Re-assessment

Resources required for an organization to perform a CBA IPI vary according to the scope of the assessment. The figures in Table 2 may be used as guidelines in planning the resource requirements for an assessment. These figures were gathered from early implementations of CBA IPI assessments. However, subsequent assessments have shown that the assessments can be done more efficiently, so we consider these estimates to be conservative and on the high end. For a typical assessment team consisting of 8 team members plus the assessment team leader, investigating 4 representative projects, and interviewing 40 functional area representatives, a CBA IPI is estimated to require approximately 200 person-days, beginning with assessment planning through writing the final report and hand-off to the team who will plan the actions for continuing process improvement activities.

Assessment Phase	Resource Required	Full-time Equivalent
Planning	Assessment team leader	10-20 days
	Sponsor	3-5 days
	Site coordinator	10-20 days
Pre-Onsite Activities	Assessment team leader	10-14 days
	Site coordinator	10-14 days
	Assessment team members	6-8 days each
On-Site Activities	Assessment team leader	6-10 days
	Sponsor	4-5 hours
	Site coordinator	6-10 days
	Assessment team members	5-10 days each
	Assessment participants (project leaders, middle managers, and functional area representatives)	1.5 days each
Post-Assessment Activities	Assessment team leader	4-8 days
	Sponsor	1 day
	Site coordinator	2 days
	Assessment team members	1-2 days each
Totals	Assessment team leader	30-52 days
	Sponsor	8-11 days
	Site coordinator	28-46 days
	Assessment team members	12-20 days each
	Assessment participants (project leaders, middle managers, and functional area representatives)	1.5 days each

7 Follow-On Activities

To benefit from the assessment, the organization must have an improvement infrastructure in place. This infrastructure would be in the form of sponsorship, establishment of a software engineering process group (SEPG), CMM training for the organization, etc.

The assessment team will develop recommendations and document the assessment results immediately following the final findings briefing. Although a final report is optional but recommended, if a final report is not required by the sponsor, one of the team members should be responsible for collecting the assessment details, removing all attribution, and making them available to the action planning team. These assessment details are critical for prioritizing issues, developing the action plan, and implementing the establishing phase. There are several

benefits to the establishing phase: it provides an organized and planned approach to software process improvement; a clear understanding of costs, schedules, and resources; and the opportunity to plan for actions that the organization has the capability to achieve. It is important to move into the establishing phase as soon as possible after completion of the assessment so that momentum from the assessment can be maintained and the staff's expectations can be managed. An assessment provides expectations for improvement in an organization; follow-on activities are very important to avoid disappointment or disillusionment by the assessment participants.

8 Future Improvement of the Method

There are many feedback mechanisms in place to enable the SEI to continuously improve the method. The SEI requires Lead Assessors to provide feedback following each assessment that they lead. In addition, each sponsor and each assessment team member is asked to provide feedback to the SEI at the conclusion of an assessment. Change requests are accepted by the SEI from anyone who wishes to send one in.

There is no major change anticipated to this assessment method until after CMM V2.0 is released. However, minor modifications will be made from time to time through notices to the Lead Assessors.

9 References

- [Dunaway 96] Dunaway, Donna K. & Masters, Steve. *CMM-Based Appraisal for Internal Process Improvement (CBA IPI): Method Description* (CMU/SEI-96-TR-007). Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, April 1996.
- [Zubrow 97] Zubrow, Dave. *Process Maturity Profile of the Software Community 1997 Update*. Pittsburgh, Pa.: Software Engineering Institute, Carnegie Mellon University, April 1997.



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

IDEALSM Approach to Implementing Software Process Improvement

Chuck Myers	Software Engineering Institute	crm@sei.cmu.edu
Paul Goodman	TBL and ESPI Foundation	100675.2423@compuserve.com
Magnus Ahlgren	Q-Labs	ma@q-labs.se

SM IDEAL is a service mark of Carnegie Mellon University

The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense and operated by Carnegie Mellon University

© 1997 Carnegie Mellon University

RISK97-IDEAL-1



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Agenda

- Introduction
- Context
- Overview
- Initiating Phase
- Diagnosing Phase
- Establishing Phase
- Acting Phase
- Learning Phase
- Questions(!) and Answers(?)

© 1997 Carnegie Mellon University

RISK97-IDEAL-2



Where Do You Work?

Military?

Other Government?

Government Contractor?

Industry?

- Finance?
- Retail?
- Industrial?

Bespoke System Supplier?

Outsourcing Organization?

Academia?



How Long with SPI?

Just starting

< 1 year?

1 < but < 3 years?

3 < but < 5 years?

> 5 years?



Who's Here?

"Sponsor"

"Agent"

"Champion"

Participant

Interested/Curious

Other



Tutorial Objectives

When you have completed this tutorial, we hope you'll be able to do the following:

- **describe the IDEALSM phases and activities**
- **develop some ideas for applying IDEAL to your back-home circumstances, starting with tutorial exercise materials**



Agenda

Introduction

➡ Context

Overview

Initiating Phase

Diagnosing Phase

Establishing Phase

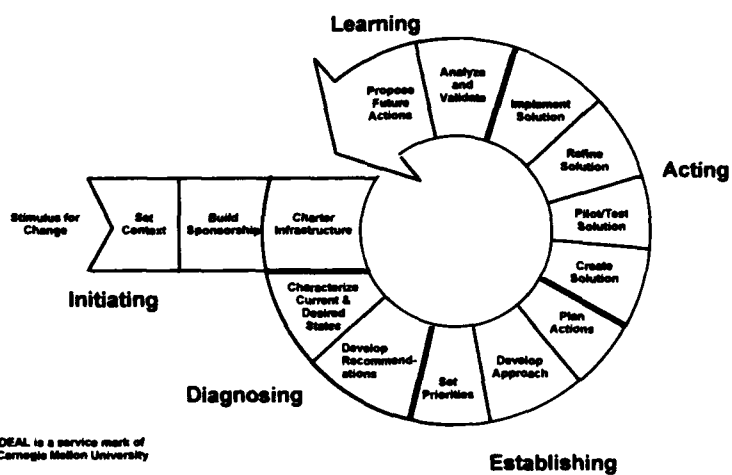
Acting Phase

Learning Phase

Questions(!) and Answers(?)



The IDEALSM Model





Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Origins

Grew out of software process improvement (SPI)

SPI adopters requested how-to information and guidance

Experienced multi-disciplinary team was formed to synthesize knowledge and experience

Tutorial developed by team presented at 1993 Software Engineering Symposium

Roadmap published early 1996

© 1997 Carnegie Mellon University

RISK97-IDEAL-9

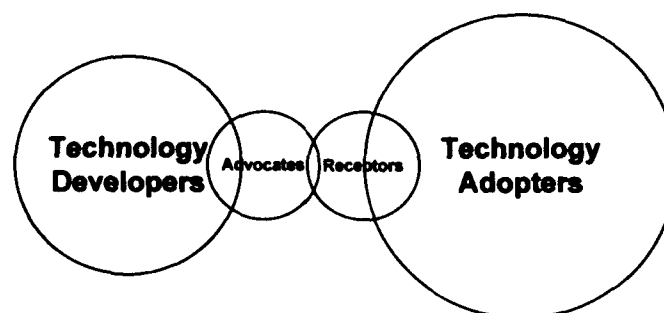


Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Technology Transition Model



© 1997 Carnegie Mellon University

RISK97-IDEAL-10



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

The Problem

Technology Adopters

- have difficulty selecting among improvement possibilities
- do not use best transition practices to introduce selected improvements

Technology Developers

- have difficulty understanding adoption difficulties that will be encountered by potential adopters
- do not provide enough information or services needed to support the adoption process
- do not get technologies into widespread practice as quickly as they could

© 1997 Carnegie Mellon University

RISK97-IDEAL-11

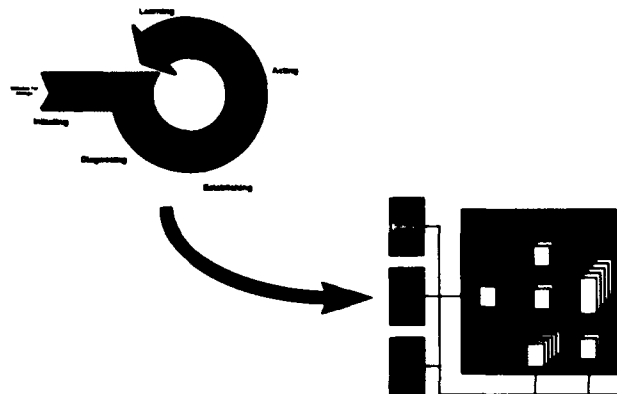


Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Focus of Technology Adoption Architectures Project



© 1997 Carnegie Mellon University

RISK97-IDEAL-12



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Current Efforts

- Document the model ("Web Report")**
- Develop IDEAL Transition FrameworkSM (ITFSM)**
- Extend model to encompass additional technical areas**
 - **Risk (1997)**
 - **Network Systems Survivability (1998)**
- Make model scalable**

SM IDEAL Transition Framework and ITF are service marks of Carnegie Mellon University
 © 1997 Carnegie Mellon University RISK97-IDEAL-13



Carnegie Mellon University
Software Engineering Institute

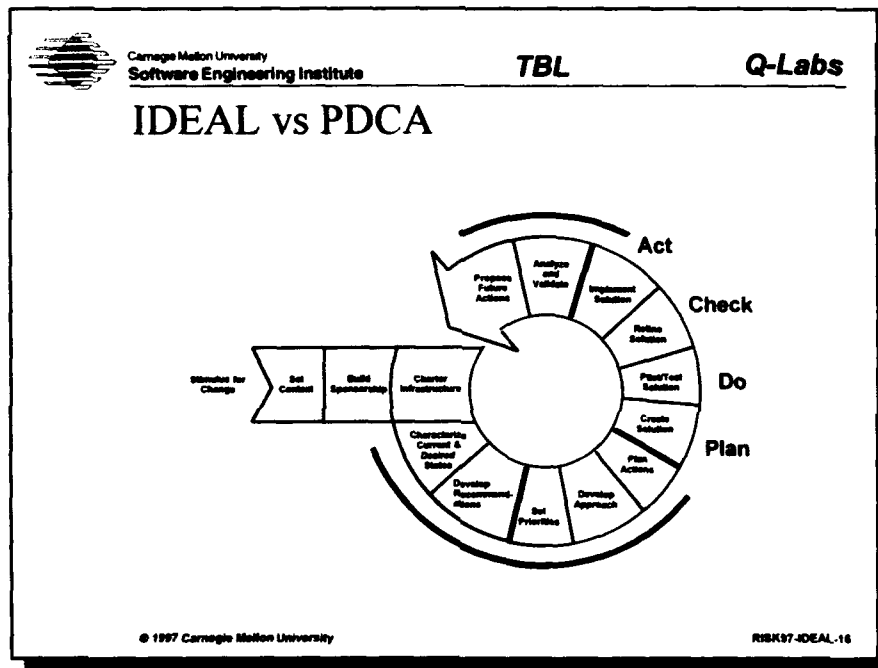
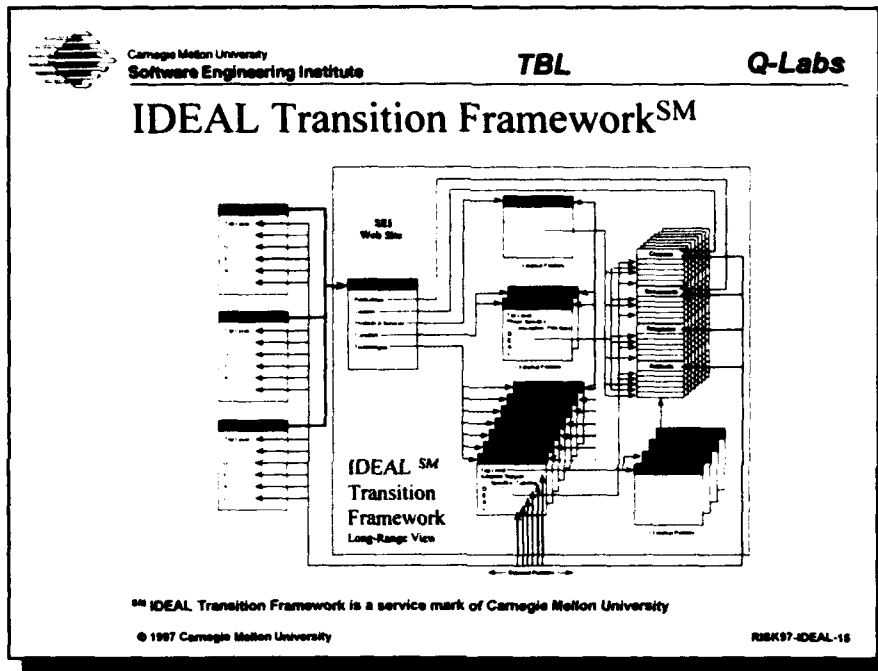
TBL

Q-Labs

The Web Report

- Serves as a technical report**
- Takes advantage of electronic, web-based technology**
 - **accessibility**
 - **flexibility**
 - **ease of update**
- Focuses predominantly on adopter needs**

© 1997 Carnegie Mellon University RISK97-IDEAL-14





Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Agenda

- Introduction
- Context
- ➡ Overview
- Initiating Phase
- Diagnosing Phase
- Establishing Phase
- Acting Phase
- Learning Phase
- Questions(!) and Answers(?)

© 1997 Carnegie Mellon University

RISK97-IDEAL-17



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

The Initiating Phase – Overview

- Know what prompted this particular change
- Identify where this change fits within the larger organizational context
- Specify business goals and objectives that will be realized or supported
- Identify impact on other initiatives and on-going work
- Secure the support required to give the change a reasonable chance of succeeding
- Adjust relevant organizational systems to support the effort
- Establish an infrastructure for managing specifics of implementation

© 1997 Carnegie Mellon University

RISK97-IDEAL-18



The Diagnosing Phase – Overview

Identify the organization's current state with respect to a related standard or reference model

Identify the organization's desired state against the same standard or model

Recommend actions that will move the organization from where it is to where you want it to be

Identify potential barriers to the effort



The Establishing Phase – Overview

Develop priorities among the recommended actions on the basis of such things as:

- **availability of key resources**
- **dependencies between the actions recommended**
- **funding**

Develop an approach that reflects priorities and current realities

Develop plans to implement the chosen approach



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

The Acting Phase – Overview

Bring together everything available to create a “best guess” solution specific to organizational needs, e.g.,

- existing tools, processes, knowledge, skills...
- new knowledge, information...
- outside help

Put the solution in place on a trial basis to learn what works and what doesn't

Revisit and modify the solution to incorporate new knowledge and understanding

Iterate as necessary

When solution is deemed workable and sufficient, implement it throughout the organization

© 1997 Carnegie Mellon University

RISK97-IDEAL-21



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

The Learning Phase – Overview

Determine what was actually accomplished by the effort

Compare what was accomplished with the intended purpose for undertaking the change

Summarize/roll up lessons learned regarding processes used to implement IDEAL

Develop recommendations concerning management of future change efforts using IDEAL

© 1997 Carnegie Mellon University

RISK97-IDEAL-22



Agenda

Introduction

Context

Overview

➡ **Initiating Phase**

Diagnosing Phase

Establishing Phase

Acting Phase

Learning Phase

Questions(!) and Answers(?)



Initiating Phase Activities

Stimulus for Change

Set Context

Build Sponsorship

Charter Infrastructure



Stimulus for Change

"Something" prompts IDEAL use.

The nature of the stimulus can vary widely:

- reaction to unanticipated events/circumstances
- edict from "on high"
- enactment of change as part of proactive continuous improvement approach.

The stimulus can have far-reaching influence on the change effort's visibility, conduct, and ultimate success.



Set Context

"Setting context" means being very clear about where this change fits within the organization's business strategy.

Setting context requires that there be consensus regarding

- the organization's core mission
- business goals and objectives
- a coherent vision for the future
- a strategy for achieving the vision

This activity may require extensive work if the above do not exist.

Context and implications often become more clear as the effort proceeds.



Build Sponsorship

Sponsorship is "senior level" support for an effort.

Sponsors are most effective if they

- give personal attention to the effort
- stick with the change through difficult times
- commit scarce resources to the effort
- change their own behavior to be consistent with the change being implemented
- modify the reward system.

The level and quality of sponsorship is often the most critical element contributing to success.

Sponsorship starts here, but it must be sustained throughout the remaining IDEAL phases.



Charter Infrastructure

Implementing significant change often requires establishment of organizational structures to manage the effort.

The infrastructure may include

- an oversight group (senior level people)
- a change agency group (staff to the oversight body)
- one or more technical working groups (TWGs)

"Chartering" these groups involves developing an explicit written agreement describing the responsibilities of each.

It usually is not possible (or desirable) to charter TWGs during the Initiating Phase (but anticipating them might be useful).

**Carnegie Mellon University
Software Engineering Institute**

TBL

Q-Labs

Agenda

- Introduction
- Context
- Overview
- Initiating Phase
- ➡ **Diagnosing Phase**
- Establishing Phase
- Acting Phase
- Learning Phase
- Questions(!) and Answers(?)

© 1997 Carnegie Mellon UniversityRISK97-IDEAL-29

**Carnegie Mellon University
Software Engineering Institute**

TBL

Q-Labs

Diagnosing Phase Activities

- Characterize Current & Desired States
- Develop Recommendations

© 1997 Carnegie Mellon UniversityRISK97-IDEAL-30



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Characterize Current & Desired States

The "desired state" is what your organization will be like after the change has been implemented.

The "current state" is what your organization looks like now *with respect to the desired state*.

Defining these organizational states is much easier if the following exist:

- a comprehensive model (e.g., capability maturity model)
- standardized appraisal instruments and methodologies associated with the model

© 1997 Carnegie Mellon University

RISK97-IDEAL-31



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Develop Recommendations

Recommendations consist of action(s) that can be taken to move the organization from present state to desired state.

© 1997 Carnegie Mellon University

RISK97-IDEAL-32



Agenda

Introduction

Context

Overview

Initiating Phase

Diagnosing Phase

➡ **Establishing Phase**

Acting Phase

Learning Phase

Questions(!) and Answers(?)



Establishing Phase Activities

Set Priorities

Develop Approach

Plan Actions



Set Priorities

Priorities addressed here are *for the change effort*.

Organizational realities preclude doing everything at once:

- resources needed for change are limited
- dependencies exist between recommended activities
- external factors may intervene
- organization's strategic priorities must be honored

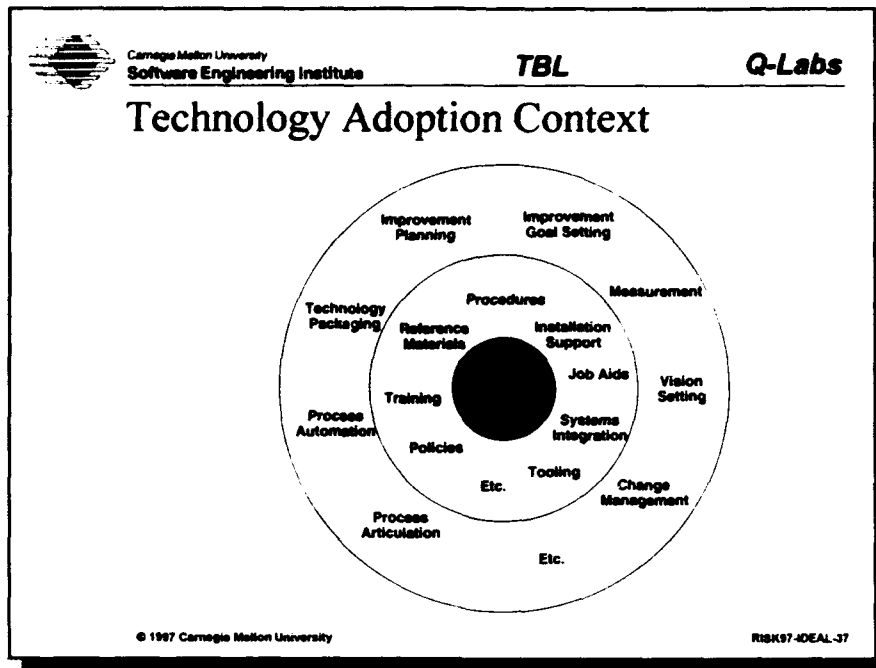
Major change almost always requires making choices.



Develop Approach

The approach taken to implementing a change accounts for many competing (and often conflicting) factors, e.g.:

- priorities
- specifics of installing the new technology
- new skills and knowledge required of the people who will be using the technology
- organizational culture
- facilitators and inhibitors
- sponsorship levels
- market forces



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Plan Actions

Plans put specifics against the approach, e.g.:

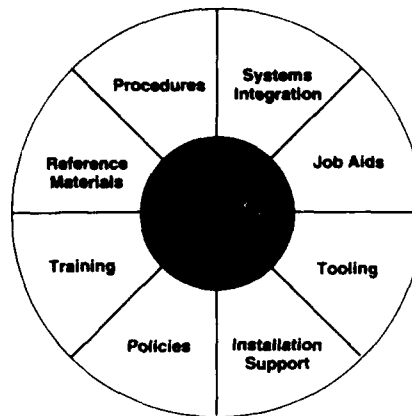
- schedule
- tasks
- milestones
- decision points
- resources
- responsibilities
- measurement and tracking
- risks & mitigation strategies

© 1997 Carnegie Mellon University

RISK97-IDEAL-38



Whole Product Concept



Agenda

- Introduction
- Context
- Overview
- Initiating Phase
- Diagnosing Phase
- Establishing Phase
- ➡ Acting Phase
- Learning Phase
- Questions(!) and Answers(?)



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Acting Phase Activities

Create Solution

Pilot/Test Solution

Refine Solution

Implement Solution

© 1997 Carnegie Mellon University

RISK97-IDEAL-41



Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Create Solution

Creating a solution means bringing key elements together in new and synergetic ways to create a "best guess" solution to address specific organizational needs.

This work is often accomplished by a technical working group.

The concept is to create *what-will-be* from *what-is*, not to make something from scratch.

© 1997 Carnegie Mellon University

RISK97-IDEAL-42



Pilot/Test Solution

"Best guess solutions"

- almost never work exactly as planned.
- should be tried out on a limited basis to learn what works and what doesn't.

The trial implementation should be conducted in an environment free of compromising factors.



Refine Solution

Once the paper solution has been put to the test, it should be modified to reflect experience and lessons learned.

Iterations of *pilot/test* and *refine* may be necessary before arriving at a solution that is deemed satisfactory.

It is important to move forward in the IDEAL cycle only when the solution is considered workable.

On the other hand, implementing a less-than-perfect solution is often more useful than waiting for perfection.

Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Implement Solution

Implementing the solution means rolling it out to the organization as a whole.

Various rollout approaches may be used, e.g.:

- top-down
- lateral
- staged

No one rollout approach is universally better than another.

For a major change, this activity may require substantial time and resources.

© 1997 Carnegie Mellon University

RISK97-IDEAL-46

Carnegie Mellon University
Software Engineering Institute

TBL

Q-Labs

Agenda

Introduction

Context

Overview

Initiating Phase

Diagnosing Phase

Establishing Phase

Acting Phase

➡ Learning Phase

Questions(!) and Answers(?)

© 1997 Carnegie Mellon University

RISK97-IDEAL-46



Learning Phase Activities

Analyze and Validate

Propose Future Actions



Analyze and Validate

To improve ability to implement change, the IDEAL experience must be revisited:

- In what ways did it/did it not accomplish its intended purpose?
- What worked well?
- What could be done more effectively or efficiently?

Records must be kept throughout the IDEAL cycle with this activity in mind.



Propose Future Actions

Recommendations for improving an organization's ability to use IDEAL need to be made based on analysis and validation.

These recommendations address a different aspect of the organization's business than those made during the Diagnosing Phase.

The people closest to the IDEAL experience rarely have the authority to make changes to it.

Proposed future actions need to be documented.



Agenda

Introduction

Context

Overview

Initiating Phase

Diagnosing Phase

Establishing Phase

Acting Phase

Learning Phase

➡ Questions(!) and Answers(?)

Exercise 1: IDEALSM Overview

How would you rate your organization with regard to how well it has accomplished the objectives of the IDEAL phases in past change implementations? What are the possible implications for your current change effort?

Initiating

☐ Good ☐ Fair ☐ Poor

Implications:

-
-
-
-

Diagnosing

☐ Good ☐ Fair ☐ Poor

Implications:

-
-
-
-

Establishing

☐ Good ☐ Fair ☐ Poor

Implications:

-
-
-
-

SM IDEAL is a service mark of Carnegie Mellon University.

Acting

☐ Good ☐ Fair ☐ Poor

Implications:

-
-
-
-

Learning

☐ Good ☐ Fair ☐ Poor

Implications:

-
-
-
-

Exercise 2: Initiating Phase

1. Briefly describe the change effort you will be focusing on during this tutorial.

2. What factors prompted your organization to take on this effort?
 -
 -
 -
 -
3. What does your organization hope to accomplish by implementing this change?
 -
 -
 -
 -
4. How will this change affect other important work your organization is doing?

5. How committed are key managers to seeing this change through to fruition?
What makes you believe that this is the case?

6. What sorts of organizational elements might need to be chartered to ensure that the effort is a success?
 -
 -
 -
 -

Exercise 3: Diagnosing Phase

1. Based upon your responses to the Initiating Phase exercise, what are some things about your organization that need to change?
 -
 -
 -
 -
 -
 -
2. What aspects about the way your organization does its business ought to be preserved?
 -
 -
 -
 -
 -
 -
3. In what ways do you expect your organization will be different after this change has been institutionalized?
 -
 -
 -
 -
 -
 -
4. How can you confirm or disconfirm these expectations?
 -
 -
 -
 -

Exercise 4: Establishing Phase

1. What elements might you want to bring together to constitute a “whole product” solution for the change your organization is implementing?
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
 -
2. Look back through your list and mark those you haven’t considered before. What impact might developing those elements have on your effort?



Peer Reviews - the key to cost-effective quality

Fran O'Hara

e-mail: franoh@worknet.ie

F.G. O'Hara and Associates

European SEPG

Amsterdam

June 16th 1997

1

F.G. O'Hara and Associates



Goals of this Tutorial

- Show why reviews are an essential part of the development process
- Present a review process that includes metrics and meets Quality Standards/SEI CMM requirements
- Skills for *effective* reviews
- Practice the process using a simple example
- Maximise your chances of successful implementation

2

F.G. O'Hara and Associates



Contents

- Why review?
- What does the process involve?
- Deployment strategy
- Case studies
- Effectiveness issues
- Metrics

3

F G O'Hara and Associates



Exercise: Process problems/issues?

- Objective:
 - ➔ Note your current process problems/issues/concerns
- How to:
 - ➔ Write down the main process problems you are facing
- Feedback
 - ➔ These will be recorded on a flipchart to be addressed during the workshop.

4

F G O'Hara and Associates



What's a Review? - the Main Phases

- **Planning/Preparation**
 - selection of participants and roles
 - kick-off meeting/informal comm.
 - reading/checking
- **Meeting**
 - reviewers raise issues, scribe logs
 - record data
 - leader role crucial
- **Edit/Follow-up**
 - author investigates issues, fixes defects
 - closure (verify?)

5

F.G. O'Hara and Associates



Contents

- Why review?
- What does the process involve?
- Deployment strategy
- Case studies
- Effectiveness issues
- Metrics

6

F.G. O'Hara and Associates



Why we need Reviews

- to improve quality
 - by finding more defects - especially major defects
 - testing 'versus' reviews?
- to improve productivity and reduce costs
 - by finding defects early in the development lifecycle (prevention/early detection) and reducing expensive rework
- to help build effective technical teams
 - communication (e.g. developers - testers)
 - education ('on the job' training)

7

F.G. O'Hara and Associates



Why we need Reviews cont.

- supports cultural shift to self-managed teams
- introduction/developing a culture of quality
- to provide data on the product and the development process
- give testers and developers a user's perspective of applications

8

F.G. O'Hara and Associates



Typical Benefits - Gilb/Graham

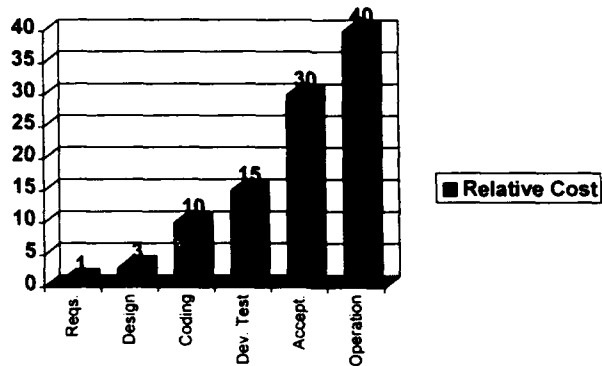
- net productivity increases of 30-100%
- net timescale reductions of 10-30%
- test execution costs and timescales reduced by a factor of 5 - 10
- maintenance costs reduced by a factor of 10

9

F.G. O'Hara and Associates



Cost of fixing defects



From an analysis of 63 projects (Boehm 1981)

10

F.G. O'Hara and Associates



Comparison of test and reviews

- **UNISYS improved their test practices**
 - ➡ **Reported that number of defects reduced by 50% with 5 levels of testing**
 - ➡ **Then implemented Inspections**
 - ➡ **Reported a 4 fold improvement in defects found over 4 years**
- **Siemens (Austria) reported efficiency in '96:**
 - ➡ **1 hour/defect with reviews**
 - ➡ **30 - 50 hours/defect with system testing of large systems**
 - ➡ **productivity x 5, 1 Million dollars/yr saved**

11

F.G. O'Hara and Associates



Formal Review Costs and RoI

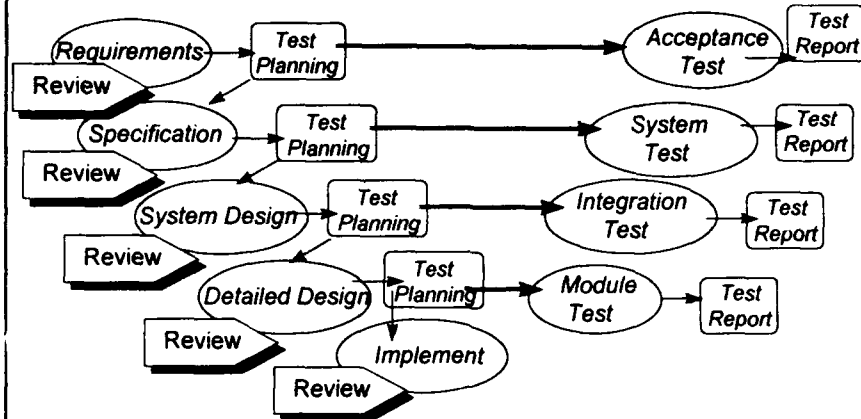
- **Relatively low initial investment required**
 - ➡ **training (plus implementation support if in-house experience not available)**
- **Ongoing allocation of resources (10-15% of development budget for formal inspections - Gilb)**
- **Capers Jones (SPR)**
 - ➡ **reported at SP '96 that 'formal reviews/inspections had the highest empirical evidence of success with almost no failures' - RoI data was 10 at 4 years!**

12

F.G. O'Hara and Associates



Where reviews fit in the process



13

F.G. O'Hara and Associates



Contents

- Why review?
- What does the process involve?
- Deployment strategy
- Case studies
- Effectiveness issues
- Metrics

14

F.G. O'Hara and Associates



What are the process goals?

- reviews are planned (i.e. in the project schedule)
- defects in work products are found and fixed
- defects are prevented?

15

F.G. O'Hara and Associates



What's a Review? - the Main Phases

- Planning/Preparation
 - selection of participants and roles
 - kick-off meeting/informal comm.
 - reading/checking
- Meeting
 - reviewers raise issues, scribe logs
 - record data
 - leader role crucial
- Edit/Follow-up
 - author investigates issues, fixes defects
 - closure (verify?)

16

F.G. O'Hara and Associates



The Review Mechanism

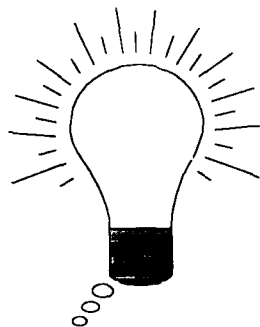
- An overview of how it works step by step

17

F G O'Hara and Associates



Put the ideas on paper



18

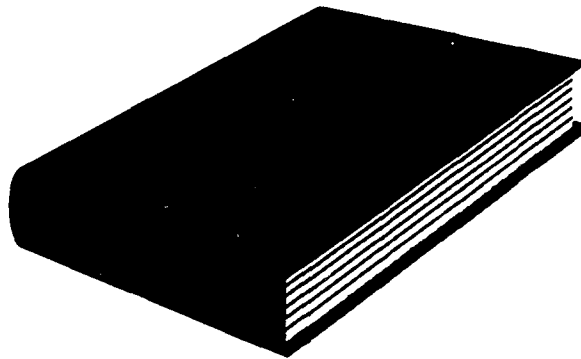
F G O'Hara and Associates

Monday 16 June

(T101d) S-9



Complete the deliverable



19

F.G. O'Hara and Associates



Appoint a review leader



20

F.G. O'Hara and Associates



Leader determines readiness for review



21

F.G. O'Hara and Associates



Identify Reviewers



22

F.G. O'Hara and Associates



Assign roles to reviewers



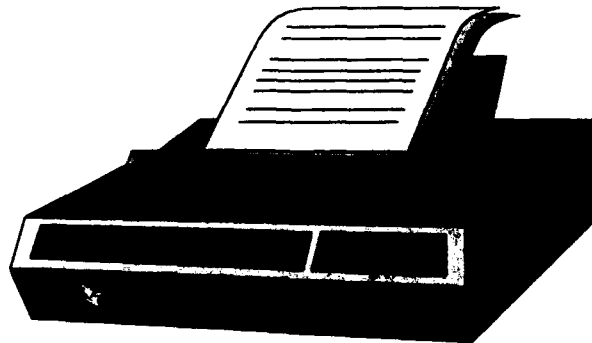
23

F.G. O'Hara and Associates



Distribute the deliverable

- with parent/source documents!



24

F.G. O'Hara and Associates



Individual checking

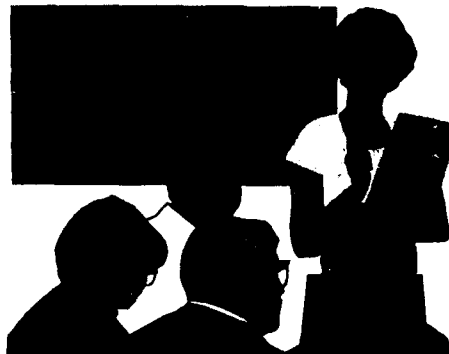


25

F.G. O'Hara and Associates



Gather for the review

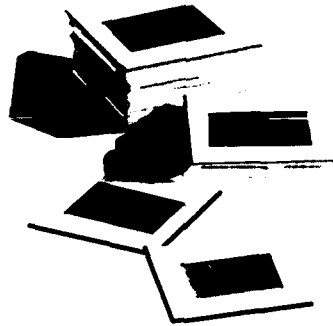


26

F.G. O'Hara and Associates



Declare the version under review



27

F.G. O'Hara and Associates



Leader gets general comments

- **This allows each person to give an overall impression.**
- **Keep comments short.**
- **The objective is to determine if this review will be a success**
- **Make a go / no go decision after everyone has made their general comment**
- **Isn't this too late?! Shouldn't general comments be used for generic issues which aren't just localised to one small part of the document?**

28

F.G. O'Hara and Associates



Go through the deliverable

- The best way is page by page.
- Allow each reviewer to make their comments in turn.
 - ➔ round robin is best
- Do not fix the problems identified
- Keep reviewer comments brief and focused on the review criteria
- Scribe to note each problem
- Collect data!

29

F.G. O'Hara and Associates



Propose an outcome - seek consensus



30

F.G. O'Hara and Associates



After the review - edit/follow-up/exit

- Author investigates issues
- Author fixes the identified problems
- Review leader checks that the solutions are adequate
- Review leader signs-off this version of the deliverable
- Review closed

31

F.G. O'Hara and Associates



Review outcomes/exit criteria

Typical outcomes of the review are:

- no change required
- review by leader only
 - ➡ verify changes
- another full review required
 - ➡ new material being added
 - ➡ based on objective count of majors/page (Gilb)
 - ➡ significant investigation on issues to be undertaken
- review incomplete

32

F.G. O'Hara and Associates



Review records

Minimum set could include:

- **List of direct issues**
- **List of related issues**
- **Review summary**

On-line?

33

F.G. O'Hara and Associates



Summary of Roles

	Leader	Author	Scribe	Reviewers
Prep	Advise, Check	Write, Organise	None	Read & Prepare
Meet	Manage	Explain	Log Issues	Comment
Post	Sign-off	Address Issues	None	May Advise

34

F.G. O'Hara and Associates



Code review - what's different?

- can be just like a document review
- additional option to paraphrase
 - ➡ someone other than the author paraphrases the code line by line
 - ➡ reviewers raise their issues as point in code is paraphrased
 - ➡ rotate role
 - ➡ others, including author, listen for differences of interpretation to intention
- code selection criteria
 - ➡ overall risk, complexity, etc.

35

F.G. O'Hara and Associates



Leader guidance

- General
 - ➡ give everyone a fair go
 - ➡ don't abuse your authority
 - ➡ think about the process and look for improvements
- Preparation/planning phase:
 - ➡ check that the following are appropriate before the meeting:
 - the review audience
 - the review questions
 - the role assignments
 - ➡ check the material (and source documents) is ready for review before being distributed (entry review)

36

F.G. O'Hara and Associates



Leader guidance cont.

- During the meeting: what is the leader's responsibility?
 - manage, control, drive, you own the review!
 - impartially chair the review
 - manage the extent of the discussion
 - make sure that issues are properly recorded
 - ensure an efficient issue logging rate is achieved
 - ask people how long they spent preparing (data!)
 - resolve conflict
 - propose the review outcome

37

F.G. O'Hara and Associates



Leader guidance - cont.

- Resolving conflict
 - get to the root of the problem - ask WHY the issue was raised
 - focus on purpose of the document (not personalities)
 - if disagreement on an issue, minute it and move on
 - remember the key points that
 - the author is responsible for the material
 - reviewers are being asked if the work product is adequate and meets criteria (not the best possible)
 - reviews are to raise not resolve issues (no design!)
 - remember the tactful phrase 'Consider if' when noting the issue

38

F.G. O'Hara and Associates



Reviewer guidance

- Preparation

- ➡ consider the effective reading approach (see 'effective reviewing' later)
- ➡ focus on the review questions - criteria/rules
- ➡ focus on any specific role assignment you may have
- ➡ use source documents and checklists as directed
- ➡ prepare adequately by scheduling in time well in advance of the review
- ➡ follow optimum checking rates intelligently
- ➡ ask for any further information you feel necessary (e.g. on-line access to code)
- ➡ re-read your comments 10 mins prior to the review

39

F.G. O'Hara and Associates



Reviewer guidance cont.

- During the review

- ➡ avoid egotism, grandstanding and soapboxing
- ➡ listen, and think
 - remember you want to find *more* major defects at the meeting
- ➡ don't be afraid to ask the dumb question - a key reviewer phrase is 'I don't understand..!'
- ➡ focus on the review questions and be constructive in your comments

40

F.G. O'Hara and Associates



Scribe guidance

- During the review
 - ➡ 'But I can read my own shorthand...'
 - ➡ Ask for people to wait while you write
 - ➡ mark up your document (different colour)
 - ➡ note the page # and speaker's initials
 - ➡ liaise with the leader to ensure you have captured the essence of each issue (maybe sit beside the leader so (s)he can check the noted issues as you go)

41

F.G. O'Hara and Associates



Author guidance

- Author
 - ➡ use the review criteria/rules when writing the document/code
 - ➡ you should be happy with the document/code *before* it enters the review process
 - ➡ clarify - don't argue or take things personally
 - ➡ follow the leader's guidance
 - ➡ remember your peers are doing you a favour

42

F.G. O'Hara and Associates



Additional guidance

- **Project manager**
 - **plan the review effort into your schedule**
 - **ensure your team have the necessary support, skills and resources to perform effective reviews**
- **SQA**
 - **ensure the planned reviews take place**
 - **audit the review process itself - help identify improvements**

43

F.G. O'Hara and Associates



The Golden Rules

- **reviews must be scheduled**
- **project plan must allocate resources for reviewing in the project schedule**
- **preparation is the key**
- **review the material not the author**
- **author is responsible for the material**
- **'is it technically adequate?' - not could it be better?**
- **author asks questions about the material which the reviewers attempt to answer**
- **raise issues, ask questions - don't design**

44

F.G. O'Hara and Associates



The Golden Rules cont.

- all reviewers are treated as peers
- management should never use results of reviews as individual performance indicators
- signature does not equal approval - a review is not a gate

45

F.G. O'Hara and Associates



Exercise: Review a Specification

- Objective:
 - ➔ Conduct a review
- Consider:
 - ➔ Remember 3 parts to every review
 - Pre-Review (Author/leader preparation, reviewers assigned, etc.)
 - Review Meeting (Minutes prepared etc)
 - Post-Review (Tidy-up etc)
- Divide into teams as directed by tutor
 - ➔ All will be reviewers but some with additional roles (eg leader, scribe...)
- Record the review issues on the templates provided

46

F.G. O'Hara and Associates



Exercise: cont'd

- You have XX minutes for this exercise
- Feedback session will be lead by review leaders.
- Discussion/Summary

47

F.G. O'Hara and Associates



Contents

- Why review?
- What does the process involve?
- **Deployment strategy**
- Case studies
- Effectiveness issues
- Metrics

48

F.G. O'Hara and Associates



Deployment strategy

- get your SEI CMM Level 2 processes under control
 - especially project planning and tracking
- introduce a basic formal review process
 - focus on identifying and fixing defects (major)
 - reviews are planned into project
- obtain cultural acceptance
- improve the process
 - migrate to an advanced process
 - additional focus on defect prevention

49

F.G. O'Hara and Associates



Basic process

- focus on identifying major/critical defects
- defect prevention - side-effect rather than a focus
- concentrate on the big questions - main criteria
 - staged use of criteria/rules and checklists
- simple and few forms
- minimise the number of meetings per review cycle
- data collection/analysis minimal and focused
- informal use of additional reviewing roles
- leaders and reviewers trained
- etc.

50

F.G. O'Hara and Associates



Advanced process

- additional focus on defect prevention (e.g. Gilb/Fagan)
- formalise more of the process
- increased quantitative control of the process
- more sophisticated sampling strategies
- increased use of rules, checklists for each doc./code type
- formalised use of additional reviewing roles
- leaders subject to initial and on-going certification
- etc.

51

F.G. O'Hara and Associates



Contents

- Why review?
- What does the process involve?
- Deployment strategy
- Case studies
- Effectiveness issues
- Metrics

52

F.G. O'Hara and Associates



Company A - Background

Background

- market leader in manufacture of PS Auto. Test Equipment and EMC Systems and Instruments
- 8 software engineers (plus 12 involved in both hardware and software)
- becoming increasingly oriented towards developing software (windows applications)
- prior code review initiative failed 'due to lack of training and experience' leading to informality and a lack of review leadership

53

F.G. O'Hara and Associates



Company A - Review Process Introduction

- introduced Peer Reviews under the TRI-SPIN program
- Training:
 - 1 day training workshop followed by 7 days consultancy support to help implement the process
- Results:
 - 'One of our very first reviews saved us considerable grief and money'
 - obtained a benefit/cost of 2:1 from first 6 month pilot phase (saving 2 hrs rework for every hour spent on reviews)

54

F.G. O'Hara and Associates



Company A - lessons learnt

- Project management improvement is a prerequisite to engineering improvement (see CMM level 2 and level 3)
 - implementation of reviews process was hampered by resource/schedule issues
- Management commitment and support at all levels is crucial
- Experienced practitioner support and just in time training are key issues with reviews
- With reviews, 'the cultural issues are probably a bigger part of things than pure techniques'

55

F.G. O'Hara and Associates



Teletronics - Background

- Teletronics Pacing Systems, Sydney
 - manufacturer of heart pacemakers and implantable defibrillators
 - real time embedded safety-critical system
 - 270 people in R&D, 35 in software
 - reviews were cornerstone of quality system - used by all disciplines
 - no review data collected
 - inefficiencies creeping in so a focused self-assessment was performed in the area of peer reviews

56

F.G. O'Hara and Associates



Teletronics - Actions taken

- results of self-assessment showed
 - ➔ cultural problems - bureaucratic, 'it's not my job', reviews seen as a gate, reviewing the author, etc.
 - ➔ process problems - inadequate preparation, conflict in meetings, purpose of review unclear, reviews find faults but not omissions, focus on style not content, etc.
- training course developed and delivered to address these cultural and process issues
- additional training on effective writing

57

F.G. O'Hara and Associates



Results and lessons learnt

- ISO9001 certification
- 0 field failures related to s/w in 1000s of years of operation (1 lab. failure related to reviews!)
- without metrics the effectiveness of review process is subjective
- formal (re)training necessary
- effective writing is a key underlying skill
- many 'solutions' devised here are actually embedded in more formal approaches

58

F.G. O'Hara and Associates



Contents

- Why review?
- What does the process involve?
- Deployment strategy
- Case studies
- **Effectiveness issues**
- Metrics

59

F.G. O'Hara and Associates



Effective reviews

- **Effective Writing**
- **Effective Reading**
- **Effectiveness tips**
- **Leader skills**

60

F.G. O'Hara and Associates



Effective writing

"What is written without effort is in general read without pleasure".

Samuel Johnson

" I do not know what I think until I see what I write" (sic)

B. Pascal

Writing is our chief way of thinking and learning

61

F.G. O'Hara and Associates



Essential questions!

- **What is the business purpose of this document?**
- **Who is my audience?**
- **What will they DO with it?**

62

F.G. O'Hara and Associates



Effective writing cont.

- **Better documents make better reviews**
 - as well better thinking and learning at the individual and organisational level
- **You have a choice**
 - write about what you know (brain dump)
 - or write about what your reader needs to know
- **Key Principles:**
 - orient everything to the reader
 - structure your thoughts clearly
 - treat writing as a process

63

F.G. O'Hara and Associates



Writing tips

- **The reader's needs affect your:**
 - content
 - structure
 - terminology
 - emphasis
- **Use pictures and diagrams**
- **Question your clarity**
- **Plan your writing - use mind-mapping**
- **Good writing has a powerful theme**
- **Use writer's divorce**

64

F.G. O'Hara and Associates



Structure

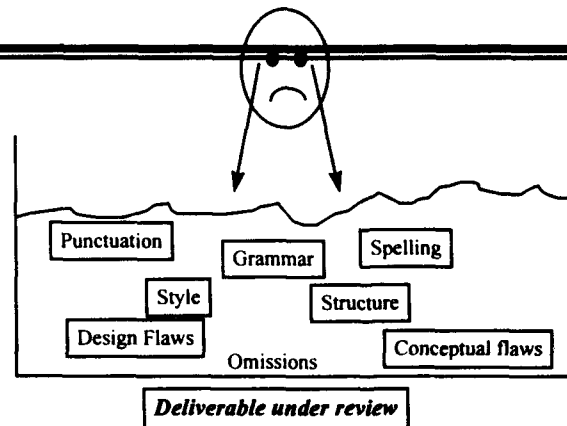
- Use good structure
 - Use headings consistently and aggressively to convey the pattern of your ideas. Make them visually prominent.
 - Talking headings - that is headings that tell you something - lift the key idea up into the heading for quick comprehension.
 - Use topic sentences in paragraphs to refer to key ideas.
 - Be careful that you highlight the topic of the sentence early in the subject matter.

65

F.G. O'Hara and Associates



The psychology of reviewing



66

F.G. O'Hara and Associates



Effective Reading

- We use patterns when we read.
- We use hierarchies to sort out importance of information.
- Readers use short-term and long-term memory to understand documents.

67

F.G. O'Hara and Associates



Effective Reading - an approach

- Concentrate on the review questions
 - ➡ Scan thorough document in entirety - 5 mins
 - ➡ Scan through again but this time more slowly - 15 mins
 - ➡ Review in detail page by page marking your questions/issues with a pen as you go
 - ➡ Scan through complete document again - 10 mins
 - ➡ p.t.o.

68

F.G. O'Hara and Associates



Effective reading - an approach cont.

- Each time you scan should ask yourself questions like
 - what's the author trying to convey
 - what's missing
 - why was that approach taken
- The scanning helps get the big picture and to focus on the conceptual issues/omissions whereas the detailed reviewing finds the 'normal' defects

69

F.G. O'Hara and Associates



Avoiding ineffective reviews - 1/5

- Preparation
 - use a pre-writing conference if appropriate
 - use risk analysis to help select items to review
 - author and leader should discuss if review criteria and review audience is appropriate (# and skills)
 - use same review audience as earlier phase e.g. code review audience should be same as design review
 - leader should scan document as a readiness for review check before it is distributed (proof read)
 - author distributes summary sheet with review criteria and roles filled in when distributing item

70

F.G. O'Hara and Associates



Avoiding ineffective reviews - 2/5

- Preparation cont.

- ➡ distribute supporting documents with the item under review e.g. design doc. and code checklist when reviewing code
- ➡ ask a secretary to schedule meeting
- ➡ distribute material at the earliest possible time to give reviewers more than enough time (as soon as reviewer agrees to participate, hand material over)
- ➡ reviewers need to schedule reviewing time in rather than leaving it till the morning of the review

71

F.G. O'Hara and Associates



Avoiding ineffective reviews - 3/5

- Preparation cont.

- ➡ if a reviewer has serious doubts about the readiness for review, don't wait till the review meeting...
- ➡ reviewers should re-read their marked up comments 10 mins prior to the review meeting

- Review meeting

- ➡ leader asks how long people spent preparing (enforce optimum checking rate!)
- ➡ leader should constantly monitor and comment on the 'height of the bar' in terms of the issues being raised during the meeting (typos, most style issues, convention deviations, etc.-pass on marked-up copy)

72

F.G. O'Hara and Associates



Avoiding ineffective reviews - 4/5

- Review meeting cont.

- raise issues don't solve them
- reviewers be constructive in your comments
- don't waste time! - everyone's responsibility but ultimately the leader's
 - conversations between individuals which are clearly specific/detailed should be done outside the meeting (note down 'to be discussed by...')
 - don't say 'I had comments A and B too', say 'No additional comments'!

73

F.G. O'Hara and Associates



Avoiding ineffective reviews - 5/5

- Follow-up

- leader checks decisions and edits of author using original document, minutes and updated version (remember 1 in 6 edits will be incomplete or incorrect according to Gilb/Graham)
- leader delegates the checking of technical issues beyond his/her knowledge
- complete edits and checking ASAP
- re-reviews use same audience and author provides traceability information from old version to help focus reviewers on changes/additions
- more?

74

F.G. O'Hara and Associates



Leader skills

- Meeting management
 - impartially chair the meeting
 - manage the extent of the discussion
 - focus on efficient issue logging rate
- Conflict resolution
 - get to root of problem - ask *why* issue was raised
 - focus on purpose of the document (not personalities)
 - if disagreement on an issue, minute it and move on
 - remember:
 - no design - raise issues don't solve them,
 - author is responsible,
 - use phrase 'Consider if...'

75

F.G. O'Hara and Associates



Contents

- Why review?
- What does the process involve?
- Deployment strategy
- Case studies
- Effectiveness issues
- Metrics

76

F.G. O'Hara and Associates



Review Metrics

- Buy-in is the biggest problemGolden rules:
 - KISS
 - explain why and how they will be used - educate
 - Avoid individual measures - productivity, quality,....
 - Easy to collect
 - graph ASAP and feedback
 - use them!
 - beware the Hawthorn effect

77

F.G. O'Hara and Associates



Review Metrics cont.

- Metrics suggestions....
 - defects/X where X = KLOC, Fn. Points, pages,...
 - efficiency
 - defects/hour
 - effectiveness
 - Defect Removal Efficiency =
Defects found/Defects that could have been found
 - cost/benefit, Return on Investment
- Basic Data required
 - time spent (preparation, meeting, follow-up)
 - size (of document/code reviewed)
 - number and type of defects

78

F.G. O'Hara and Associates



What is a "defect"?

- Definition changes in every phase of the life-cycle?
- Define a defect carefully to enable metrics gathering.
 - ➡ how do you define it? - depends on your goal!
 - ➡ can be done in terms of predicted impact on delivered product if no further review/testing was done
- The cost of detecting and correcting defects rises as you move forward in the life-cycle.

79

F.G. O'Hara and Associates



Defect definitions - Gilb/Graham

Defect = 'an error made in writing, in a document or in code which may cause a failure of any kind if the portion containing the defect is used or executed.'

'Anything which impacts or interferes with maintaining or improving quality and productivity'

'Severity is classification of an issue based on the estimated future cost to find and fix a defect at a later stage if not fixed at this stage' => minor - about the same; major - substantially greater; critical - project/product threatening

Major = would be identified by customer as needing correction - failure to meet customer expectations

80

F.G. O'Hara and Associates



Tutorial Summary

- Peer reviews are one of the most powerful engineering techniques available to help improve quality and productivity and reduce costs
- Good management control over projects facilitates successful introduction of peer reviews
- Adopt as formal a process as your culture will allow (consider the deployment strategy presented)
- Use metrics to help control and monitor the process and the work products being reviewed

81

F.G. O'Hara and Associates



Selling reviews to management

- refer to industry data and other case studies
- visit other companies
- analyse your cost of rework/corrective maintenance (from problems uncovered by testing and from field reports)
- do a pilot of the process collecting data for RoI or cost/benefit calculations - prove it!
- make explicit any necessary assumptions to allow you to put the case in terms of the bottom line....£s, \$s, etc.

82

F.G. O'Hara and Associates



Further Reading

- **Watts Humphries**
Managing the Software Process (Addison Wesley)
ISBN : 0-201-18095-2
- **Tom Gilb and Dot Graham**
Software Inspection (Addison Wesley), 1993
ISBN : 0-201-63181-4
- **Michael Fagan**
Design and Code Inspections to Reduce Errors in
Program Development, IBM Systems Journal, vol
15, no. 3, 1976, pp182-211
Advances in Software Inspections, IEEE Trans. on S/w
Eng., vol 12, no 7, 1986, pp.744-751

83

F.G. O'Hara and Associates



Further Reading cont.

- **Tervonen et. al.**
Monitoring Software inspections with prescriptive
metrics, 5th Euro. conf. on S/w Quality, 1996
- **Zopf et. al**
Toward statistical process control in software
development, 5th Euro conference on s/w quality,
1996
- **Edward Kit**
Software Testing in the real world - improving the
process (Addison-Wesley) 1995
ISBN 0-201-87756-2

84

F.G. O'Hara and Associates



Additional topics

- Rules of thumb
- SEI CMM and Peer Reviews
- Criteria, Rules and checklists

85

F.G. O'Hara and Associates



Rules of thumb

- suggested minimum notice periods related to size of work product
- meetings should not take more than 1 hour
- see Tervonen reference e.g.
 - avg. LOC inspected < 40/500 LOC (fn./chunk)
 - complexity metrics $CC < 15/100$, $V < 100/8000$ (fn./c.)
 - avg. prep. rate < 150 - 200 LOC/hr
 - avg. insp. rate < 150 - 200 LOC/hr
 - avg. for designs < 250 - 400 LOT/hr
 - prep./insp. time 1.25 - 1.4

86

F.G. O'Hara and Associates



Rules of thumb cont.

⇒ Tervonen reference cont..

→avg. effort/KLOC	50 hrs/KLOC
→avg. effort/fault det.	0.5 - 1 hr/fault
→tot. faults det./KLOC	40 faults/KLOC
→reinspect or exit	0.25 majors/page
→DRE	74%, 61%, 55%

87

F.G. O'Hara and Associates



SEI Capability Maturity Model* and Peer Reviews

*Special permission to reproduce segments of the 'Capability Maturity Model-SM for Software', V1.1 and 'Key Practices of the Capability Maturity Model', V1.1 (c) 1987 by Carnegie Mellon University, is granted by the Software Engineering Institute. CMM and Capability Maturity Model are service marks of Carnegie Mellon University.

88

F.G. O'Hara and Associates



CMM: The key Process Areas assigned to Process categories

Processes Categories	Management Software project planning, management, etc.	Organizational Senior management review, etc.	Engineering Requirements analysis, design, code, test, etc.
Levels			
5. Optimizing		Technology Change Management Process Change Management	Defect Prevention
4. Managed	Quantitative Process Management		Software Quality Management
3. Defined	Integrated Software Management Intergroup Coordination	Organization Process Focus Organization Process Definition Training Program	Software Product Engineering Peer Reviews
2. Repeatable	Requirements Management Software Project Planning Software Project Tracking & Oversight Software Subcontract Management Software Quality Assurance Software Configuration Management		
1. Initial 89	Ad Hoc Processes		F.G. O'Hara and Associates



CMM Peer Reviews - Key Practices

- Goal 1: Peer review activities are planned
- Goal 2: Defects in the S/W work products are identified and removed
- Commitment 1: The project follows a written organizational policy for performing peer reviews.
- Ability 1: Adequate resources and funding are provided for performing peer reviews on each S/W work product to be reviewed.
- Ability 2: Peer review leaders receive required training in how to lead peer reviews.
- Ability 3: Reviewers receive required training in the objectives, principles and methods of peer reviews.



CMM Peer Reviews - Key Practices cont.

- **Activity 1:** Peer reviews are planned and the plans are documented.
- **Activity 2:** Peer reviews are performed according to a documented procedure.
- **Activity 3:** Data on the conduct and results of the peer reviews are recorded.
- **Measurement 1:** Measurements are made and used to determine the status of peer review activities.
- **Verification 1:** The SQA group audits the activity.

91

F.G. O'Hara and Associates



Review criteria, rules and checklists

92

F.G. O'Hara and Associates



Review criteria

- what is the purpose of the review?
- author asks focusing questions which all reviewers attempt to answer
- improves effectiveness especially if no rules/checklists in place
- can be informal or standardised for each type of document/code

93

F.G. O'Hara and Associates



Typical (high level) review criteria

- Is it technically correct?
- Does it address all the requirements?
- Are the levels of testing appropriate?
- Is it achievable within time constraints?
- Has network time/space/cost been considered?
- Is it reusable?
- Is the context clear?
- Are service level agreements included?
- Does it implement the design as specified?
- Does it conform to coding standards and naming conventions?

94

F.G. O'Hara and Associates



Rules

- 'a writing standard (which should have directed the software engineering task which produced the product document from the source document)' - Gilb
 - a procedure for writing a document, a recommendation for best known practice, defining acceptable s/w engineering behaviour, etc.
 - are used by author and reviewers
 - can be the subject of issues raised (resulting in mods and improvements to the rules themselves)
 - e.g. how to write a test plan - contents, etc.
- ➡ see Gilb Appendix D for examples of 'rule sets'

95

F.G. O'Hara and Associates



Checklists

- used by reviewers to help find more major defects
- Gilb; interpretations of rules which help checkers find more defects
- can contain examples of typical defects
- helps capture experience/knowledge



96

F.G. O'Hara and Associates



Strategy for implementation of criteria, rules, checklists

- **start with review criteria**
 - **use existing procedural documentation instead of rules**
 - **develop basic review criteria for different types of document**
 - **raise related issues**
- **add checklists**
 - **brainstorm,**
 - **analyse defect tracking system,**
 - **do for each type of document/code**
- **add rules or expand criteria**

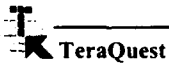
97

F.G. O'Hara and Associates

The People CMM

A Brief Introduction

Dr. Bill Curtis
TeraQuest Metrics
Austin, Texas

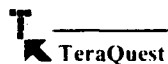


1

P-CMM Overview 1-day
© 1996 TeraQuest

Agenda

Key people issues in software	3
What is the People CMM?	7
Level 2	21
Level 3	29
Level 4	37
Level 5	44
Applying the People CMM in improvements	49



2

P-CMM Overview 1-day
© 1996 TeraQuest

Key People Issues in Software

Overwhelming impact of individual differences: 20 to 1

Knowledge is the raw material of software development—technology can not make up for a lack of knowledge.

Increasingly complex software challenges—people have become our primary software assets.

Importance of an environment that supports a skilled, knowledgeable, competent workforce

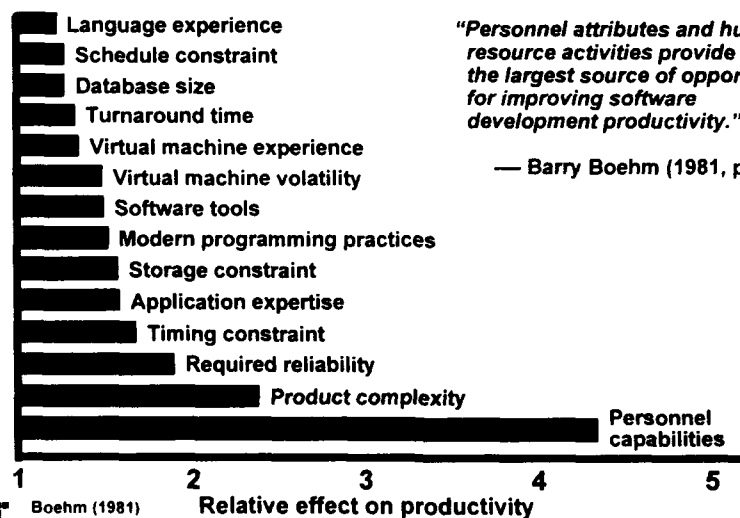
Addressing people issues as part of an integrated organizational improvement approach

TeraQuest

3

P-CMM Overview 1-day
© 1996 TeraQuest

Individual, Not Organizational Capability



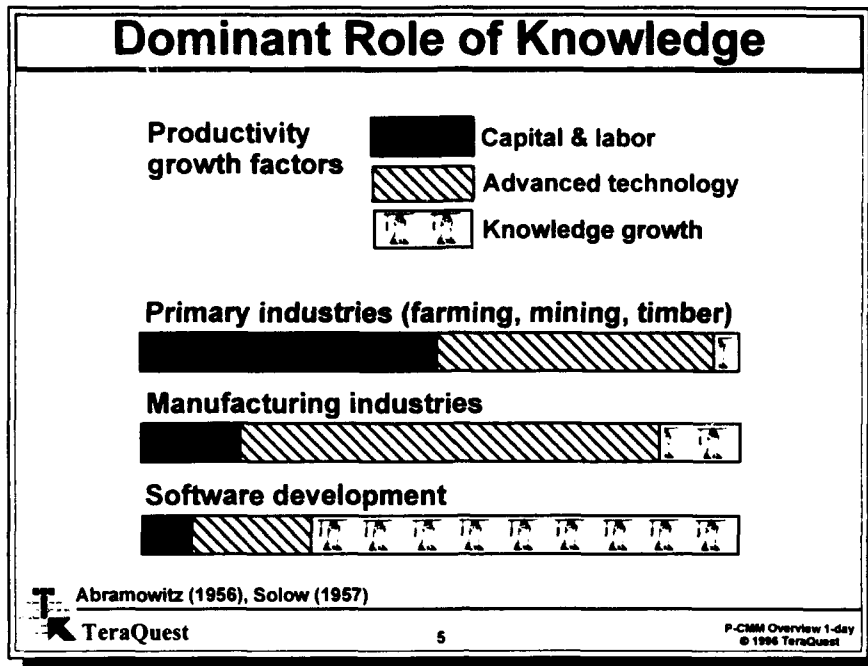
"Personnel attributes and human resource activities provide by far the largest source of opportunity for improving software development productivity."

— Barry Boehm (1981, p.666)

TeraQuest

4

P-CMM Overview 1-day
© 1996 TeraQuest



Improvement Program Shortfalls

Improvement programs for software organizations have focused on process or technology, not on people.

Many software organizations want to include improved people management in their improvement programs, but do not know how.

High maturity software organizations have found their growth required significant changes in managing people that were not accounted for in the CMM.

 TeraQuest

6

P-CMM Overview 1-day
© 1996 TeraQuest

What Is the P-CMM ?

A conceptual model based on state-of-the-art people-related practices to help software organizations:

- characterize the maturity of their people-related practices
- set priorities for improving their staff capability
- integrate improvement in people-related practices with process improvement
- establish a culture of software engineering excellence that attracts the ablest minds

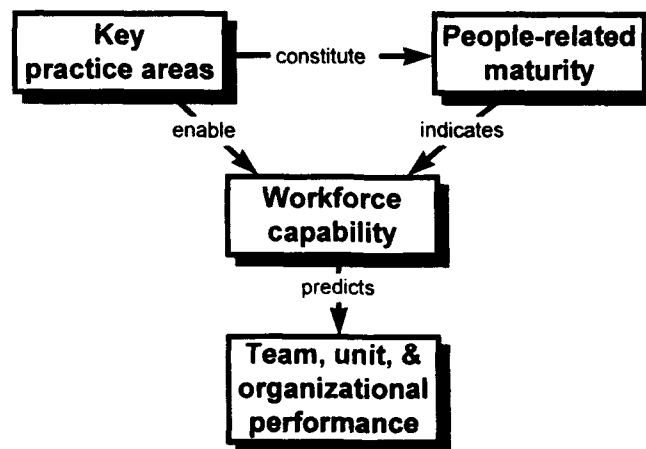
 Curtis, Hefley, & Miller (1995)
TeraQuest

Capability Maturity Model, CMM, and IDEAL are service marks of Carnegie Mellon University.

7

P-CMM Overview 1-day
© 1996 TeraQuest

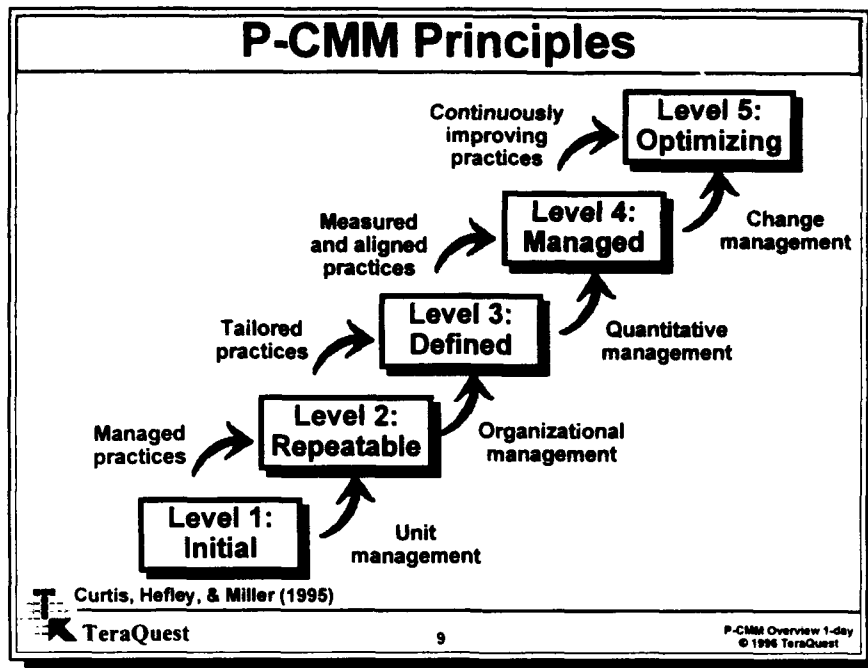
P-CMM Framework



 TeraQuest

8

P-CMM Overview 1-day
© 1996 TeraQuest



P-CMM Architecture

5 Optimizing	Continuous knowledge and skills improvement	Continuous workforce innovation Coaching Personal competency development
4 Managed	Effectiveness measured and managed, high performance teams developed	Organizational performance alignment Organizational competency management Team-based practices Team building Mentoring
3 Defined	Competency-based workforce practices	Participatory culture Competency-based practices Career development Competency development Workforce planning Knowledge and skills analysis
2 Repeatable	Management takes responsibility for managing their people	Compensation Training Performance management Staffing Communication Work environment
1 Initial		

T Curtis, Hefley, & Miller (1995)
 TeraQuest 10 P-CMM Overview 1-day © 1996 TeraQuest

Level 2—Repeatable

Executive management establishes importance of people-related values and activities.

Unit managers take responsibility for performing basic practices, laying a disciplined foundation.

Practices enhance individual contributions to unit performance.

Units know and manage their skill needs.

Successful practices can be repeated and transferred to other units, establishing a learning mechanism.



TeraQuest

11

P-CMM Overview 1-day
© 1996 TeraQuest

Level 2 Key Process Areas

Instill basic discipline

Level 2—Repeatable

**Compensation
Training
Performance Management
Staffing
Communication
Work Environment**

Level 1—Initial



TeraQuest

12

P-CMM Overview 1-day
© 1996 TeraQuest

Level 3—Defined

The organization identifies the knowledge and skills needed in its business and tailors its practices to develop them.

Practices reward knowledge and skills growth and careers are planned around them.

The organization develops strategic and near-term plans for people-related activities.

A participatory culture is established, laying a foundation for team development.

A common culture emerges based on development of core competencies across the organization.



TeraQuest

13

P-CMM Overview 1-day
© 1996 TeraQuest

Level 3 Key Process Areas

Identify core competencies and align people-related activities with them



Level 3—Defined

Participatory Culture
Competency-Based Practices
Career Development
Competency Development
Workforce Planning
Knowledge and Skills Analysis

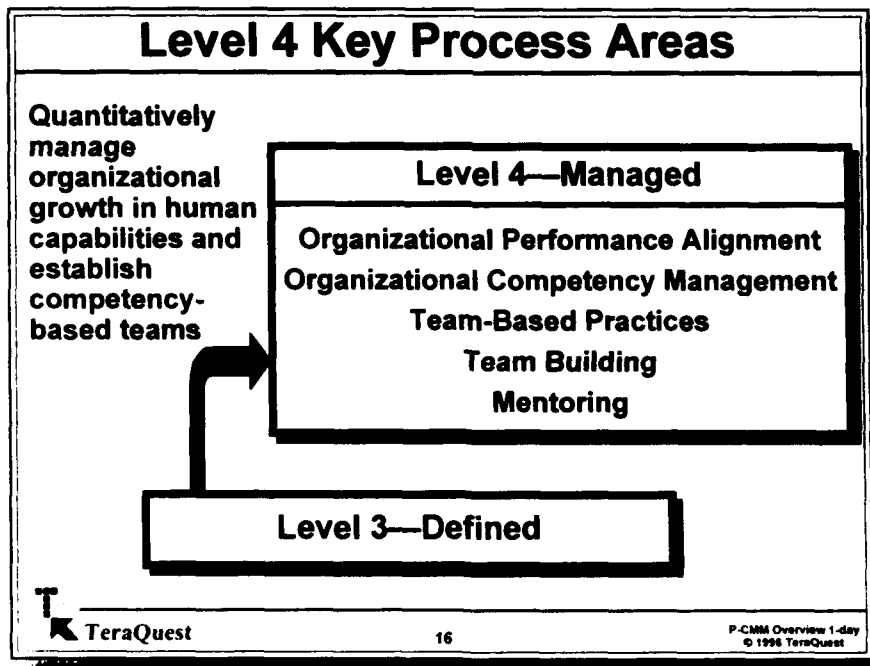
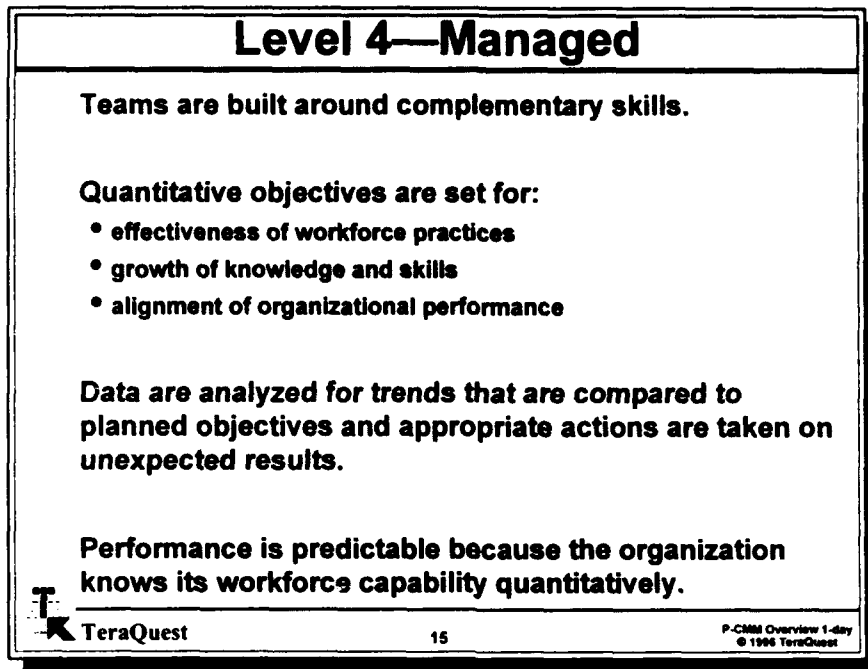
Level 2—Repeatable



TeraQuest

14

P-CMM Overview 1-day
© 1996 TeraQuest



Level 5—Optimizing

Individual staff members focus on developing their own competencies with quantitative feedback.

Coaches work with staff to improve skill performance.

The entire organization is focused on continuously improving the organization's human capability.

Improved practices and innovative methods are evaluated and piloted.

A culture of performance excellence emerges based on continuous improvement at all levels.



TeraQuest

17

P-CMM Overview 1-day
© 1996 TeraQuest

Level 5 Key Process Areas

Continuously improve methods for developing personal and organizational competence



Level 5—Optimizing

Continuous Workforce Innovation
Coaching
Personal Competency Development

Level 4—Managed



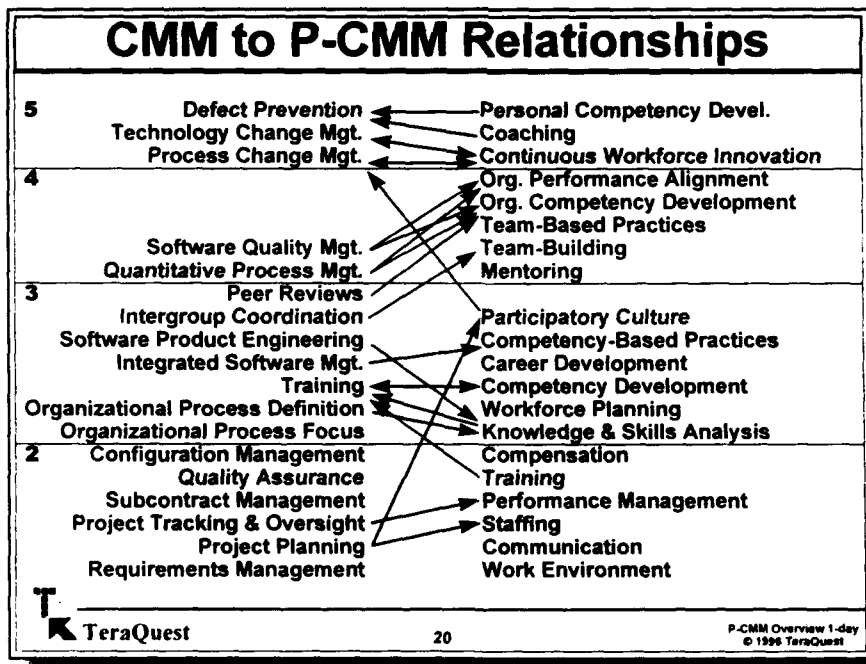
TeraQuest

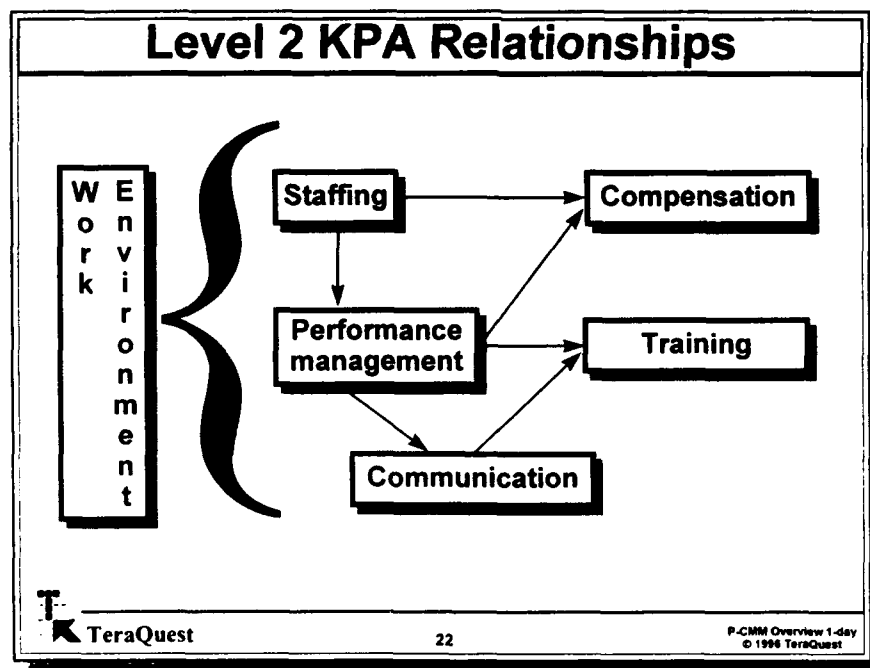
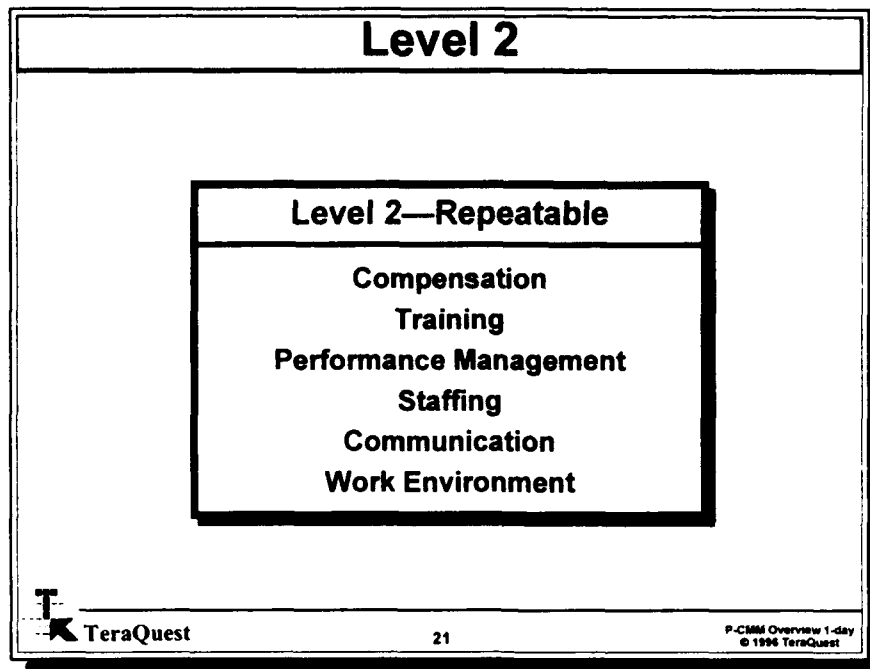
18

P-CMM Overview 1-day
© 1996 TeraQuest

P-CMM Conceptual Architecture				
Levels	Process categories			
	Developing capabilities	Building teams and culture	Motivating and managing performance	Shaping the workforce
5 Optimizing	Coaching Personal competency development	Continuous workforce innovation		
4 Managed	Mentoring	Team building	Organizational performance alignment Team-based practices	Organizational competency management
3 Defined	Competency development Knowledge and skills analysis	Participatory culture	Competency based practices Career development	Workforce planning
2 Repeatable	Training Communication	Communication	Compensation Performance management Work environment	Staffing

TeraQuest 19 P-CMM Overview 1-day © 1996 TeraQuest





Work Environment Goals

Purpose: Establish and maintain physical working conditions that allow individuals to perform their tasks efficiently and to concentrate on their tasks without unnecessary or inappropriate distractions.

Goal 1 An environment that supports the performance of business processes is established and maintained.

Goal 2 The resources needed by the workforce to perform their assignments are made available.

Goal 3 Distractions in the work environment are minimized.



TeraQuest

23

P-CMM Overview 1-day
© 1996 TeraQuest

Communication Goals

Purpose: Establish a social environment that supports effective interaction and ensure that the workforce has the skills to share information and coordinate their activities efficiently.

Goal 1 A social environment that supports task performance and coordination among individuals and groups is established and maintained.

Goal 2 Information is shared across levels of the organization.

Goal 3 Individuals develop skills to share information and coordinate their activities.

Goal 4 Individuals are able to raise grievances and have them addressed by management.



TeraQuest

24

P-CMM Overview 1-day
© 1996 TeraQuest

Staffing Goals

Purpose: Establish a formal process by which talent is recruited, selected, and transitioned into assignments in the organization.

Goal 1 The organization actively recruits for qualified talent.

Goal 2 The most qualified candidate is selected for each position.

Goal 3 Selected candidates are transitioned into their new positions.

Goal 4 Members of a unit are involved in its staffing activities.



25

P-CMM Overview 1-day
© 1996 TeraQuest

Performance Management Goals

Purpose: Establish objective criteria against which unit and individual performance can be measured, to provide performance feedback, and to enhance performance continuously.

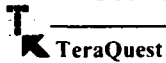
Goal 1 Job performance is measured against objective criteria and documented.

Goal 2 Job performance is regularly discussed to identify actions that can improve it.

Goal 3 Development opportunities are discussed with each individual.

Goal 4 Performance problems are managed.

Goal 5 Outstanding performance is recognized.



26

P-CMM Overview 1-day
© 1996 TeraQuest

Training Goals

Purpose: Ensure that all staff members have the skills required to perform their assignments.

Goal 1 Training in the critical skills required in each unit is provided.

Goal 2 Individuals receive timely training that is needed to perform their assignments.

Goal 3 Training opportunities are made available to all individuals.



27

P-CMM Overview 1-day
© 1996 TeraQuest

Compensation Goals

Purpose: Provide all individuals with remuneration and benefits based on their contribution and value to the organization.

Goal 1 Compensation strategies and activities are planned, executed, and communicated.

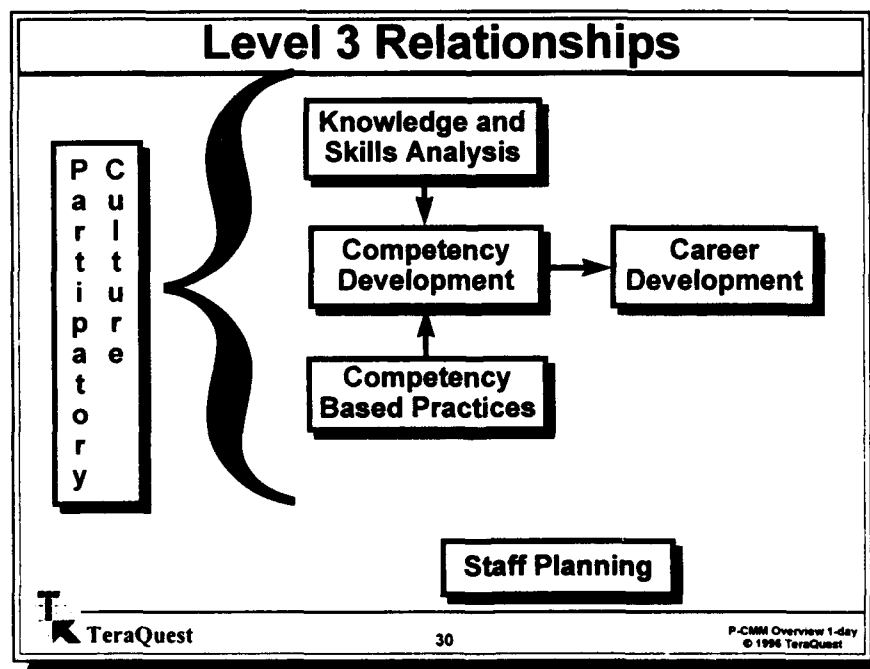
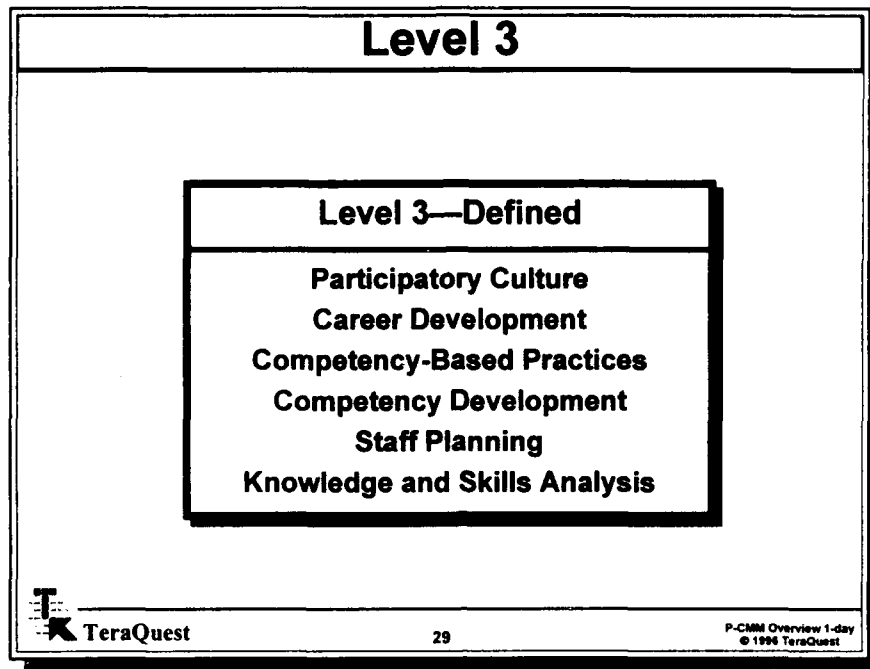
Goal 2 Compensation is equitable relative to skill qualifications and performance.

Goal 3 Adjustments in compensation are made periodically based on defined criteria.



28

P-CMM Overview 1-day
© 1996 TeraQuest



Knowledge and Skills Analysis Goals

Purpose: Identify the knowledge and skills required to perform core business processes so that they may be developed and used as a basis for workforce practices.

Goal 1 The core competencies required to perform the organization's business processes are known.

Goal 2 Knowledge and skills profiles exist for each business process.

Goal 3 Core competencies are updated for anticipated future needs.



TeraQuest

31

P-CMM Overview 1-day
© 1996 TeraQuest

Workforce Planning Goals

Purpose: Coordinate workforce activities with current and future business needs at both the organizational and unit levels.

Goal 1 The organization develops a strategic plan for long-term development of the competencies and workforce needed for its business operations.

Goal 2 Near-term and competency development activities are planned to satisfy both current and strategic workforce needs.

Goal 3 The organization develops talent for each of its key positions.

Goal 4 The organization tracks performance in achieving its strategic and near-term workforce development objectives.



TeraQuest

32

P-CMM Overview 1-day
© 1996 TeraQuest

Competency Development Goals

Purpose: Constantly enhance the capability of the workforce to perform their assigned tasks and responsibilities.

Goal 1 The organization knows its current capability in each of the core competencies required to perform its business processes.

Goal 2 The organization develops capabilities in its core competencies.

Goal 3 Individuals develop their knowledge and skills in the organization's core competencies.



TeraQuest

33

P-CMM Overview 1-day
© 1996 TeraQuest

Career Development Goals

Purpose: Ensure that all individuals are motivated and are provided opportunities to develop new skills that enhance their ability to achieve career objectives.

Goal 1 Career development activities are conducted with each individual.

Goal 2 The organization offers career opportunities that provide growth in its core competencies.

Goal 3 Individuals are motivated to pursue career goals that optimize the value of their knowledge and skills to the organization.



TeraQuest

34

P-CMM Overview 1-day
© 1996 TeraQuest

Competency-Based Practices Goals

Purpose: Ensure that all workforce practices are based in part on developing the knowledge and skills of the workforce.

- Goal 1** Workforce practices are tailored to motivate individuals and groups to improve their knowledge and skills in the core competencies of the organization.
- Goal 2** Workforce activities are adjusted to support development in the core competencies of the organization.
- Goal 3** Compensation and reward strategies are tailored to motivate growth in the core competencies of the organization.



35

P-CMM Overview 1-day
© 1996 TeraQuest

Participatory Culture Goals

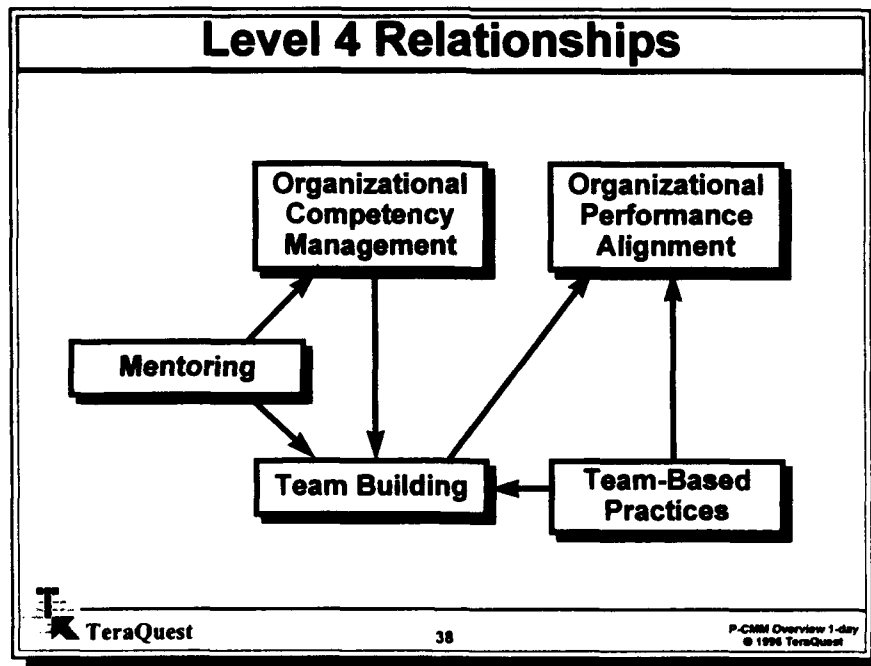
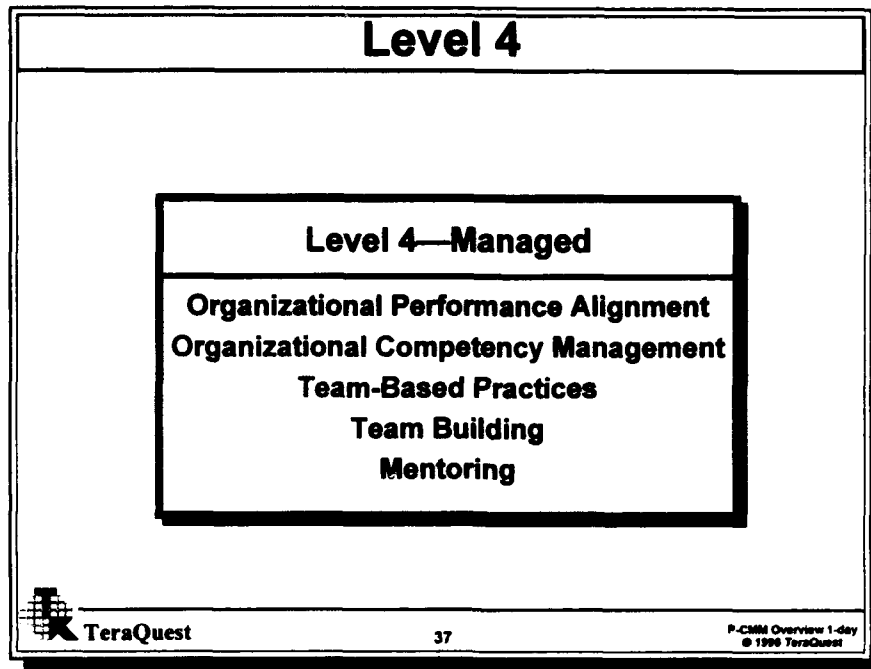
Purpose: Ensure a flow of information within the organization, to incorporate the knowledge of individuals into decision-making processes, and to gain their support for commitments.

- Goal 1** Communication activities are enhanced to improve the flow of information within the organization.
- Goal 2** Decisions are made at the lowest appropriate level of the organization.
- Goal 3** Staff members participate in decision-making processes relative to their work.



36

P-CMM Overview 1-day
© 1996 TeraQuest



Mentoring Goals

Purpose: Use the experience of the organization's workforce to provide personal support and guidance to other individuals or groups.

Goal 1 Mentoring activities are matched to defined objectives.

Goal 2 Mentors are selected and prepared for their responsibilities.

Goal 3 Mentors are made available for guidance and support other individuals or groups.



TeraQuest

39

P-CMM Overview 1-day
© 1996 TeraQuest

Team Building Goals

Purpose: Capitalize on opportunities to create teams that maximize the integration of diverse knowledge and skills to perform business functions.

Goal 1 Teams are formed to improve the performance of interdependent tasks.

Goal 2 Team assignments are made to integrate complementary knowledge and skills.

Goal 3 Team members develop their team skills.

Goal 4 Team members participate in decisions regarding their work.

Goal 5 The organization provides standard processes for tailoring and use by teams in performing their work.



TeraQuest

40

P-CMM Overview 1-day
© 1996 TeraQuest

Team-Based Practices Goals

Purpose: Tailor the organization's workforce practices to support the development, motivation, and functioning of teams.

Goal 1 The organization adjusts its workforce practices and activities to motivate and support the development of team-based competencies within the organization.

Goal 2 Workforce activities are tailored to support the needs of different types of teams within the organization.

Goal 3 Team performance criteria are defined and measured.

Goal 4 Compensation and reward systems are tailored to motivate improved team performance.



TeraQuest

41

P-CMM Overview 1-day
© 1996 TeraQuest

Organizational Competency Management Goals

Purpose: Increase the capability of the organization in its core competencies, and to determine the effectiveness of its competency development activities in achieving specific competency growth goals.

Goal 1 Measurable goals for capability in each of the organization's core competencies are defined.

Goal 2 Progress toward achieving capability goals in the organization's core competencies is quantified and managed.

Goal 3 The knowledge and skills building capability of the organization's competency development activities is known quantitatively for each of its core competencies.



TeraQuest

42

P-CMM Overview 1-day
© 1996 TeraQuest

Organizational Performance Alignment Goals

Purpose: Enhance alignment of performance results at the individual, team, unit, and organizational levels with the appropriate goals, and to quantitatively assess the effectiveness of workforce practices on achieving alignment.

Goal 1 Measurable goals for aligning individual, team, unit, and organizational performance are defined.

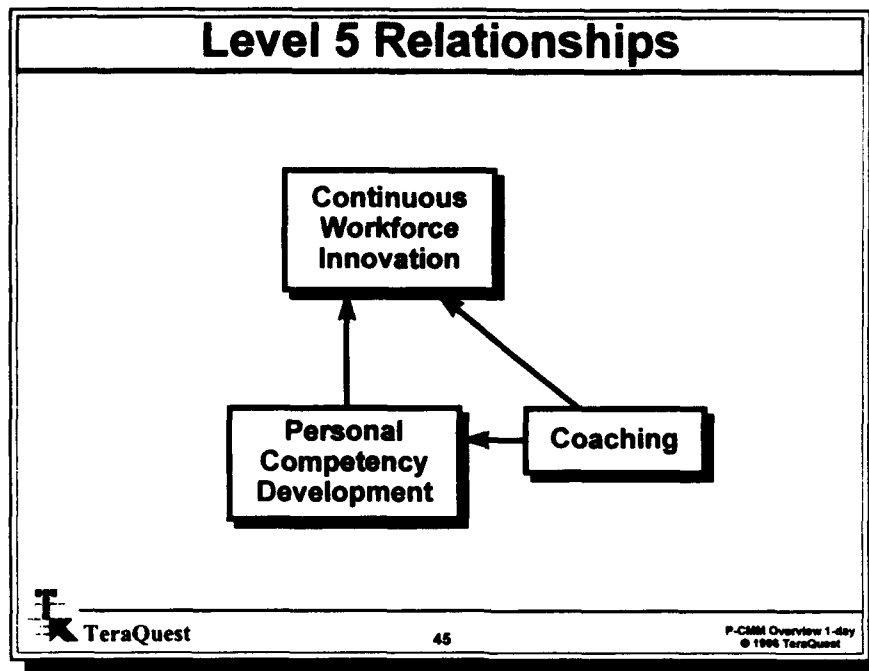
Goal 2 Progress toward achieving performance alignment goals is quantified and managed.

Goal 3 The capability of workforce activities to align individual, team, unit, and organizational performance is known quantitatively.

Level 5

Level 5—Optimizing

Continuous Workforce Innovation
Coaching
Personal Competency Development



Personal Competency Development Goal

Purpose: Provide a foundation for professional self-development.

Goal 1 Individuals know their capability in each of the competencies involved in their work.

Goal 2 Individuals continuously improve their knowledge and skills in the competencies involved in their work.

Goal 3 Participation in improving personal competencies is organization-wide.

TeraQuest 46 P-CMM Overview 1-day © 1996 TeraQuest

Coaching Goals

Purpose: Provide expert assistance to enhance the performance of individuals or teams. Coaches engage in close relationships with individuals or teams in order to guide development of skills that improve performance.

Goal 1 Coaches are selected for their expertise and prepared for their responsibilities.

Goal 2 Coaches work with individuals to improve their personal competency and performance.

Goal 3 Coaches work with teams to improve their team-based competencies and performance.



TeraQuest

47

P-CMM Overview 1-day
© 1996 TeraQuest

Continuous Workforce Innovation Goals

Purpose: Identify and evaluate improved workforce practices and technologies, and implement the most promising ones throughout the organization.

Goal 1 Innovative workforce practices and technologies are evaluated to determine their effect on improving core competencies and performance.

Goal 2 The organization's workforce practices and activities are improved continuously.

Goal 3 Participation in improving the organization's workforce practices and activities is organization-wide.



TeraQuest

48

P-CMM Overview 1-day
© 1996 TeraQuest

Applying P-CMM in SPI Programs

Identify the P-CMM effort as a component of the overall software process improvement (SPI) program to senior management, not as a separate program conducted by the Human Resources group

Blend people management and ongoing process improvement activities

Develop a strong partnership between Human Resources and software management

- Leverage existing HR personnel and resources
- Consider placing an HR person on the SEPG



TeraQuest

49

P-CMM Overview 1-day
© 1996 TeraQuest

P-CMM Assessment Goals

Provide profile of workforce practices:

- collect data on current workforce practices
- identify strengths and weaknesses
- determine satisfaction of key process area goals

Guide improvement actions:

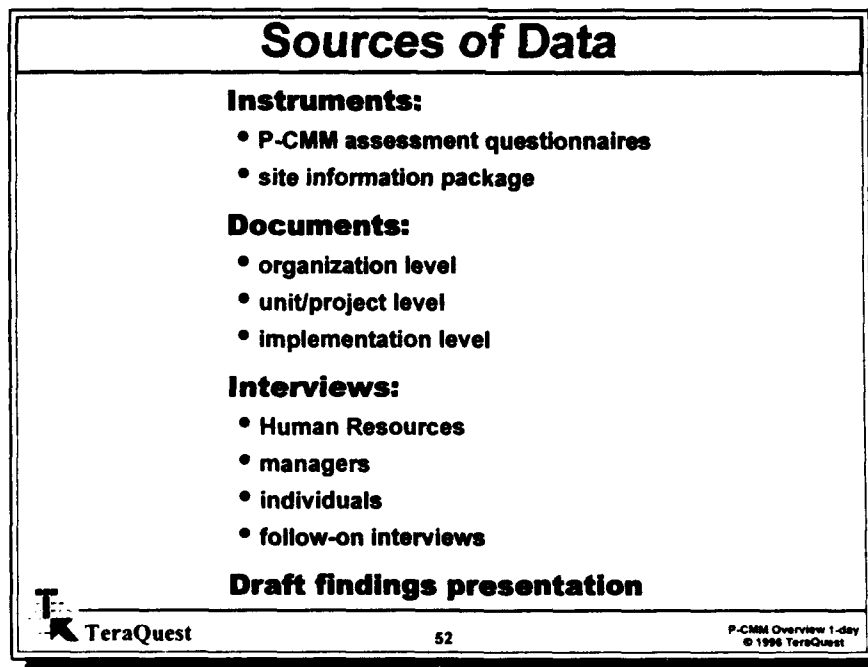
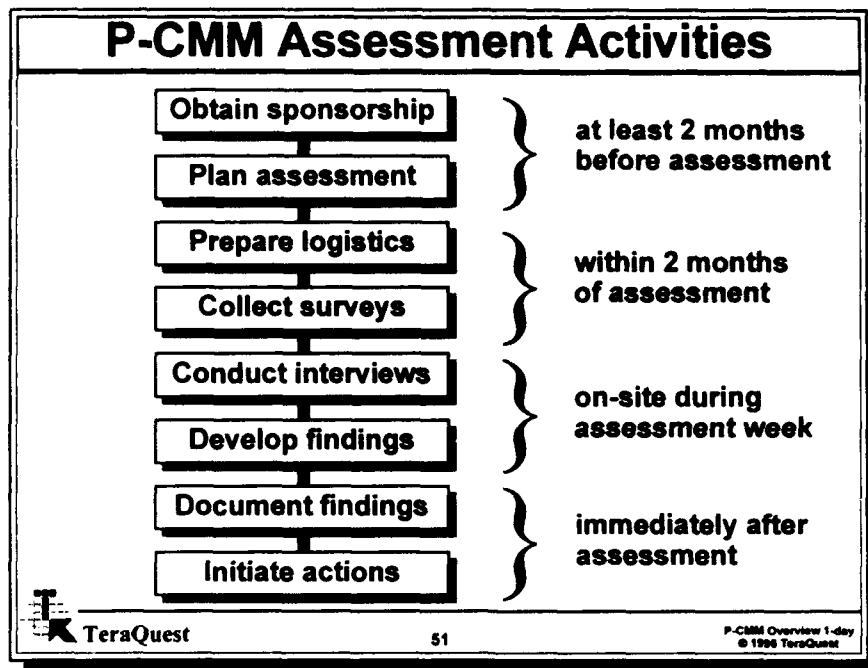
- generate consensus and buy-in
- establish ownership of results
- provide a framework and catalyst for action
- reinforce sponsor leadership and commitment



TeraQuest

50

P-CMM Overview 1-day
© 1996 TeraQuest



Example P-CMM Survey Result

44. To what extent have you received the training you need to perform your work responsibilities?

22	146	279	83	41	54
complete extent	great extent	some extent	little extent	no extent	don't know

Comments:

"Training was provided too late to be useful on my job"

"Training was too generic—unrelated to our applications"

"Found mentoring by senior programmers to be more beneficial than classroom training"

"By the time the training was provided I had transferred to another project and needed knowledge of a different application"

"Why doesn't someone train my manager?"

"Training helped a lot, could have used more"

"Apprenticeship to senior programmer equalled 2 years of college"



TeraQuest

53

P-CMM Overview 1-day
© 1996 TeraQuest

P-CMM Assessment Onsite Activities

1. 1 hr. Team coordination	6. 2 hrs. Complete HR interviews	11. 1.5 hrs. Workforce interviews	16. 1 hr. Legal review	21. 1.5 hrs. Final findings presentation
2. 1 hr. Participant's briefing	7. 2 hrs. Consolidate HR data	12. 1.5 hrs. Workforce interviews	17. 2 hrs. HR review	22. 2 hrs. Debrief sponsor
3. 2 hrs. Survey analysis	8. 1.5 hrs. Manager interviews	13. 1 hr. Consolidate workforce data	18. 1.5 hrs. Manager review	23. 2 hrs. Team wrap-up
4. 2 hrs. Interview planning	9. 1.5 hrs. Manager interview	14. 2 hrs. Preliminary findings	19. 1.5 hrs. Workforce review	4. 2 hrs. Interview planning
5. 2 hrs. Initial HR interview	10. 1.5 hrs. Consolidate manager data	15. 2 hrs. Preliminary briefing	20. 4 hrs. Final findings development	5. 2 hrs. Initial HR interview



TeraQuest

54

P-CMM Overview 1-day
© 1996 TeraQuest

Example P-CMM Assessment Finding

Performance management activities are performed inconsistently and are occasionally ineffective:

- individuals in several departments have no objective performance criteria defined for their responsibilities
- many managers do not maintain continuing awareness of the performance of those they supervise
- performance feedback often occurs too long after the actual performance to provide useful guidance for improvement
- in many cases no action is taken on poor performers
- many categories on the performance appraisal form are not relevant to technical jobs
- most managers have not been trained in performance management techniques
- discussion of improvement activities and career options is inconsistent across managers



TeraQuest

55

P-CMM Overview 1-day
© 1996 TeraQuest

SEI P-CMM Assessment Activities

Draft assessment method:

- under review
- being piloted

Assessment pilots:

- Citibank (3-96)
- GDE Systems (4-96)
- SEI (6-96)
- US Army (7-96)
- Boeing (1-97)

Release of method description in 1997:

- no lead assessor program planned
- self-assessment training available



TeraQuest

56

P-CMM Overview 1-day
© 1996 TeraQuest

Obtaining a Copy of the P-CMM

Descriptions:

<http://www.sei.cmu.edu/products/publications/95.reports/95.mm.001.html>

<http://www.sei.cmu.edu/products/publications/95.reports/95.mm.002.html>

Reports:

<ftp://ftp.sei.cmu.edu/pub/documents/95.reports/pdf/mm001.95.pdf>

<ftp://ftp.sei.cmu.edu/pub/documents/95.reports/pdf/mm002.95.pdf>

Other information:

SEI Customer Relations

1-412-268-5800

customer-relations@sei.cmu.edu



TeraQuest

57

P-CMM Overview 1-day
© 1996 TeraQuest

Contacting TeraQuest

TeraQuest
P.O. Box 200490
Austin, Texas 78720-0490
USA
1-512-219-9152 (phone)
1-512-219-0587 (fax)
<http://www.teraquest.com/>

Process assessment
Process improvement
Software architecture
Software measurement
Risk analysis
Project mgt. training
SE-CMM services
P-CMM services

Don Oxley	oxley@teraquest.com	1-512-219-9152	President
Dr. Bill Curtis	curtis@acm.org	1-512-219-0286	Chief Scientist
Dr. Joyce Statz	statz@teraquest.com	1-703-219-0358	VP



TeraQuest

58

P-CMM Overview 1-day
© 1996 TeraQuest

DEALING WITH THE UNDERWORLD



Dealing with the Underworld

Origin/Quality Management

Jeroen Brinkman

Mike Morrell

Origin/In-Product Software

Wilko van Asseldonk

Dealing with the Underworld

Theory: Mike

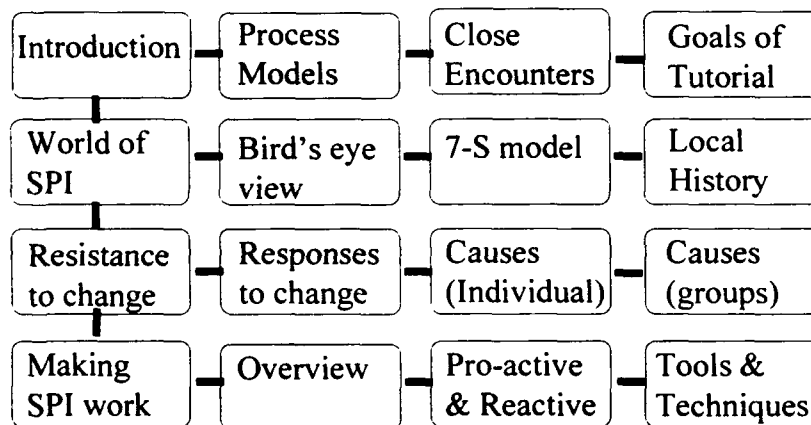
--- Break ---

Practice (1): Wilko

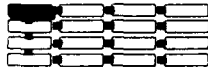
Practice (2): Jeroen

Panel session: Mike, Wilko, Jeroen

THEORY - ROAD MAP



Dealing with the Underworld



INTRODUCTION



Introduction - Process Models



Introduction - Process Models

Current Processes

Improved Processes



Introduction - Process Models



"This is taking too much time"

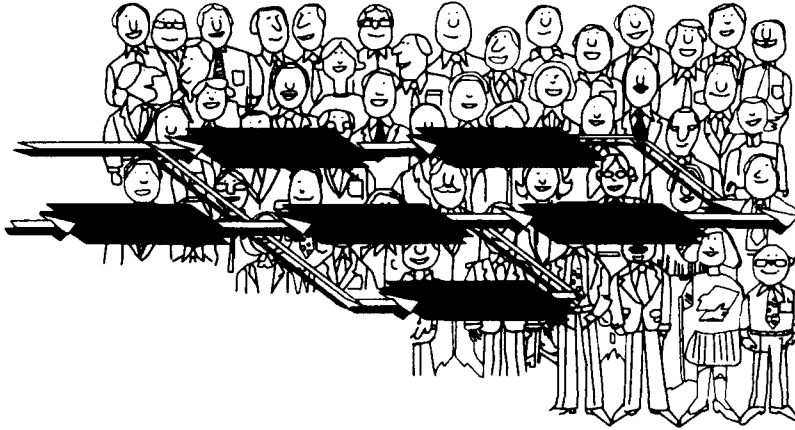
"We don't understand what you want us to do"

"This is never going to work - it's much too complicated"

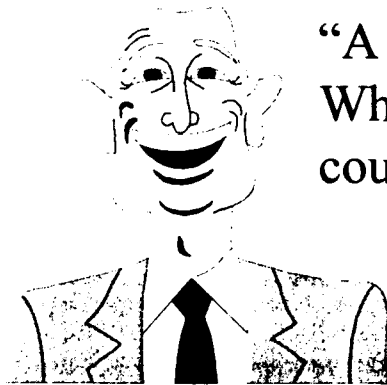
"Well it may work for *other* organisations, but our department is a special case..."

"What's wrong with the way things are at the moment, anyway?"

Introduction - Process Models



Close Encounters



“A SPI project?
What a great idea,
count me in!!”

Close Encounters of the 1st kind

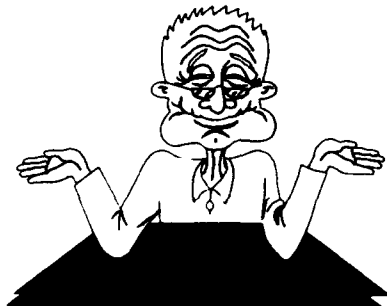


"This new tracking process won't interface to our financial systems"

"This tool will be too difficult to use for small projects"

"This method of handling requirements is going to slow us down too much"

Close Encounters of the 2nd kind



"Well it may work for other projects, but . . ."

"I don't think that it'll work in this department - my staff will never go for it"

CONFLICT WITH ESTABLISHED PRACTICES, CULTURE, ETC.

Close Encounters of the 3rd kind



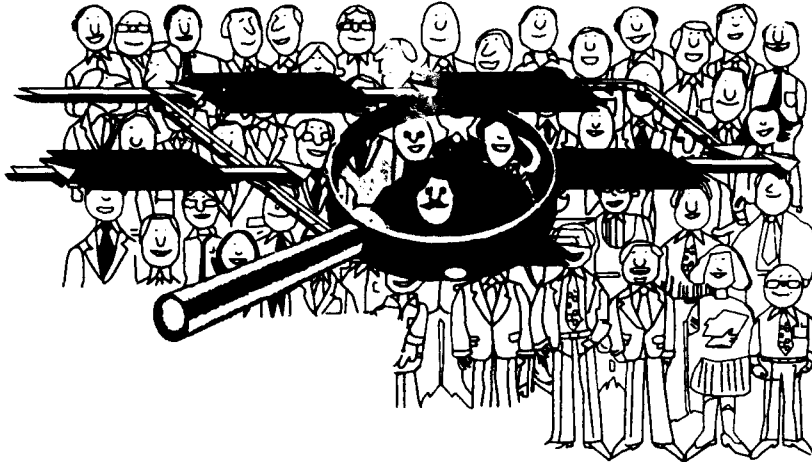
*The
Underworld*

Close Encounters of the 3rd kind

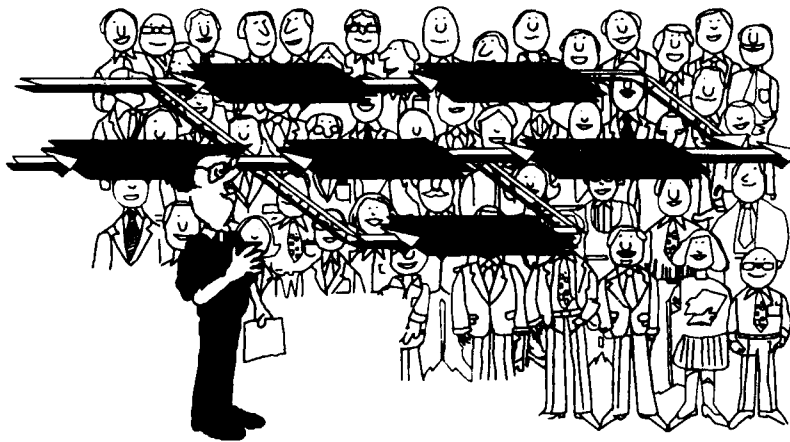


- What's happening here?
- Why is it happening?
- How do I deal with it?

Introduction - Goals of Tutorial



Introduction - Goals of Tutorial



Dealing with the Underworld



THE WORLD OF SPI



Bird's-eye view Organisational issues

Social issues



Economic issues



Political issues

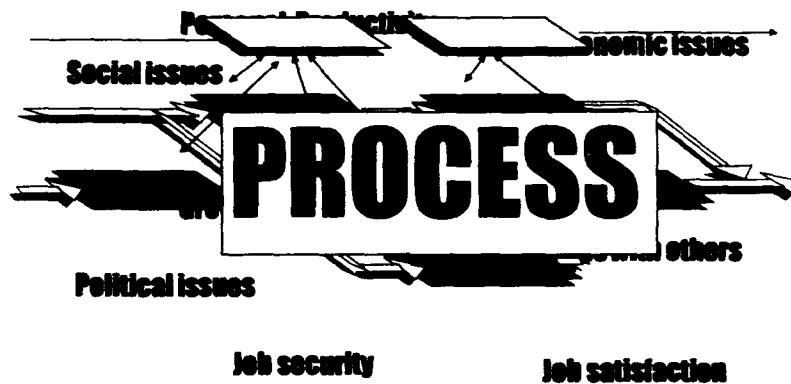


Bird's-eye view: Individual issues

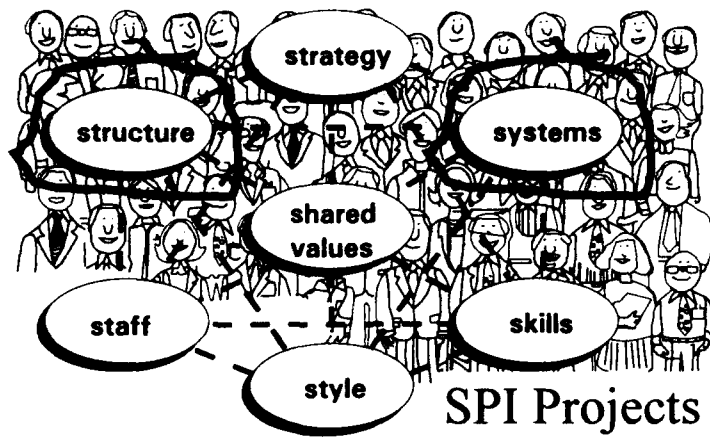


- Job security
- Job satisfaction
- Growth prospects
- Relationships with others
- Personal Productivity
- Professional Interests
- Etc.

Bird's-eye view: SPI issues



World of SPI: The 7-S model



World of SPI: Local history

Then . . .



Now



- Reorganisations
- Major projects
- Periods of hardship
- Influential people
- Etc.

The world of SPI (summary)

- The success of SPI depends on more than 'PROCESS'
- There are wider organisational issues and individual issues at stake
- The 7-S model (or equivalent) helps to get a 'bird's-eye' view
- You can learn a lot from local history.

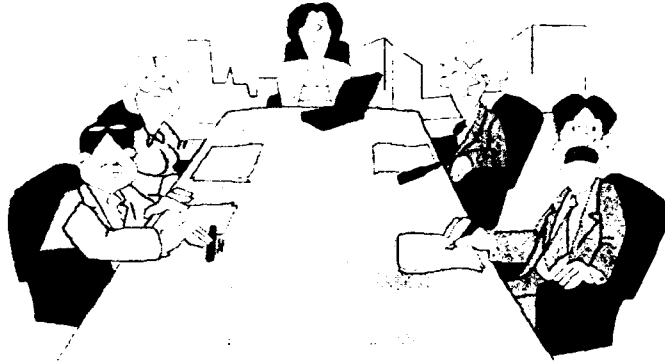
Dealing with the Underworld



RESISTANCE TO CHANGE

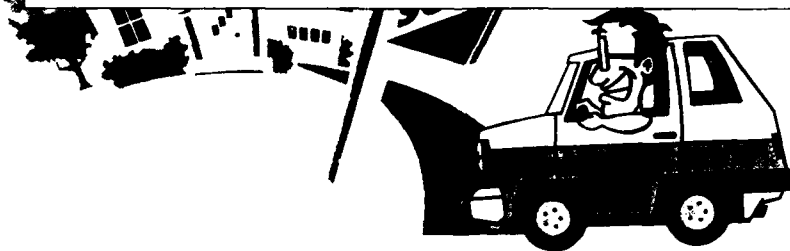


A new SPI project . . .



Responses to change - Old Habits

**RESISTANCE IS OFTEN
INDEPENDENT OF WHETHER
SOMEONE IS FOR OR AGAINST THE
CHANGE**



Responses: Open Resistance



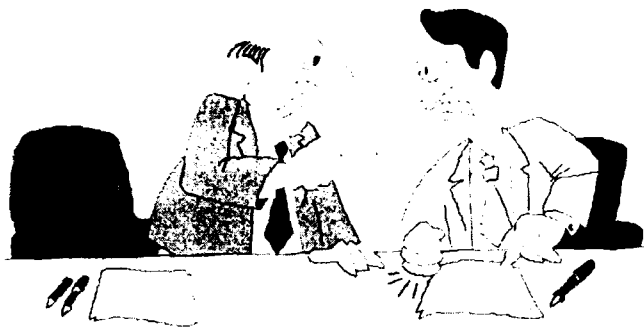
- "We're developing our own (better) method . . ."
- "We manage to work around the new procedures . . ."
- "The old way was much more efficient . . ."
- "Actually, nothing much has really changed"

Responses: Hidden resistance

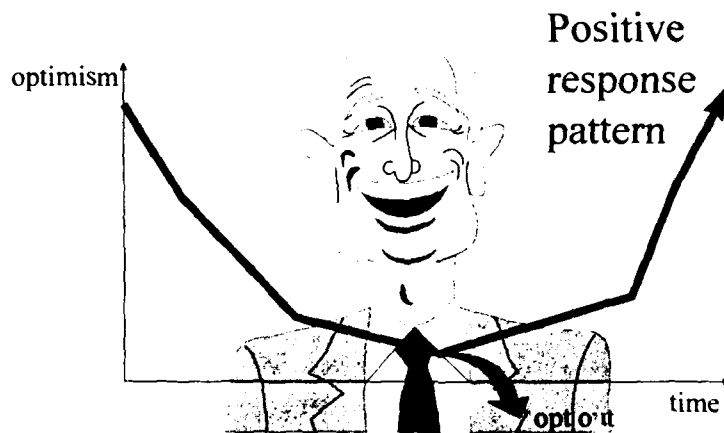


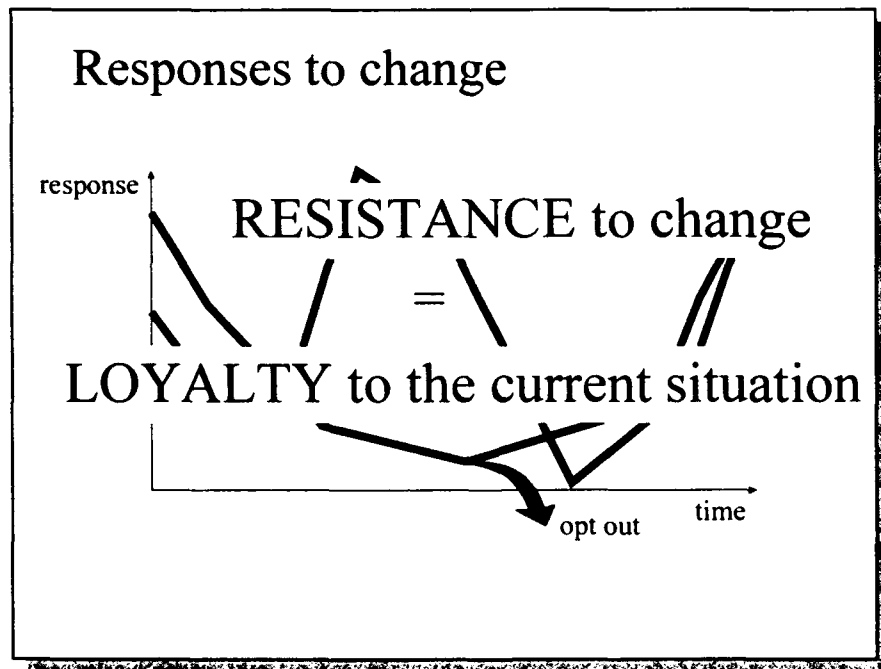
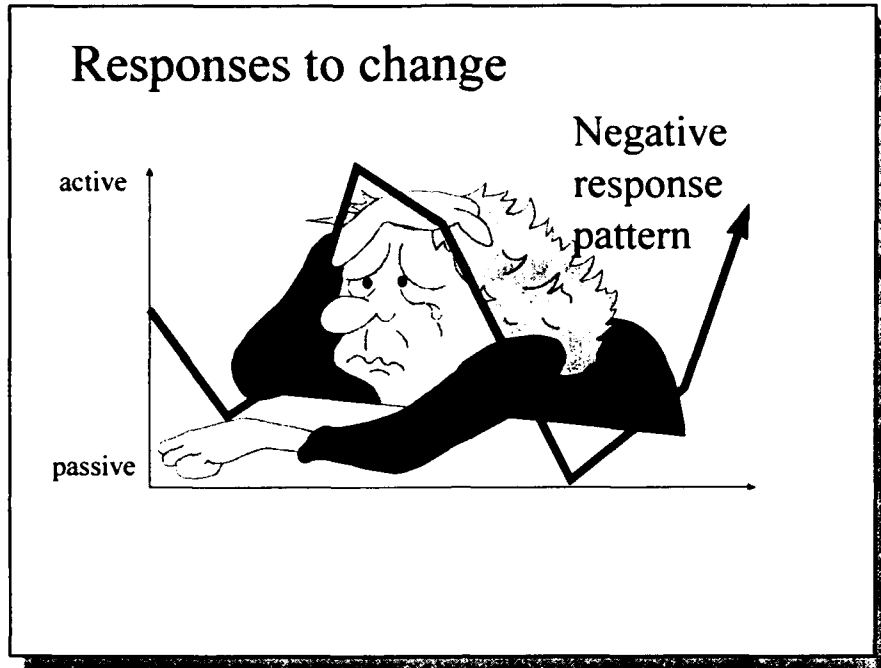
- "I'd really like to help you out but . . ."
- "I have so little time at the moment . . ."
- "Let's talk about it when I come back from vacation ."
- "It's so difficult to get people together . . ."
- "I'm sure you can manage without me . . ."

Responses: Hidden Resistance (?)

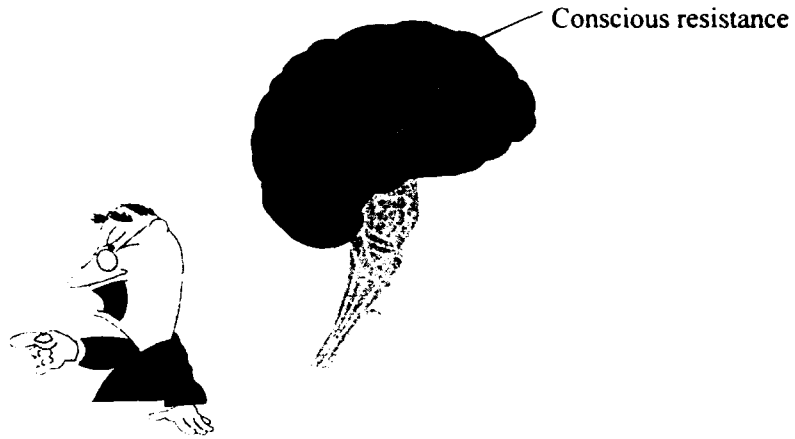


Responses to change

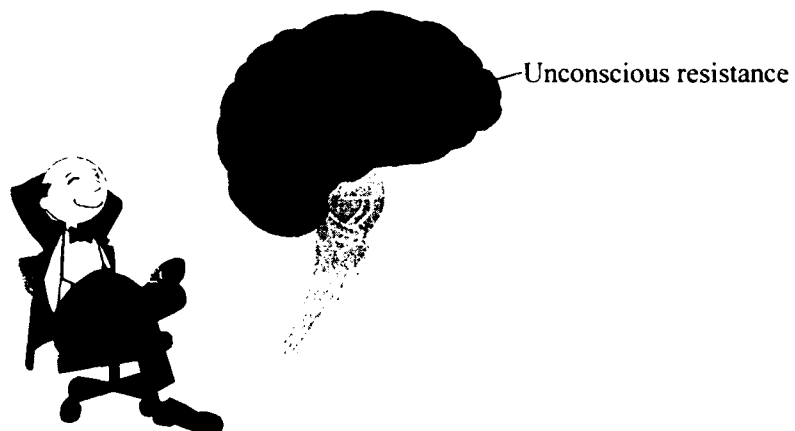




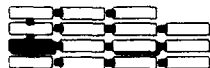
Responses to change



Responses to change



Resistance



Causes

Causes of resistance (individuals)

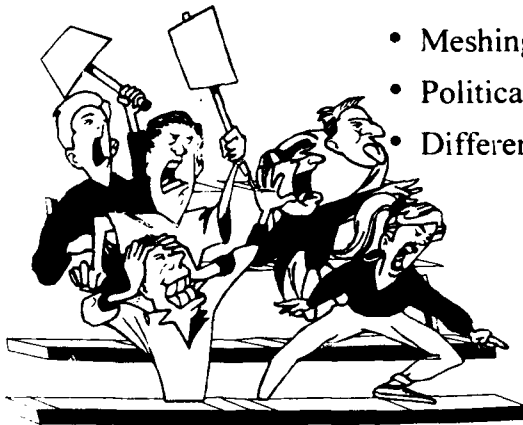


- Lack of support for goals
- Fear of losing freedom
- Fear of losing stability/control
- Avoidance of discomfort
- Lack of personal resources

Causes of resistance (individuals)

*People prefer the
'certainty of misery' to
the 'misery of
uncertainty'*

Causes of resistance (groups)



- Meshing of individual issues
- Political differences
- Different frames of reference

Resistance (summary)

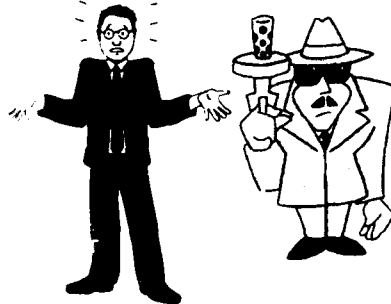
Resistance is . . .

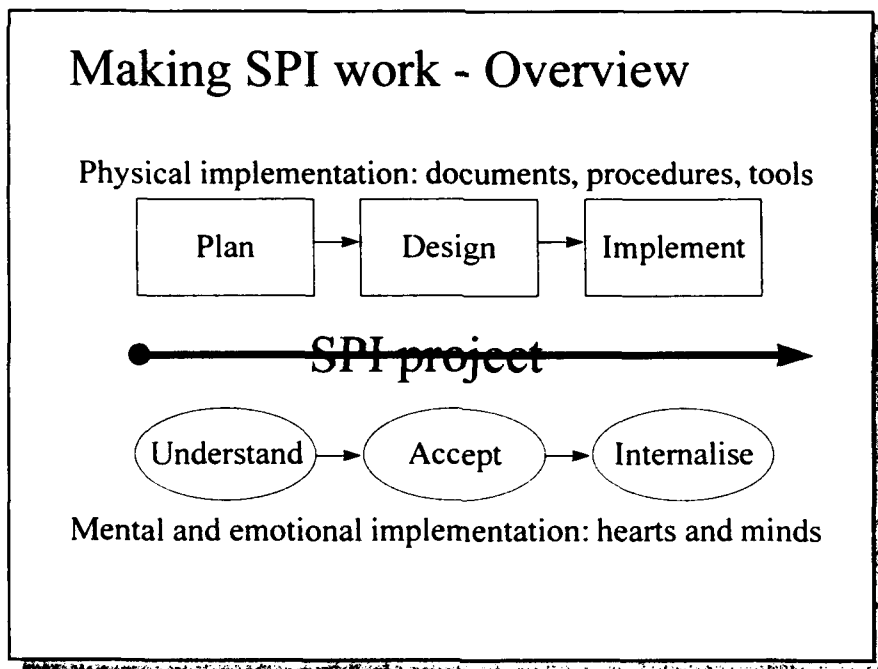
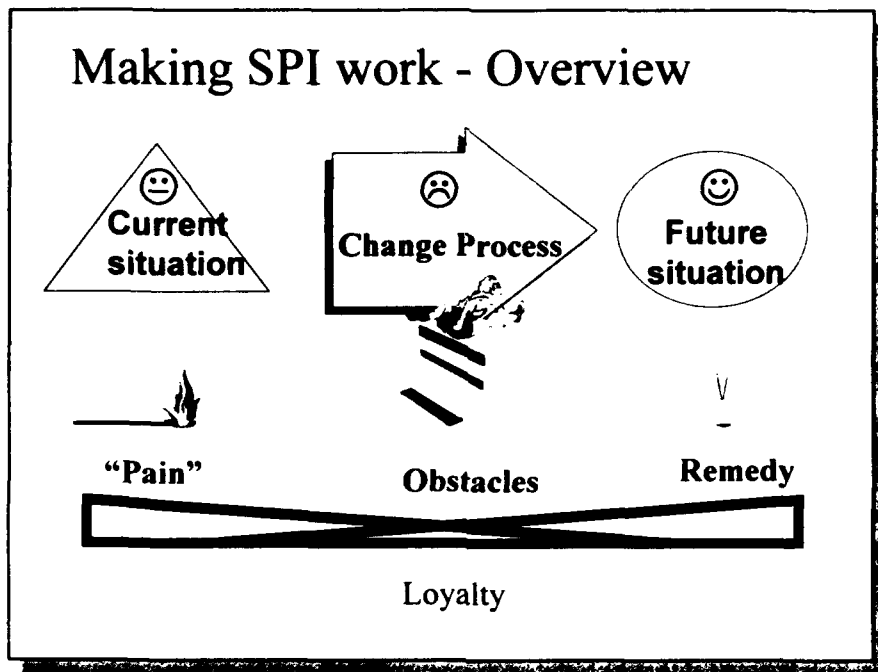
- loyalty to the current situation
- a normal and healthy response
- dependent on one's personal frame of reference
- determined by one's *ability* to adapt to a change
- determined by one's *willingness* to adapt to a change
- easy to see, if you're looking out for it

Dealing with the Underworld

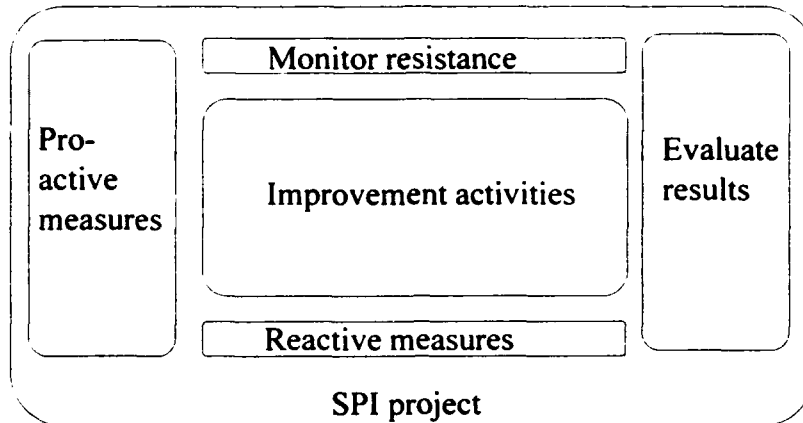


MAKING SPI WORK





Making SPI work - Overview



Making SPI work: Pro-actively

- Make sure people understand the need for change
- Make sure people understand the consequences
- Be prepared to pay the price
- Let people understand what's going to happen
- Give people influence
- Get real "buy-in"



Making SPI work: Monitoring

“The proof of the pudding
is in the eating”

(so let people eat)



Making SPI work: Monitoring

Look out for hidden
resistance

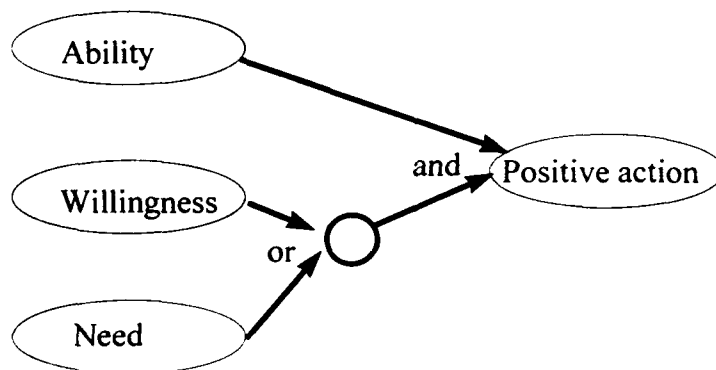


Making SPI work: Monitoring

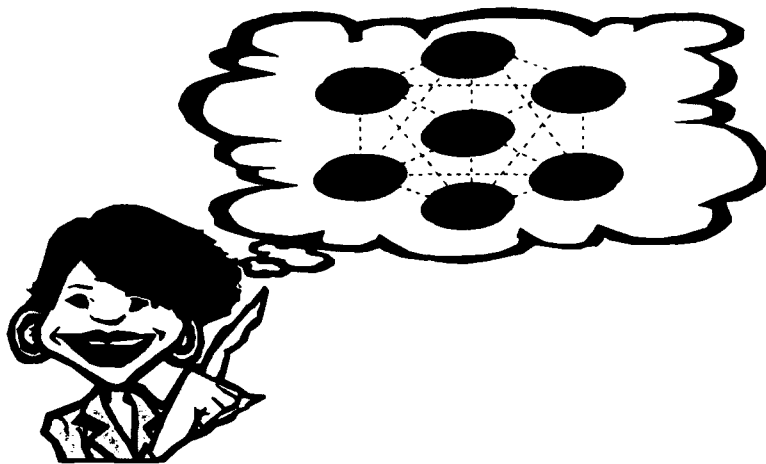


Welcome it
- it's telling you something!

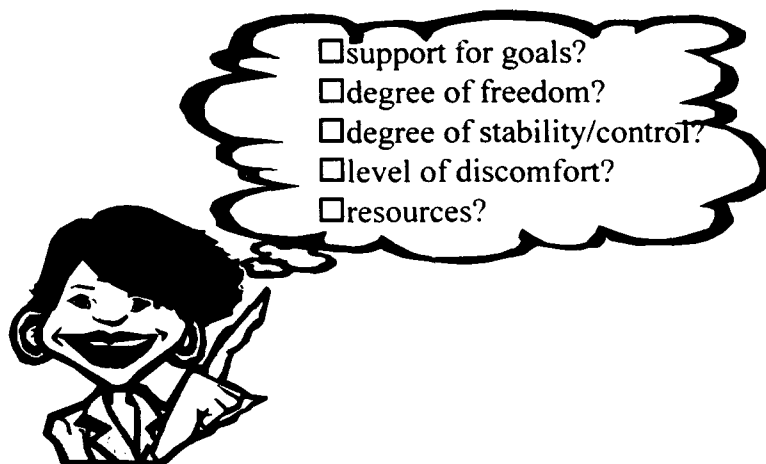
Making SPI work: Monitoring



Making SPI work: Monitoring



Making SPI work: Monitoring



Making SPI work: Re-actively

Open
Support

Open
Resistance

Hidden
Resistance



- Confront people with the *consequences* of the resistance
- Get 'under the surface'
- Get people to talk 'straight from the heart'
- Draw the cause of the resistance out into the open
- Agree ways of dealing with the resistance that are acceptable.

Making SPI work: Re-actively

Open
Support

Open
Resistance

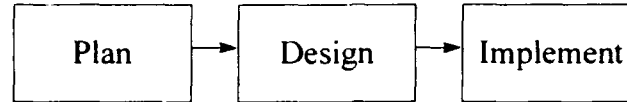
Hidden
Resistance



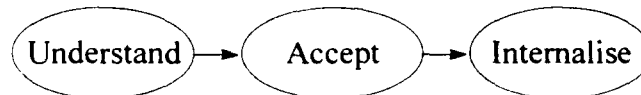
- Discuss the need for the change
- Discuss consequences
- Discuss 'the price'
- Discuss what's going to happen
- Check degree of influence
- Obtain a real 'buy-in'

Making SPI work: Evaluate results

Physical implementation: documents, procedures, tools



● **SPI project** ●



Mental and emotional implementation: hearts and minds

Tools - communication

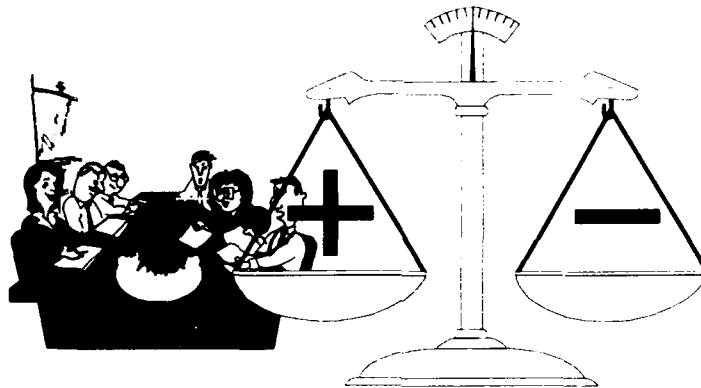
How do you really feel about ...?

Well, to be honest...

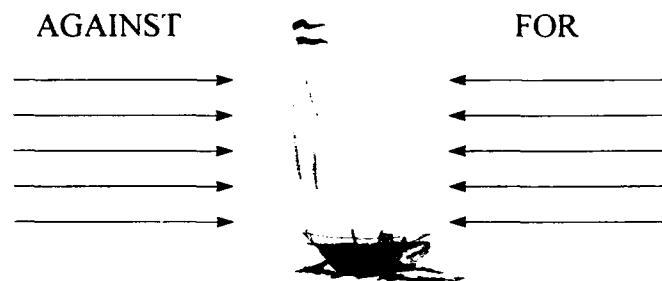


- Presentations
- Publication
- Workshops
- Surveys
- Ask the right questions
- Listen "between the lines"

Balancing the pro's and con's



The winds of change



Three wishes . . .



“If you got three wishes, what would you change around here?”

Why?”

Dealing with the Underworld

THE WHOLE STORY



The whole story (Summary)

- 'The Underworld' is hidden resistance to change
- It has more to do with organisational and individual issues than with 'PROCESS'
- It tells us that our proactive measures are not complete
- We need to draw it out into the open and take reactive measures
- Top Quality communication skills are a "must"
- Communication techniques and tools can help

The whole story

Real and lasting SPI takes place only when (hidden) resistance is confronted and transformed.

This is the heart of SPI



Accelerated SPI

Wilko van Asseldonk

Topics

- Nature of the SPI Project
- Major Changes to the People Involved
- Forms of Resistance Encountered
- Lessons Learned
- Questions

Context

- Local for local, development and production
- Slow SPI programme:
 - assessment 1994
 - STP / LTP with low targets
 - low management awareness
- Cut back production

What Changed?

- Decision to start product for existing market
- New management (BU, development, etc.)
- Local → Global: aggressive marketing program
- Growth of development department (40 → 120)
- Sharply increased complexity and software content
- Significant development lead time reduction
- Quality risk reduction (mass production)
- Accelerated SPI program: CMM Level 3 in 9 months

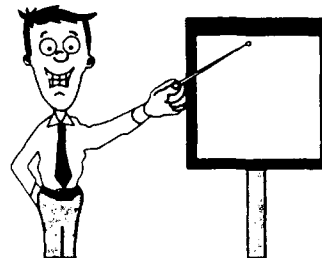
Nature of the SPI Project

- Accelerated SPI: “change or die”
 - situation allows extreme end of spectrum
 - specific preconditions should be met

	Physical	Mental & Emotional
Process Approach	✓	✓
External Support	✓	✓
Management Sponsorship	✓	✓
As “Fast & Hard” as Product Development		✓
Extensive Communication		✓

Approach: Characteristics

- Prescriptive during definition



Approach: Characteristics

- Prescriptive during definition
- Deployment by local champions

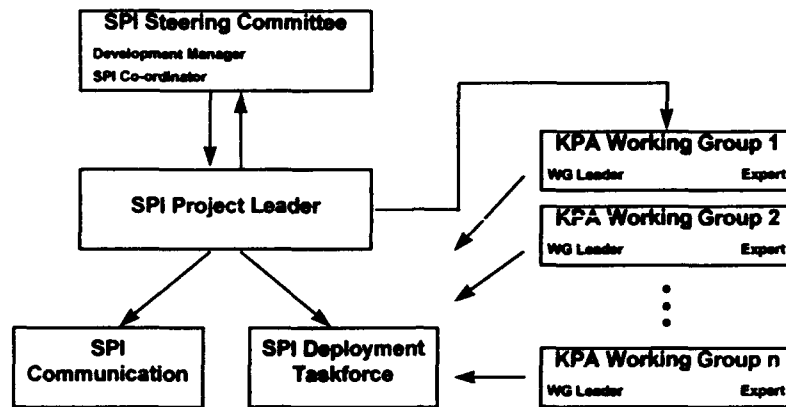


Approach: Characteristics

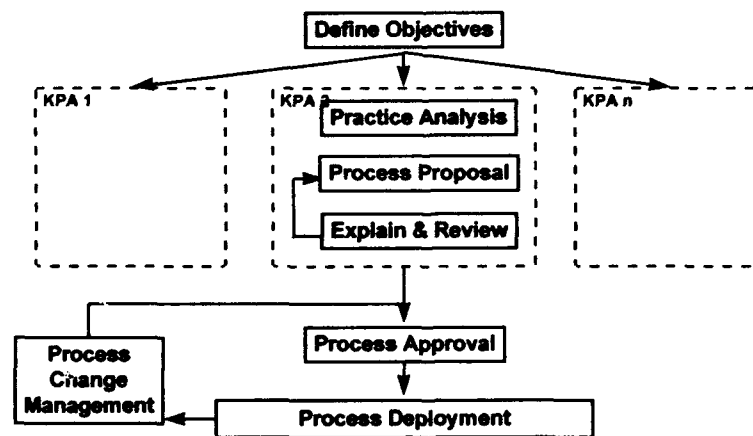
- Prescriptive during definition
- Deployment by local champions
- PR/CR for process to transfer ownership



Approach: Organisation



Approach: Process



External Support

- Process experts (expert status)
- Physical implementation
 - Process definition
- Mental & emotional implementation
 - Start-up and support deployment
 - SPI by walking around

Management Sponsorship

- Budget



Management Sponsorship

- Budget
- Support
 - show awareness and commitment
 - include SPI in appraisal
 - process focus on project reviews

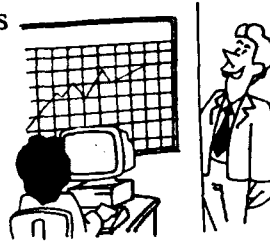


Management Sponsorship

- Budget
- Support
 - show awareness and commitment
 - include SPI in appraisal
 - process focus on project reviews
- Involvement
 - Steering Committee
 - Participate in working groups

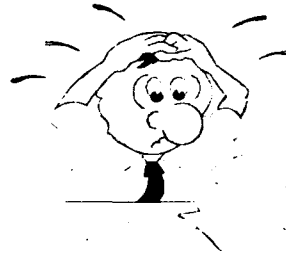
Management Sponsorship

- Budget
 - show awareness and commitment
 - include SPI in appraisal
 - process focus on project reviews
- Support
 - Steering Committee
 - Participate in working groups
- Involvement
 - Project parameter flexibility



“Fast & Hard”

- Development projects “suffered” ⇒
SPI project should suffer



“Fast & Hard”

- Development projects “suffered” ⇒
SPI project should suffer
- Growth of organisation size

“Fast & Hard”

- Development projects “suffered” ⇒
SPI project should suffer
- Growth of organisation size
- Fast & adequate >> slow & perfect



Extensive Communication

- Bulletin Board
- On-line information
- Feedback on change proposals
- SPI newsletter (commercial)

Results

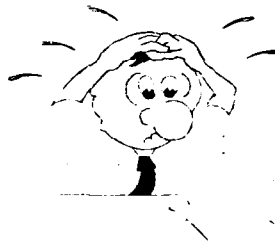
- Planned:
 - Level 3 by July 1996
- Actual (independently verified):
 - Level 2 potentiality by May 1996
 - Level 3 potentiality by October 1996
 - Enormous shift towards process mind set

Resistance Prevented

- Starting situation:
 - Growth of organisation: no history of way of working
 - Chaos, anarchy, state of survival and the promise of SPI
- Pro-active
 - Communicate situation
 - Regular “Process Focus Days” for management
 - Transfer ownership of process

Resistance Encountered

- Overload: “Doing SPI as well is too much”
 - “We will do the work, you just have to follow”



Resistance Encountered

- No Time: "I do not have time for that bureaucracy"
 - Explain how it can help them relieve pressure
 - Explain it as an investment in the future
 - Explain that targets will rise again in the future



Resistance Encountered

- Management: "process is not relevant at our level"
 - discuss relevance during "Process Focus Days"
 - Explain that management develops the process, the engineers will develop the product



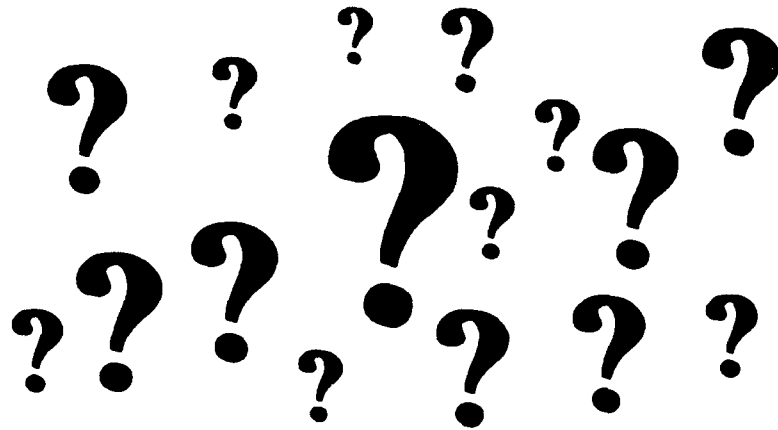
Lessons Learned

- SPI can be accelerated, but:
 - Special approach in a special situation
 - Need crisis (organisation in survival mode)
 - No failure history
 - Requires careful guidance of:
 - Mental & emotional change process
 - Approach: organisation & process
 - Resistance
 - Focus on mental & emotional implementation

Lessons Learned

- Management sponsorship is crucial (again)
 - Initiate SPI project
 - Support & involvement
 - Align change with shop floor level
- Know where you are in the change management spectrum

Questions



DEALING WITH THE UNDERWORLD

Case

Jeroen Brinkman RE

Change maker

The situation 1

strategy

Changing markets, electronic products.

systems

One big dedicated logistical application under construction.

structure

System development & -exploitation, little kingdoms, no clearly defined functions.

shared values

Burn-out syndrome, lots of distrust towards the management, learned passivity

The situation 2

staff

Group of 18 people.

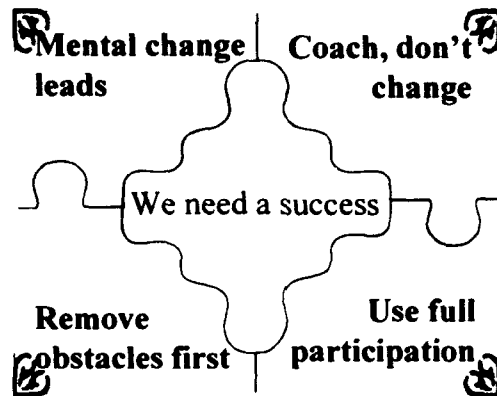
skills

Lack of skills & internal procedures.

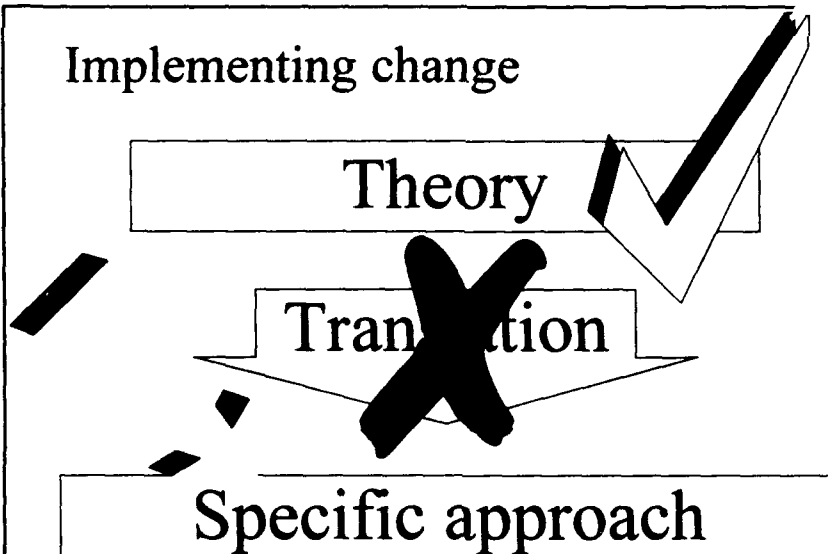
style

Department (barely) managed as a cost-unit instead as a business-investment.

Change strategy



Implementing change



Buy-In

Buy-in

Self-
assessment

Increase
pain

Suggest
solutions

Remove
obstacles

First
actions

Buy-In

1 Kick-off session:

- we have to do a self-assessment!
- when is that a success?
- are there any dos and/or don'ts?
- after the self assessment we'll come back to you!

Buy-in

Self-
assessment

Increase
pain

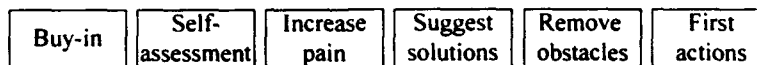
Suggest
solutions

Remove
obstacles

First
actions

Self-assessment

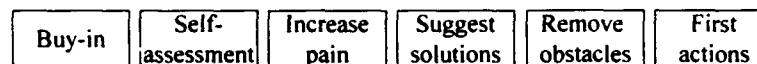
- 2 Self-assessment:
 - quality of work;
 - culture within the department;
 - perception of management-skills;
 - change potential.
- 3 Interviews with each individual:
 - problems + leisure of (future) work.



Increase pain

- 4 Feed-back session self-assessment:
 - current situation not OK;
 - there is a bad and a worst option;
 - it's not the management but the market that drives the change;

➡The results were fully accepted.



Suggest solutions

5 We showed there was a possible solution:

- improve procedures and skills AND;
- restructure the department AND;
- improve the management-control AND;
- manage and coach the change process;

➡ avoid the burning-platform syndrome.

Buy-in	Self-assessment	Increase pain	Suggest solutions	Remove obstacles	First actions
--------	-----------------	---------------	-------------------	------------------	---------------

Remove obstacles

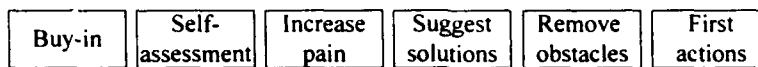
6 Two-day workshop in order to:

- confirm sponsorship by upper management;
- confirm the urge to change by customers;
- define a mission and vision;
- learn to choose reachable goals;
- discuss the way the manager manages;
- and have a good time.

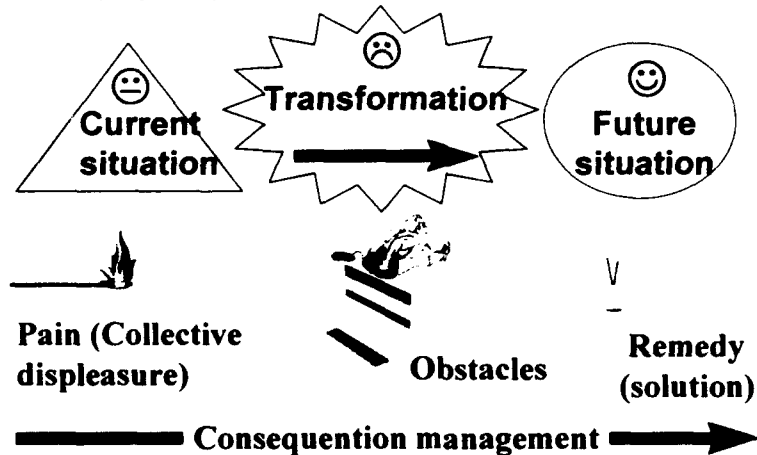
Buy-in	Self-assessment	Increase pain	Suggest solutions	Remove obstacles	First actions
--------	-----------------	---------------	-------------------	------------------	---------------

First actions

- 7 A mix of actions was taken:
- Quick wins: Install a help-desk;
 - Test acceptance: Implement time-sheets;
 - Secure future changes: Restruct department using empowerment & self-steering teams;
- + Two other actions, suggested by the employees.
- ➡ Employees were made responsible.



Reference model



Lessons learned

- Don't use audits, instead use a self assessment.
- Talk (up to a group of 50 people) with each individual.
- Start with mental change, physical change will follow.
- Communication-skills are essential for success.
- Use a participative approach, it needs an initial investment, but will pay back later.
- The eating is the proof of the pudding, so let them eat!
- Use the power of a well prepared workshop.

Communication & resistance

The what-if communication strategy to clarify resistance:

I: I'm afraid of the change

Q: Afraid for what aspect of the change precisely?

A: I'm afraid that my work will become less pleasurable.

Q: Suppose the pleasure of your work won't change, for instance because we add pleasurable tasks to your work, would there be anything else you'll be afraid of?

A: Yes, that I'll lose influence within the company.

Communication & resistance 2

Q: Suppose you won't lose your influence, for instance because we give you an important role in the change-process, would there be anything else you'll be afraid of?

A: No, I don't think so.

Q: So, if the pleasure of your work won't change and you won't lose your influence, then you would not be afraid of the change anymore?

A: No, then the change is OK for me.

Communication & resistance 3

General rules:

- Repeat this process until there is nothing left.
- The last reason mentioned is most likely the most important one.
- Let people be precise.
- Always use examples.
- Conclude with a final check.

Check your own change

Ask the people if they know the answers on the following two questions:

1 What makes that we need a change?

2 What happens if nothing changes?

Questions?



Dealing With the Underworld - Accelerating SPI

1. Introduction

As quality management consultants, we are often asked to advise on and to supervise SPI projects carried out internally within Origin and also by our clients in government, business and industry. Where applicable, we use models such as CMM, TICKIT, and ITIL to pinpoint areas for improvement, draw up improvement plans and to manage improvement projects.

But process models tell us generally what is happening (or what is not happening) in the organisation. Process models do not tell us *why* these things happen and why other (desirable) things don't. Process models don't tell us why - despite 'awareness sessions', agreed improvement plans and adequate support - some improvements have a tendency not to happen as we have planned. To actually *achieve* real and lasting improvements, we need to look *beyond* the clinical world described by the process models and confront the murky, shifting realities of 'The Underworld'.

'The Underworld' is the term that we have given to those aspects of corporate life which are seldom visible using rational logic (or process models) alone. They are the aspects governed by corporate and local cultures, by styles of management, by personal values, beliefs and perceptions, by assumed priorities, by hidden reward systems, and by political tussles. Like the 90% of an iceberg that's underwater, out of sight but steered by the deeper ocean currents, the underworld has a major influence on what we normally see on the surface of the corporate sea, including the success or failure of software process improvement plans. We may be puzzled, for example, why improved planning procedures didn't get the support we'd expected. We may despair at the difficulty that a management team has in keeping project tracking procedures in place, or we may be at a loss to explain why a particular project manager is so disinterested in requirements management.

In this tutorial, we'll be looking more closely at the influence that the 'underworld' plays in carrying through software process improvements - on an individual level, but also at department and company level. We'll present some of the methods and that we have used to make 'the underworld' more visible. And we'll discuss some 'change management' methods that can be used to manage 'the underworld'.

2. Close Encounters

Most if not all SPI projects start enthusiastically - well at least someone is enthusiastic even if it's only the management. Things always seem to go well initially until somewhere along the line we start to hear things like:

"I don't see how this new tracking process is going to interface to our financial systems..."

"I have a friend who works for company XXXXXX which uses this new tool and they say it's difficult to use for small projects"

"Well, this idea for handling requirements sounds great in principle, but in practice..."

In other words, we start to encounter some resistance to the project. The above reactions are typical of discussions on the process itself. We encounter resistance to the SPI project that is caused purely by doubts about the suitability of the proposed improved process. We can call this a *Close Encounter Of The First Kind*.

We can also encounter a form of resistance that - if looked at closely - really has little to do with the actual *process* being proposed but has more to do with potential conflicts between the proposed process and the established working practices, the culture of the company, personal freedom of staff etc. When this resistance is voiced openly and honestly and can be discussed openly, we can call this a *Close Encounter of the Second Kind*. The following reactions typify this kind of encounter:

"Well sure it may work for *other* projects, but we have special circumstances..."

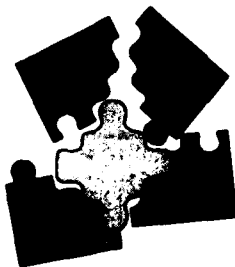
"I'm afraid it just won't work in this department, my senior staff will never go for it..."

We sometimes encounter a third form of resistance that is difficult to put one's finger on. We may just have the vague feeling that someone who *should* be involved in a SPI project is showing remarkably little interest. Even more common is that key people who promised to invest time and energy in a project are nowhere to be seen when the time comes to actually do the work - though they may continue to voice public support for the project. We may also encounter long and/or emotionally charged discussions about issues that seem trivial. This *Close Encounter of the Third Kind* is an encounter with what we have come to call 'The Underworld'. Something - initially hidden from our understanding - is causing unexpected resistance to a SPI project. Moreover, this resistance is not expressed openly but manifests itself in unexpected delays in delivering promised results, cancelled appointments, inertia in starting activities, seemingly endless meetings, misunderstandings, etc.

3 The birds-eye view of SPI

Before we put on our 'change management' boots and wade into 'the underworld' let's zoom out for a moment and take a bird's-eye view of what SPI is and where it fits into the general course of organisational development. Whatever the specific goals of a SPI project, all projects have a common intention of improving the way people work individually and (more often) work together to produce a desired result. Models such as CMM, ITIL and TICKIT are useful as guidelines for analysis of the current process to highlight weak spots. Using such models, we tend to focus on the activities that are, or are not carried out in an IT-organisation. Some models also go a step further and identify whether there is structural evidence of commitment, resourcing and monitoring for processes. Assessments based on such models therefore give us a 'snapshot' image of an organisation's processes. For the purpose of process analysis, this is fine. The snapshot lets us fill in the missing pieces to get a picture of how the improved process map should look. This helps us to create a vision of what we want to achieve (the desired result). What the analysis does not tell us however, is how to get there. But to achieve the end result, we also need an effective *change process* to get us there. We can symbolise this in figure 2 below.

Current Processes



Improved Processes



Change Process

Figure 1: The Change Process

The next logical question is: what do we need to *change* in the organisation (other than the process itself) in order to achieve the desired result?

What do we see from a bird's-eye view of an organisation down below? Lots of little people walking around, talking to each other in meetings and on the phone. Working individually and in groups to create products, develop ideas, build relationships, achieve individual and common goals. This is not *all* 'process'. It is in fact a small community with all the issues of the wider community, such as political, economical and social issues. People who work in this small community each have their own individual concerns: job security, growth prospects, relationships, setting priorities, etc. When we choose to focus on 'process', we tend for the sake of simplicity to ignore these issues in the work community.

For the purposes of planning and managing major SPI projects however, it is often useful to take this bird's eye view and include the wider issues. Just think about this for a moment. Can 'process' exist completely independently of these other organisational issues? We may decide not to focus attention on them - but of course they're there just the same. One of the main themes of this tutorial is that the organisation's 'process' is in fact strongly interrelated with other aspects of organisational development - for example with the quality of staff. Another obvious relationship is the one between 'process' and 'culture'. 'Pretty obvious', you may think, 'and so what?' Well, so if we start trying to change 'process', we are sooner or later going to find that it's being held firmly in place by 'quality of staff' or 'culture' or by something else. A useful model that can help us to take this bird's eye view and understand where 'process' (and therefore SPI) fits into the wider scope of things is the 7S-model developed by McKinsy, as shown below.

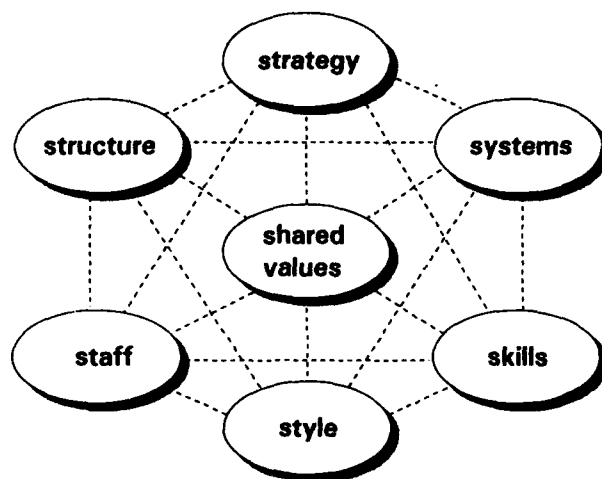


Figure 2: The 7S-model

The 7-S model recognises seven main aspects of an organisation that are interrelated: strategy, structure, systems, staff, skills, shared values and management style. These characteristics are described below.

Strategy

What are the organisation's key goals and targets and how does it go about realising these? How does an IT organisation need to develop in order to continue to meet the future IT needs of the business? What is its policy on IT systems? Outsourcing? Strategic Alliances? Which opportunities and threats are foreseen? What are the relevant strengths and weaknesses?

Structure

What organisational structures enable and prevent things getting done easily? What is the process structure? What are the characteristics of the formal and informal organisation structures? Is decision-making centralised or decentralised? How is power distributed in the organisation and how effectively is it used? What communication structures are in place and how effective are these

Systems

'Systems' are basically the rules, regulations and procedures in use (including automated systems) that support the *structure* of the organisation. Examples of such systems are the staff assessment system, the reward system, the project cost monitoring system, the project tracking and reporting system, the software development systems. During SPI projects we often look at the *systems* in use (since these are highly visible) to tell us about the structures.

Shared Values

Shared Values - another term for 'organisational culture' - are the unwritten rules of an organisation. The implicit - but dominant - assumptions, principles and values that determine decisions and behaviour. In other words - 'the way we do things around here'. Shared values can include, for example, the belief that decentralised entrepreneurship always strengthens the organisation as a whole in the long term - even if this means internal competition for market segments.

Staff

This item refers to the quality of human resources, primarily in terms of motivation and training but in also in terms of whether staffing levels are adequate. It also includes the principles by which staff are selected, assessed and rewarded.

Skills

'Skills' refers to the 'core competences' of the organisation. What are its strengths and weaknesses? Where do people tend to be good in? Which types of skills are planned for and monitored? Which are not? For example, do project leaders have a good grounding in project management skills or do they tend to be good technical people who lead by example?

Style

'Style' refers to the dominant management style, for example Authoritarian? Participative? Critical? Supportive? Product-oriented, People-oriented, etc. Management style usually reflects the underlying culture and makes this visible.

As figure 2 shows, all 7 Ss are in principle interdependent. For example, the systems support the structures that have evolved together with the management style and the shared values. The interrelations between these will tend to ensure that they remain consistent. Natural forces will come into play to resist attempts to change 'process' in ways that are inconsistent with the other

- influencing the *direction* of the future development of the IT organisation, taking into account the legacies of the past
- influencing the *process* of organisational development

5. Resistance to change

We started out by wondering what was going on in The Underworld. Then we zoomed out to look at the context of SPI in the wider issues of organisational development using the '7-S model'. Then we went on to consider the effect of local history on the current organisational and on the potential for further development. Now it's time to get back to 'The Underworld' in SPI.

Most, if not all SPI projects start with the focus on "process". So we start out for example, wanting to improve the 'software design' process, or the 'risk management' process or the 'verification and validation' processes. The 'configuration management' process seems to be a popular area for improvement at other conferences this year.

How does a SPI project actually get started? On the advice of experts? By taking a vote amongst project leaders? Because management think that this is the most beneficial improvement step that they can take? Any of these may be possible. Our experience is that most large scale SPI projects are started by 'IT management' with little real understanding or support at the lowest levels - with the exception of a few internal 'experts'. So what happens when we try to develop and implement process improvements? We invariably get a varied response from different people. We sometimes get an immediate positive response or an immediate negative response. Often we can get little real response at all. This usually means that the implications of the project are not fully understood yet. It's possible of course that the implications are well understood by staff but that it's not seen as 'such a big deal'. This may be the perfect improvement project - all the 'gain' for no 'pain'. But it's worth asking yourself whether the improvements that *you* think are going to happen are the same ones that the 'no big deal' people think are going to happen!

These are the immediate responses, on the basis of which we may decide to take some actions. But what if we asked the same questions a few weeks later? A few months later? A year later? Chances are that the responses would be different. Research has shown that people's responses to a change tend to follow one of two patterns: the 'positive response pattern' or the negative response pattern'. We're not going to go into too much detail on these patterns in the tutorial (because we don't tend to use them much in practice anyway). But the main thing to note is that responses tend to change and to some extent are predictable. This means that we can't expect to change a negative response into a positive response overnight - certainly on the basis of logical arguments alone. It also means that we shouldn't treat an initial positive response as the 'final' response. Strongly positive responses are likely to become less positive as time goes by and we can positively influence negative responses.

An inherent part of both response patterns is "resistance" to the change. For widely different reasons (that we'll examine later) all people have an in-built tendency to resist attempts to change them. Perhaps the most innocent and widely seen form of resistance is simply 'inertia'. People tend to continue to think and act as they have been used to doing in the past. Even if people genuinely support a new way of working, when the time comes when they individually have to choose between the old way and the new way there is always a force at work that pulls them back towards the old way. This force is usually termed 'habit'. It's the same force that causes people who've recently moved house to leave work and absent-mindedly drive back to their old neighbourhood. It's the same force that causes people who work in a company that has changed names answer the phone with the old company name - perhaps months after the

change. Habits become ingrained and take time (and energy) to wear off. So this kind of resistance is largely unconscious and is independent of whether someone is *for* or *against* the proposed change. This kind of resistance is always present.

But there is of course another kind of resistance that is related to the perceived *nature* of the change. People - often for good reason - can be sceptical as to the benefits of the proposed change. They may see disadvantages for the organisation or for themselves. They may just generally feel uneasy about the proposed change. A common reaction is to take a passive but somewhat defensive position: "We'll wait and see how things turn out". This person is in fact saying: "I'm not making any firm commitment now. If I see that it's beneficial and safe to support the change, then I'll support it. But I'm reserving the right to resist the change if it turns out that the disadvantages outweigh the advantages." Another way of putting it is that the person is 'sitting on the fence'.

For example, let's assume that an experienced system developer has serious misgivings about proposed new configuration management procedures. She foresees a high overhead in checking work in and out, expects delays in solving urgent bugs during integration and has a feeling of having to submit to 'the system' (which will be administrated by someone junior to her).

She may show open resistance in discussions on the subject, voicing arguments publicly and privately against the procedures. We then speak of 'overt resistance'. It's out in the open, is known, and there's no doubt or confusion about where this person stands on the issue. A second form of resistance is 'covert resistance' which occurs when the resistance is not voiced openly - or at least not publicly - but nevertheless influences the response of an individual towards the proposed change. Covert resistance is therefore a much more difficult form of resistance to deal with. And this brings us to the central theme of this tutorial:

COVERT RESISTANCE IS THE MEANS BY WHICH 'THE UNDERWORLD' MAKES ITSELF KNOWN.

Think about the different ways in which we encounter 'covert resistance' every day:

- people don't seem to have the time they promised to commit to the project
- halfway through the project, people start thinking about "an even better way" of doing things
- when it comes down to details, managers just can't seem to agree on new procedures or how to use new tools
- people spend more time thinking how to work *around* the new procedures than working with them
- under time pressure, people resort back to the old ways of doing things
- people somehow manage to create the superficial impression that they've adopted the new working procedures while in fact nothing much has changed.

6. Interpreting 'Resistance Messages'

Let's now look more closely at the underlying causes of both overt and covert forms of resistance. To start with we must recognise that the people may or may not be conscious of their 'resistance' and the underlying causes of this. *Unconscious* resistance to change can be a purely emotional or intuitive reaction which may even run counter to the person's *conscious* response to a SPI proposal, for example in the case of a 'supporter' who unfortunately never seems to be able to find time to help out on the project. Although unconscious resistance often shows itself as covert resistance, it can also be shown overtly - for example in emotional outbursts at meetings that are largely unsupported by factual arguments.

The person may also be consciously aware of the causes for his or her resistance. Ideally, this type of resistance is shown as 'overt' resistance. The person is able and willing to explain and discuss the reasons for the resistance. Of course there is also the possibility - at least in theory - that the person - although consciously aware of the causes of his/her resistance - chooses to opt for covert resistance. He/she may not feel confident enough to bring the resistance into the open (if this involves a personal risk) - or may simply calculate that covert resistance will in the end be more effective. The last type of resistance - covert resistance - of which it seems that the person in question is perfectly aware but is unwilling to make open, makes for a difficult atmosphere in which to introduce change. This type of resistance appears to others as 'underhand' or 'manipulative' and quickly leads to feelings of mistrust and to allegations of 'political manoeuvring'.

Some of the main causes of resistance by individuals are listed below:

*Lack of political
Support*

The person has a relatively low level of political support for the goals of the project, in relation to other (potential) projects. This is the most straightforward cause of resistance to change: in the view of the person concerned, the benefits do not outweigh the costs (including perhaps the 'cost' of averting attention to a more pressing improvement).

*Perceived
insecurity,
instability, loss of
control*

The person has negative feelings associated with his/her perception of the nature of the change. He/she may feel potentially threatened by the expected *results* of the change or by the expected *change process*. Because people have a basic need for security, stability, and control over their environment (The 'Maslow' hierarchy), they naturally resist - consciously or unconsciously - threats to this security, stability and control. For example, a manager who rose to his position because of his ability to get things done in the 'informal organisation' - may resist moves to formalise processes and procedures and to do things 'by the book'. People also naturally resist major changes in situations where the consequences for them personally are not foreseen.

*Avoidance of
discomfort*

Another natural and basic human reaction is to avoid the discomfort caused with having to unlearn old habits and learn new habits. It requires considerable effort to unlearn old habits, learn to work with new procedures and tools, develop new working relationships with people etc. In periods of change, people can no longer use the 'automatic pilot' for doing routine things. They have to concentrate on doing things, they make mistakes and must correct them. People vary as to the level of discomfort that they can accept while continuing to function properly. And they vary in terms of the length of time for which they can put up with the discomfort before wanting to feel the benefits or at least feel comfortable again.

*Perceived lack of
personal resources*

People may feel a lack of personal resources to cope with the change process. It's OK to claim that you've no time to deal with change 'under so much pressure'. It's less OK to publicly admit that you don't have the intellectual wherewithal to fully understand the implications of the change. It's even less OK to admit that you don't feel competent to run projects under the new management procedures. Attempts to introduce new process management procedures can fail if managers are not comfortable with the underlying concepts of process management.

Fear of failure

Everybody wants to support a winning team. People may therefore hesitate to give public support to an improvement project if they expect that the project will fail. There may be a history of such failures in the organisation. There may be a perceived lack of sponsorship/commitment for this specific project, in which case people may avoid taking personal risks.

Resistance by groups of people has its roots in the factors listed above for individuals, but it is obviously strengthened due to its being 'bundled'. For example, when a whole project team decides to boycott departmental testing procedures, with support from the project leader, the individual resistance patterns are meshed, and therefore stronger than an equivalent number of purely individual resistance patterns.

On a wider scale, the 7-S elements of the organisation tend to work together to maintaining the current equilibrium. For example, attempts to change only the *systems* of the organisation, will run into resistance at the point at which the desired change runs contrary to the *structure* and /or *shared values*. Consider for example, a situation where a senior manager gives (without realising it) a conflicting message to the employees during an important presentation. On the one hand he may give explicit support to a SPI project that will result in a standard, defined, software development process to be implemented for all projects. If later in the presentation he then goes on to praise one of the project teams that were able to deliver a software package earlier than planned at the request of the customer, he may unknowingly support the tradition of throwing caution - and standard departmental procedures- to the wind to get 'something' shipped when required. Most employees will realise that the real rewards are given for fast delivery - not for adhering to standard procedures. This 'shared value' - that is inconsistent with the goals of the SPI project - is thereby unwittingly reinforced.

The impact of Shared Values (corporate, departmental or project culture) cannot be overestimated, because most people in the organisation become unaware of them after they've been there some time. When SPI project goals are inconsistent with the culture, the culture always wins and the project always (eventually) fails.

To summarise this central part of the tutorial, we can say that "resistance" is:

- usually a perfectly normal, healthy aversion to losing control, stability or security
- always dependent on one's frame of reference
- determined by one's perceived ability and willingness to adapt to a change, where 'willingness' depends on the perceived personal benefits and costs

7. Making SPI work

OK, this is the bit we've been working up to: how do we actually deal with 'The Underworld' in real life? Given that we now understand why resistance to change occurs, which patterns it follows and how it manifests itself, what can we do about it? The short answer is probably 'very little'. Before you despair, let us quickly add that although we can't do much about it, we can deal with it more productively when we understand it and accept it rather than simply ignoring it or trying to fight it head on. The following measures can help in minimising resistance.

Explain the need for change

We can ensure that the *need for change* is fully understood by those involved. In our experience this is the simplest action that managers can take and is also the action that is most often forgotten or badly handled. We need to communicate the need for change clearly at levels of the organisation. The bigger the change, the more discomfort or uncertainty

it will involve and the greater the 'pain' has to be. The term 'pain' is used here to mean that:

- there is a current or future problem threatening the business, or
- there is current or future business opportunity that we must take advantage of.

We then need to show clearly how the proposed change will 'ease the pain'.

Understand the consequences of the change

We can ensure that the true consequences of the change for individual staff are well understood by the sponsors of the SPI project and (successively) by each layer of management that must give support to the change. These implications may for example include changes to the organisational culture, to established working patterns, or to current social relationships.

Be prepared to pay the price of change

We can ensure that the true costs of the changes to individuals and to the organisation as a whole are made known to those responsible and that those responsible are prepared to pay the price of managing and implementing the change. This commitment to paying the price of implementing the change must be sustained and made visible within the organisation. Commitment is visible when the SPI goals are pursued consistently and creatively and when short-term benefits that are inconsistent with the SPI goals are rejected. Staff must be left in no doubt of the support at management levels for the change. Once a critical mass has been established, many of the 'fence sitters' are likely to jump off.

Tell people in detail what is going to happen and when - then wait for this to sink in

We can minimise the ambiguity and uncertainty for the people involved by ensuring that the micro-implications for the individual are well understood. We must also allow the time and where possible the direct involvement of people - to allow emotional commitment to take place as well as intellectual commitment. The implications of the change take time to really 'sink in'.

Avoid 'soaked sponges'

We can also ensure that staff don't have to assimilate too many major changes in too short a period of time. Admittedly, what changes are 'major' and what periods of time are 'too short' are very subjective. A general attitude of passive acceptance coupled with a feeling of powerlessness is a possible indicator that staff have been subjected to too many changes too quickly and simply do not have the emotional energy to actively respond to this one.

Reward supportive behaviour

To reinforce new habits, we can ensure that 'positive behaviour' is recognised and is consistently rewarded. Also negative behaviour should not go unnoticed. Obviously, rewards should be withheld from those who fail to make the adjustments.

Invest sufficient resources to implement the change

We can ensure that sufficient resources are invested for the change to be implemented successfully. Under 'resources' we may think of tooling, training, time for assimilation, reduction of normal working pressure, etc. In particular we must ensure that each person required to support the change feels - as far as possible - confident that he/she is in a position to do so. In establishing resource budgets to implement the change, it is

often helpful to bear in mind the potential costs of failure to implement the change.

Go out looking for resistance and welcome it when you meet it

We can use some tools and techniques to help track down areas of resistance. When we know what we're up against, we can develop measures to reduce the resistance. We can for example hold surveys to measure the impact of a proposed change, to measure the support, to highlight areas of resistance. We can organise discussions to try to bring resistance out into the open.

Despite the measures described above, we will continue to encounter 'The Underworld'. However, we should recognise it for what it is - a sign that the person is unhappy with some aspect of the proposed change - for one or more of the good reasons discussed earlier. Rather than ignoring or fighting the resistance, we should first do all we can to encourage the resistance into the open. When we know the specific reasons for the resistance, we become more able to deal with it constructively. If this fails, then we can creatively apply the measures above to deal with the possible underlying causes of the resistance. This may reduce the resistance - or in applying the measures, we may succeed in drawing the cause of the resistance out into the open.

So to summarise, how can we deal with the underworld?

- Take preventative measures to minimise the resistance
- Go looking for it and recognise it when you meet it
- Respect it - it's trying to tell you something
- Confront it and try to bring it out into the open
- Don't suppress it (it will just come back at you later)
- Come to terms with it and - ideally - transform it (acceptance)

We should note that 'transforming resistance' sometimes means making concessions to our original plans. It is in our experience better to have 80% of the originally planned 'solution' securely implemented in the organisation than to hold out in the hope that resistance to the 100% will wear off.

In our experience, real and significant SPI takes place (only) when the resistance is confronted and transformed. This transformation of resistance lies in our view at the *heart* of SPI.

8. Literature

Although the themes of this tutorial arose from our practical experience, we have drawn from existing literature to explain some of the concepts. In particular, we recognise the following important sources for this tutorial and recommend these for further reading:

1. Managing at the speed of change: how resilient managers succeed and prosper where others fail
Daryl Conner
ISBN 0-679-40684-0
2. Kwaliteit & Service: handboek voor organisatievernieuwing en leiderschap
P.J.M. van Esch
ISBN 90-204-2012-7



Carnegie Mellon University
Software Engineering Institute

Software Process Improvement: Business Impacts and Value

European SEPG
June 16, 1997
Dave Zubrow



Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pa 15213

SM CMM, Capability Maturity Model, and IDEAL are service marks of Carnegie Mellon University
Sponsored by the U. S. Department of Defense

©1997 by Carnegie Mellon University

1



Carnegie Mellon University
Software Engineering Institute

Objectives

**Present data on the impacts of software
process improvement and technology
investments**

**Discuss issues and considerations for the
measurement of improvement**

©1997 by Carnegie Mellon University

2



Carnegie Mellon University
Software Engineering Institute

Outline

Benefits of Software Process Improvement

Views on CMM Based SPI

Project Management: The First Challenge

Additional Consideration for Improvement

Measuring Impacts

Exercise: Making the Case

Closing

©1997 by Carnegie Mellon University

3



Carnegie Mellon University
Software Engineering Institute

Impacts of Software Process Improvement

Studies

- SEI study of general results
- LOGOS International study of general results
- AFIT study on cost and schedule performance
- SEI survey of predictability, performance, and customer satisfaction
- Case studies
- Estimates of impacts

©1997 by Carnegie Mellon University

4



Carnegie Mellon University
Software Engineering Institute

SEI Study of Benefits

Participating Organizations:

Bull HN

**GTE Government
Systems**

Hewlett Packard

Hughes Aircraft Co.

Loral Federal Systems

Lockheed Sanders

Motorola

Northrop

Schlumberger

**Siemens Stromberg-
Carlson**

Texas Instruments

U. S. Air Force

**Tinker AFB Air
Logistics Center**

**U. S. Navy Fleet
Combat Direction
Systems Support
Activity**

See Herbsleb, J., et al "Benefits of CMM-based SPI:
Initial Results." SEI 94-TR-13, 1994.
©1997 by Carnegie Mellon University

5



Carnegie Mellon University
Software Engineering Institute

Data Profile

Data met criteria for inclusion in this study

Data reported over multiple years

**Not all organizations provided all types of
results**

Caveats

- **unmeasured or unreported results**
- **other activities may have contributed to
results**
- **do not know if results are typical**

©1997 by Carnegie Mellon University

6



Carnegie Mellon University
Software Engineering Institute

Types of Results

Cost
Productivity
Schedule
Defects
Business value

©1997 by Carnegie Mellon University

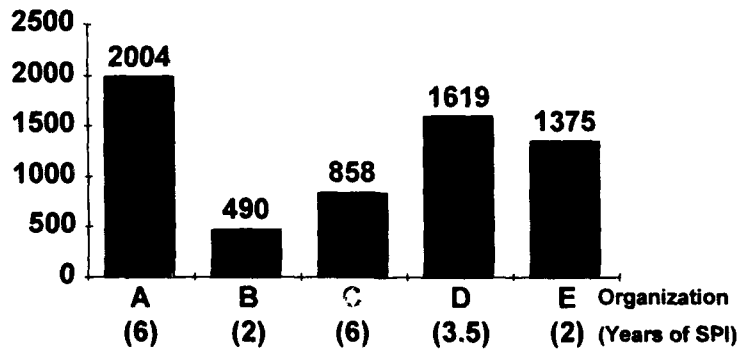
7



Carnegie Mellon University
Software Engineering Institute

Yearly Expenditure on SPI

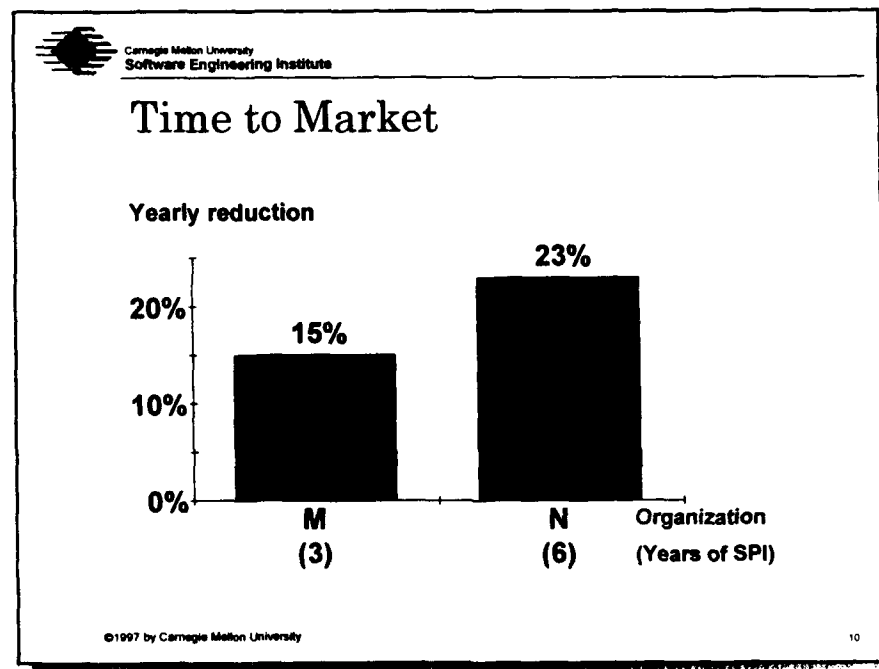
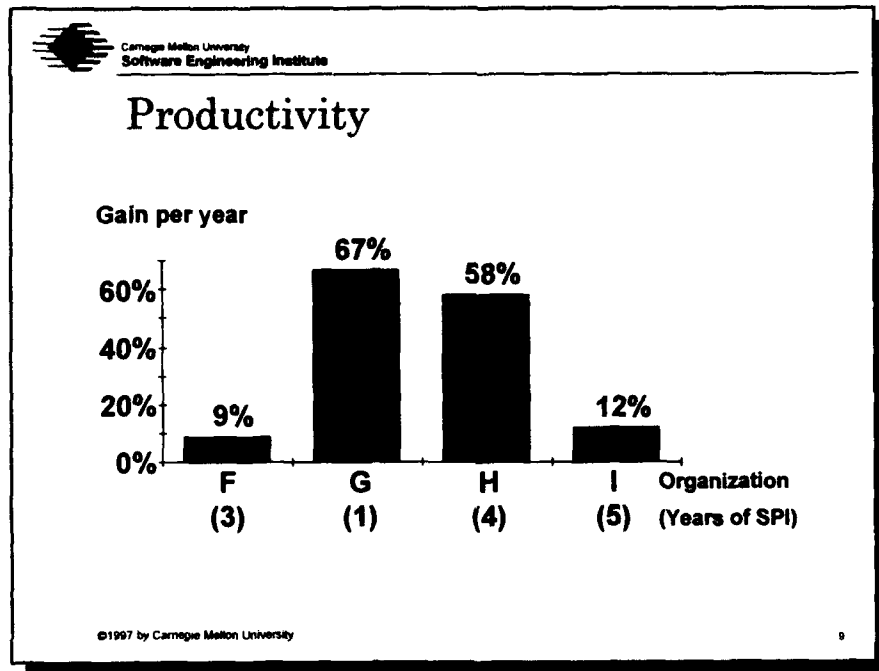
Dollars per
software engineer per year



©1997 by Carnegie Mellon University

8

Software Impacts and Value

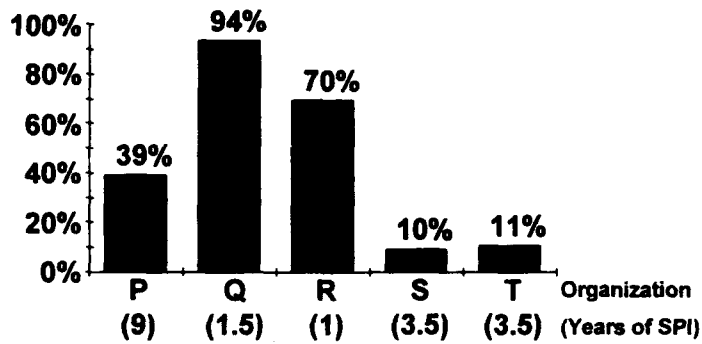




Carnegie Mellon University
Software Engineering Institute

Defects

Reduction per year



©1997 by Carnegie Mellon University

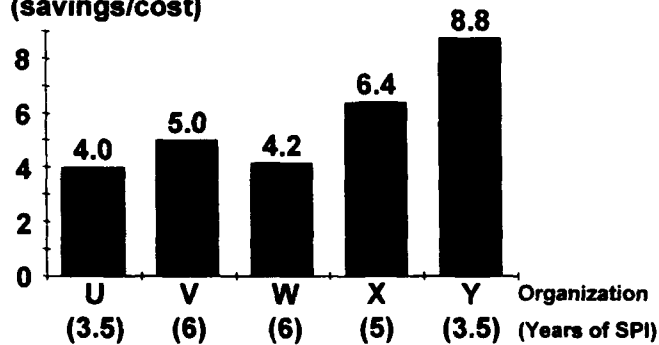
11



Carnegie Mellon University
Software Engineering Institute

Business Value

Business value ratio
(savings/cost)



©1997 by Carnegie Mellon University

12



Carnegie Mellon University
Software Engineering Institute

Summary of SEI-94-TR13

cost of SPI per software engineer/yr.	\$490-2004	\$1375	5
gain in productivity per year	9%-67%	35%	4
reduction in time to market per year	15%-23%	---	2
reduction in post-release defects per year	10%-94%	39%	5
business value ratio (benefit/cost)	4.0-8.8:1	5.0:1	5

©1997 by Carnegie Mellon University

13



Carnegie Mellon University
Software Engineering Institute

LOGOS International Study*

Background

- Study commissioned by the US Air Force
- Quantifiable results on benefits and ROI

Method

- Industry interviews with representatives from 22 industry and government organizations
- 70 questionnaires mailed with a response rate of 29%
- 33 organizations covered in all

* See Brodman and Johnson "Return on investment (ROI) from software process improvement as measured by US industry," *Software Process, Pilot Issue*, pg. 35-47, 1995

©1997 by Carnegie Mellon University

14



LOGOS International Study

Maturity Level of organizations in the study

ML 1 = 4 (24%)
ML 2 = 7 (41%)
ML 3 = 4 (24%)
ML 4 = 1 (6%)
ML 5 = 1 (6%)

Note: some organizations did not report their maturity level.



Results of Process Improvement Initiatives - Benefits

Metric Category	Measurement	Benefits Realized by Various Software Organizations*
Productivity	Increase in Productivity	10-20%, 90-100%, 50%, 15-20%, 5%, 130%, 12%, 2.5-6.3%, 35%
Quality	Reduction in Defects	10%, 80%, 50-70%, 50%
	Reduction in Error Rate	45%
	Product Error Rate	From 2.0 down to 0.11 per thousand source lines of code From .72 down to 0.13 per thousand non-commented source statements
Cost	Project Dollars Saved to Dollars Invested	1.5 to 1.2 to 1.4 to 1.6 to 1.7 to 1.1 to 1.26 to 1.5 to 1
	Project Dollars Saved	\$2 million to 3.4 million
	Code Problems During Integration	20% of original value
	Decrease in Cost of Retesting	50%
	Cost Savings of Metrics Program	50-300% 40-290%
Schedule	Within Estimate	5% of estimate
	On-time Deliverables	From 51% up to 94% on time
	Project Completion	From 50% down to 1% late
	Savings in Schedule	10%, 20%
Effort	Reduction in Rework	5 to 10%
		From 40% down to 25% of effort
		From 41% down to 11% of project cost
	Savings in Test Time	10 tests hours per one analysis hour

* Benefits are shown as a range of results within a single organization, results from different organizations are separated by commas. All organizations are not represented. Source: Brodman and Johnson, CrossTalk, April 1996



Carnegie Mellon University
Software Engineering Institute

LOGOS International Summary

Wide variation in results across the organizations

Identified different definitions of ROI

- **government = savings**
- **contractors = productivity**
- **commercial = decreased time to market, improved quality**

Positive benefits for CMM-based SPI do exist

©1997 by Carnegie Mellon University

17



Carnegie Mellon University
Software Engineering Institute

Critical Assertions of the CMM

Predictability and performance by maturity level

Key process areas

Factors that influence success

Moving up the maturity scale
Usability and implementation

©1997 by Carnegie Mellon University

18



Carnegie Mellon University
Software Engineering Institute

Study on Cost and Schedule Performance*

Test for correlation between CPI and SPI and software process maturity level

- $CPI = BCWP / ACWP$
- $SPI = BCWP / BCWS$

Data

- 52 observations of cost, schedule, and maturity
- 31 Projects
- 11 DoD contractors
- Used both SPA and SCE data

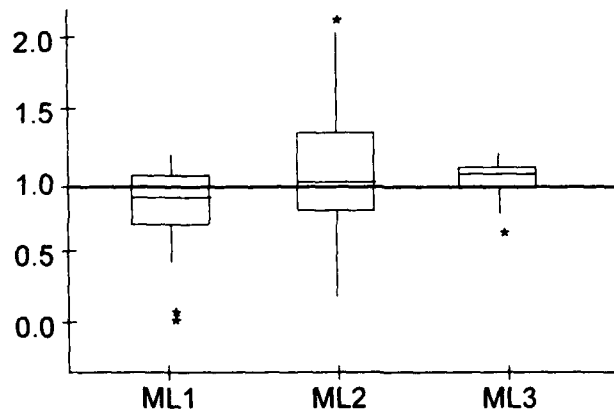
*See Lawlis, Flowe, & Thordahl. "A correlational study of the CMM and software development performance." Crosstalk, 8, September 1995, pp 21-25
©1997 by Carnegie Mellon University

19



Carnegie Mellon University
Software Engineering Institute

Cost Performance by Maturity Level



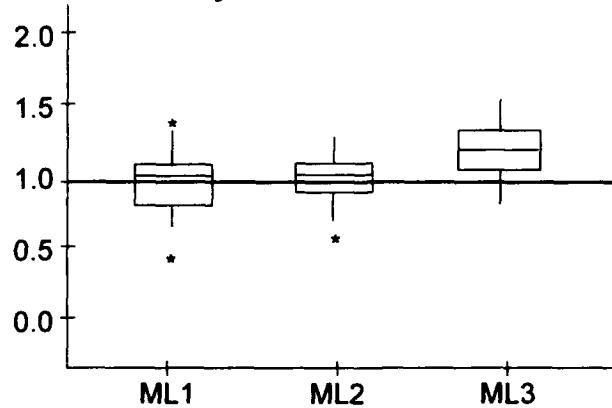
*See Lawlis, Flowe, & Thordahl. "A correlational study of the CMM and software development performance." Crosstalk, 8, September 1995, pp 21-25
©1997 by Carnegie Mellon University

20



Carnegie Mellon University
Software Engineering Institute

Schedule Performance by Maturity Level



*See Lawks, Flowe, & Thordahl "A correlational study of the CMM and software development performance." Crosstalk, 8, September 1995, pp 21-25
©1997 by Carnegie Mellon University

21



Carnegie Mellon University
Software Engineering Institute

CPI and SPI Study Summary

Correlations in the expected direction

- least mature organizations more likely to have difficulty adhering to cost and schedule baseline
- more mature organizations more likely to have on-baseline cost and schedule performance

Reduced variation in CPI with increasing maturity

Reduced variation in SPI in moving from ML1 to ML2

©1997 by Carnegie Mellon University

22



Carnegie Mellon University
Software Engineering Institute

Post-Appraisal Survey* Goals

Address previous shortcomings

- representative data
- broader view of each organization

Information about

- value of appraisals
- **predictability and performance**
- factors that distinguish success and failure

See Goldenson and Herbsleb, "After the appraisal: A systematic survey of process improvement, its benefits, and factors that influence success." SEI 95-TR-009.

©1997 by Carnegie Mellon University

23



Carnegie Mellon University
Software Engineering Institute

The Post-Appraisal Survey

Appraisals in SEI database from 1992 and 1993

Responses from 138 people, 56 appraisals

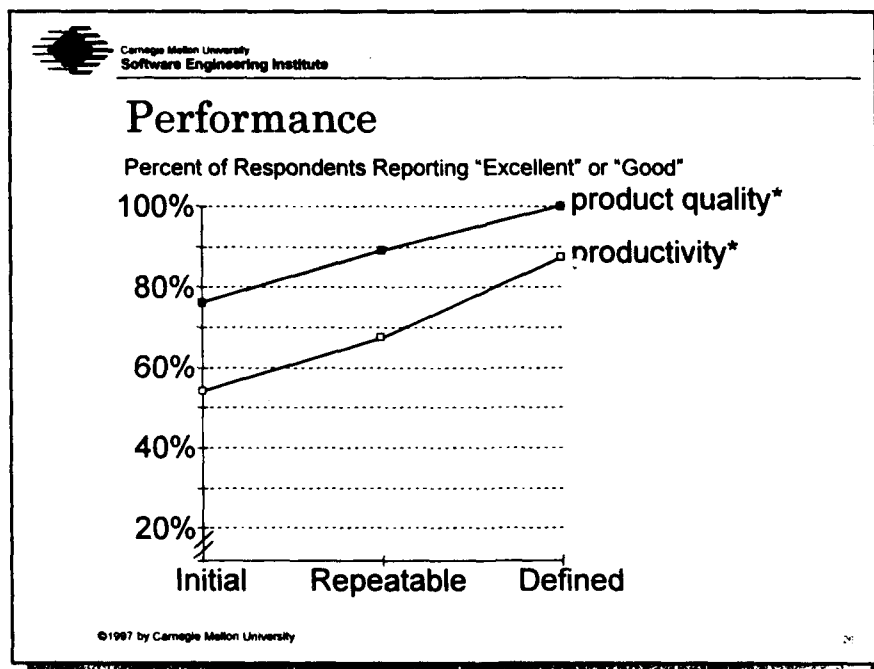
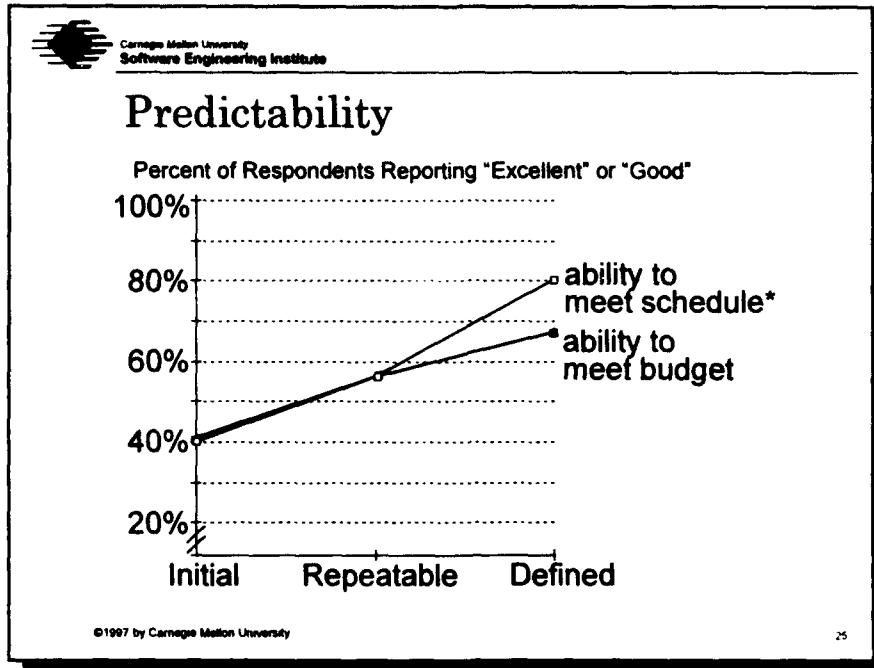
Have 83% return rate (138 of 167)

Questionnaires received from...

- SEPG member (44)
- senior technical person (47)
- project manager (47)

©1997 by Carnegie Mellon University

24

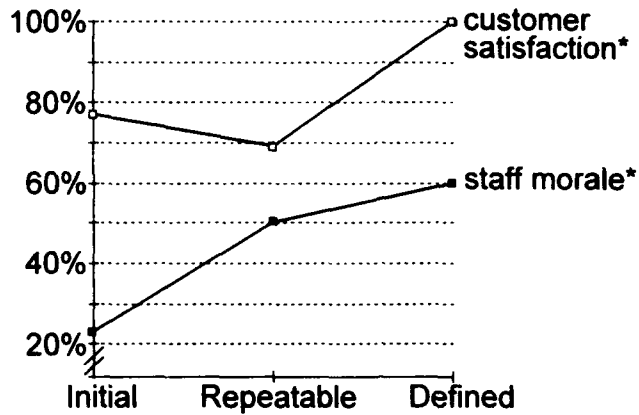




Carnegie Mellon University
Software Engineering Institute

Intangibles

Percent of Respondents Reporting "Excellent" or "Good"



©1997 by Carnegie Mellon University

27



Carnegie Mellon University
Software Engineering Institute

Some Recent Case Studies

Boeing Space Transportation Systems

**Hewlett Packard Software Engineering Systems
Division**

SAIC Health Technology Group

Bellcore

©1997 by Carnegie Mellon University

28



Carnegie Mellon University
Software Engineering Institute

Boeing Space Transportation Systems

Reduced Defects and Rework

- Later phase defects reduced from 31% to 4%

Improved Productivity by 62%

Improved Cycle Time by 36%

Improved Product Performance

- more accurate satellite delivery

Improved Customer Satisfaction by more than 10%

Source: John D. Vu, SEPG 1997 Presentation

©1997 by Carnegie Mellon University

29



Carnegie Mellon University
Software Engineering Institute

Hewlett Packard

Challenge: ML 3 in 36 months

Improvement run as a project

Deployment/adoption progress monitored and measured

Benefits in 12 months

- reduced cycle time from 21 to 14 months
- open major defects reduced from 4.6 to 1.6
- fewer missed deadlines
- ROI of 9:1 (savings of \$2M/yr)

Source: Lowe and Cox, "Implementing the capability maturity model for software development," Hewlett Packard Journal, August 1996.

©1997 by Carnegie Mellon University

30



Carnegie Mellon University
Software Engineering Institute

SAIC Health Technology Group

Time series over multiple releases

Production: rate of code development and modification up 30%

Developer productivity: 12% per year

1-Year Error Rate reduced by 71%

Approximately 50% improvement in customer satisfaction

Source: Lane and Zubrow, "Integrating measurement with improvement: an action-oriented approach," Proceedings of the International Conference on Software Engineering, 1997

©1997 by Carnegie Mellon University

31



Carnegie Mellon University
Software Engineering Institute

Bellcore

Dramatic reduction in defects

- 10x lower than industry average
- exceeds best in class

Customer satisfaction improved from 60% in 1992 to 91% in 1996

9 hr cut over

- Add 888 to 800 system
- No reported defects

Source: Bird-of-a-feather session, SEPG 97 and Bellcore press release, Feb 5, 1997.

©1997 by Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Estimates of Improvement

	ML1	ML2	ML3	ML4	ML5
Defects Delivered ¹ (per function point)	.75	.44	.27	.14	.05
Cost ² in \$m (hypothetical 200 KLOC project)	5.4	1.3	.7	.4	.15

1 Jones, C. "Software benchmarking," IEEE Computer, pg 102-103, October 1995.

2 Rifkin, S. "The Business Case for Software Process Improvement," Master Systems, 1993

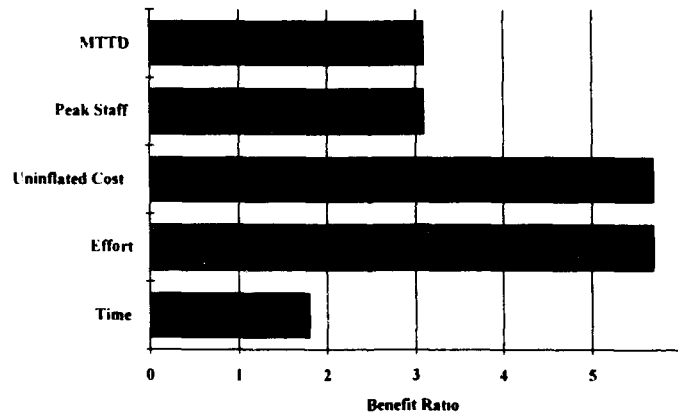
©1997 by Carnegie Mellon University

33



Carnegie Mellon University
Software Engineering Institute

ML1 vs ML 3 Benefit Ratio*



* Putnam, L. "The economic value of moving up the SEI Scale," Quantitative Software Management, Inc, 1993.

©1997 by Carnegie Mellon University

34



Carnegie Mellon University
Software Engineering Institute

Summary of CMM-based SPI Impacts and Studies

**Empirical studies show positive impacts on
organizational performance**

Wide variance in results reported in the literature

**Evidence developing to support performance
differences across maturity levels**

Empirical data still in a relatively immature state

**Many estimates and extrapolations of change
across maturity levels**

©1997 by Carnegie Mellon University

35



Carnegie Mellon University
Software Engineering Institute

Outline

Benefit of Software Process Improvement

→ **Views on CMM Based SPI**

Project Management: The First Challenge

Additional Consideration for Improvement

Measuring Impacts

Exercise: Making the Case

Closing

©1997 by Carnegie Mellon University

36

...and it impacts with value



Carnegie Mellon University
Software Engineering Institute

Views on CMM Based SPI

Some debate

"After the appraisal..." (SEI 95-TR-009)

©1997 by Carnegie Mellon University

37



Carnegie Mellon University
Software Engineering Institute

CMM – The Right Medicine?

Absolutely not... say James Bach and Tom DeMarco

Of course... say Bill Curtis and Watts Humphrey

Yes, but...

©1997 by Carnegie Mellon University

38



Carnegie Mellon University
Software Engineering Institute

Everybody Knows That . . .

**The CMMSM causes
runaway
bureaucracy.**

**CMM-based SPI
squelches creativity.**

**Appraisals neglect
important issues.**

**Appraisals are not
worth the expense.**

CMM and Capability Maturity Model are service marks of Carnegie Mellon University.

©1997 by Carnegie Mellon University

39

**Routine processes are
handled more efficiently.**

**Technical people are
freed for technical tasks.**

**Appraisals provide
essential focus and
prioritization of issues.**

**Appraisals are well
worth the investment.**



Carnegie Mellon University
Software Engineering Institute

Post-Appraisal Survey* Goals

Address previous shortcomings

- representative data
- broader view of each organization

Information about

- value of appraisals
- **predictability and performance**
- factors that distinguish success and failure

See After the appraisal: A systematic survey of..... SEI 95-TR-009.

©1997 by Carnegie Mellon University

41



Carnegie Mellon University
Software Engineering Institute

Survey Questions

Asked questions about

- the appraisal
- **problems and successes in addressing findings and recommendations**
- the organization and the SPI effort

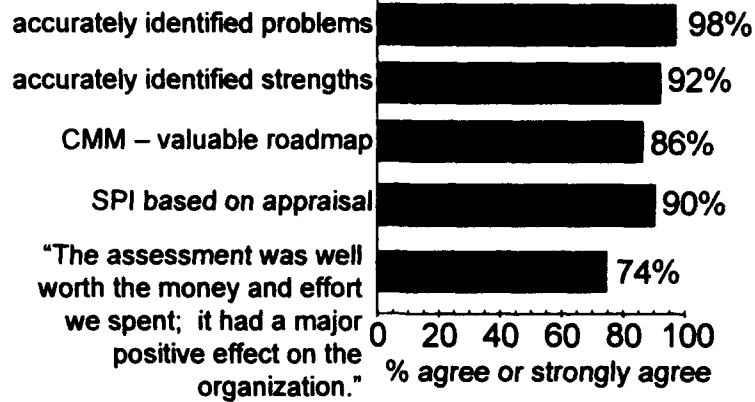
©1997 by Carnegie Mellon University

42



Carnegie Mellon University
Software Engineering Institute

Appraisal Worthwhile?



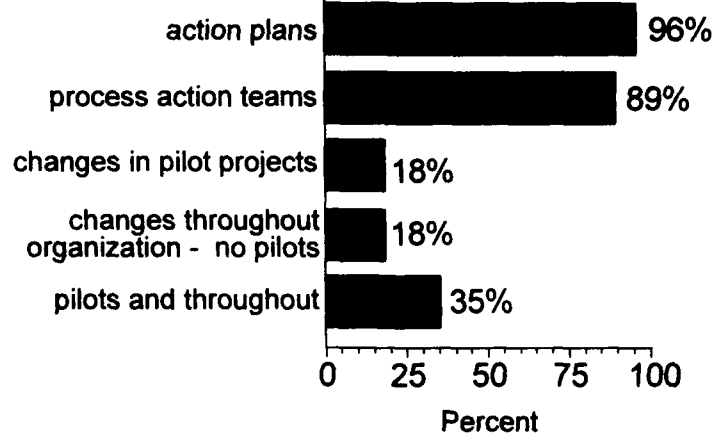
©1997 by Carnegie Mellon University

43



Carnegie Mellon University
Software Engineering Institute

SPI Progress



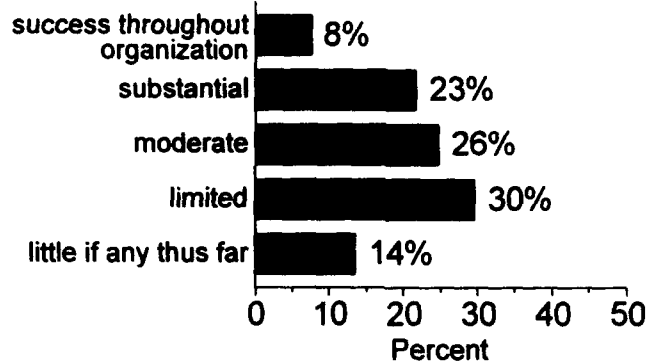
©1997 by Carnegie Mellon University

44



Carnegie Mellon University
Software Engineering Institute

Degree of SPI Success



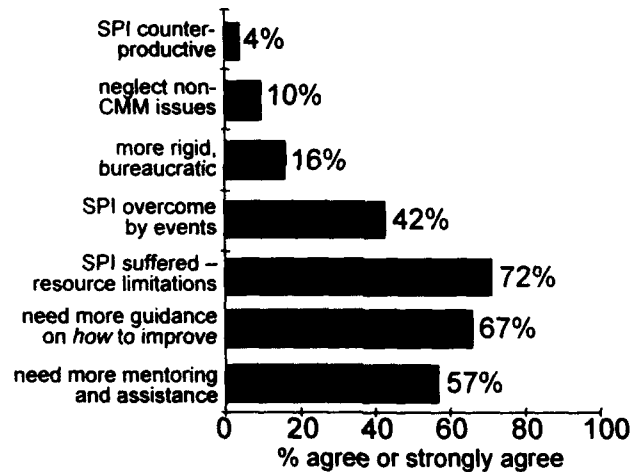
©1997 by Carnegie Mellon University

45



Carnegie Mellon University
Software Engineering Institute

Potential SPI Problems



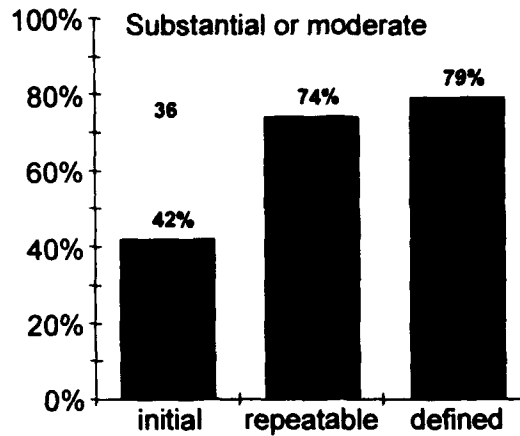
©1997 by Carnegie Mellon University

45



Carnegie Mellon University
Software Engineering Institute

Willingness to Take Risk



©1997 by Carnegie Mellon University

47



Carnegie Mellon University
Software Engineering Institute

Highly Successful Efforts...

- Senior management actively monitors SPI progress**
- Clearly stated, well understood SPI goals**
- Staff time / resources dedicated to process improvement**
- Clear, compensated assignment of responsibility**
- SEPG staffed by highly respected people**
- Technical staff is involved in improvement**

©1997 by Carnegie Mellon University

48



Carnegie Mellon University
Software Engineering Institute

Less Successful Efforts...

Organizational politics

Turf guarding

Cynicism from previous unsuccessful improvement experiences

Belief that SPI "gets in the way of real work"

Need more guidance on *how* to improve, not just what

©1997 by Carnegie Mellon University

49



Carnegie Mellon University
Software Engineering Institute

Outline

Benefit of Software Process Improvement

Views on CMM-based SPI

→ **Project Management: The First Challenge**

Additional Consideration for Improvement

Measuring Impacts

Exercise: Making the Case

Closing

©1997 by Carnegie Mellon University

50



Carnegie Mellon University
Software Engineering Institute

Project Management Processes are a Problem

**"Project management issues emerge as the
main reasons for runaway projects." KPMG**

**"[T]he most important software productivity
and quality improvements today are
management...driven." SRI International**

**"What we have found is that most projects fail
because of people and project management
concerns...." R. Thomsett**

**"Software project management in 1994 is very
amateurish work." C. Jones**

©1997 by Carnegie Mellon University

51



Carnegie Mellon University
Software Engineering Institute

Getting to the Repeatable Level

Conventional wisdom

- Get project management under control first

Questions to be answered

- Which KPAs are the biggest challenges?
- What are the differences between successful
and unsuccessful organizations?

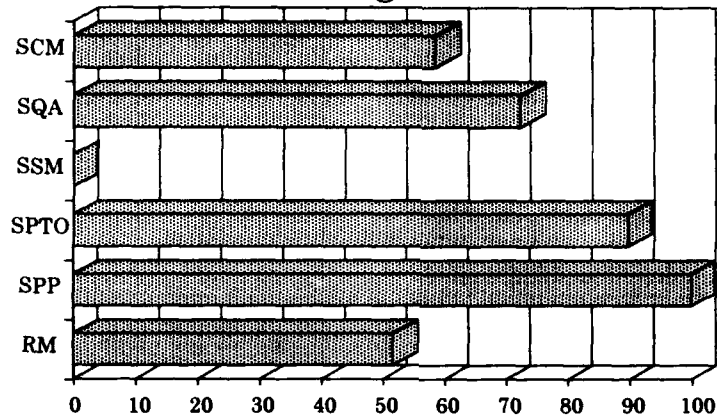
©1997 by Carnegie Mellon University

52



Carnegie Mellon University
Software Engineering Institute

KPAs Included in Findings of Initial Level Organizations*



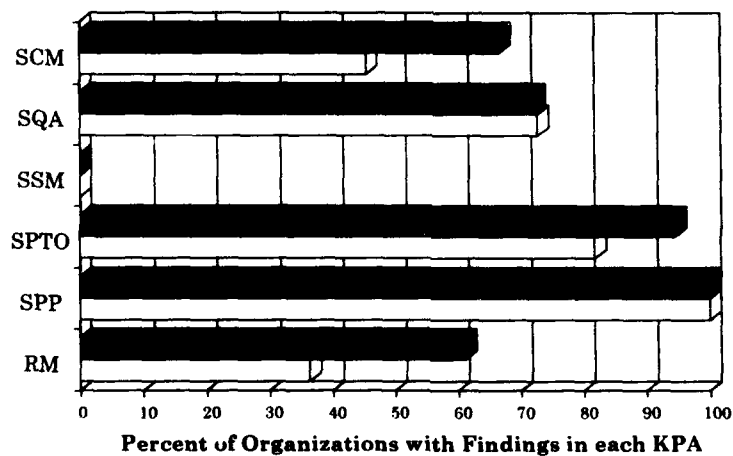
Percent of Organizations with Findings in each KPA
Source: Hayes, W. and Zubrow, D. "Moving on Up: Data and experiences doing CMM-based SPI," SEI 95-TR-008
©1997 by Carnegie Mellon University

53



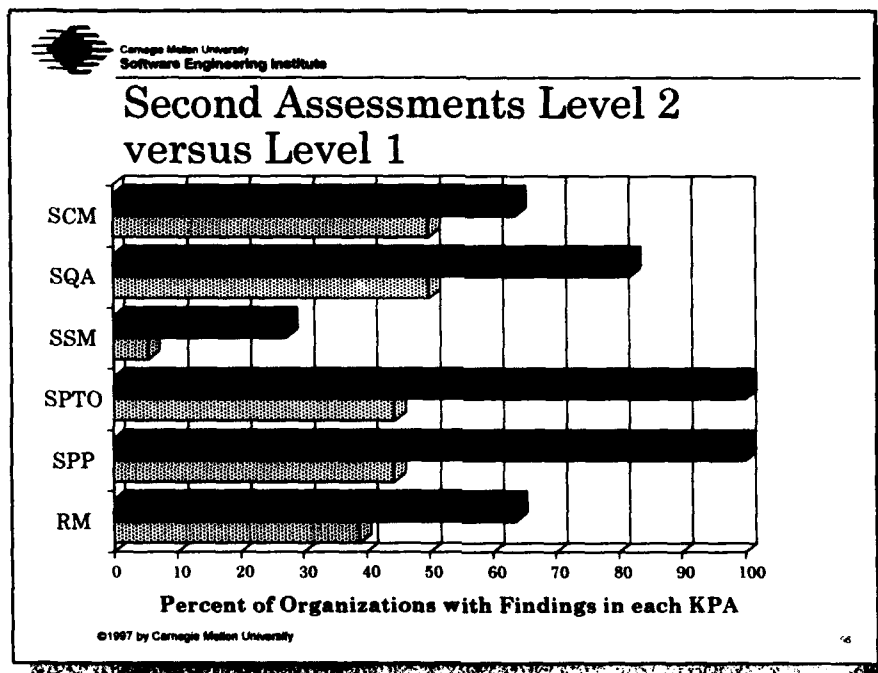
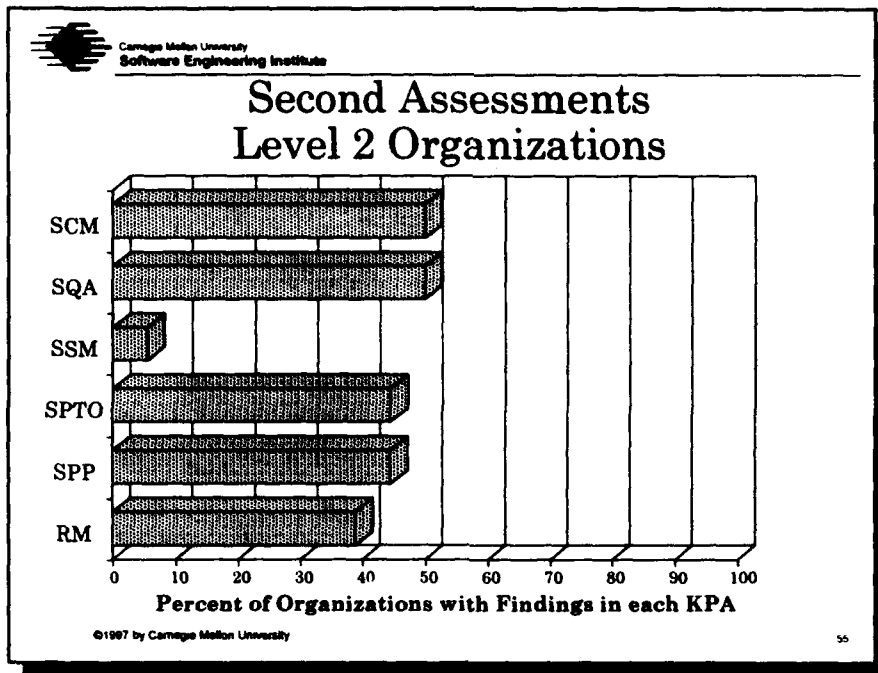
Carnegie Mellon University
Software Engineering Institute

Differences Between First Assessments



Percent of Organizations with Findings in each KPA
©1997 by Carnegie Mellon University

54





Carnegie Mellon University
Software Engineering Institute

Sample Findings Software Project Planning

Major Issues

"Projects are initiated without proper planning."

Minor Issues

"Estimating techniques for Object-Oriented software at embryonic stage."

"Support disciplines (e.g., CM, SQA) sometimes not included in small bids."

©1997 by Carnegie Mellon University

57



Carnegie Mellon University
Software Engineering Institute

Sample Findings Project Tracking & Oversight

Major Issues

"Inadequate tracking of actuals to estimates."

Minor Issues

"Perception that many measures are collected but not always used for project tracking and replanning."

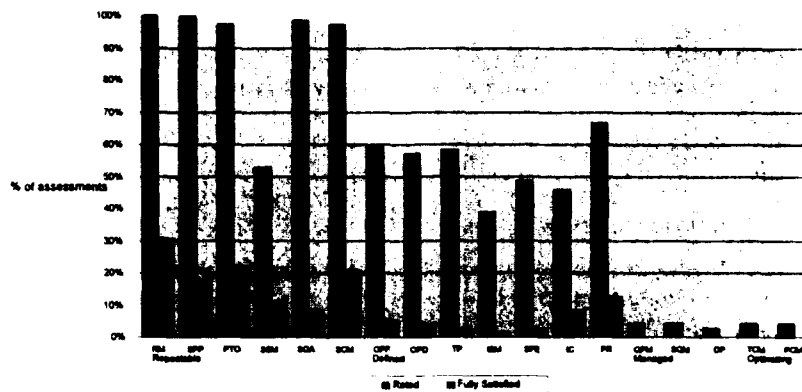
©1997 by Carnegie Mellon University

58



Carnegie Mellon University
Software Engineering Institute

Key Practice Area (KPA) Profiles for Organizations Assessed at Level 1 (going for 2)



Based on 72 IPI assessments. Source: Community Maturity Profile Update, November 1996.

©1997 by Carnegie Mellon University

59



Carnegie Mellon University
Software Engineering Institute

Outline

Benefit of Software Process Improvement

Views on CMM Based SPI

Project Management: The First Challenge

→ **Additional Consideration for Improvement**

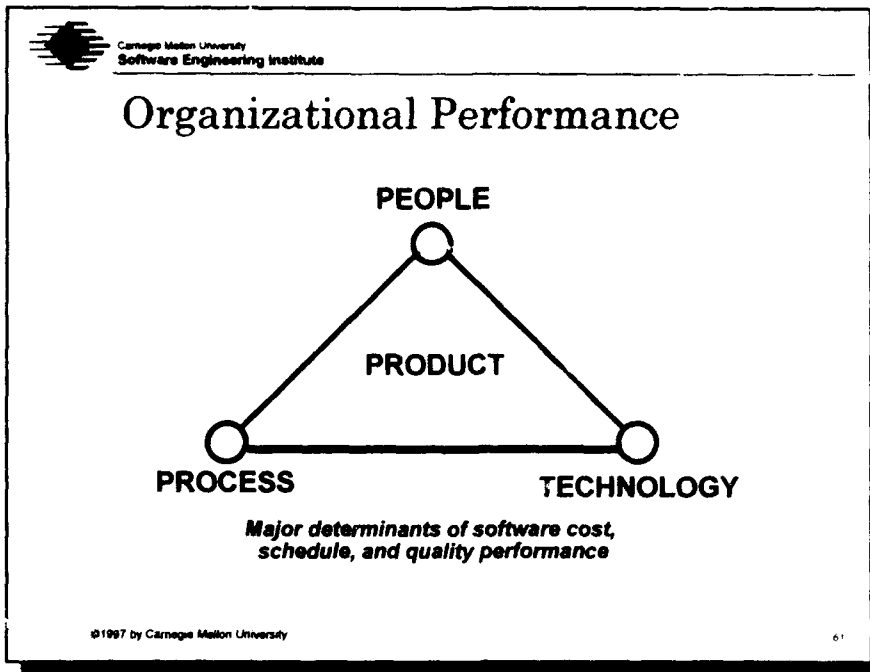
Measuring Impacts

Exercise: Making the Case

Closing

©1997 by Carnegie Mellon University

60



 Carnegie Mellon University
Software Engineering Institute

Impacts of Technology Investment

SRI Case Tool Report

Software Productivity Research Estimates

©1997 by Carnegie Mellon University 62

business impacts and value



Carnegie Mellon University
Software Engineering Institute

SRI CASE Tool Study

Case studies of the investment and use of tools

Short reviews of CASE technologies

Investigation of management issues

Source: Dewey, R. "Software engineering technology and management: the search for high performance solutions"
SRI International, Report no. 762, 1988.

©1997 by Carnegie Mellon University

63



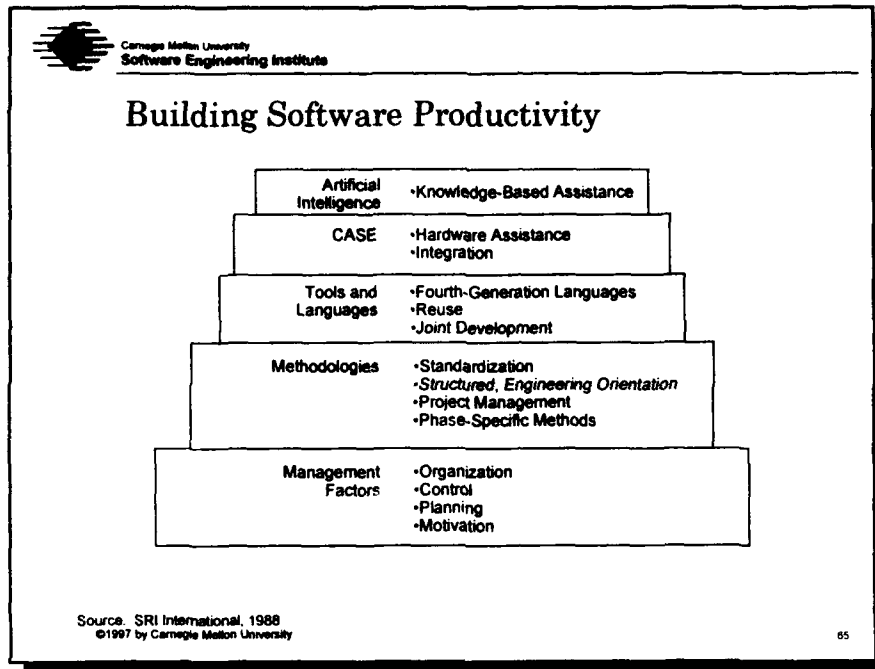
Carnegie Mellon University
Software Engineering Institute

Productivity Results - Selected Cases

<u>Company</u> <u>/Organization</u>	<u>Actions</u>	<u>Reported Results</u>
Bell Canada	Reengineered COBOL payroll application	44% reduction in maintenance costs
Morgan Stanley	Uses fourth-generation language (4GL): Natural	2-to-1 productivity gain over COBOL
University of Auckland	Uses 4GL: LINC	10-to-1 gain per function point over COBOL and PL/1
Du Pont	Uses Cortex's Application Factory	6-to-1 productivity gain over other methods
AION Corp.	Uses its own artificial intelligence product, ADS	5- to 15-to-1 gains reported by customers
Milliken	Uses Transform by Transform Logic	Minimum of 2-to-1 gain over other methods
Federal Express	Uses Teamwork by Cadre	2-to-1 productivity gain over other methods
Bank of America	Uses Nomad-2	10-to-1 productivity gain where used
CNA Insurance Cos.	Uses IBM's Joint Application Design (JAD)	50% productivity gain
Higher Order Software	Uses its own product, USE.IT	4-to-1 productivity gain in controlled test
Hartford Insurance	Uses CASE workstations	2-to-1 productivity gain
Raytheon Missile	Has developed libraries for reuse of commercial stations	60% productivity gain
Lockheed	Uses Ada Environment	50% productivity gain over older methods
Hughes Aircraft	Has developed libraries for reuse	50% productivity gain

Source: Dewey, R. "Software engineering technology and management: the search for high performance solutions"
SRI International, Report no. 762, 1988.
©1997 by Carnegie Mellon University

74



People Impacts and Approaches

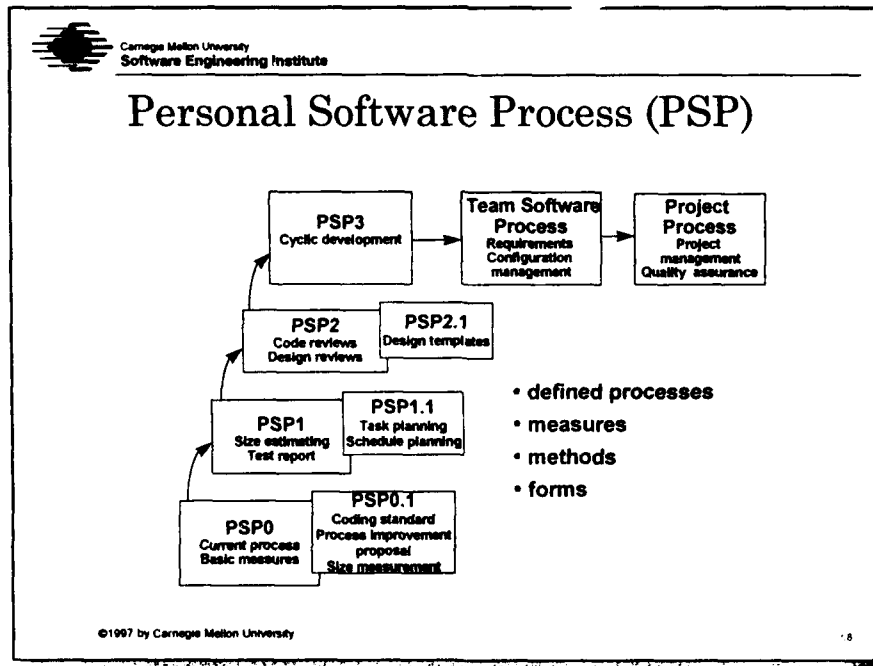
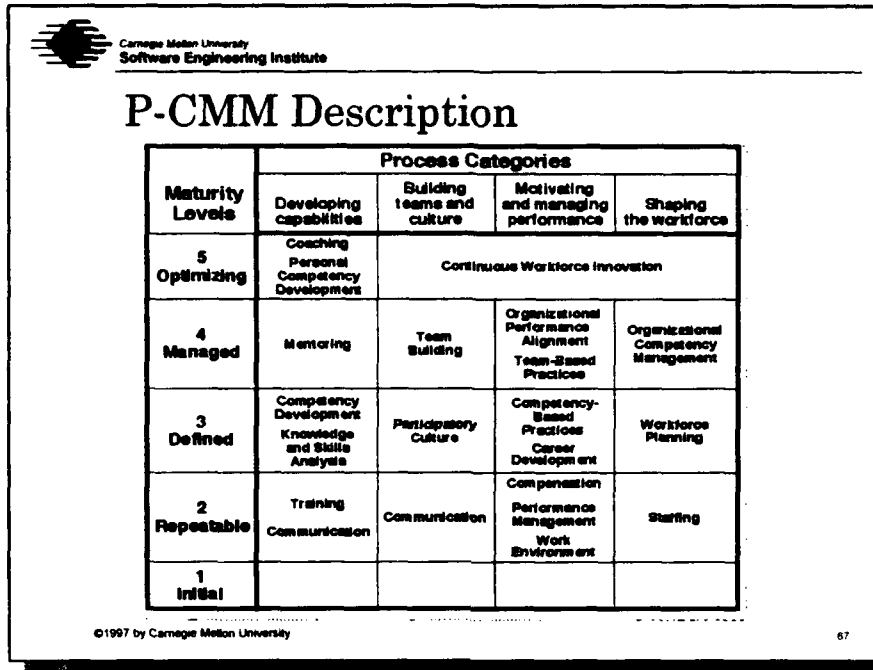
"Ignorance of performance management issues is probably the single greatest explanation for the failure of ... technologies" Yourdon, *Decline and Fall of the American Programmer*, 1992, pg, 67.

Approaches to people issues

- People CMM to address management issues
- PSP to improve engineer performance

©1997 by Carnegie Mellon University

66





Carnegie Mellon University
Software Engineering Institute

PSP Overview

The PSP is a disciplined process for engineers to use for software development and other structured personal tasks

It is extendable to larger-scale team or project development.

Engineers learn to practice disciplined methods that dramatically improve their performance.

©1997 by Carnegie Mellon University

59



Carnegie Mellon University
Software Engineering Institute

Programmer Productivity

Wide variation

- Sackman, Erikson and Grant (1968) 8:1
- Boehm (1981) 4:1
- McGarry (1982) 8:1
- Jones (1985) 50%
reduction in maintenance costs
- PSP results show wide variation initially

©1997 by Carnegie Mellon University

60



Carnegie Mellon University
Software Engineering Institute

The PSP Helps Engineers

With PSP engineers:

- **gather process measurements**
- **define their own processes**
- **manage and improve their processes**
- **follow a ML5 personal process.**

This helps them to:

- **make commitments they can meet**
- **better manage their work**
- **resist unreasonable commitment pressures**

©1997 by Carnegie Mellon University

71



Carnegie Mellon University
Software Engineering Institute

The PSP Helps Organizations

The PSP supports organizational process improvement in several ways.

- **improves overall engineering capability**
- **gets engineers to participate in organizational process improvement**
- **accelerates organizational improvement**

The PSP is an appropriate technology for:

- **large and small organizations**
- **organizations at ML2 and above**
- **strong ML1 organizations**

©1997 by Carnegie Mellon University

72



Carnegie Mellon University
Software Engineering Institute

Average PSP Improvement

From data on 104 engineers through 1995

- total number of defects injected declined by 58%
- defects found in test were reduced by 72%
- average productivity improved by 21%

While these numbers varied by individual:

- most improved defect levels significantly
- most had higher productivity

New results confirm significant impacts on individual engineers

©1997 by Carnegie Mellon University

73



Carnegie Mellon University
Software Engineering Institute

Summary of Improvement

Technology investment

- need foundation in process and management to reap benefits

People factors

- wide variation in performance
- can be enhanced through managerial attention and individual training

Both technology and people impacts contribute to and are enabled by a strong process capability

©1997 by Carnegie Mellon University

4



Carnegie Mellon University
Software Engineering Institute

Outline

Benefit of Software Process Improvement

Views on CMM Based SPI

Project Management: The First Challenge

Additional Consideration for Improvement

→ **Measuring Impacts**

Exercise: Making the Case

Closing

©1997 by Carnegie Mellon University

75



Carnegie Mellon University
Software Engineering Institute

A dilemma

"[practitioners tend to] believe that most of these 'breakthroughs' are at best unproven, and at worst malicious, counterproductive mischief." *R. Glass*

"[U]ntil some method of standardizing the collection and analysis of data is defined, there will be no way of determining how accurately process improvement return can be predicted or measured." *Brodman and Johnson*

©1997 by Carnegie Mellon University

76



Carnegie Mellon University
Software Engineering Institute

But....

The demand for experiential data is great

- **justify proposals**
- **evaluate proposals**
- **benchmarking**
- **decision making**

Evaluating the quality of data can be tricky

©1997 by Carnegie Mellon University

77



Carnegie Mellon University
Software Engineering Institute

Measuring Impacts

The Cost Benefit Equation

Making causal connections

Do the results apply to my organization

Advice to the Analyst

Advice to the Consumer

©1997 by Carnegie Mellon University

78



Carnegie Mellon University
Software Engineering Institute

A Cost Benefit Equation

ROI = Savings / Cost of Improvement

Savings = ?

Cost of Improvement = ?

How do we deal with time?

©1997 by Carnegie Mellon University

79



Carnegie Mellon University
Software Engineering Institute

How do you measure savings?

Savings = costs not incurred

Estimate of what would have been spent

Intangible and non-quantified benefits

Externalities

©1997 by Carnegie Mellon University

80



Carnegie Mellon University
Software Engineering Institute

Cost of Improvement

What do you include?

How long do you include it?

Can the costs of improvement and costs of operations be segmented?

©1997 by Carnegie Mellon University

81



Carnegie Mellon University
Software Engineering Institute

Accounting for time

How should costs and benefits be aligned?

Over what time period are savings assigned to costs?

How are future savings estimated?

Are improvements amortized?

©1997 by Carnegie Mellon University

82



Carnegie Mellon University
Software Engineering Institute

What caused the change?

**Where in the process are measurements made?
Are the measures being analyzed based on the
same definition?**

**Do we really know what's different in the
process? Did the people change? Did the
technology change?**

©1997 by Carnegie Mellon University

63



Carnegie Mellon University
Software Engineering Institute

Do the results apply to my organization?

**Did participants in the study have a similar
process? use similar technology?**

Do they build a similar product?

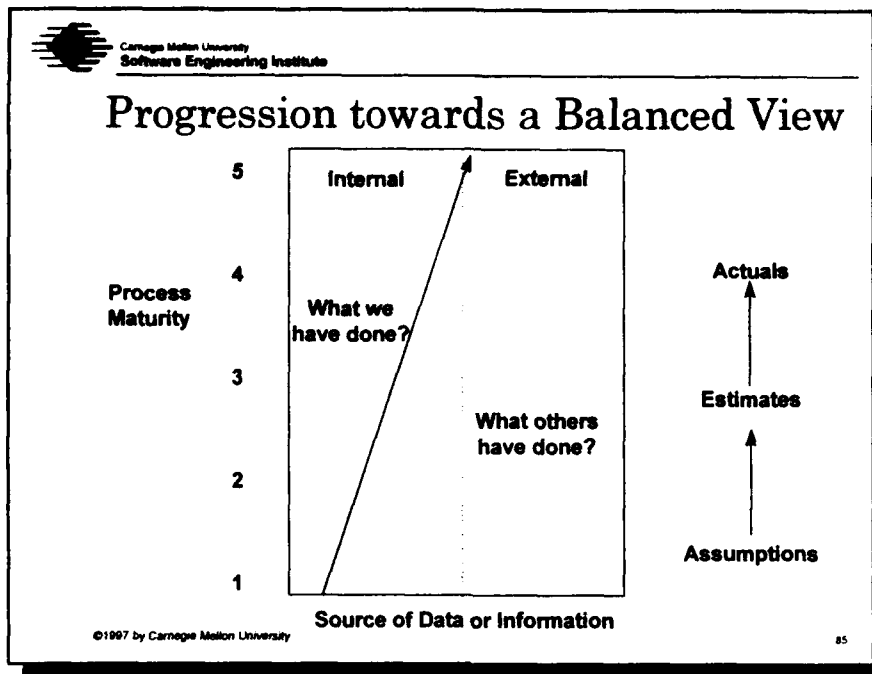
**Are they subject to similar regulations? similar
market pressures?**

**Are the participants of a similar size? a similar
age? similar health?**

How specific is your need?

©1997 by Carnegie Mellon University

64



 Carnegie Mellon University
Software Engineering Institute

Advice to the Analyst

- Specify a model in advance and get agreement**
- Be creative about measures**
- Know and understand the data**
- Differentiate between results and inferences or interpretations**
- Seek to improve your models, measures, and data**
- Match the investment in measurement and analysis to the magnitude of decisions supported**

©1997 by Carnegie Mellon University

86



Carnegie Mellon University
Software Engineering Institute

Advice to the Consumer

Are the data and analytical methods explained in sufficient detail so that you understand them?

Do the data and the results support the conclusions drawn?

- speaking beyond the data
- correlation vs causality

What else might explain the results?

- alternative models, rival hypotheses

Does the confidence level of the results match the magnitude of the decision to be made?

©1997 by Carnegie Mellon University

87



Carnegie Mellon University
Software Engineering Institute

Outline

Benefit of Software Process Improvement

Views on CMM Based SPI

Project Management: The First Challenge

Additional Consideration for Improvement

Measuring Impacts

→ **Exercise: Making the Case**

Closing

©1997 by Carnegie Mellon University

88



Carnegie Mellon University
Software Engineering Institute

Exercise: Making the Case

Divide into two groups: analysts and decision makers

Analysts: Present the case for establishing a CMM-based SPI initiative using the slides from the workshop and/or ones you create

Decision Maker: Comment and critique the case and make a decision

Time: 15 minutes to prepare, 10 minutes to present, 5 minutes for comment and critique, 10 minutes for debrief

©1997 by Carnegie Mellon University

89



Carnegie Mellon University
Software Engineering Institute

Outline

Benefit of Software Process Improvement

Views on CMM Based SPI

Project Management: The First Challenge

Additional Consideration for Improvement

Measuring Impacts

Making the Case

→ Closing

©1997 by Carnegie Mellon University

90



Carnegie Mellon University
Software Engineering Institute

Closing - 1

Growing literature and evidence documenting positive impacts from CMM-based SPI

- not just improved process maturity
- significant returns on investment

Important to consider people and technology as well

Evidence is mounting but basis for prediction and generalization is still immature

Wide variation in the results of studies and estimates

©1997 by Carnegie Mellon University

91



Carnegie Mellon University
Software Engineering Institute

Closing - 2

Need a transition strategy as well as the CMM

Be explicit about models and use them to focus measurement

Document your assumptions and strive to replace them with data

Use data to know your organization

Be a smart consumer of SPI and technology innovation data and analyses

©1997 by Carnegie Mellon University

92

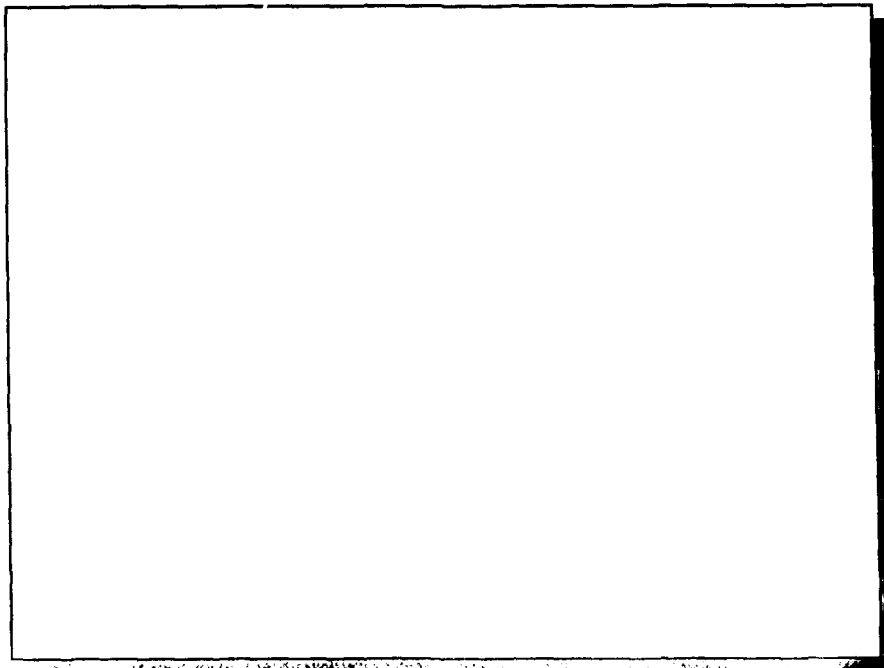


Carnegie Mellon University
Software Engineering Institute

Questions?

Thank you!

©1987 by Carnegie Mellon University 92



Effective Implementation of CMM L2 & L3

Effective implementation of
L2 and L3

Magnus Ahlgren & Christophe Debou

European SEPG '97, Amsterdam

Q-Labs

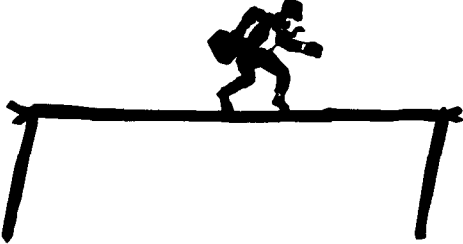
Magnus Ahlgren & Christoph Debou

1

Effective Implementation of CMM L2 & L3

Software Issues

- What would you consider being the biggest software issue your organisation is currently facing?



Q-Labs

Magnus Ahlgren & Christoph Debou

2

Effective Implementation of CMM L2 & L3

Purpose

- Give understanding of principles behind process improvement
- Give insight in differences between process improvement moving towards level 2 and moving towards level 3
- Provide tips and guidelines for your process improvement effort
- Give examples of unorthodox ways of increasing process maturity
- Give possibility to reflect on your own situation

Q-Labs

Magnus Ahlgren & Christoph Debow

3

Effective Implementation of CMM L2 & L3

Agenda

- Introduction
 - CMM Overview
 - Characteristics of levels 2 And 3
- Improvement programme cycle
 - Level 2
 - Level 3
- Example of effective implementation of CMM practices

Q-Labs

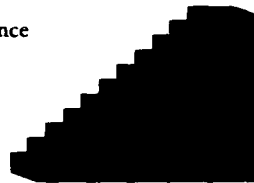
Magnus Ahlgren & Christoph Debow

4

Effective Implementation of CMM L2 & L3

Capability maturity Model Objectives

- CMM is TQM with a software view, is based on state-of-the-art SW Engineering practices and helps organisations:
 - Characterise the maturity of their process
 - Establish goals for process improvement
 - Set priorities for immediate actions
 - Envision culture of software excellence



LEI/043.97/0332 A 02/06/97

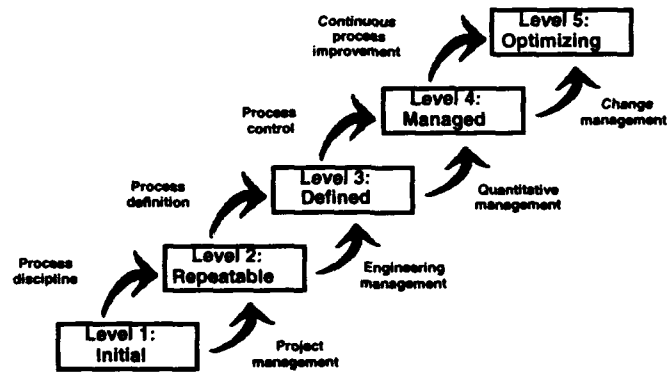
Q-Labs

Magnus Ahlgren & Christoph Debaux

5

Effective Implementation of CMM L2 & L3

The Capability Maturity Model



LEI/043.97/0332 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debaux

6

Effective Implementation of CMM L2 & L3

CMM architecture

Level	Focus	Key Process Areas
Optimizing (5)	Continuous process improvement	Defect prevention Technology change mgmt Process change mgmt
Managed (4)	Product and process quality	Quantitative process mgmt Software quality management
Defined (3)	Defined engineering process	Organization process focus Organization process definition Training program Integrated software management Software product engineering Intergroup coordination Peer reviews
Repeatable (2)	Project management and commitment process	Requirements management Software project planning Software project tracking and oversight Software subcontract management Software quality assurance Software configuration management
Initial (1)	Heroes.	

Q-Labs

Magnus Ahlgren & Christoph Debaux

Effective Implementation of CMM L2 & L3

CMM Level 2 Key Process Areas

- Requirements Management
- Software Project Planning
- Software Project Tracking and Oversight
- Software Subcontract Management
- Software Quality Assurance
- Software Configuration Management

Q-Labs

Magnus Ahlgren & Christoph Debaux

Effective Implementation of CMM L2 & L3

CMM Level 3 Key Process Areas

- Organisation Process Focus
- Organisation Process Definition
- Training Programme
- Integrated Software Management
- Software Product Engineering
- Intergroup Co-ordination
- Peer Reviews

Q-Labs

Magnus Ahlgren & Christoph Debow

Effective Implementation of CMM L2 & L3

Levels 2 and 3 characteristics (1)

Level 2

- Stabilising the environment
- Management establishes process discipline
- Processes are protected within projects
- Successful processes are repeated
- A culture of commitment is being built
- Basic management discipline is installed
- Technical practices are not fully defined

Level 3

- Installing a common process
- Organisation establishes a process framework
- Organisational process is tailored to projects
- Best practices are transferred across the organisation
- A culture of engineering is built
- Management and technical practices are integrated, documented, used and living

Q-Labs

Magnus Ahlgren & Christoph Debow

Effective Implementation of CMM L2 & L3

Levels 2 and 3 characteristics (2)

Level 2

- Some historical data is available
- Improvements are focused on projects
- Product baselines are established and controlled
- Process capability for meeting schedule



Level 3

- Organisation-wide database is available and used
- Formal focus on process improvement exists (SE/PG)
- People are systematically trained to perform their task
- Process capability for meeting schedule, cost and functionality targets



Q-Labs

Magnus Ahlgren & Christoph Debow

11

Effective Implementation of CMM L2 & L3

Improvement Cycles

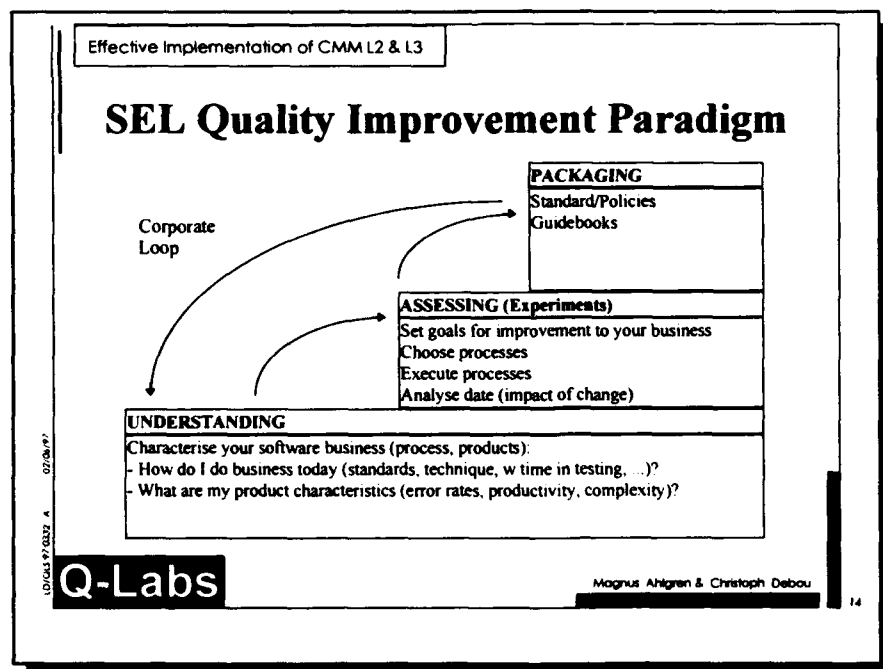
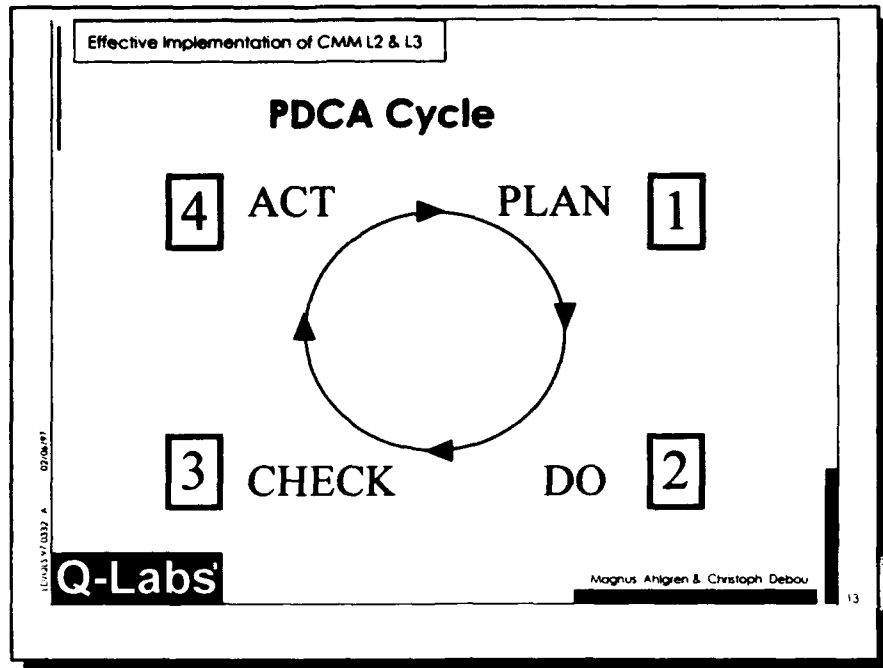
- Shewhart PDCA cycle
- SEL Quality Improvement Paradigm: Experimental Software Engineering (1985)
- ami (Application of Metrics in Industry): goal-oriented and metrics driven improvement cycle (1992)
- SEI IDEAL (1995)
- SPICE Guide for Use in Process Improvement (1995)

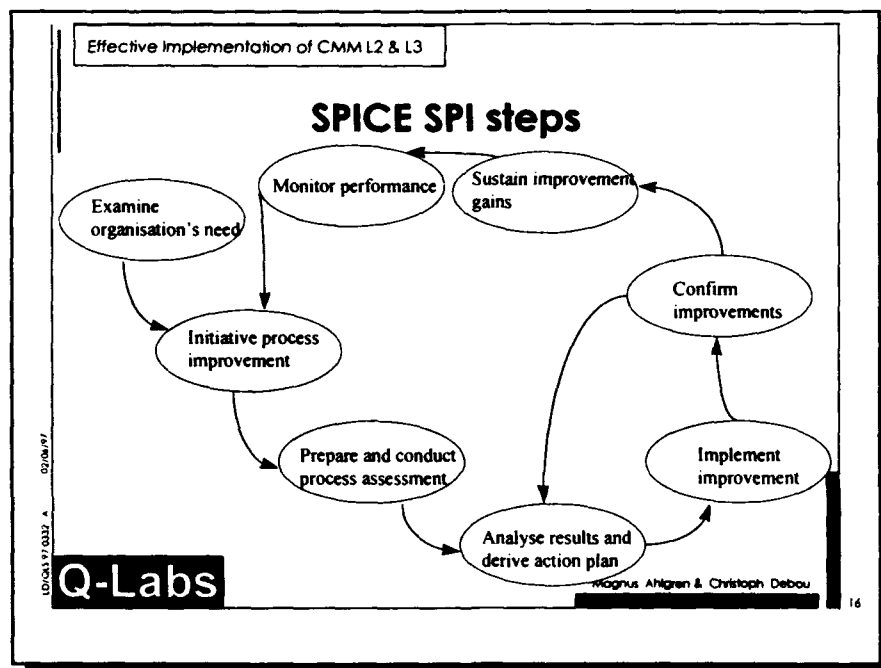
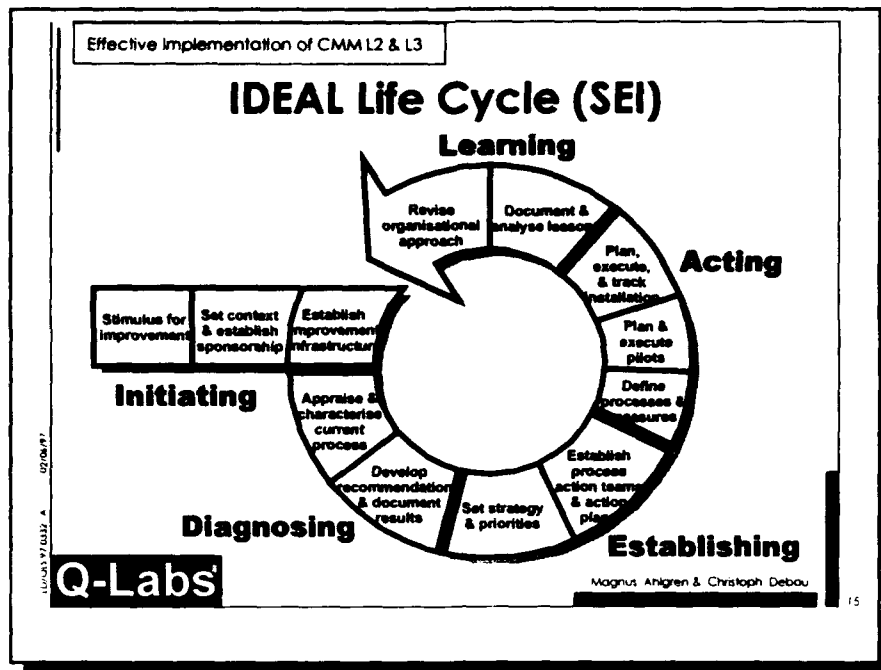


Q-Labs

Magnus Ahlgren & Christoph Debow

12





Effective implementation of CMM L2 & L3

Conclusion

- All those strategies/paradigms are completely sound,
 - only their level of abstraction only the action planning phase differs, specially in techniques to be used.
- Select a paradigm/Apply a set of principles that makes sense in your context
- Level 2:
 - Realistic planning is crucial
 - A life cycle with pre-defined milestones is required
 - People is the Key issue
- Level 3:
 - A more well defined process for process improvement is possible
 - Roles and activities better defined

Q-Labs

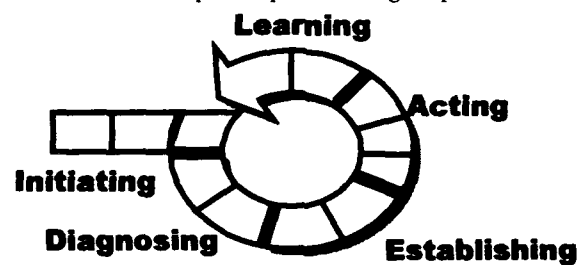
Magnus Ahlgren & Christoph Debaux

17

Effective implementation of CMM L2 & L3

IDEAL - framework for this presentation

This is NOT a presentation of the IDEAL life cycle
 The presentation contains process improvement best practices
 IDEAL framework will help to keep track during the presentation



Q-Labs

Magnus Ahlgren & Christoph Debaux

18

Effective Implementation of CMM L2 & L3

Initiating

- Mobilising for change
- Building a shared vision
- Generating the improvement strategy
- Establish business measures
- Training and orientation
- Establishing the improvement infrastructure

ID: 0117/0112 A 02/04/97

Q-Labs

Magnus Ahlgren & Christoph Debow

19

Effective Implementation of CMM L2 & L3

Mobilising for change

- Mobilisation is the means of creating shared motivation and commitment to achieve a common set of goals.
- It governs the process of mustering the mental energy needed to feed the change process.

Source (Gouillart and Kelly - 1995)

ID: 0117/0112 A 02/04/97

Q-Labs

Magnus Ahlgren & Christoph Debow

20

Mobilising for change

- Building motivation and understanding that change/improvement is necessary for future business
- Changeability is perceived to be one of the most important characteristics for a company to survive
- The ability to create mobilisation will be a key driver to increase changeability
 - identify and/or envision the challenge
 - make the challenge concrete
 - communicate the challenge to all involved

ID004170317 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debow

21

Example of challenges

- Perceived problems, e.g.
 - Measurable , e.g. deliveries X months late, number of defects in delivered products
 - Guts feeling
 - Customer complaints
- Future scenarios
 - if obvious problems do not exist
 - to prepare for future competitiveness and market desire

ID004170317 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debow

22

Effective Implementation of CMM L2 & L3

Ericsson corporate motivation

- SW represents an increasing share of total R&D expenditure
- The cost of poor software quality is increasing as a percentage of R&D, 1993 estimates for just TR handling - USD 250 million
- The competition has identified SW as a key issue, Motorola introduces the 6 sigma concepts into SW, also claiming to be at CMM level 3 with all their design centres 1995
- SW is the number one enabler to address new market opportunities, e.g. PCS, multimedia etc.

ID:G11 97/0332 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debaou

23

Effective Implementation of CMM L2 & L3

Ericsson Communicatie B.V.

- Ericsson Telecommunicatie B.V. situation
- Big trouble
- Almost lost its major customer
- Close to be thrown out of AXE-10 standard design
- Reasons for status
- Late deliveries
- High fault density
- Turbulent management situation
- Changes to requirements not adequately managed
- CMM level 1 at assessment in 1994

ID:G11 97/0332 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debaou

24

Effective Implementation of CMM L2 & L3

Incentives for measurable motivation

- Management get many requests for investments
 - Increased marketing organisation
 - IT equipment
 - CASE tools
- Executives want numbers
 - if you make sense they will listen
 - they must justify investment in improvements
- There is generally data
 - may be just "laying" around

Q-Labs

Magnus Ahlgren & Christoph Debau

25

Effective Implementation of CMM L2 & L3

Provide a road map

- Gather obvious data that affect cost, schedule or quality
 - w of missed delivery dates
 - w of schedule overrun
 - # of defects per KSLOC or FP delivered to customer
 - cost of servicing customer complaints
 - w of development time spent in rework
 - cost of fixing defects late
- Be careful when presenting data
 - explain the limitations
 - be conservative in extrapolating

Q-Labs

Magnus Ahlgren & Christoph Debau

26

Effective Implementation of CMM L2 & L3

Future scenarios

- Adequately long horizon, e.g. year 2002 scenario
- Develop several scenarios for different potential futures
- Involve different persons with different background
 - managers, project managers, designers, testers, requirements people
 - external sources of information (readings, consultants)
- Investigate
 - possible developments in the domain
 - what it takes to be competitive in that domain

11/04/03 9:03:32 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debaux

27

Effective Implementation of CMM L2 & L3

Achieving Mobilisation

- Understand current challenges
- Identify what challenges that can be addressed by process improvements
 - many challenges has to do by other means e.g. product strategies, market strategies and re-organisations
- There must always be a bottom line business need for the process improvement activities
- Communicate, communicate, communicate

Understand WHY you want to do SPI

11/04/03 9:03:32 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debaux

28

Effective Implementation of CMM L2 & L3

Achieving Mobilisation

- Level 2
 - Build a business case
 - Management must be committed to business case
 - Use data if available, if not get “factual” viewpoints and build scenarios, guts feeling
 - Management must communicate business case frequently
 - Towards level 2 the business case is crucial
- Level 3
 - Re-evaluate the business case
 - Define and use data to support the business case
 - Make a cost analysis for the “new” process organisation
 - Towards level 3 the business case becomes more factual, less guts feeling

Q-Labs

Magnus Ahlgren & Christoph Debow

29

Effective Implementation of CMM L2 & L3

Exercise on Mobilisation

- What do you consider are your main drivers/major incentives for process improvements, consider:
 - Perceived problems
 - Future scenarios

Q-Labs

Magnus Ahlgren & Christoph Debow

30

The shared vision

- The vision provides a shared mental framework that gives form to a new and better future, i.e. a mental focus and a sense of purpose (source: Gouillart and Kelly, 1995)
- A vision is a picture of the future you seek to create, described in the present tense, as if it were happening now
 - shows where we want to go
 - what we will be like when we are there
 - Comes from the latin videre "to see"
 - The link to seeing is significant

LUND 1997-03-02 A 02/04/97

Q-Labs

Magnus Ahlgren & Christoph Debau

31

Example visions

- Motorola: In 1998 Motorola shall be recognised as the world's pre-eminent software supplier.
- Ericsson: 1997 Ericsson will be the world leading supplier of telecom software
- Coca Cola: Put the product within arm's reach for anyone in the world
- Pepsi Cola: Defeat Coca Cola

LUND 1997-03-02 A 02/04/97

Q-Labs

Magnus Ahlgren & Christoph Debau

32

Creating the vision statement

- A strategic intent is the core of the vision
 - the firm's ambition in life
 - captures the imagination of the entire organisation
 - extend boundaries within the realm of the possible
- The leader's task is to give the vision statement life
- Guidelines for creating a vision statement
 - be bold - guts and nerve is required
 - be broad - encompass business as whole and imply extensions into new businesses
 - ...but not too broad - it is possible to be too ambitious
 - Look a long way ahead - 5 to 10 years

ID: A371337 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debow

33

A broad business idea - Telia (Swedish PTT)

- Telia develops quality of life, environment, and competitiveness for people and organisations by connecting them through easy-to-use telecommunication based information services.

Note, this is a translation from Telia's business idea on Swedish, wording is not validated by Telia

- Official Telia statement in Swedish: Telia utvecklar livskvalitet, miljö och konkurrenskraft för människor och organisationer genom att förena dem via lättanvända telekombaserade informationstjänster

ID: A371337 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debow

34

Effective Implementation of CMM L2 & L3

Building the shared vision

- Level 2
 - Optional, but important to commit management to the SPI activities
 - Make the site executive write down his/her vision
 - Discuss and reformulate the vision in the management team
 - Communicate vision and conduct workshops where staff
- Level 3
 - A prerequisite, since now we start building common process assets and infrastructure
 - Develop a suggestion for a vision in the management team
 - Staff reevaluate the suggested vision statement
 - Build action plans based on vision

Q-Labs

Magnus Ahlgren & Christoph Debau

35

Effective Implementation of CMM L2 & L3

Shared vision - exercise

- Please write what you consider being the vision for your organisation
 - write down the one you have or,
 - write down the one you think you should have

Q-Labs

Magnus Ahlgren & Christoph Debau

36

Effective Implementation of CMM L2 & L3

Process improvement - a business strategy

- Software Process Improvement is a business strategy
 - It is a tool for managers to improve efficiency
 - It requires investments
 - It requires management decisions on trade offs
- Software Process Improvement is one of many potential strategies:
 - Will software process improvement contribute to solving our business challenges?
 - Does process improvement contribute to our vision?

Q-Labs

Magnus Ahlgren & Christoph Debaux

37

Effective Implementation of CMM L2 & L3

Generating the improvement strategy (1)

- Strategic objectives
 - What constitutes a successful improvement programme
 - reduced cycle time
 - CMM level
 - happy customers
- Organisational span
 - organisational units to be included
 - likely allies for SPI that must be established
- Improvement infrastructure

Q-Labs

Magnus Ahlgren & Christoph Debaux

38

Generating the improvement strategy (2)

- Implementation strategy
 - life cycle for the SPI-programme
 - selection of model(s) for guiding improvements (CMM, Quality awards, measurements)
- Guidance for planning
 - budget
 - primary milestones (e.g. Performing a CMM-based assessment)
 - primary deliverables

ISO/IEC 15504-4:2015

Q-Labs

Magnus Ahlgren & Christoph Debat

39

Improvement strategy

- | | |
|--|--|
| <ul style="list-style-type: none"> • Level 2 <ul style="list-style-type: none"> – Strategic objectives may not be quantifiable – Put demands on the organisation for improvement – Planning and budgeting of improvements are rather rough, estimate about 3w of budget | <ul style="list-style-type: none"> • Level 3 <ul style="list-style-type: none"> – Strategic objectives precise and preferably quantifiable – Planning and budgeting more precise, often based on previous experiences – Life cycle model is detailed and defined, mechanisms for implementation and institutionalisation exists |
|--|--|

ISO/IEC 15504-4:2015

Q-Labs

Magnus Ahlgren & Christoph Debat

40

Effective Implementation of CMM L2 & L3

Establish business measures

- Establish concrete measures to assess status of business, e.g. by building a Balanced Scorecard
- All measures should be based on a clear purpose, (goal-oriented)
- Measures is the strongest vehicle to motivate process improvements

10001970032 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debow

41

Effective Implementation of CMM L2 & L3

The Balanced Scorecard

- Gives a comprehensive picture of an organisations
 - Status
 - Trends
- Connects financial measures/perspectives with other, e.g.
 - Customer
 - Internal Business
 - Innovation and Learning
 - Process
- States Goals and Measures for each perspective

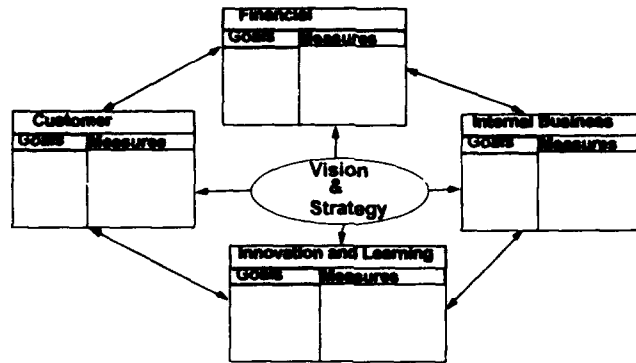
10001970032 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debow

42

Building the balanced scorecard

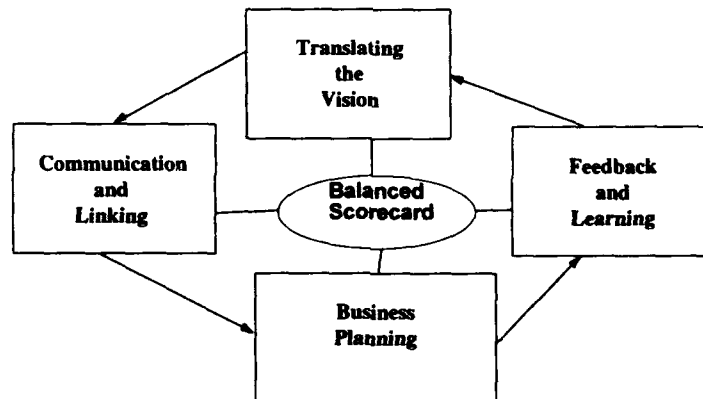


Q-Labs

Magnus Ahlgren & Christoph Debat

43

Balanced Scorecard - role



Q-Labs

Magnus Ahlgren & Christoph Debat

44

Balanced scorecard in use

- Helps organisations:
 - Understand the system entities affecting their way of doing business
 - Correlates these entities so that the system can be understood
 - Moves organisations beyond short-term financial decisions
 - Understand the intangible assets
- Use the goals of the Balanced Scorecard to
 - Derive goals for improvement program
 - Derive goals for e.g. GQM goal setting

10/01/97 03:37 A

Q-Labs

Magnus Ahlgren & Christoph Debow

45

The Goal-Question-Metric paradigm

- The GQM Paradigm involves the following principles:
 - Set explicit measurement-goals
 - Acquire quality models from people involved
 - Consider context
 - Derive appropriate metrics
 - Analyse data according to goals
 - Let data be interpreted by people involved



10/01/97 03:37 A

Q-Labs

Magnus Ahlgren & Christoph Debow

46

Effective Implementation of CMM L2 & L3

Measurements for SPI

- Measurement to track progress
 - Where are we today?
 - Long-term goals
 - Short-term goals
- Improvement program
 - Are we improving?
 - Fewer defects from previous phases
 - Cutting lead-time
 - Is it cost effective?
- Why are we improving (or not)?
 - What factors influenced the outcome?

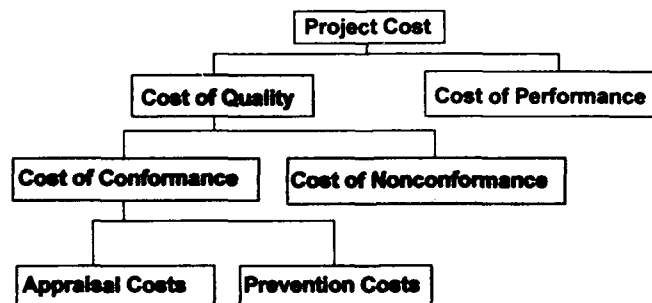
Q-Labs

Magnus Ahlgren & Christoph Debow

47

Effective Implementation of CMM L2 & L3

Crosby's Cost of Quality Model



Q-Labs

Magnus Ahlgren & Christoph Debow

48

Effective Implementation of CMM L2 & L3

Raytheon's Cost Analysis Method

- Crosby's cost of quality model:
 - Performance - cost of building it right the first time
 - Nonconformance - cost of rework
 - Appraisal - cost of testing
 - Prevention - cost of preventing nonconformance
- Change in average% project time by cost type:

	Perform	Noncomf	Apprais	Prevent
1988	34%	41%	15%	7%
1990	55%	18%	15%	12%
1992	66%	11%		
1994	76%	6%		

Q-Labs

Magnus Ahlgren & Christoph Debow

49

Effective Implementation of CMM L2 & L3

Opportunities for ROI

- Lower maturity levels (according to CMM)
 - cost and schedule overruns
 - project terminations
 - expensive rework
 - 30% at TRW (Boehm, 1987)
 - 40% at NASA-SEL (McGarry, 1987)
 - 33% at IIP (Duncker, 1992)
 - 41% at Raytheon (Dion, 1993)

Q-Labs

Magnus Ahlgren & Christoph Debow

50

Effective Implementation of CMM L2 & L3

Business measures

- Level 2
 - Define business measures, collect and analyse data, ex build a Balanced Scorecard (optional)
 - Define goal-oriented measures for process improvement and start measuring
- Level 3
 - Define/Refine the Balanced Scorecard
 - Analyse data and refine measures for monitoring SPI progress

Q-Labs

Magnus Ahlgren & Christoph Debow

51

Effective Implementation of CMM L2 & L3

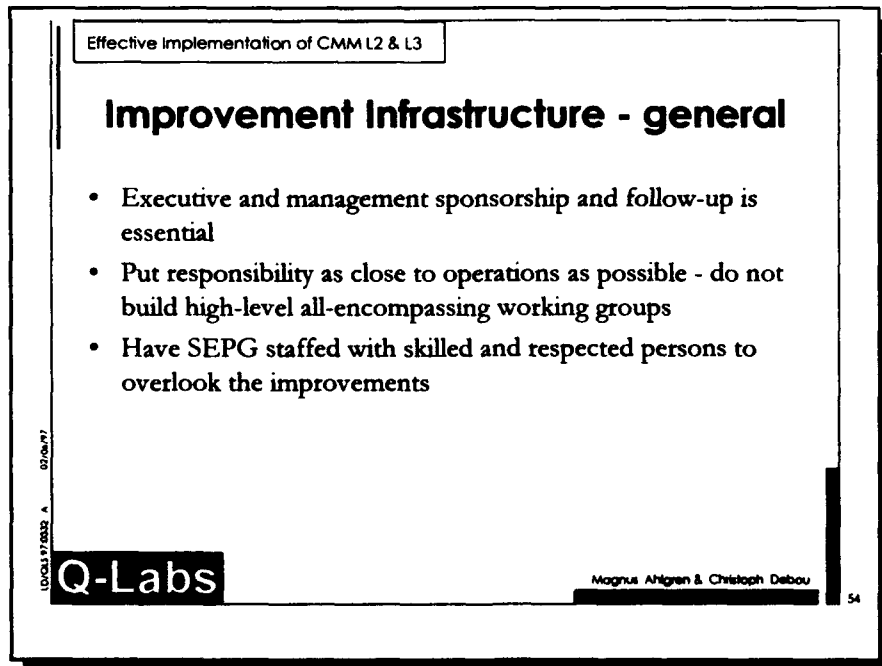
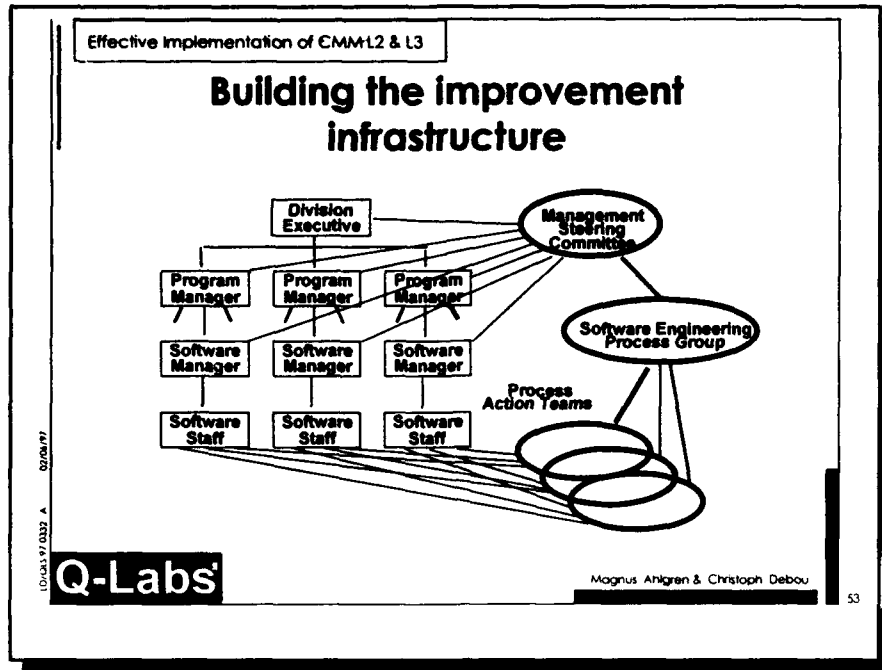
Structuring the operations for improvement

- Infra-structure with long-term focus is necessary to drive improvement
 - What roles and responsibilities must be defined
 - How to interact with sponsors and management
 - Budget and resources
 - Activities external and internal to projects
 - Infra-structure for reuse of experience
 - Operational in time

Q-Labs

Magnus Ahlgren & Christoph Debow

52



Effective Implementation of CMM L2 & L3

Improvement Infrastructure

- **Level 2**
 - Keep it simple, but clear
 - Distribute responsibility to projects and product lines
 - Project-based improvements
 - Product line steering board
 - Steering group for and a directive for its work is established
 - Little organisational co-ordination (SEPG)
 - SEPG's role mainly in coaching projects
- **Level 3**
 - Reinforced SEPG co-ordination
 - Shared responsibility between organisation and product lines
 - Organisational-wide steering Committee including
 - Organisation-wide/Product line working groups to identify best practices, package them and support the deployment.
 - Process maintenance taken into account

Q-Labs

Magnus Ahlgren & Christoph Debou

55

Effective Implementation of CMM L2 & L3

Improvement Infrastructure - Exercise

- Describe your current improvement infrastructure and identify issues/improvement areas
- Discussion how to organise improve infrastructure with regards to:
 - organisational coverage
 - roles and responsibilities

Q-Labs

Magnus Ahlgren & Christoph Debou

56

Effective Implementation of CMM L2 & L3

Training and orientation

- Level 2
 - Invite expert for 1-day motivation seminar
 - Executive and managers gets trained in CMM, SPI and their role (1- 2 days)
 - Improvement co-ordinator gets trained in CMM (3 days) and in his role in SPI (2-3 days)
 - Staff gets trained in CMM and SPI (1/2 - 2 days depending on role)
 - Staff gets orientation SPI initiative and intention
- Level 3
 - Executives gets trained in CMM level 3 (1 day)
 - Staff gets trained in CMM level 3 (1/4 - 2 days depending on role)
 - Staff gets orientation SPI initiative and intention

Q-Labs

Magnus Ahlgren & Christoph Debou

57

Effective Implementation of CMM L2 & L3

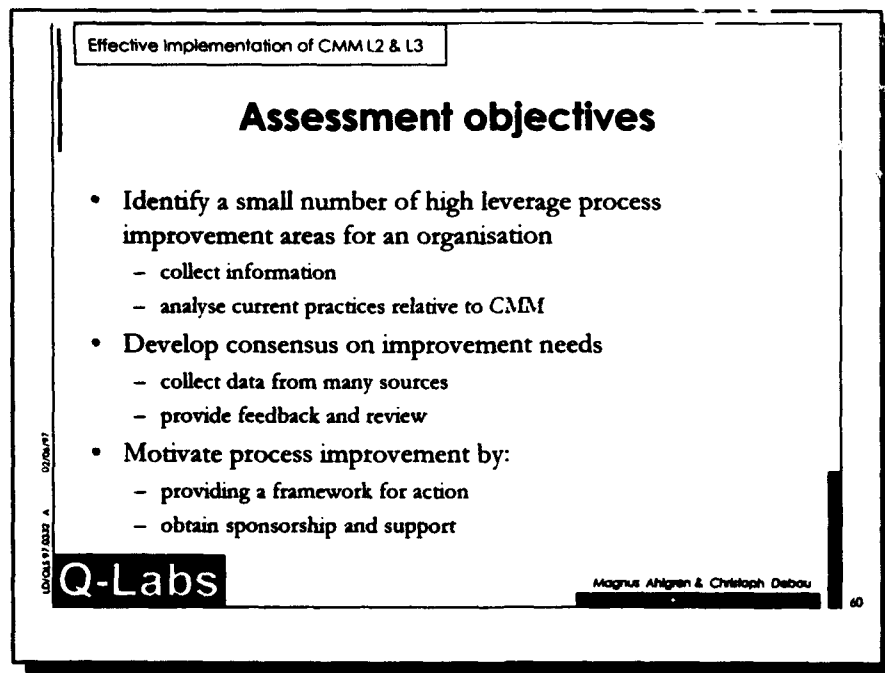
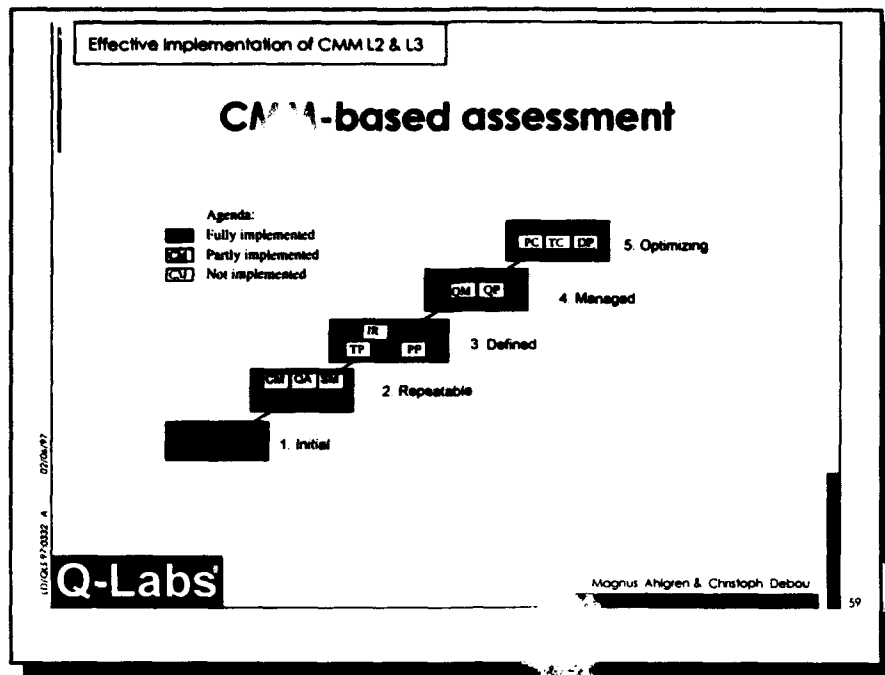
Diagnosing

- Performing the assessment
- Management participation

**Q-Labs**

Magnus Ahlgren & Christoph Debou

58



Effective Implementation of CMM L2 & L3

Different types of assessments

- **Mini-assessment**
 - A group of carefully selected people make a walk-through of a maturity model-based questionnaire and answer these questions
- **CMM Light assessment**
 - A large group of carefully selected individuals from different competency areas fill out questionnaires
- **SEI-style assessment**
 - Project leader fill out questionnaires
 - Interviews are made with project leaders
 - Discussions are performed between practitioners from different competency area

1D-QJ 197-0332 A 02/04/97

Q-Labs

Magnus Ahlgren & Christoph Debou

61

Effective Implementation of CMM L2 & L3

Perform assessment

- **Level 2**
 - Perform a CMM-based assessment tailored to the organisation
 - The bulk of the information is from interviews and discussions
 - Focus is on projects and on CMM KPA Activities to Perform
- **Level 3(re-assessment)**
 - Perform a CMM-based assessment tailored to the organisation
 - The information from interviews and discussions is expanded with document review
 - Focus is on both project and organisation and large attention is paid to ALL CMM common features

1D-QJ 197-0332 A 02/04/97

Q-Labs

Magnus Ahlgren & Christoph Debou

62

Effective Implementation of CMM L2 & L3

Management participation in assessment

- Level 2 and level 3
- The site executive and other managers with responsibility for software development need to be present to demonstrate their commitment to the assessment result
 - at assessment start up (optional, depending on assessment type)
 - during management interview (optional, depending on assessment type)
 - during presentation of assessment result (compulsory)

Q-Labs

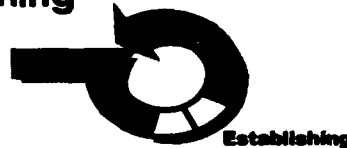
Magnus Ahlgren & Christoph Debou

63

Effective Implementation of CMM L2 & L3

Establishing

- Prioritise improvements
- Establish process action teams
- Planning of improvements



Q-Labs

Magnus Ahlgren & Christoph Debou

64

Effective Implementation of CMM L2 & L3

Prioritise Improvements

- The assessment result contains findings in about 7 Key Process Areas
- It is recommended that an organisation focuses improvements in about 3 areas
- Prioritisation is necessary

**Q-Labs**

Magnus Ahlgren & Christoph Debou

65

Effective Implementation of CMM L2 & L3

Strategy for prioritisation (1)

- Use the CMM as a general model for prioritisation
 - level 2 before level 3
 - consider internal relation at levels
 - ensure common sense and consensus around CMM requirements
- Consider the following when prioritising
 - What will give the largest leverage on business figures
 - What is easy to implement
 - Are there any immediate success stories
 - What improvements would get most buy in from people
- Consider differences between product lines

Q-Labs

Magnus Ahlgren & Christoph Debou

66

Effective Implementation of CMM L2 & L3

Process for prioritisation meeting

- Rank the findings areas
- Rank each of the findings
- Develop recommendations for the 3/4 most highly ranked findings
- Rank recommendation based on selected criteria, e.g.
 - ranking of finding(s) it tries to solve
 - effect on business criteria, e.g. quality
 - ease of implementation
- Let management steering committee review and agree to suggested prioritisation

Q-Labs

Magnus Ahlgren & Christoph Debou

67

Effective Implementation of CMM L2 & L3

Prioritisation

- | | |
|---|--|
| <ul style="list-style-type: none"> • Level 2 <ul style="list-style-type: none"> – Conduct a recommendations meeting – Keep it simple – Focus on selecting Key Process Areas for improvement – Review prioritisation with management – “CMM level 2 within 12 months” may be a lousy goal but a good driver | <ul style="list-style-type: none"> • Level 3 <ul style="list-style-type: none"> – Conduct (a set of) prioritisation meeting (s) – Stronger focus on the actual recommendations – More strategic thinking is recommended – Review prioritisation with management – “CMM level 3” within 12 months is a quite lousy goal, more complex goals could be handled |
|---|--|

Q-Labs

Magnus Ahlgren & Christoph Debou

68

Effective Implementation of CMM L2 & L3

Establish Process Action Teams

- Select key staff to implement the improvements in certain Key Process Areas
 - respected
 - knowledgeable in the area(s)
 - knowledgeable of the organisation
 - good communicator
 - energetic
- Assign PAT members for a substantial amount of time with designated responsibility

Q-Labs

Magnus Ahlgren & Christoph Debou

69

Effective Implementation of CMM L2 & L3

Establish Process Action Teams

- | | |
|--|---|
| <ul style="list-style-type: none"> • Level 2 <ul style="list-style-type: none"> – Build loosely co-ordinated PATs – Get experts in the various Key Process Areas – Give PATs orientation in findings – Train persons in “their” CMM Key Process Area(s) – Get persons that can coach projects/product lines | <ul style="list-style-type: none"> • Level 3 <ul style="list-style-type: none"> – Establish PATs that are co-ordinated by an SEPG – Tie experts to the PATs work in a network alike form – Give PATs orientation in findings – Train persons in CMM Key Process Areas |
|--|---|

Q-Labs

Magnus Ahlgren & Christoph Debou

70

Effective Implementation of CMM L2 & L3

Planning of improvements

- Planning is essential
- Base the planning on
 - the prioritised areas for improvements (prioritised recommendations)
 - ambition with regards to speed
 - availability of resources
 - the activities that are expected to be executed
- Establish clear mile stones

L2: 4157/032 A 02/04/97

Q-Labs

Magnus Ahlgren & Christoph Debou

71

Effective Implementation of CMM L2 & L3

Planning of improvements

- | | |
|---|---|
| <ul style="list-style-type: none"> • Level 2 <ul style="list-style-type: none"> – Commit resources – Identify areas to build in communality – Identify type of activities <ul style="list-style-type: none"> • Support to projects • Common activities – Assign effort and time schedule – Clear goals: Ex. All projects started during Q4 1994 should fulfil the Software Project Planning KPA | <ul style="list-style-type: none"> • Level 3 <ul style="list-style-type: none"> – Commit resources – Define clear deliverables – Define detailed plans with activity descriptions (see IDEAL model) <ul style="list-style-type: none"> • process development • development of measurements • pilot execution • practice dissemination |
|---|---|

L2/0337/032 A 02/04/97

Q-Labs

Magnus Ahlgren & Christoph Debou

72

Effective Implementation of CMM L2 & L3

Acting

- Coaching
- Defining process assets
- Follow-up assessments
- Training and orientation
- Management follow-up



LD/QL17/0132 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debou

73

Effective Implementation of CMM L2 & L3

Improvement approach

- Level 2
 - Select business-critical projects as SPI customers
 - SPI co-ordinator/PAT works together with project leaders to define lists of prioritised improvements (follow SW life cycle)
 - PAT work is more of project task force instead of WG
 - Training off staff in CMM and its implementation
- Level 3
 - Improvement Plan's scope is the whole organisation
 - Focus on identifying and transferring best practices across projects cross-projects
 - Cross-projects PAT's approach is envisageable
 - Cross-project Software Engineering training to achieve synergy and common culture

LD/QL17/0332 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debou

74

Improvement approach (2)

- Level 2
 - Focus effort on key issues (Management ones)
 - Regular incremental introduction of sets of concrete practices into projects
 - Initial support by external consultants when resources or skills are lacking.
- Level 3
 - Focus on all CMM aspects (all common features)
 - Planned development, piloting and deployment of practices
 - Initial support by external consultants when resources or skills are lacking.

Q-Labs

Magnus Ahlgren & Christoph Debow

75

Coaching

- Level 2
 - The most important activity is coaching projects
 - The SEPG/PAT sit down with project manager and the project team to interpret CMM and how it can be applied
 - The coaching activities are based on individual skills (Ad hoc)
 - CMM Kick-Off with project members at project start-up
- Level 3
 - Coaching is integrated in the improvement approach
 - A process is defined based on best practices and thereafter transferred
 - Structured technology transfer contains coaching of new practices

Q-Labs

Magnus Ahlgren & Christoph Debow

76

Effective Implementation of CMM L2 & L3

Coaching activities at level 2, ex (1)

- Support projects to define and apply project level processes, procedures and standards
- Work with project management on defining and implementing an estimation and planning process
- Institutionalise the commitment process
- Help each project to create a SW dev. plan with sufficient details
- Help project leaders to decide what project progress indicators to track

ID: Q1.197.0332 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debou

77

Effective Implementation of CMM L2 & L3

Coaching activities at level 2, ex (2)

- Coach QA and CM responsables in supporting projects
- Coach Project leaders in understanding the needs for QA and CM activities
- Support requirement management process including the CCB

ID: Q1.197.0332 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debou

78

Effective Implementation of CMM L2 & L3

Coaching activities at level 3

- Assist SW project team in the management of project interfaces within the project and across the organisation
- Support the identification and definition of training requirements for the organisation and projects
- Support the effective performance of peer reviews
- Support the definition of roles, responsibilities, methods, procedures and tools for each life cycle activity
- Facilitate the creation of the organisational standard software process from project best practices

Q-Labs

Magnus Ahlgren & Christoph Debou

79

Effective Implementation of CMM L2 & L3

Definition of process

- Level 2
 - Processes are defined at project level and are transferred to the next project in the same product line
 - If cost effective, common process assets may be developed
 - Focus is managerial processes
- Level 3
 - Common process assets are developed ex.
 - Common way of documenting processes
 - Best practices are identified and integrated
 - Focus is managerial and engineering processes

Q-Labs

Magnus Ahlgren & Christoph Debou

80

Level 3 processes assets

- Organisation's standard SW process including standards, procedures, templates, for all life cycle and support activities
- Architecture of process elements: interfaces, interdependencies
- Software life cycle
- Tailoring guidelines
- SW process database: actual measurement data
- Library of documents on past projects that can use as examples for future projects

15/CAL 17/0337 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debow

81

Follow-up assessments

- Perhaps the most important vehicle to monitor and put pressure on on-going improvement activities
- Focus is:
 - Evaluate the appropriateness of on-going and completed improvement actions w.r.t. initial findings/impact on the organisation
 - Identify new improvement actions in line with business goals
- Minimise disturbance in the organisation compared to initial assessment

15/CAL 17/0332 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debow

82

Effective Implementation of CMM L2 & L3

Follow-up assessments - approach

- Should be done and reported
 - per project
 - per product line
 - across organisation
- Results should be visibly displayed

IDG1197.0332 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debau

84

Effective Implementation of CMM L2 & L3

Follow-up assessments - method

- Follow-up assessments should contain the following components
 - CMM-based questionnaire for valid CMM level(s)
 - Feedback mechanism for consistency and training
 - Reporting
- Different methods exists
 - Motorola's
 - SEI's Interim Profile
 - Q-Labs

IDG1197.0332 A 02/06/97

Q-Labs

Magnus Ahlgren & Christoph Debau

84

Effective Implementation of CMM L2 & L3

Training and orientation

- Level 2
 - Training of staff is a key component for success
 - Example training
 - Coaching
 - CMM assessment
 - Specific training:
 - All project leaders in CMM
 - different roles in performing their role and interpreting CMM (ex. quality assurance and configuration management)
 - staff should receive orientation in CMM implementations
- Level 3
 - Training on processes will be co-ordinated across the organisation
 - Staff should receive orientation on how the process organisation at level 3 works
 - Example training
 - Coaching
 - CMM assessment

Q-Labs

Magnus Ahlgren & Christoph Debow

85

Effective Implementation of CMM L2 & L3

Management follow-up

- Communicate and reinforce the importance of the improvement effort
- Visible support (Budget, personal engagement, decisions, management by walking around)
- Continuous follow-up to ensure the SPI project is remaining focus
 - The improvement programme (and follow-up assessments) should be standard item on their agenda
 - Reward good role models

Q-Labs

Magnus Ahlgren & Christoph Debow

86

Management follow-up

- Level 2
 - Visible management follow up is essential, since process is a new ballgame
 - Visible comparison of and competition between project and product lines - reward success
 - Symbolic leadership is important in resource conflicts with projects
- Level 3
 - Management follow up is still an important issue
 - Some follow-up can be delegated to the SEPCG, since basic commitment to process should be installed
 - Management needs to oversee organisational changes at level 3
 - Reward sharing and usage of best practices

Q-Labs

Magnus Ahlgren & Christoph Debaux

87

Information

- Identify and use various means for communication
 - intranet
 - newsletter
 - meetings
 - bulletin boards
- Inform, inform, inform, e.g.
 - Plans and status in improvement activities
 - Result from CMM follow-up assessments
 - Changes in business figures
 - People, projects and product lines that gets awarded

Q-Labs

Magnus Ahlgren & Christoph Debaux

88

Effective Implementation of CMM L2 & L3

Improvement implementation - exercise

- Each participants gives Top 5 hinders for improvement implementation.

10/04/97 02:00:07 A

Q-Labs

Magnus Ahlgren & Christoph Debow

89

Effective Implementation of CMM L2 & L3

Effective implementation

- Supporting Level 2 implementation
 - Team based practices
 - Incremental development
- Supporting level 3 implementation
 - The experience factory concept

10/04/97 02:00:07 A

Q-Labs

Magnus Ahlgren & Christoph Debow

90

Team Definition

A team is a group of preferably 3-6 people that work together interdependently as a whole over a period of time to achieve a common goal, where the team members are jointly responsible for the result, and strives to maximise performance through innovative methods.



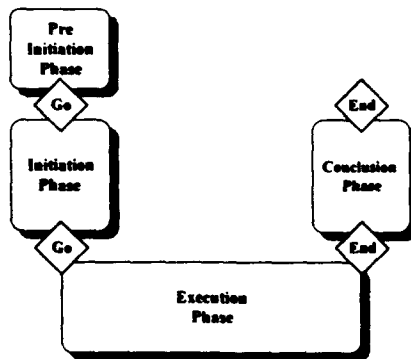
Team member
Team leader
External resource

Q-Labs

Magnus Ahlgren & Christoph Deibau

91

The Team Assignment Life Cycle



Q-Labs

Magnus Ahlgren & Christoph Deibau

92

Effective Implementation of CMM L2 & L3

The Team Assignment Life Cycle

Pre Initiation Phase

Team A: [REDACTED]

Team B: [REDACTED]

Team C: [REDACTED]

One Assignment:

Initiation Phase

Execution Phase

Conclusion Phase

Q-Labs

Magnus Ahlgren & Christoph Debow

93

Effective Implementation of CMM L2 & L3

Three main players in a Team Based Organisation

```

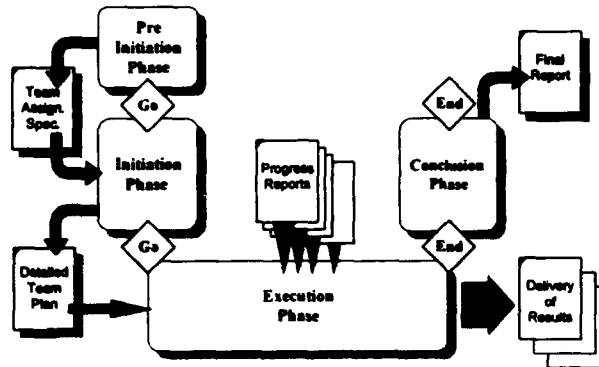
graph TD
    AO([Assignment Orderer]) <--> RR([Resource Responsible])
    AO <--> AP([Assignment Performer])
    RR <--> AP
    subgraph Labels
        PSM[Project sponsor ($) Project manager] --- AO
        LM[line manager] --- RR
        TDT[The team doing the task] --- AP
    end
        
```

Q-Labs

Magnus Ahlgren & Christoph Debow

94

The Team Assignment Life Cycle

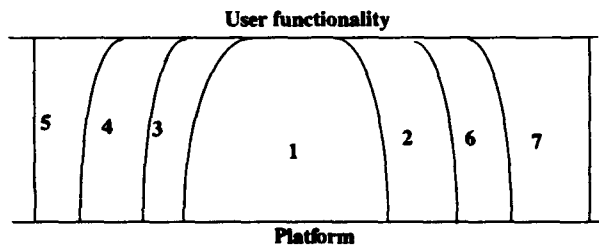


Q-Labs

Magnus Ahlgren & Christoph Debow

95

Incremental development



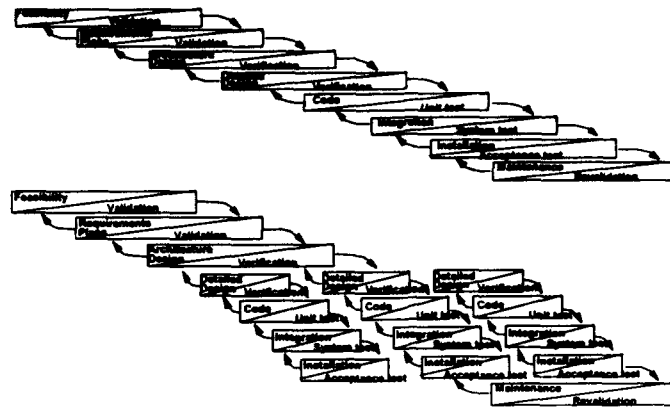
Each accumulation of developed increments is a complete user executable system

Q-Labs

Magnus Ahlgren & Christoph Debow

96

Incremental development



Q-Labs

Magnus Ahlgren & Christoph Debaux

97

Reasons for incremental development

- Lead-time—too long start up and completion phases
- More time for analysis, implementation and test
- Unstable or unknown requirements
- Smaller more manageable parts
- Early quality feedback
- No big bang integration
- Early test of platform and performance
- Enforces planning, co-ordination and follow up

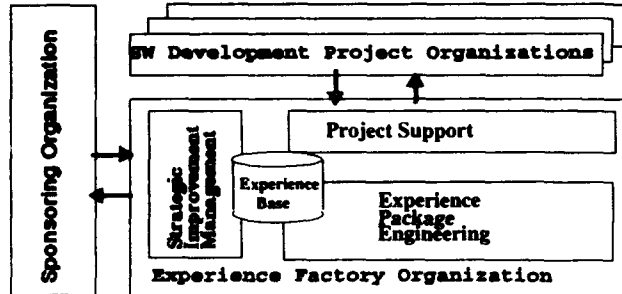
Q-Labs

Magnus Ahlgren & Christoph Debaux

98

Experience Factory Model

The EF organisational model reflects the need for extra-project resources (i.e. strategic improvement management, project support, and experience package engineering):

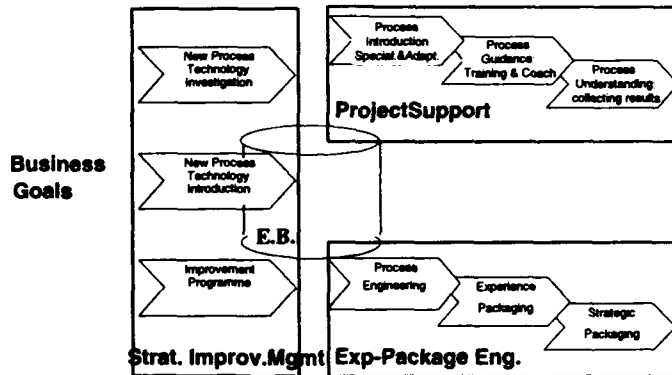


Q-Labs

Magnus Ahlgren & Christoph Debow

99

Experience Factory processes



Q-Labs

Magnus Ahlgren & Christoph Debow

100

EF contributions to CMM level 3

- Supporting projects
- Guiding and understanding process use
- Creating memory by structuring experiences
- Retrieving and storing experiences
- Assessing new technology

Q-Labs

Magnus Ahlgren & Christoph Debow

101

'Controlling Software Contracts'

Charles Symons
Guild of Independent Function Point Analysts
17th June 1997



Controlling Software Contracts
ESEPG June 97
© 1997 Copyright The Guild of Independent Function Point Analysts Ltd

1

We will aim to discuss:-

- The Customer and Supplier view of software contracts; how value is added
- Performance measures needed and performance trade-offs
- Conventional software contracting and the 'SCUD' process
- Some case studies
- Lessons for the Supplier; relevance to Software Process Improvement
- Lessons for the software-contracting Customer
- Challenges for the future



Controlling Software Contracts
ESEPG June 97
© 1997 Copyright The Guild of Independent Function Point Analysts Ltd

2

Customer and Supplier objectives overlap, but differ

Customer	Supplier
▪ Better price/performance than in-house – cost, speed, 'quality' – delegate risk ▪ Business understanding	▪ Profit ▪ Customer satisfaction ▪ Differentiate from other suppliers, eg – very low cost – move up value chain



But both parties need the partnership to succeed

GIFPA

Controlling Software Contracts
ESEPG June 97

© 1997 Copyright The Guild of Independent Financial Planners Ltd

3

The Supplier can add value at various levels

Application	Implementation	Business
<u>Functionality</u> • User Specified • Pre-packaged <u>Quality</u> • Ease of Learning • Ease of Use • Responsiveness • Accuracy • Portability • Maintainability	<u>Operational Extent</u> • Sites Accessible • Departments implemented • Users Supported • Extent of Business • Data Automated	<u>Value to Business</u> • Business Survival (<i>plus</i>) • Value of Business Process Supported (<i>plus</i>) • Value of Decisions supported (<i>minus</i>) • Cost of Ownership

Increasing added value →

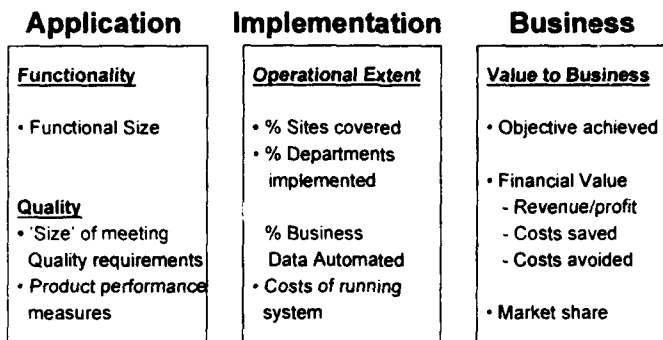
GIFPA

Controlling Software Contracts
ESEPG June 97

© 1997 Copyright The Guild of Independent Financial Planners Ltd

4

Measures of value vary correspondingly



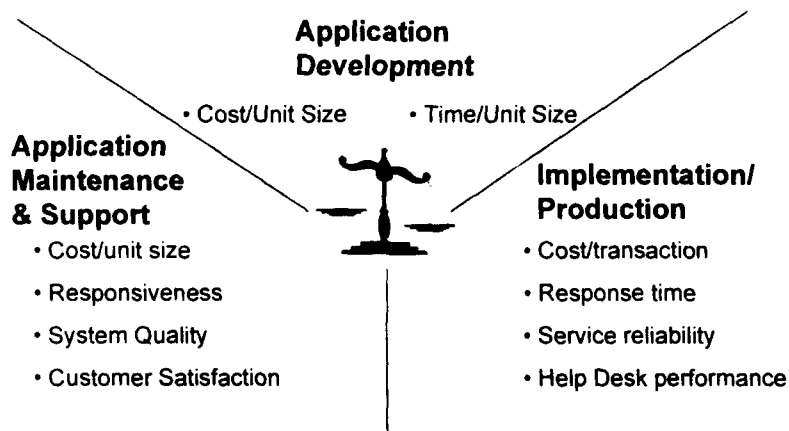
Increasing added value →

GIFPA

Controlling Software Contracts
ESEPG June 97
© 1997 Copyright The Guild of Independent Functional Point Analysts Ltd

5

Performance trade-offs must be understood



GIFPA

Controlling Software Contracts
ESEPG June 97
© 1997 Copyright The Guild of Independent Functional Point Analysts Ltd

6

Some measure of software size is essential for estimating and for measuring performance

Candidates:

- × Source Lines of Code
- ✓ 'IFPUG' Function Points
- ✓ 'MkII' Function Points
- ✓ Local FP-like or SLOC-based estimating rules
- ✓ Environment-specific 'Standard-Hour' formulae



Controlling Software Contracts
ESEPQ June 97

© 1997 Copyright The Centre of Independent Function Point Analysis Ltd

7

Are the measures adequate for the task?

Market Needs

(Examples)

- Supporting the \$3.2B EDS/Xerox outsourcing contract
- Helping resolve a £xM out-of-court settlement of a software dispute

vs

Current Function Point Measures

- Accuracy known?
- Subjective rules
- Theoretical basis? (IFPUG)
- Effort and time to collect data
- Limited to 'data-rich' software

Just about.....within limits.....with care!



Controlling Software Contracts
ESEPQ June 97

© 1997 Copyright The Centre of Independent Function Point Analysis Ltd

8

Conventional software contracting is difficult for both Customer and Supplier (1)



1. Customer Task:

- Customer effort, time, skill to define Requets. and ITT
- Supplier(s) cost, time to bid for Development
- Supplier(s) cannot add much value

GIFPA

Controlling Software Contracts
ESEPQ June 97
© 1997 Copyright The Guild of Independent Function Point Analysts Ltd

9

Conventional software contracting is difficult for both Customer and Supplier (2)



2. Supplier Task:

- Customer selects Supplier on basis of limited data
- Difficult for Customer to control Supplier
- (Preferred scenario for Supplier)

GIFPA

Controlling Software Contracts
ESEPQ June 97
© 1997 Copyright The Guild of Independent Function Point Analysts Ltd

10

**The 'SCUD' process offers hope.
('Software Charged by Unit Delivered')**

Feasibility	Requirements Determination	Design, Build, Test, etc
--------------------	-----------------------------------	---------------------------------

- Customer issues outline requirements; invites bids on basis of 'Price/FP'
- Supplier selected on price and other factors
- Customer pays on FP's delivered (counted by an independent expert)
- Process saves considerable time and cost for both parties

GIFPA

Controlling Software Contracts
ESEPG June 97

© 1997 Copyright The Guild of Independent Function Point Analysts Ltd

11

But use of 'SCUD' needs great care



- Balanced performance measures, not just 'price/FP'
- Understanding FP limitations, eg complex rules
- Unstable requirements; controlling changes
- Supplier bids assuming a particular package; detailed analysis shows it does not meet the needs

**Process pioneered in Australia (Govt. of Victoria)
Introduced in US and UK Outsourcing contracts**

GIFPA

Controlling Software Contracts
ESEPG June 97

© 1997 Copyright The Guild of Independent Function Point Analysts Ltd

12

Case 1: Outsourced Development

**UK
Retailer**

- Benchmarked development productivity was one-third industry average
- Negotiating target was for supplier to get to 'upper quartile' productivity within 4 years
- Subsequent analysis showed speed of delivery had been 2 - 3 times industry-average
- Negotiating targets re-focused on balanced performance measures to maximise business benefit

GIFPA

Controlling Software Contracts
ESEPG June 97

© 1997 Copyright The Guild of Independent Function Point Analysts Ltd

13

Case 2: Outsourced Software Maintenance (1)

UK Utility

"We used to have them on fixed price, and we could never find them. Now we have them on Time & Materials, and we can't get rid of them"

Solution: Develop local 'standard-hour' estimating matrices for various types of work. All tasks controlled by users

GIFPA

Controlling Software Contracts
ESEPG June 97

© 1997 Copyright The Guild of Independent Function Point Analysts Ltd

14

Case 3: Outsourced Software Maintenance & Support (2)

**Multinational
Manufacturer**

- World-wide legacy portfolio & staff outsourced
- Portfolio FP size determined in 10 weeks by external experts (careful sampling and approximation)
- Price is based on \$/FP supported, using external benchmarks - seems to work OK

GIPPA

Controlling Software Contracts
ESEPO June 97

© 1997 Copyright The Guild of Independent Front-End Analysts Ltd

15

Case 4: Command & Control Software Supplier

- Invitations to Tender are bulky, but of uneven detail
- Solutions will be mostly 'COTS + glue', at multiple levels
- Desperate need of Supplier for early means of estimating approximate size of requirement
- Solutions:
 - COTS-specific rough estimating rules
 - Local 'standard-hours'- based measures for specific tasks
 - FP's useful for some parts of solution

GIPPA

Controlling Software Contracts
ESEPO June 97

© 1997 Copyright The Guild of Independent Front-End Analysts Ltd

16

Case 5: A Legal Precedent - Beware!

UK Local Government

- £1.3M contract, but defects cost Customer an additional £1.3M
- Court ruled that software is 'goods', and must be 'fit for purpose'
- Cost to supplier £12M compensation; bad publicity
- Causes:
 - poor definition of requirements
 - technical staff not properly involved
 - poor estimation and feedback



Controlling Software Contracts
ESEPG June 97

© 1997 Copyright The Guild of Independent Function Point Analysts Ltd

17

Lessons for Suppliers

- Life has been too easy
- Customers are becoming more sophisticated and litigious
- Building experience in software metrics and estimating is a matter of survival
- Understand how your customer measures Value



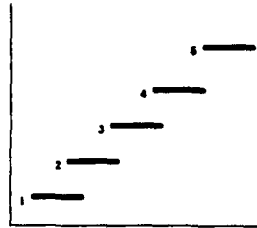
Controlling Software Contracts
ESEPG June 97

© 1997 Copyright The Guild of Independent Function Point Analysts Ltd

18

Relevance to Software Process Improvement

- The SEI CMM requires mastery of software metrics at Level 4
- Start at Level 2 when you have repeatable processes
- Apply measures to re-usable products (COTS, objects)



GIFPA

Controlling Software Contracts
ESEPG June 97

© 1997 Copyright The Guild of Independent Foreign Patent Analysts Ltd

19

Lessons for Customers

- Keep strong contract management skills in-house
- Need to master performance metrics to ensure Value for Money (measurement can be outsourced)
- The 'SCUD' method is attractive - but use with care!
- Understand your suppliers; recognise their needs



GIFPA

Controlling Software Contracts
ESEPG June 97

© 1997 Copyright The Guild of Independent Foreign Patent Analysts Ltd

20



Challenges for all Software 'Engineers'!

- Existing software estimating and performance measures are only just adequate for the needs of business software contracting
- Improvements needed:
 - Sound theoretical basis, aligned with modern software development concepts, valid for all domains
 - Build on Industrial Engineering experience of work measurement
 - Integrated cycle : Estimate - measure - refine (re-calibrate), etc

GIFPA

Contracting Software Contracts
ESEPQ June 97

© 1997 Copyright The Guild of Independent Project Management Analysts Ltd

21

CONTROLLING SOFTWARE CONTRACTS

Charles Symons

Director

Guild of Independent Function Point Analysts

*(Paper presented at the European SEPG Conference, Amsterdam, June 1997.
Numbers refer to the slides in the presentation of the same title.)*

1. Introduction

Competitive pressures in the private sector and cost pressures in the public sector are forcing more and more software development and maintenance to be contracted out, either via single contracts, or by outsourcing all or part of the services. Yet the means of controlling these contracts, particularly for the customer, and even to some extent for the supplier, are still relatively crude and not well understood.

This paper explores the critical issues of how software contracts can be controlled, and the difficulties involved from the customer and supplier viewpoints. Customer and supplier objectives are examined, and ways in which the supplier can add value. The types of performance measures needed are described, and the importance of performance trade-offs is discussed. The difficulties of conventional software contracting processes are described, as well as a new 'SCUD' process which appears to overcome some of the difficulties. Finally, several case studies are used to illustrate the main messages of this paper, and some concluding lessons are drawn for suppliers, customers, and the Software Engineering community at large.

(Note that the focus of this paper is on controlling software contracts from a performance measurement viewpoint, with an emphasis on protection of the customer's interests. There are many other contractual issues which need to be addressed when outsourcing the supply of software, such as ownership of intellectual property rights, protection of the rights of staff transferred, retention of key skills, contract termination, etc., etc. These no less important issues are beyond the scope of this paper.)

2. Customer and supplier objectives

If organisations choose to buy software externally rather than develop in-house, the minimum they will expect from their supplier is a service at a better price-performance ratio than they could achieve with their own resources. Unfortunately, few software customers have a clear idea what they want in terms of 'performance'. They will certainly want lower costs, but they may also want faster delivery, better 'quality' of the delivered software, which could mean greater flexibility, greater ease of use, or less defects than would be expected from an in-house solution. Increasingly there is an expectation of off-loading development risk to the supplier. And the customer will expect the supplier to demonstrate some track-record of understanding the customer's business.

These are mixed and demanding objectives, and not often well thought out. For example, they are not often prioritised.

In contrast, the supplier's objectives are generally clearer. First, he needs to make a profit. Second, he will want to satisfy the customer, for a satisfied customer will be the best source of repeat business, and failure to satisfy could be extremely expensive. If he is bidding competitively, the supplier will also want to differentiate himself in some way from the competition. This might be achieved by being a low-cost supplier, by offering some unique deal or, increasingly, by trying to move up the customer's value chain.

Customer and supplier objectives therefore differ in various ways, but they also overlap in that both want to achieve customer satisfaction. This is commonly expressed in both parties wanting a 'partnership' that must be made to work. However, if one party does not behave as expected, or becomes dominant, the relationship is unlikely to be satisfactory to the other party. Software contracting is therefore successful if both parties 'win' in achieving all their objectives. But the supplier starts out with the distinct advantage that software contracting is his business. By comparison, most customers are relatively inexperienced. True partnership is therefore a difficult balance to achieve.

3. How software suppliers 'add value'

When software is supplied, value is added at three levels (Slide 4)

- Application software adds value by providing information-processing functionality. This may be specified by the user, or come pre-packaged from the supplier. The customer will also specify quality requirements - ISO standard 9126 provides a good list. The more demanding these are, the greater the value added in achieving them.
- When the application software is implemented in production on an infrastructure, value is further added in varying degrees, for example in the extent to which the system is accessible and the extent of data implemented and, of course, by the routine processing of the application.
- All the software development and processing ultimately adds value by enabling achievement of business objectives, such as helping to lower costs, or making better decisions which provide competitive advantage. The net value accrued at this level is the benefits obtained by the business, offset by the costs of ownership of the application.

For each of these levels, we need different types of measures to determine the 'value added' (Slide 5). At the Application level, the 'size' of the functionality delivered is a primary measure of value, but other measures, for example of the various quality attributes of the software product, are also needed to get a full picture. Measurement of value added at the Implementation level is fairly obvious and straightforward. At the Business level, on the other hand, whilst measures of business performance are well-understood, disentangling the contribution of the software is often difficult.

It is important to recognise that software suppliers are increasingly trying to move up the customer's value chain, in order to grow their business and to get a greater grip on their customers. Hence some major outsourcing suppliers are proposing contracts in which they are paid on the basis of the improvement in business performance arising from their services, as opposed to the more classic method of being paid for the software delivered. This seems to work satisfactorily when the customer wants to outsource a whole business process where information technology plays a major role, and the supplier is paid on, say, a cost per transaction processed. But the more ambitious schemes of aiming to be paid on the general improvement in profitability arising from development and implementation of improved systems seem fraught with difficulty. A general down-turn in prices, for example due to increased competition, could seriously undermine profitability, no matter how much had been invested in improved systems.

From hereon, we will concentrate on the classic task of controlling software contracts at the Application level, with some reference to the interaction with the Implementation level. Customers should be aware, however, of the ambitions of the outsourcing supply industry, and of the greater difficulties of controlling value for money at these higher levels of added value.

4. Software contracting performance measures and trade-offs

Some of the most important performance measures needed at the Application and Production levels are shown in Slide 6. The vital point here is that significant trade-offs in performance are possible, yet customer organisations rarely have the data available to make informed choices, and may even be unaware of the possibilities and risks.

Perhaps the commonest customer failing when negotiating a software contract is to concentrate on the immediate development cost, whilst ignoring the fact that the life-time cost of ownership of a system is dominated by the production and on-going maintenance and support costs. Alternatively, a business may be prepared to pay more for a development if the system can be delivered very quickly to provide competitive advantage. Effort (and hence cost) and time are tradable to a considerable extent. Unfortunately there are conflicting views in the industry on the best description of the trade-off relationship.

The golden rule for the customer is to define all his performance objectives, prioritise them, and then seek to define the set of performance goals which provides the best balance for meeting those objectives. It is very unlikely that pulling hard on just one performance lever will be enough to achieve all your objectives.

To be able to do this properly requires the customer organisation to have built up an understanding of the factors which influence performance in software delivery, and how they interact. This takes time, and is knowledge which even major software suppliers do not always seem to possess in all parts of their organisation.

Any software measurement programme must be founded on measures of software size. The first question asked in estimating any development effort is 'how big is it?'

And software size is the key component of performance measures such as 'productivity' (size/effort), 'delivery rate' (size/elapsed time), etc. There are several choices for this measure. (Slide 7)

- Source Lines of Code ('SLOC') are still very widely used, although they suffer from several well-known deficiencies. Most important of these is that the actual number of SLOC is not known until the software is developed. SLOC can therefore only be used in estimating if they can be predicted from another measure obtained much earlier in the software life-cycle
- Function Point Analysis ('FPA') was first proposed by Allan Albrecht of IBM in the late 1970's to overcome the main weaknesses of SLOC. He developed a composite index of counts of the functions required (inputs, outputs, inquiries, logical files and interfaces), and of the degree of influence of some 14 quality and technical requirements. The definition of the resulting index in units of 'Function Points' continues to be refined by the International Function Point User Group. IFPUG Function Points have become the most widely adopted measure of software size in the business information systems world.
- 'MkII' Function Points were developed by the present author in the late 1980's to overcome certain perceived weaknesses in the IFPUG index, including basing the size measure on concepts which had meantime come into use in requirements specifications, namely logical transactions and entities. The MkII index is also very easily adapted to apply to object-oriented models. As well as offering a means of software sizing, the method also has an integrated estimating method.
- Both Function Point methods evolved out of the business information systems world, and are mainly used for so-called 'data-rich' software. But conventional FPA does not work well for software whose characteristics are dominated by, for example, complex functionality (e.g. as in scientific and engineering programs, rule-based systems, operating systems, etc.), or which have major real-time constraints, such as in telephony software. In these software domains, SLOC are commonly used as the measure of size. For estimating, therefore, many local FP-like sizing methods have been developed to enable early prediction of SLOC.
- Finally, some development organisations have recognised the applicability of the concepts of 'standard-hours' from the world of Industrial Engineering, as a means of measuring the work associated with software development tasks or deliverables. These measurements allow monitoring of, for example, productivity 'against standard', and the measures can be used for estimating.

How good are these measures? The answer is that judged against the needs of the software contracting industry, they are not as good we would all like them to be. But they are all we have. (Slide 8)

As an illustration, multi-billion dollar software outsourcing contracts are being controlled, and legal disputes are being resolved, on the basis of Function Point measures. Yet these measures are known to have certain deficiencies and do not work reliably in all software domains. Used with great care, they can give adequate means

GITTS, 1992.

of estimating and control, at least in the business software world. But clearly there is a need for better measures.

5. Conventional software contracting and the 'SCUD' process

There are two approaches to contracting software which are commonly followed (Slides 9 and 10). Neither is particularly satisfactory, especially from the customer viewpoint.

In the first approach, the customer defines his requirements in detail, and then draws up an Invitation to Tender ('ITT'), which is sent to prospective suppliers. The suppliers then have to study the ITT, develop their estimates, and write detailed proposals. This process is time-consuming and expensive for both parties. Preparing a detailed statement of requirements and an ITT requires considerable skill, and if potential suppliers have not been involved, good ideas may have been missed. The suppliers are constrained by the ITT, and may have limited opportunity to differentiate themselves and add value.

An alternative process, therefore, is for the customer to issue only an outline statement of requirements. Suppliers may be asked to bid to complete the detailed requirements (perhaps for a fixed price), with an indicative price to complete the development subsequently. This process has the advantage of requiring less time and effort for both the customer and suppliers through the bidding phase, and it allows the chosen supplier to bring his experience to contribute to the definition of the system.

However, the customer has less control in this process. Once the supplier is installed and is helping to shape the system definition, the customer's bargaining position is considerably weakened, if subsequently the price to complete the system seems to have risen above that which was indicated at the time of the initial bid. Where this process is followed and the supplier is permanently installed, as in an outsourcing contract, the process is even less satisfactory for the customer. For this process to be satisfactory, the customer needs other performance measures than just fixed or indicative prices.

A process which overcomes the sort of weaknesses described above has emerged recently in Australia, and is finding its way into outsourcing contracts. It is known as the 'SCUD' or 'Software Charged by Unit Delivered' process (Slides 11 and 12).

Here, as in the second process described above, the customer issues an ITT with only an outline statement of requirements. The ITT requires, however, that the suppliers bid to complete the system through all its phases, at a fixed price per Function Point, eg in units of \$/FP. The preferred supplier is selected on the basis of his quoted unit price in \$/FP, and of course the usual other factors.

As the detailed requirements are developed by the customer and the chosen supplier, the first 'Baseline Function Point Count' is established. This, combined with the agreed unit price, enables the customer to decide if the overall cost is going to be affordable. When the scope is decided, development can proceed. The final price paid is determined by a final FP count for the delivered system, and the quoted unit

price in \$/FP. An independent third party is employed by both the customer and supplier to determine the FP counts at each stage, and the impact of changes made.

This process appears to offer distinct advantages to both the customer and potential suppliers. The time and effort required for the bidding round is much reduced, and the customer has the assurance from the start that he is getting a good market unit price (it can be compared against benchmarks). The customer has the possibility of controlling the overall price to be paid by keeping a tight hold on the scope of his requirements, and the process moves much of the risk from customer to supplier.

As with most such processes, it appears deceptively simple at first sight. In practice, both customer and supplier need to be aware of potential pitfalls, and to manage the process with care. The most obvious point, as has been discussed above, is that other performance measures must be agreed in addition to the basic \$/FP, and the limitations of these measures must be understood. For example, if the system has complex processing rules, which are not properly reflected in the FP measure of size, then the \$/FP and the project estimates must be compensated to allow for this.

Managing changes could also become problematic. If the customer is very indecisive, and introduces changes more often than the supplier allowed for, then there is potential for conflict. In an extreme situation, one can imagine a supplier quoting his \$/FP on the assumption that a particular package will meet the requirements. On more detailed examination, it could turn out that the package will not meet the requirements, and another solution at an entirely different \$/FP might be needed.

All these potential problems can be overcome if considered carefully in advance, but the process needs to be managed with care. The process is being used by the Government of the State of Victoria in Australia to procure software, and is being introduced into outsourcing contracts in the USA and in the UK.

6. Some Case Studies

6.1 Outsourced Development

A UK retailer was negotiating the outsourcing of all its application development and maintenance services. A major goal was to reduce costs. Targets were therefore set for the bidders that within a given period they should achieve 'upper quartile' performance in application development productivity, measured in FP/man-months, according to a particular benchmarking service. The retailer had had no previous experience of measuring performance measurement in software activities.

In order to establish a baseline, some measurements of current productivity were made. These showed, for the small sample of projects, that productivity was around one-third of industry-average. This result was an unpleasant revelation to the retailer, but looked at positively, it indicated the scale of performance improvement and cost-reduction which should be achievable from outsourcing.

As a check, we asked that the 'Delivery Rate', in FP/elapsed week, be measured for these same projects. This performance parameter showed that the speed of delivery for these same projects had been very much *higher* than industry-average. It turned out that the projects used for the measurement sample, had been set the objective to deliver as fast as possible, no matter what the cost, as the resulting systems were essential for competitive advantage. Speed of delivery was therefore very high, which was achieved by pouring resources into the project. Hence productivity was low, and the quality of the delivered software was also low due to the high speed of development. But the business goals and benefits were achieved as planned, and the projects were a great success from a business viewpoint.

If the outsourcing contract performance goals had been defined only on the basis of development productivity, then the retailer could have been bound into a situation where the supplier could meet his goals quite easily, but the retailer would suffer from lengthier development times and less business flexibility. This case illustrates the importance of understanding performance trade-offs, and the need to establish a balanced set of measures in line with business goals.

6.2 Outsourced Software Maintenance

The IT Director of a UK Utility Company expressed his frustration with the service he was getting from his supplier of outsourced application maintenance services in the memorable words:

"We used to have them on fixed price, and we could never find them. Now we have them on Time and Materials, and we can't get rid of them."

These words illustrate perfectly the difficulties of controlling on input measures of resources consumed, rather than on output measures of work delivered. The answer was to develop some simple estimating formulae, specific to the local applications and environment, by which customer and supplier could rapidly agree the cost of any standard task.

Types of standard maintenance tasks were defined at various levels of complexity. The latter was expressed in, for example, the number and size of files or screen fields that had to be changed. The effort required to complete each standard task was established in units of 'standard-hours'. So when any user needed a maintenance change, a cost could be given according to the agreed formulae. Performance improvement targets could be set for the supplier, in terms of target reductions in the standard-hours for specified tasks, and hence also reductions in costs.

6.3 Outsourced software maintenance and support

In this case a major multinational manufacturer outsourced his world-wide legacy application maintenance and support to a single supplier. The key performance parameter for payments was agreed to be \$/FP supported. This implied measurement of the size of the portfolio of some hundreds of thousands of FP's, a formidable task. The problem was solved by a combination of very careful sampling of the portfolio,

and the use of approximation techniques of FP sizing, which enabled the task to be completed within acceptable timescales, cost and accuracy.

The contract is now understood to be working smoothly. Independent benchmarks are used periodically to establish external performance comparators and, with experience, other performance measures have been added, notably product quality measures.

6.4 A command and control software supplier

This supplier of military command and control systems faces severe estimating and bidding challenges. ITT's are voluminous, but of uneven detail. Some requirements are specified at a low level of detail, whilst other simple statements may cover a lot of complex functionality. Increasingly, the solutions to such requirements are built mainly from 'COTS' (or Commercial Off-The Shelf) software from various sources, and other re-usable code. The supplier's task is to provide the 'glue' software to bind the COTS software into a coherent system. This may be required at various 'levels', for example at the man-machine interface level, the application level, the middleware, and the operating systems of clients and servers.

It is clear that a variety of environment and COTS-specific estimating and performance measurement methods are needed. A particular challenge when an ITT is received is to form a view very rapidly on the size of the contract on offer, so that the bid strategy can be determined. This requires simple estimating 'rules of thumb' based on parameters such as the number of major entities or entity-groups, and COTS components required. These can be developed by keeping and analysing records of previous bids and projects, both of components delivered and of the effort involved. FPA may be valuable for certain components, for example where bespoke software is required.

6.5 A legal precedent

A recent case in the UK of a dispute between a customer and software supplier which was resolved in court, provides a number of lessons and set an important legal precedent. The case concerned a Local Government organisation, which had commissioned software at a cost of £1.3M to handle a new local tax. The software proved unreliable, and as a result the customer's costs doubled.

The court determined that software is 'goods' (that is, not the result of services provided) and hence has to be 'fit for purpose'. This means that it is the supplier's responsibility to ensure that the software 'behaves as advertised'. The supplier lost the case, had to pay substantial damages, and suffered adverse publicity. It turned out that the contract had been managed badly from several points of view (Slide 17), including a poor definition of requirements and poor estimating.

7. Some lessons and conclusions

In general, this area of performance measurement to help control software contracts, especially outsourcing contracts, is rather under-developed. A combination of customer inexperience and weaknesses of the performance control methods has probably contributed to the seeming acceptance that budget and cost over-runs, and poor quality products are commonplace in the software supply industry.

But the game is changing. Suppliers are probably aware that customers are becoming more sophisticated, and certainly are more inclined to go to law or arbitration to settle disputes. So suppliers must build their experience in software performance measurement and estimating. (This seems to be an under-developed subject for even some of the best-known names in the industry!) As the software world becomes more competitive and professional, good estimating will become a matter of survival.

TEI's Capability Maturity Model requires mastery of software metrics only at Level 4, the 'Managed Process' level. But it may take years to develop experience in performance measurement and to build a base of measures sufficient to help improve performance and estimating. Software producers should certainly have introduced performance measurement by the time they have reached maturity Level 2, the 'Repeatable Process' level. By definition, if the process is repeatable, then past performance measures can be used reliably for future estimating, and can be applied to re-usable software components.

Although customers may feel they can get overall better Value for Money by contracting out or outsourcing their software development, there are limits to what responsibilities can be shifted to the supplier, even with the most constructive partnership. Above all, customers must retain the ability to manage contracts and suppliers. They must also at least understand the subject of performance measurement if they are to retain control. (The measurement work itself can be sub-contracted to independent experts.) With this understanding, and an understanding of supplier objectives and needs, they can take on methods such as 'SCUD' for software contracting with confidence.

Finally, there are lessons for the whole software engineering community. The methods of measuring software are only just adequate for controlling certain types of development and maintenance activities. By comparison with the advances made in software engineering generally, the subject of software metrics makes progress at snail's pace.

There is a real need for improved software metrics, compatible with modern software development methods, and with a sound theoretical basis. The metrics must be capable of continuous improvement via the 'estimate - measure - refine' feedback cycle. They must be made to work in a coherent manner across all software domains. Undoubtedly there is much to learn from ideas on work measurement in Industrial Engineering - provided you have repeatable processes! This is one of the biggest challenges we face, if we are to justify the title of 'Software Engineers'.



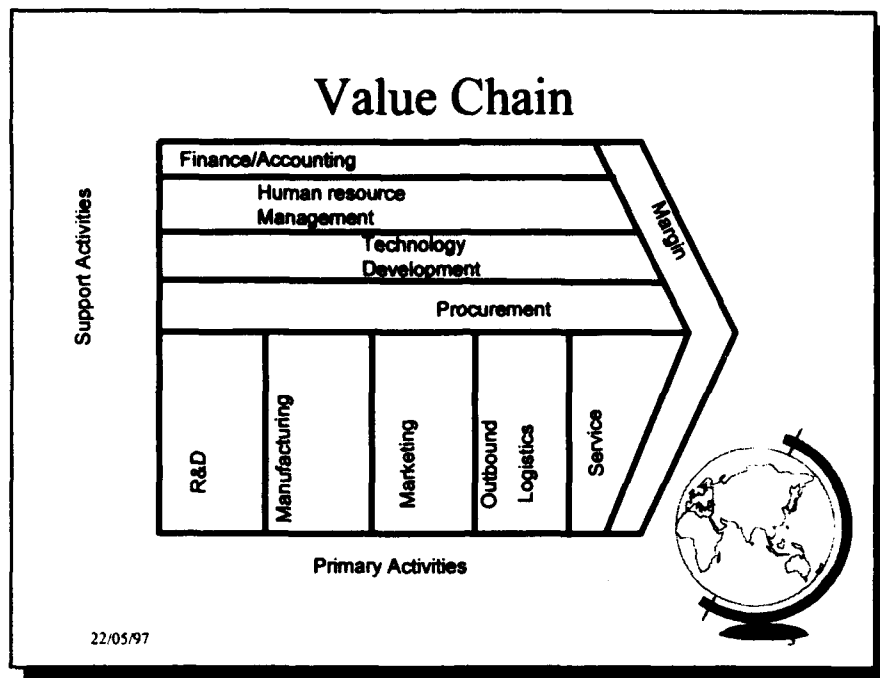
Metrics in Small Companies

Coupling a Metrics Programme to
your Business Model

Value System



22/05/91



Value Chain(Porter)

♦ The Customer value is defined by the following equation:

$$V=(Q+F)/P \quad \text{where}$$

- ♦ V = value to the Customer
- ♦ Q = perceived Quality
- ♦ F = features being valued
- ♦ P =Price to be paid

22/05/97

Preferences(1)

- ♦ EI: Extroversion or Introversion
 - To focus the dominant (favourite) process on the outer world or on the world of ideas
- ♦ SN: Sensing or Intuition
 - To use one kind of perception instead of the other when either could be used



22/05/97

Preferences(2)

- ♦ TF: Thinking or Feeling
 - To use one kind of judgement instead of the other when either could be used
- ♦ JP: Judgement or Perception
 - To use the judging or the perceptive attitude for dealing with the outer world



22/05/97

Distribution of Temperaments

♦ Guardian	(SJ)	40-45%
♦ Artisan	(SP)	35-40%
♦ Idealist	(NF)	8-9%
♦ Rational	(NT)	6-7%



22/05/97

Distribution of Types within Occupational and Academic Group

Occupations	ST %	SF %	NF %	NT %
Accountants	<u>64</u>	23	4	9
Bank employees	<u>47</u>	24	11	18
Sales, customer relations	11	<u>81</u>	8	0
Creative writers	12	0	<u>65</u>	23
Research scientists	0	0	23	<u>77</u>
Fields of Graduate Studies				
Theology	3	15	<u>67</u>	25
Law	31	10	17	<u>42</u>
Fields of College Studies				
Finance & Commerce	<u>51</u>	21	10	18
Nursing	15	<u>44</u>	34	7
Counselling	6	9	<u>76</u>	9
Science	12	5	26	<u>67</u>
Health related professions	13	36	<u>44</u>	7
Education	13	<u>42</u>	39	6
Journalism	15	23	<u>42</u>	20
P.E. and Health	32	34	24	10

22/05/97

Improvement Opportunities

- ◆ Increase Q
 - CMM, TQM, PSP
 - by improved documentation
- ◆ Increase F
 - CMM, PSP
- ◆ Decrease P
 - subcontracting.
 - ◆ downstream activities like maintenance and services.



22/05/97

Quality

- ◆ Quality which is obliged
 - minimal, legally
- ◆ Quality which should
 - which the Customer expects
- ◆ Quality which is possible
 - the Quality the Customer values



22/05/97

Quality in service

♦ Reasons why Customers change Vendors or service organisations

- die, or retire 1%
- change job or location 3%
- give assignment to friends or acquaintance 5%
- tries competition 9%
- is dissatisfied with product or service 14%
- changes due to indifferent attitude of supplier 68%



22/05/97

Quality of Documentation

♦ Ability of Documentation on Customer Perception of Product Quality

– IEEE transactions on professional Communication
Volume 36, number 3, 1996.

- performance 73.19 % correct
- usability 72.25
- maintainability 70.33
- capability 69.23
- installability 68.82
- reliability 64.76



22/05/97

Finally

- ◆ set your priorities
- ◆ metricate
 - GQM
 - AMI
 - ◆ assess
 - ◆ analyse
 - ◆ metricate
 - ◆ improve



22/05/97

Summary

- ◆ The value chain expresses value in terms of perceived Quality; features being valued, and price.
- ◆ Perception and value is different for many people.
- ◆ Remaining aware of this, keeps you closely to your Market.



22/05/97

Quantitative Management of Software Process Improvement

Christof Ebert

European
SEPG'97

Alcatel Telecom, Switching Systems Division, SEPG, Francis-Willemsplein 1, B-2018 Antwerp, Belgium, e-mail: christof.ebert@alcatel.be

SMD - S12 SEPG
European SEPG '97

23.04.97 Slide 1

Environment

▼ Alcatel Telecom

- One of the world's leading suppliers in telecommunications equipment

▼ Alcatel 1000 S12

- Digital switching system
- Used in over 60 countries with over 110 Mio digital lines and wide range of functionality (small local exchanges, transit and international exchanges, IN)

▼ Architecture

- Distributed processors and real-time software
- Realized in CHILL, C, Assembler
- Share of software is currently in the range of 80 %

SMD - S12 SEPG
European SEPG '97

23.04.97 Slide 1

Tuesday 17 June

(T201a-4) S-1

Motivation

CMM Level	Requirements	Design	Coding	Module Test	Integration + Syst. Test	Field
Defined 3	2%	5%	28%	30%	30%	<5% 2F/KLOC
Repeatable 2	1%	2%	7%	30%	80%	10% 3F/KLOC
Initial 1	0%	0%	5%	15%	80%	15% 5F/KLOC

Source: Siemens '95, Jones '96, own data

SMD - S12 MEPG
European SEPUG '97

23.04.97 Slide 3

Internal Processes vs. Customer Viewpoint

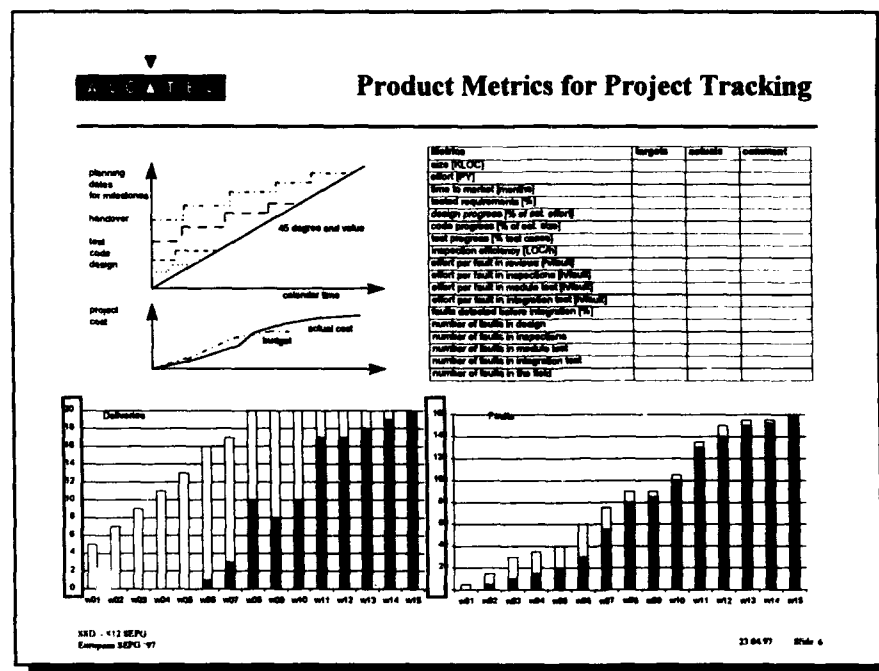
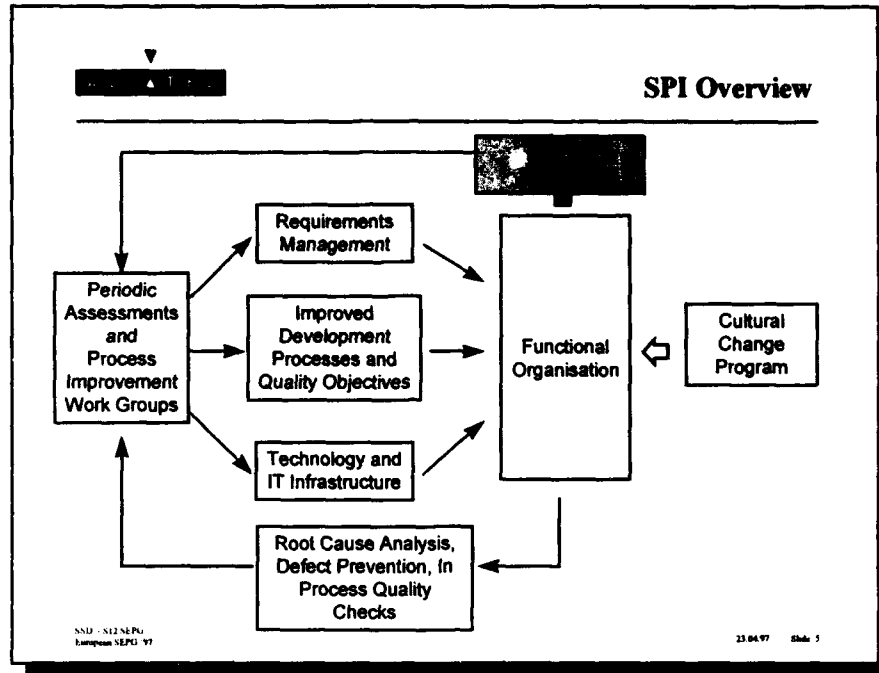
	Quality	Productivity	Deadlines	Employees
Internal process view	fault rate fault density cost per fault root causes	FP / person year FP / calendar month tool usage	percentage of work products within the 10% time frame	skills willingness overtime absence
External customer view	customer satisfaction delivered product quality functionality	cost per feature	delivery accuracy of final product to contracted date	satisfaction with contact persons (sales, after sales, engineers)

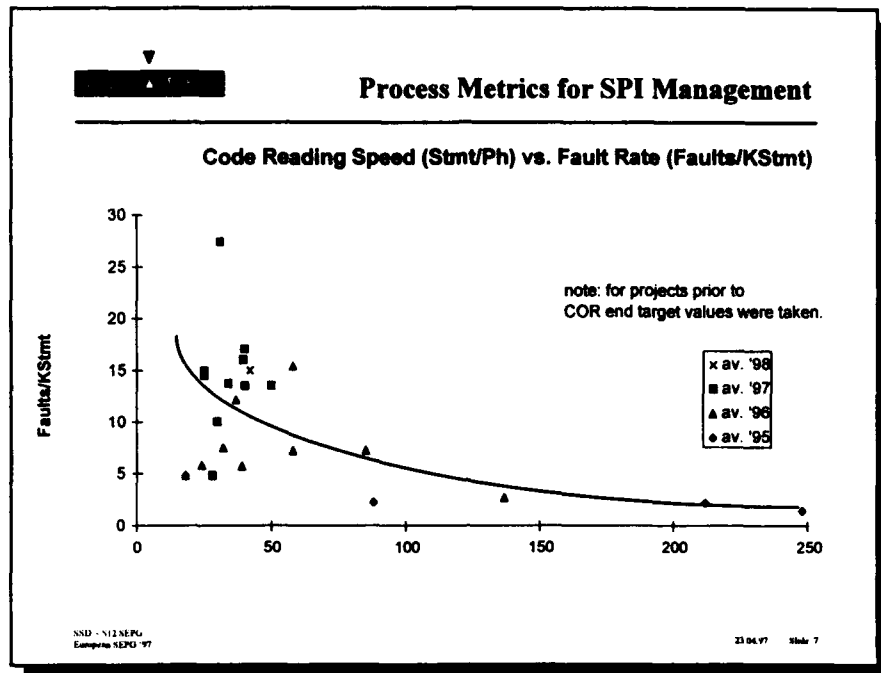
SMD - S12 MEPG
European SEPUG '97

23.04.97 Slide 4

Tuesday 17 June

(T201a-4) S-2





Evolution of a Metric Program

Year 0	- Assessment and setup of SPI program - Justification and outline of metrics program - Simple metrics for cost and quality baselines - Fault flow, fault reduction, reliability models
Year 1	- Effort estimation - Efficiency improvement - Customer satisfaction
Year 2	- Activity based costing, productivity, value creation - Benchmarking - Employee satisfaction, morale, motivation - Metrics for reuse and maintainability

SND - V12 SEP97
Empress SEP97 '97

23 04 97 Slide 8

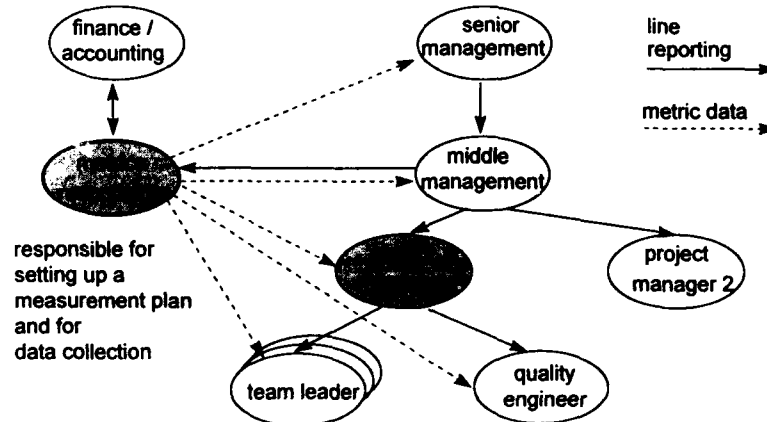
Metrics Teams and Metrics Responsibilities

- ▼ Roles and responsibilities of a measurement program must be well-defined
- ▼ Independent metrics teams are created to support all metrics related activities in a location
 - Training and coaching of management and practitioners
 - Alignment of tools or forms for data collection and analysis
 - Rationalisation and standardisation
 - Creation of a history database
- ▼ In addition the metrics responsible of each project ensures that the metric program is implemented and understood
 - Support for data collection across functions
 - Data analysis

XSD - S12 SEPJ
European SEPJ 97

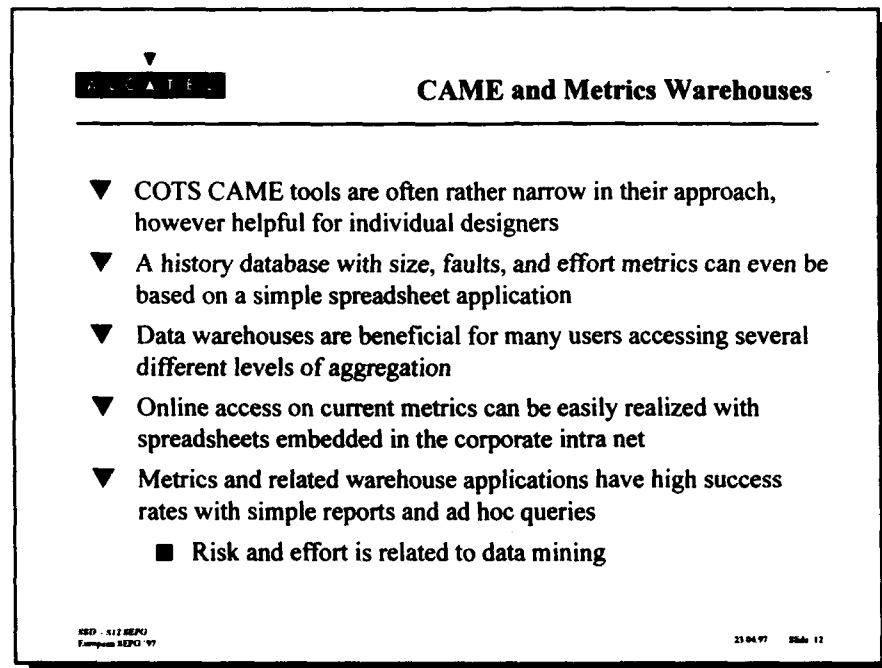
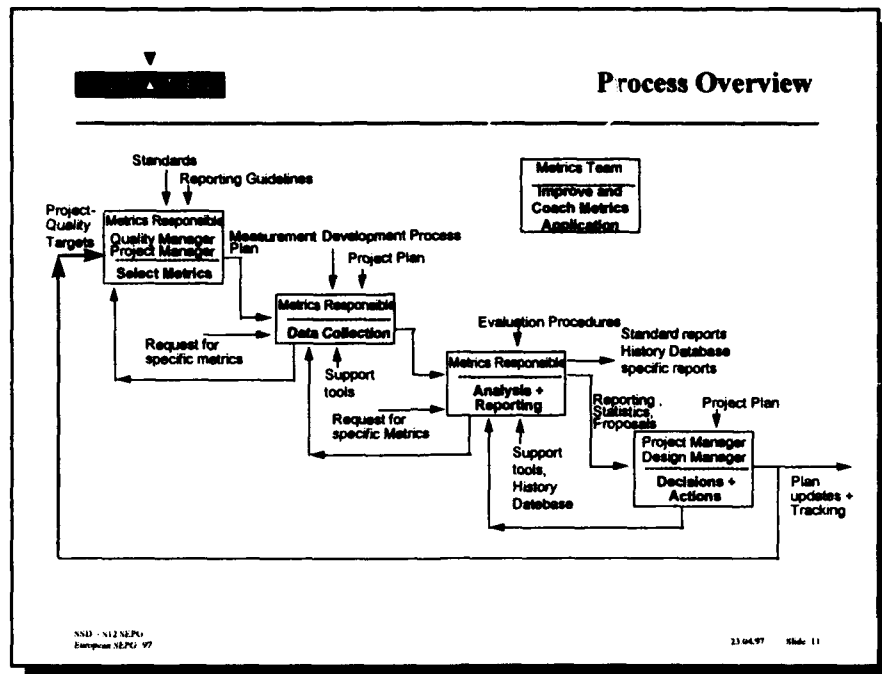
23 04 97 Slide 9

Responsibilities within Metrics Program



XSD - S12 SEPJ
European SEPJ 97

23 04 97 Slide 10



ROI Potentials

▼ CMM Level 1

- Uncontrollable cost and delivery dates
- Cancelled projects
- Firefighting

▼ CMM Level 2,3

- Correction effort
- Time to market
- Development effort
- Software quality
- Efficiency

SND - N12 SEPJ
European SEPJ '97

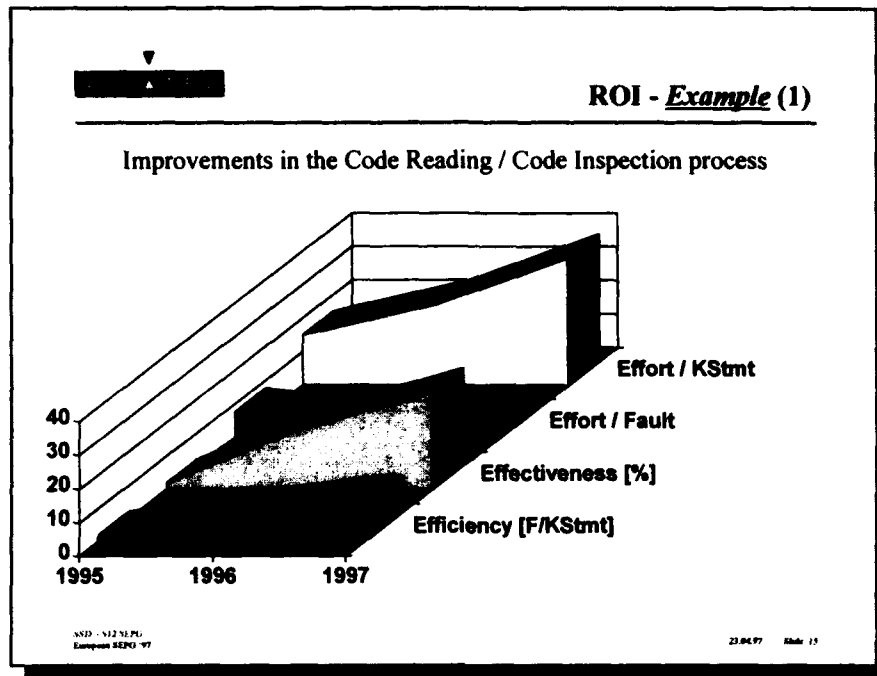
23.04.97 Slide 13

Hidden ROI Potentials

- ▼ Customer satisfaction
- ▼ Improved market share because of better quality, delivery accuracy and lower per feature costs
- ▼ Opportunity costs
- ▼ Reduced maintenance costs in follow-on projects
- ▼ Improved reusability
- ▼ Employee satisfaction
- ▼ Resources are available for new projects instead of waisting them for firefighting

SND - N12 SEPJ
European SEPJ '97

23.04.97 Slide 14



ROI - *Example (2)*

	1995	1996	1997
Reading Speed [Stmt/Ph]	142	68	39
Effort per Kstmt	15	24	36
Effort per Fault	7.5	3	3
Faults per KStmt	2	8	13
Effectiveness [% of all]	2	18	29
Project: 70Kstmt; 3150 Faults *			
Effort for Code Reading / Insp. [Ph]	1050		2660
Faults found in Code Reading / Insp.	140		910
Remaining faults	3010		2240
Corr. effort after Code Reading / Insp. [Ph]			
(based on 15 Ph/F average corr. effort)	45150		33600
Total correction effort [Ph]	46200		36260
ROI = saved total effort / add. det. effort			6.2

** effects of defect-preventive activities over time not considered for this example*

ASD - S12 SEPG
European SEPG '97

23.04.97 Slide 16

Experiences with ROI Calculation

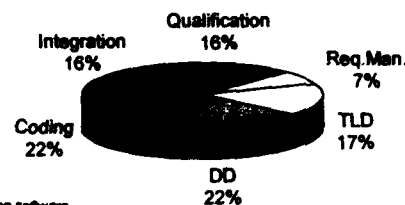
- ▼ It is better to collect effort figures during a project than afterwards
- ▼ Activities related to distinct effort figures must be defined (activity based costing helps a lot)
- ▼ Cost and effort must not be estimated for ROI calculation
- ▼ Detailed quality cost are helpful for root cause analyses and related defect prevention activities
- ▼ Tangible cost savings are the single best support for a running an improvement program
- ▼ Cost of nonperformance is a perfect trigger for a SPI program

SPD - S12 SEP0
European SEP0 '97

23 Sep 97 Slide 17

Activity-Based Models

- ▼ Estimation and allocation of tasks is based on percentage distribution to the overall effort
- ▼ Advantage: more accurate estimation than holistic models because effects of process and other specific factors can be related to each activity rather than on the entire process



Example for embedded real-time software development

SPD - S12 SEP0
European SEP0 '97

23 Sep 97 Slide 18

Benefits of SPI Related to Metrics

- ▼ Improved tracking and control
- ▼ Earlier identification of deviations from targets and plans
- ▼ Accumulation of history data
- ▼ Tracking of process improvements and deviations from processes
- ▼ Metrics link improvement strategies, pilot results, and various process reengineering efforts to the day-to-day business

SSD - 512 SEP
European SEP 97

23 04 97 Slide 19

Key Success Factors

- ▼ Motivate the metrics program with concrete and achievable improvement goals
- ▼ Start small and immediately
- ▼ Collect objective and reproducible data
- ▼ Establish focal points for metrics
- ▼ Define and align the software processes
- ▼ Get support from management (instead of abuse of the data)
- ▼ Communicate success stories
- ▼ Slowly enhance the metric program
- ▼ After all the focus is on projects and costs and not on metrics

SSD - 512 SEP
European SEP 97

23 04 97 Slide 20

Quantitative Management of Software Process Improvement

Christof Ebert, Alcatel Telecom, Switching Systems Division, Antwerp

Abstract: Collecting and analyzing metrics is critical to objectively identifying and quantifying process improvements. Progress metrics are particularly relevant for having insight in projects and at the same time into process improvements. This article focuses on introducing and maintaining a corporate metrics program in a highly distributed large organization. Experiences are shared that show how the metrics program closely relates and thus supports an ongoing SPI initiative. Results from Alcatel Telecom's Switching System Division are included to show practical impacts.

Keywords: process improvement, project control, software metrics, process improvement

1 Introduction

*Not everything that counts can be counted
and not everything that can be counted counts.*

Albert Einstein

Quantitative data is crucial for understanding software development processes and to steer any reengineering activity. Quantitative management of a software process improvement (SPI) activity is not much different from managing a project. Unless supported by metrics, it is impossible to fully understand what is happening and what will be the outcomes of prospective changes. Quantitative management of SPI is thus concerned with identifying, measuring, accumulating, analyzing and interpreting project and process information for strategy formulation, planning and tracking activities, decision-making, and cost accounting. As such it is more than only ensuring overall technical correctness of a project as earlier defined [1]. Objectives are derived from these activities:

- Decision-making: What should I do?
- Attention-directing: What should I look at?
- Performance evaluation: How well am I doing?
- Planning: What can we reasonably achieve in a given period?
- Target-setting: How much can we improve in a given period?

SPI programs are different in nature and target environment; they depend on the business goals and the competitive situation of the company, its products and the underlying processes. We will focus in this article on the SPI program of Alcatel Telecom's Switching Systems Division. Telecommunication switching systems are among the biggest challenges in current software development because they are distributed both during development time and during runtime. Due to their considerable size (several MLOC), such systems are developed within locally distributed development units by globally operating companies. The *Alcatel 1000 S12* is a digital switching system that is currently used in over 40 countries world-wide. It provides a wide range of functionality (small local exchanges, transit exchanges, international exchanges, network service centers, or intelligent networks) and scalability (from small remote exchanges to large local exchanges). Its typical size is about 2.5 MLOC of which a big portion is customized for local network operators. The code used for *S12* is realized in Assembler, C and CHILL. In terms of functionality, *S12* covers almost all areas of software and computer engineering. This includes operating systems, database management and distributed real-time software.

The paper is organized as follows. Ch.2 gives a brief overview of metrics used for decision making

in software project management. Ch.3 relates metrics activities to SPI. Starting a corporate metrics program in a distributed organization as part of a SPI initiative is introduced in Ch.4. Experiences are provided that hold especially from the perspective of organizational learning. Ch.5 tries to introduce briefly to ROI calculation as it is requested often for management decisions in a SPI program. Finally, Ch.6 summarizes the results and gives an outlook on how metrics and SPI will evolve over time in an organization.

2 Metrics and Decision Making

The single best technology for getting some control over deadlines and other resource constraints is to set formal objectives for quality and resources in a measurable way [2,3]. Planning and control activities cannot be separated. Managers control by tracking actuals against plans and acting on observed deviations. Controls should focus on significant deviations from standards and at the same time suggest appropriate ways for fixing the problems. Typically these standards are schedules, budgets, and quality targets established by the project plan. All critical attributes should be established both measurable and testable to ensure effective tracking. The worst acceptable level should be clear although the internal target is in most cases higher.

The influence of metrics definition and application from project start (e.g. estimation, target setting and planning) to steering (e.g. tracking and budget control, quality management) to maintenance (e.g. failure rates, operations planning) is very well described in the related IEEE Standard for Developing Software Life Cycle Processes [4]. This standard also helps in defining the different processes that constitute the entire development process including relationships to management activities.

Although the corporate metrics program has been set up and is maintained as part of the Division's SPI program, most benefits that we recorded are indeed related to project management:

- Improved tracking and control of each development project based on uniform mechanisms;
- Earlier identification of deviations from the given targets and plans;
- Accumulation of history data from all different types of projects that are reused for improving estimations and planning of further projects;
- Tracking process improvements and deviations from processes.

Metrics are obviously the key to successfully managing a SPI program because they link the improvement strategies, pilot results and various process reengineering efforts to the day-to-day business that after all keeps the company alive.

One of the main targets of any kind of measurement is that it should provide an objective way of expressing information, free of value judgments. This is particularly important when the information concerned is "bad news", for instance related to productivity or cost, and thus not necessarily be well received. Often the observed human tendency is to ignore any criticism related to one's own area and direct attention to somebody else's. Test articulates that "the design is badly structured", while operations emphasize that "software has not been adequately tested". Any improvement activities must therefore be based on hard numerical evidence. The first use of metrics is most often to investigate the current state of the software process. Table 1 relates the most relevant metrics to different levels of process maturity. Basically application of metrics is mainly restricted due to non-repeatable processes and thus a limited degree of consistency across projects.

With such premises it is feasible to set up not only a release-oriented phase end target but also phase entry criteria that allow for rejection to module test or inspections if the system quality is inadequate. Related test process metrics include test coverage, number of open fault reports by se-

verity, closure delay of fault reports, and other product-related quality, reliability, and stability metrics. Such metrics allow judgments in situations when due to difficulties in testing decisions on the nature and sequence of alternative paths through the testing task should be made, while considering both the entire testing plan and the present project priorities. For example, there are circumstances in which full testing of a limited set of features will be preferred to a incomplete level of testing across full (contracted) functionality.

3 Setting Objectives - The Case for Process Improvement

A primary business goal for any telecommunication systems supplier today is to control and reduce software development cost. Software development increasingly contributes to product cost. It has been shown within various companies that the SEI CMM is an effective roadmap for achieving cost-effective solutions. Many of the problems that lead to project delays or failures are technical, however the critical ones are managerial. Software development depends on work products, processes and communication. It is only structured if a structure is imposed and controlled. This is where the CMM fits in. Figure 1 which was compiled from different industrial databases gives an overview of fault detection within organizations according to their respective maturity level (effects on ROI are further explained in Ch.5). Obviously what is done right in software development is done early. There is little chance for catching up when things are discovered to be going wrong later in the development.

Since the CMM provides both a guideline for identifying strengths and weaknesses of the software development process and a roadmap for improvement actions, Alcatel Telecom also based its improvement activities on the CMM. Periodic assessments are conducted in all major development sites. The direct findings are analyzed according to their prospective impacts on Alcatel's business goals and then prioritized to select those areas with highest improvement potential. Based on this ranking a concrete planning of improvement actions is repeatedly refined, resulting in an action plan with detailed descriptions of improvement tasks with responsibilities, effort estimates, etc. Checkpointing assessments of the same type are repeatedly done to track the implementation of the improvement plan.

The improvement program within S12 development consumes roughly 5% of the total development effort for activities such as process control, pilots, tools improvements and enhanced tracking activities. The metrics program is one core part of the entire improvement program (Figure 2). Several achievements during the first part of the S12 improvement program can be attributed to increasing visibility of project status, improved awareness of work products quality and setting improvement targets for results of each major development phase.

Our experiences with SPI include:

- Defining a common process framework that is applicable for all types of projects (perhaps after some predefined degrees of tailoring) and thus building a common decision framework that allows for sharing experience and learning from other projects' lessons, both positive and negative.
- Focus on quality which reflects that customers when faced with a choice will always select quality when the functionality of the options is nearly equivalent.
- Freeze requirements at a distinct point during design and decide on phased deliveries instead of struggling with ever changing requirements on a fixed schedule and budget.
- Involve customers in the process and its improvement since in many cases they are willing to share their knowledge and experience.
- Understand the needs of the customer instead of trying to stick to specifications that are typi-

cally unclear to both sides upfront. For the same reason customer satisfaction is much more relevant from a SQA viewpoint than mere specification conformance.

Different groups typically work towards individually controlled goals that build up to business division level goals and corporate goals. An example for improved maintainability indicates this hierarchy. A business division level goal could be to improve maintainability within legacy systems, as it is strategically important for all telecommunication suppliers. Design managers might break that further down to redesigning exactly those components that are at the edge of being maintainable. Project managers on the other hand face a trade-off with time to market and might emphasize on incremental builds instead. Clearly both need appropriate indicators to support their selection processes which define the way towards the needed metrics related to these goals. Obviously one of the key success criteria for SPI is to understand the political context and various hidden agendas behind technical decisions in order to make compromises or weigh alternatives.

Objectives related to individual processes must be unambiguous and agreed by the respective groups. This is obvious for test and design groups. While the first are reinforced for finding defects and thus focus on writing and executing effective test suites, design groups are targeting to delivering code that can be executed without defects. In case of defects they must be corrected efficiently, which allows for setting up another metric for a design group which is the backlog of faults it has to resolve.

It is thus important for process metrics to consider different viewpoints and their individual goals related to promotion, projects and the business. Most organizations have at least four: the practitioner, the project manager, the department head, and corporate executives. Their motivation and typical activities differ much and often create confusing goals which at the worst level are resolved on the practitioner level. Reuse for instance continuously creates trade-off discussions. When a project incurs expenses due to keeping components maintainable and to promote their reusability, who pays for it and where is it recorded in a history database that compares efficiency (e.g. bang per buck) of projects and thus of their management?

Managing and tracking SPI can be done on different levels of abstraction. Senior management is interested in the overall achievements based on what has been invested in the program (see Ch.5). Related metrics include the effectiveness of fault detection because the obvious relationship to cost of quality is directly related to the most common business goal of cost reduction. Lead-time reduction and effort reduction is related to reduced rework and as such also related to less defects and early detection. On the project level SPI management includes a variety of process metrics that compare efficiencies and thus relate on the microscopic level to achieving the business goals (e.g. Figure 3).

4 Setting Up a Metrics Program in a Distributed Organisation

This section shares some selected experiences we made while setting up a globally distributed metrics program in the different locations of the Switching Systems Division. The following key success factors could be identified:

- Start small and immediately (see initial timetable in Table 2). It is definitely not enough only to select goals and metrics. Tools and reporting must be in line; and all of this takes its time. It must however be clearly determined what needs to be measured before deciding based on what can be measured. Use external consultants where needed to get additional experience and authority.
- Motivate the metrics program with concrete and achievable improvement goals. Unless targets are achievable and clearly communicated to middle management and practitioners they will

clearly feel metrics as yet another instrument of management control. Goals must be in line with each other and on various levels. Business goals must be broken down to project goals and those must be aligned with department goals and contents of quality plans. Clearly communicated priorities might help with individual decisions.

- Provide training both for practitioners who after all have to deliver the accurate raw data, and for management who will use the metrics. The cost and effort of training is often stopping its effective delivery. Any training takes time, money, and personnel to prepare, update, deliver, or receive it.
- Establish focal points for metrics in each project and department. Individual roles and responsibilities must be made clear to ensure a sustainable metrics program that endures initial SPI activities (see below for an explanation of Figure 4).
- Define and align the software processes to enable comparing metrics. While improving processes or setting up new processes, ensure that the related metrics are maintained at the same time. Once estimation moves from effort to size to functionality, clearly the related product metrics must follow.
- Collect objective and reproducible data. Ensure the chosen metrics are relevant for the selected goals (e.g. tracking because to reduce milestone delay) and acceptable for the target community (e.g. it's not wise to start with productivity metrics).
- Get support from management. Enduring buy-in of management can only be achieved if the responsibility for improvements and the span of necessary control are aligned with realistic targets. Since in many cases metrics beyond test tracking and faults are new instruments for parts of management this group must also be provided with the necessary training.
- Avoid abuse of metrics by any means. Metrics must be "politically correct" in a sense that they should not immediately target persons or satisfy needs for personal blames. Metrics might hurt but should not blame. Certainly limited visibility and access to the metrics helps in creating credibility among practitioners (Table 3). Before introducing metrics however, it is even more important to indicate the application of the metrics (such as individual improvements with individual data) based on a supportive climate. It is often helpful to change perspective towards the one providing raw data: is the activity adding value to her daily work? Statistical issues might not automatically align with emotional priorities. Remember that their perception is their reality.
- The targets of any improvement program must be clearly communicated and perceived by all levels as realistic enough to fight for. Each single process change must be accompanied with the respective goals and supportive metrics that are aligned. Those affected need to feel that they have some role in setting targets. Where goals are not shared and the climate is dominated by threats and frustration, the metrics program is more likely to fail.
- Communicate success stories where metrics enabled better tracking or cost control. This includes identifying metrics advocates that help in selling the measurement program. Champions must be identified at all levels of management, especially at senior level, that really use metrics and thus help to support the program. Metrics can even tie in an individual's work to the bigger picture if communicated adequately.
- Slowly enhance the metric program. This includes defining "success criteria" to be used to judge the results of the program. Since there is no perfect metrics program it is necessary to determine something like a "80% available" acceptance limit that allows to declare success when that is achieved.
- Don't overemphasize the numbers. It is much more relevant what they bring to light, such as emerging trends or patterns. After all the focus is on successful projects and efficiency improvement and not on metrics.

Metrics need to make sense to everybody within the organization who will be in contact with them. Therefore, the metrics should be piloted and evaluated after some time. Potential evaluation questions include:

- Are the selected metrics consistent with the original improvement targets? Do the metrics provide added value? Do they make sense from different angles and can that meaning be communicated without many slides? If metrics are considering what is measurable but don't support improvement tracking, they are perfect for hiding issues but should not be labeled metrics.
- Do the chosen metrics send the right message about what the organization considers relevant? Metrics should spotlight by default and without cumbersome investigations of what might be behind. Are the right things being spotlighted?
- Do the metrics clearly follow a perspective that allows comparisons? If metrics include ambiguities or heterogeneous viewpoints they cannot be used as history data.

4.1 The Metrics Process, Roles and Responsibilities

Obviously the introduction of metrics to projects has to follow a stepwise approach that must be carefully coached. Each new metric that needs tools support must be piloted first in order to find out whether definitions and tools description are sufficient for the collection. Then the institutionalization must be planned and coached in order to obtain valid data. For that reason following three roles are established:

1. Project metrics responsables within each single project as a focal point for engineers or the project management in the project (Figure 4). They ensure that the metric program is uniformly implemented and understood. The role includes support for data collection across functions and analysis of the project metrics. The latter is most relevant for a project manager because he must be well aware of progress, deviations and risks with respect to quality or delivery targets. By creating the role of a project's metric responsible we guaranteed that the responsibility was clearly assigned as of project start, while still allowing for distributed (functional) metrics collection.
2. Local metric teams in each location to be a focal point for all metrics related questions in a location, to synchronize metrics activities, to emphasize on commonality, and to collect local requirements from the different projects. Besides being the focal point for the metrics program in a single location they provide training and coaching of management and practitioners on metrics, their use and application. In addition they ensure that heterogeneous tools are increasingly aligned or that tools and forms for data collection and analysis are made available to the projects and functional organization.
3. A central (business division) metric team created by representatives of the local metric teams of the organization. They altogether ensure that rationalization and standardization of a common set of metrics and the related tools and charts are accelerated. Upon building a corporate metrics program and aligning processes with the software process improvement activities, the creation of a history database for the entire organization is an important prerequisite of improving estimates.

This structure guarantees that each single change or refinement of metrics and underlying tools, but also needs from projects can be easily communicated through the whole organization. Use of teams should however be done cautiously. While a team has the capability to take advantage of diverse backgrounds and expertise, the effort is most effective when there are not more than 3 people involved. Larger teams spend too much time backtracking on metrics choices. We also found that when potential users worked jointly to develop the metrics set with the metrics support staff, the program was more readily accepted.

The related metrics process is applicable in the day-to-day project environment. It is based on a set of defined metrics and rather supports the setting up and tracking of project targets and improvement goals (Figure 5):

1. Based on a set of predefined corporate metrics the first step is to select metrics suitable for the project.
2. Then the raw data is collected to calculate metrics. Be aware of the operational systems that people work with that need to supply data. If the data is not available easily chances are high that the metric is inaccurate and people tend to ignore further metrics requests. People might then even comply to the letter of the definition but not to the spirit of the metric.
3. Metrics are then analyzed and reported through the appropriate channels. Analysis includes two steps. First data is validated to make sure it is complete, correct, and consistent with the goals it addresses. Don't assume that automatic metrics are always trustworthy. At least perform sample checks. The real challenge is the second step which investigates what is behind the metrics. Some conclusions are straightforward, while others require an in-depth understanding of how the metrics relate with each other. Consolidating metrics and aggregating results must be done with great caution, even if apples and apples might fit neatly, so to speak. Results are useless unless major observations are reported back to the people who make improvements or decisions.
4. Finally the necessary decisions and actions are made based on the results of the analysis. The same metrics might trigger different decisions depending on the target audience. While senior management may just want to get an indication how well the improvement program is doing, the local SEPG leader might carefully study process metrics to eliminate deficiencies.

4.2 Metrics Selection and Definition

Each metric's definition should ensure consistent interpretation and collection across the organization. Capturing precise metrics information not only helps with communicating what's behind the figures but also builds the requirements for automatic tools support and provides basic material for training course development. We have used the following sections within the definition template:

- Name and identifier of the metric;
- Brief description;
- Relationships to goals or improvement targets;
- Definition with precise calculation;
- Tools support (links and references to the supporting tools, such as databases, spreadsheets, etc.);
- Visualization with references to templates;
- Collection period and reporting frequency;
- Alarm levels for interpretation;
- Configuration control with links to storage of (periodically collected) metrics.

A project-specific measurement plan links the generic metrics definition to concrete projects with their individual project goals and responsibilities. Additional metrics to be used only in that project are referred to in the measurement plan. The measurement plan is linked to the quality plan to facilitate alignment of targets.

Often terminology must be reworked especially when the projects are scattered in a distributed organization, such as Alcatel Telecom. The standard definition might include several sections applicable for different project types. Project size as well as design paradigms influence definitions and metric calculation, even if the metric goals and underlying rationales are the same, such as with deliverables tracking.

5 ROI Calculation

ROI is difficult to calculate in software development. This is not so much any more due to not having collected effort figures but rather by distinguishing the actual effort figures that relate to investment (that would have otherwise not been done) and the returns (as difference to what would have happened if not having invested). For many years ROI data was reported but in most cases not backed up by real data, or where real data existed, it was counter to the current mainstream viewpoints [2]. Only recently several studies have been published that try to compare results of software process improvement activities [5,6].

For calculating ROI effects the following rules should be considered:

- Samples should consider projects before and after the start of the improvement program to be evaluated with ROI.
- Controlling should be able to provide history data (e.g. effort).
- Aggregated or combined effort or cost figures must be separated (i.e. prevention, appraisal cost, cost of nonperformance, cost of performance - which are typically spent in any case).
- Include only those effects which trace back to root causes that were part of the original improvement planning.
- Check cost data on consistency within one project and across projects.

ROI is most efficiently presented according to the following flow:

1. Current results (these are the potentials; i.e. problems, cost; causes);
2. Known effects in other (competing) companies (i.e. improvement programs in other companies; benchmarking data; cost-benefit estimation for these companies);
3. ROI calculation (calculate cost of quality per month for several sample projects; calculate the savings since start of the improvement program; extrapolate these savings for all affected projects which is benefit; compare the benefit with the cost of the improvement program which is ROI; never include cost of performance since this is regular effort).

We will try to provide insight in a ROI calculation. The data which is used for calculations results from average values that have been gathered in the our history database (Table 4). The history database currently represents 50 projects with an average size of 70 new or changed KStmt and roughly 2 MStmt reused code per project.

We will compare the effect of increased effort for combined code reading and code inspection activities as a key result of our improvement program. The summary shows that by reducing the amount of code to be inspected per hour by more than a factor three, the efficiency in terms of faults detected increased significantly. As a result the percentage of faults detected during coding increases dramatically. While reading speed reflects only the actual effort spent for fault detection, the effort per KStmt includes both detection and correction, thus resulting in around 3 Ph/Fault which seems stable.

Given an average sized development project and only focusing on the new and changed software without taking into account any defect-preventive results over time, the following calculation can be derived. Effort spent for code reading and inspection activities increases by 1610 Ph. Assuming a constant average combined appraisal cost and cost of nonperformance (i.e. detection and correction effort) after coding of 15 Ph/Fault, the total effect is 9940 Ph less spent in 1997. This results in a ROI value of 6.2 (i.e. each additional hour spent during code reading and inspections yields 6.2 saved hours of appraisal and nonperformance activities afterwards).

We made the following experiences with ROI calculations:

- It is better to collect the different effort figures during a project than afterwards.

- Activities related to distinct effort figures must be defined (activity based costing helps a lot).
- Cost and effort must not be estimated, but rather collected in projects (typically the inputs to the estimation are questioned until the entire calculation is not acceptable any more).
- Detailed quality cost are helpful for root cause analyses and related defect prevention activities.
- Tangible cost savings are the single best support for a running improvement program.
- Cost of nonperformance is a perfect trigger for a SPI program.
- Obvious management and software engineering practices are typically no ROI topics.
- There are many "hidden" ROI potentials that are often difficult to quantify (e.g. customer satisfaction; improved market share because of better quality, delivery accuracy and lower per feature costs; opportunity costs; reduced maintenance costs in follow-on projects; improved reusability; employee satisfaction; resources are available for new projects instead of wasting them for firefighting).
- There are also hidden investments that must be accounted (e.g. training, infrastructure, coaching, additional metrics, additional management activities, process maintenance).
- Without any metrics managers need not to be convinced (poor managers entirely believe in qualitative reasoning and call it "intuition" as an excuse for lack of motivation (without metrics no improvement and without improvement no metrics).

Not all ROI calculations are based on monetary benefits. Depending on the business goals it can as well be directly presented in reduced lead time or higher efficiency and productivity.

6 Conclusions

Software process improvement is now a big issue on the agenda of all organizations with software as a core business. As such it is also a major research topic, that may continue to grow in importance well into the 21st century. However, some software technologies have a shorter lifetime and for sure the management attention is focused rather on short-term achievements with impact to the score card. Unless tangible results can be achieved in the related short timeframe, interest in SPI will quickly wane.

We have presented the introduction of a corporate metrics program as part of the SPI initiative of Alcatel Telecom's Switching Systems Division. Any data presented here on effort or value of SPI is only approximate and closely linked to the environment it was extracted from. It is obvious by now that the more advanced the corporate process maturity, the more various and specific the metrics that are used. While maintaining the metrics program it must be emphasized that metrical needs evolve over time. The critical precondition of any SPI program is the focus on metrics and their effective exploitation.

So far many of the industrial discussions and articles related to SPI are based on facts, while research is targeting theories and small-scale examples. Both is valid - from the respective viewpoint. It would however be helpful to bridge the gap with corporate studies related to answering the two important questions:

- What does it cost to improve software processes?
- How long will it take to make tangible improvements?

Answering such questions of course needs some focus on areas such as quality improvement, better productivity, shorter lead-time, or higher customer satisfaction.

References

- [1] Rook, P.: Controlling Software Projects. *Software Engineering Journal*, Vol. 1, No. 1, pp. 7 - 16, 1986.
- [2] Fenton, N. E. and S.L. Pfleeger: *Software Metrics: A Practical and Rigorous Approach*. Chapman & Hall, London, UK, 1997.
- [3] Stark, G., R.C. Durst and C.W. Vowell: Using Metrics in Management Decision Making. *IEEE Computer*, Vol. 27, No. 9, pp. 42 - 48, 1994.
- [4] IEEE Standard for Developing Software Life Cycle Processes. IEEE Std 1074-1991, IEEE, Piscataway, USA, 1992.
- [5] McGibbon, T.: A Business Case for Software Process Improvement. *DACS State-of-the-Art Report*. Rome Laboratory, <http://www.dacs.com/techs/roi.soar/soar.html#research>, 1996.
- [6] Jones, T.C.: Return on Investment in Software Measurement. *Proc. 6. Int. Conf. Applications of Software Measurement*. Orlando, FL, USA, 01. Nov. 1995.

Table 1: Appropriate Metrics for Different CMM Levels

CMM	Description	Metrics
5	Continuous improvements are institutionalized	Process metrics for the control of process change management
4	Products and processes are quantitatively managed	Process metrics for the control of single processes
3	Appropriate techniques are institutionalized	Defined and established product metrics; automatic metric collection
2	Project management is established	Defined and reproducible project metrics for planning and tracking (fault status, effort, size, progress); few process metrics for SPI progress tracking
1	Process is informal and ad hoc	Few project metrics (size, effort, faults); however metrics are inconsistent and not reproducible

Table 2: Time Table for Setting up a Corporate Metric Program

Activity	Elapsed time	Duration
Initial targets set up	0	2 weeks
Creation and kick-off of metric team	2 weeks	1 day
Goal determination for projects and processes	3 weeks	2 weeks
Identifying impact factors	4 weeks	2 weeks
Selection of initial suite of metrics	5 weeks	1 week
Report definition	6 weeks	1 week
Kick-off with management	6 weeks	2 hours
Initial tool selection and tuning	6 weeks	3 weeks
Selection of projects / metric plan	6 weeks	1 week
Kick-off with project teams / managers	7 weeks	2 hours
Collection of metric baselines	7 weeks	2 weeks
Metric reports, tool application	8 weeks	continuously
Review and tuning of reports	10 weeks	1 week
Monthly metric-based status reports within projects	12 weeks	continuously
Application of metrics for project tracking and process improvement	16 weeks	continuously
Control and feedback on metric program	24 weeks	quarterly
Enhancements of metric program	1 year	continuously

Table 3: Visibility, Access, and Timing of Metrics

Private data for practitioner	Private data for project team	Corporate data
Immediate access (i.e. minutes)	Hourly or daily access	Weekly or monthly access
- Fault rates of individuals - Fault rates in module before integration - Fault rates during coding - Number of local compile runs - Effort spent for single module	- Fault rates in subsystems - New/changed code per module - Estimated effort and new/changed size per module - Number of repeated reviews and inspections - Fault rates during design	- Fault rates in project - Failure rates in project - New/changed code in project - Effort per project - Effort per delivered code size (efficiency) - Effort per fault - Effort per phase and in processes - Elapse time per process and phases

Table 4: Process Improvements with Focus on Code Reading / Inspections (see also Figure 3; defect preventive activities are not considered for this example)

	1995	1996	1997
Reading Speed [Stmt/Ph]	142	68	39
Effort per Kstmt	15	24	36
Effort per Fault	7.5	3	3
Faults per KStmt	2	8	13
Effectiveness [% of all]	2	18	29
Project: 70Kstmt; 3150 Faults			
Effort for Code Reading / Insp. [Ph]	1050		2660
Faults found in Code Reading / Insp.	140		910
Remaining faults	3010		2240
Corr. effort after Code Reading / Insp. [Ph] (based on 15 Ph/F average corr. effort)	45150		33600
Total correction effort [Ph]	46200		36260
ROI = saved total effort / add. det. effort			6.2

Figure 1: Typical benchmark effects of detecting faults earlier in the life cycle

CMM Level	Requirements	Design	Coding	Module Test	Integration Syst. Test	Field
Defined 3	2%	5%	28%	30%	30%	<5% 2F/KLOC
Repeatable 2	1%	2%	7%	30%	50%	10% 3F/KLOC
Initial 1	0%	0%	5%	15%	65%	15% 5F/KLOC

Figure 2: The S12 Software Improvement Activities and Emphasis on the Metrics Program

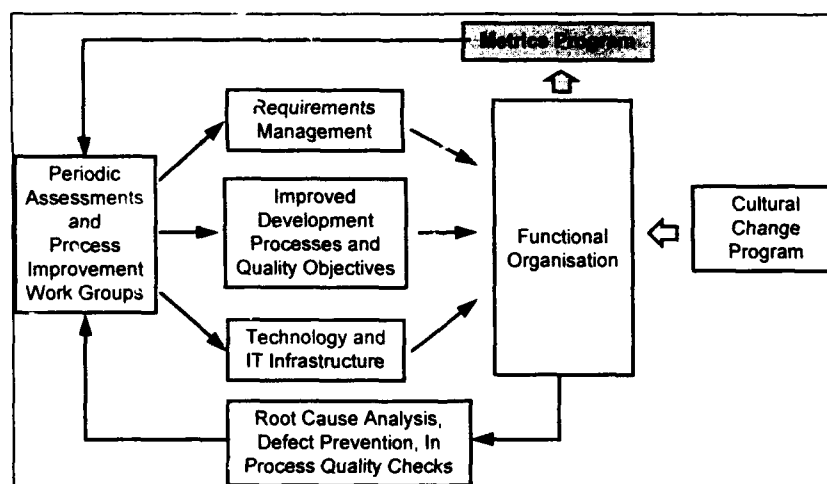


Figure 3: Metrics for SPI Management; Example: Code Reading Speed vs. Related Fault Rate in different Years of the SPI Program

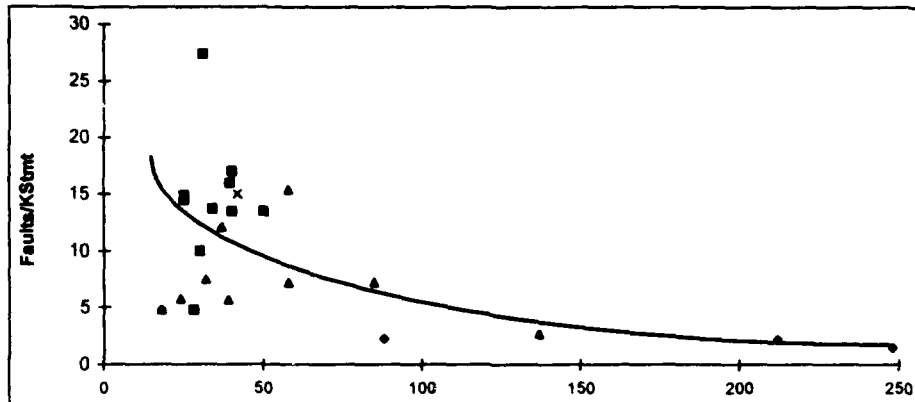


Figure 4: Roles and Responsibilities within Metrics Program

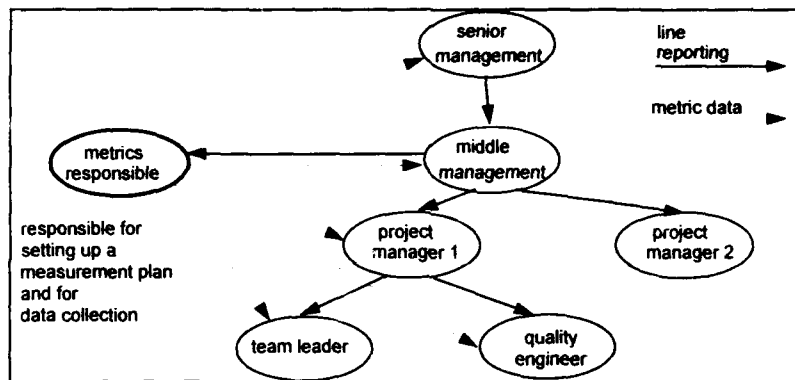
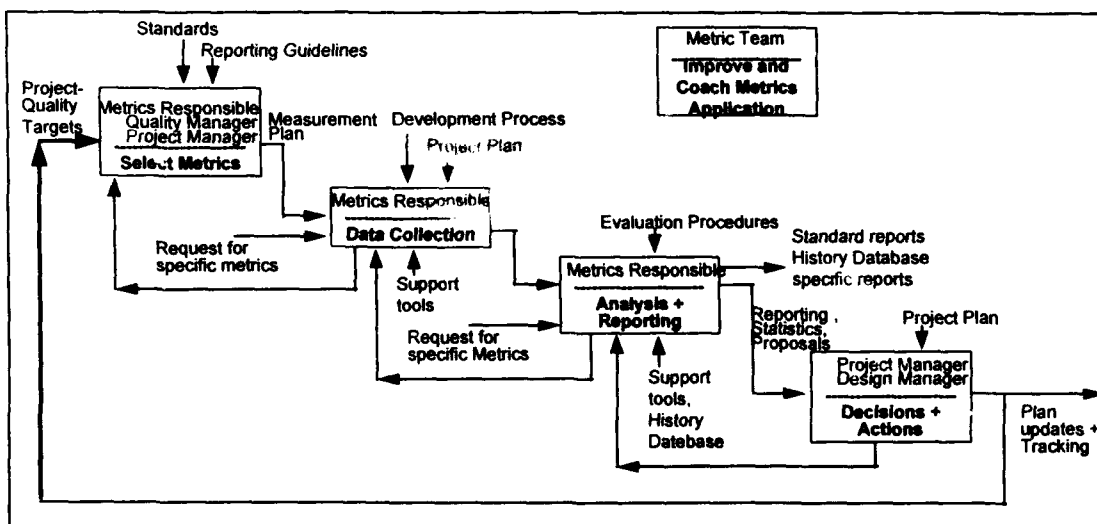


Figure 5: Integrated Measurement Process Overview



*European SEPG 1997
Measurement Symposium*

**Process Improvement by
Software Measurement**

*Current and Future Directions for
Goal/Question/Metric method*

Rini van Solingen,
Schlumberger RPS, The Netherlands
vansolingen@bladel.rps.slb.com

Egon Berghout
Delft University of Technology
e.w.berghout@twi.tudelft.nl

Schlumberger

European SEPG 1997 - Measurement Symposium

Schlumberger Retail Petroleum Systems

Objectives of this presentation

- Present current state of GQM method
- Introduce practical GQM techniques
- Make direct GQM application possible
- Point out future improvements GQM

Schlumberger

European SEPG 1997 - Measurement Symposium

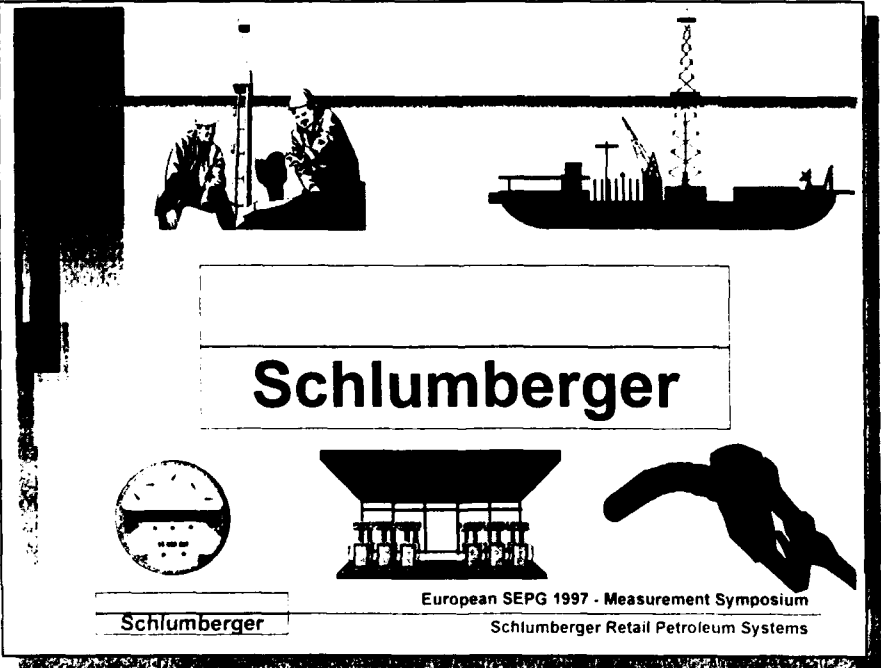
Schlumberger Retail Petroleum Systems

Outline of this Presentation

- **Introduction**
 - Schlumberger Retail Petroleum Systems
 - Goal/Question/Metric paradigm
- **Six step GQM method**
 - Abstraction Sheets
 - Feedback Sessions
- **Future Direction of GQM**
- **Conclusions**

Schlumberger

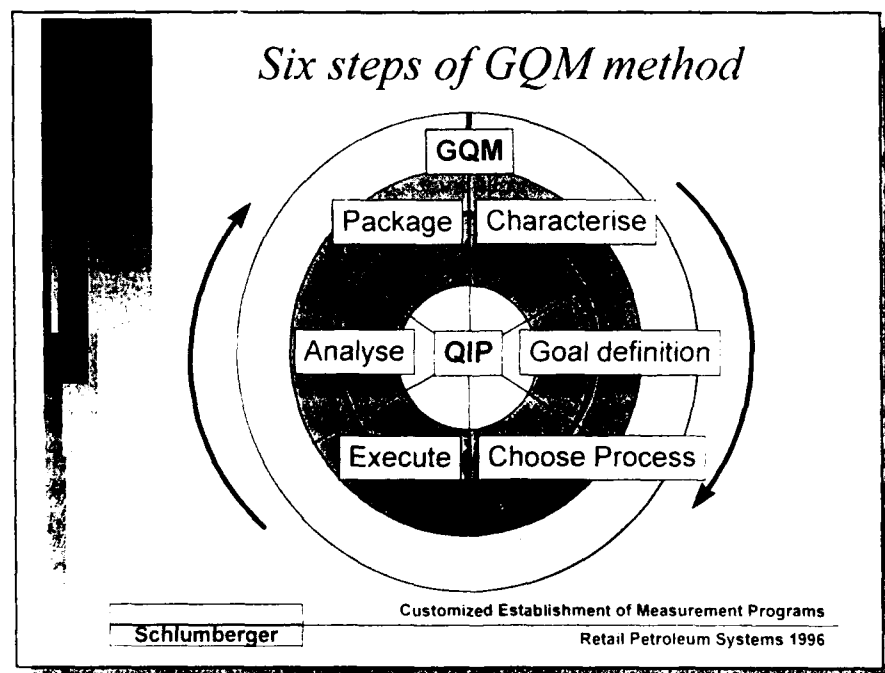
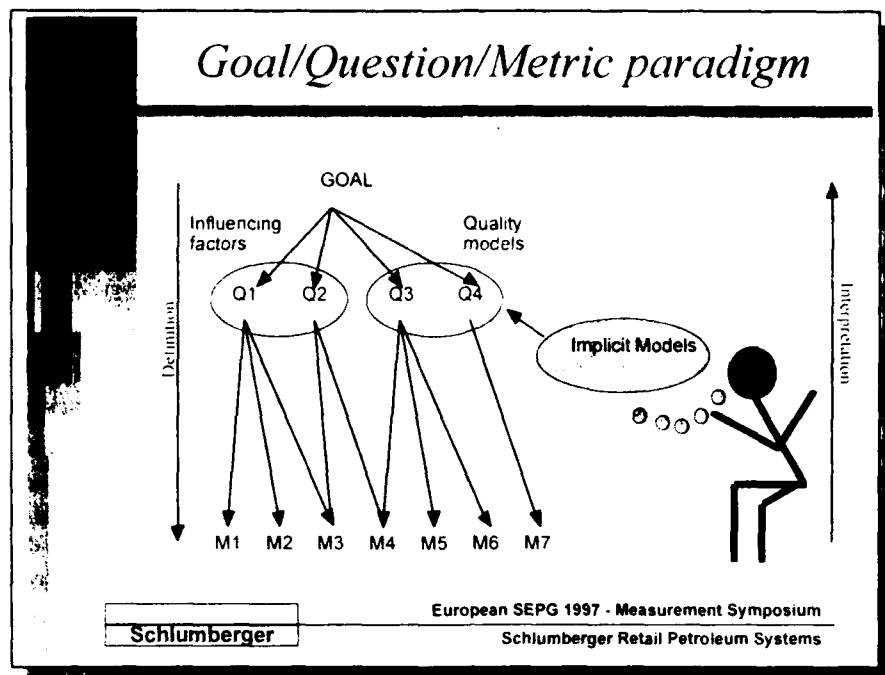
European SEPG 1997 - Measurement Symposium
Schlumberger Retail Petroleum Systems



Schlumberger

Schlumberger

European SEPG 1997 - Measurement Symposium
Schlumberger Retail Petroleum Systems



Six steps of GQM method

- **Step-1: Characterise**
- **Step-2: Define measurement goals**
- **Step-3: Make GQM deliverables**
- **Step-4: Execute project & measurement**
- **Step-5: Interpretation & Feedback**
- **Step-6: Package**

Schlumberger

European SEPG 1997 - Measurement Symposium
Schlumberger Retail Petroleum Systems

Step-1: Characterise

- **Baseline organisation and project**
- **Identify project objectives**
 - Product oriented
 - Process oriented
- **Use assessment outcomes**
 - CMM, BOOTSTRAP, SPICE
- **Identify available models, tools and techniques**

Schlumberger

European SEPG 1997 - Measurement Symposium
Schlumberger Retail Petroleum Systems

Step-2: Define Measurement Goals

- Capture strategic and business goals
- Refine high-level goals
- Define and prioritise measurement goals
- Goal template:

- Analyse:	reviewing process
- For the purpose of:	understanding
- With respect to:	effectiveness
- From the viewpoint of:	sw-developers
- In the following context:	Slb-project A

Schlumberger

European SEPG 1997 - Measurement Symposium
Schlumberger Retail Petroleum Systems

Step-3: Make GQM deliverables

- Define or review process models
- Interview project people
- Make GQM plan
 - Measurement objectives
 - Goal - Questions - Metrics
 - Hypotheses
- Make Measurement plan
 - Data collection procedures
 - Data collection forms, tools, etc.

Schlumberger

European SEPG 1997 - Measurement Symposium
Schlumberger Retail Petroleum Systems

Step-4: Execute Project & Measurement

- **Kick-off data collection**
- **Track data collection closely**
 - Measurement champion
 - Copy to QA or manager
- **Refine Measurement plan if necessary**
- **Give weekly feedback on performance**
 - Number of data points
 - X-weeks to feedback session

Schlumberger

European SEPG 1997 - Measurement Symposium
Schlumberger Retail Petroleum Systems

Step-5: Interpretation & Feedback

- **Feedback Sessions**
 - Critical success factor
 - High involvement of project team
 - Define: decisions, conclusions, and actions
- **Interpretation by project team**
 - GQM team is facilitator
 - Moderator sometimes necessary
- **Charts and tables**
 - Basic set
 - Additional analysis

Schlumberger

European SEPG 1997 - Measurement Symposium
Schlumberger Retail Petroleum Systems

Step-6: Package

- **Packaging of required knowledge**
- **Packaging on measurement goal**
 - Capture learning points, relations, influences
 - Distribute results
 - Disseminate frequently
 - Inform management (slides)
- **Packaging of GQM material**
 - Prepare for future use
 - Capture learning points and improvements

Schlumberger

European SEPG 1997 - Measurement Symposium
Schlumberger Retail Petroleum Systems

Abstraction Sheets

- **Structured interviewing**
- **Capture definitions, assumptions and models**
- **Four related quadrants:**
 - Quality Focus
 - Baseline Hypothesis (estimates)
 - Variation factors
 - Impact on Hypothesis
- **One page abstraction of GQM plan**

Schlumberger

European SEPG 1997 - Measurement Symposium
Schlumberger Retail Petroleum Systems

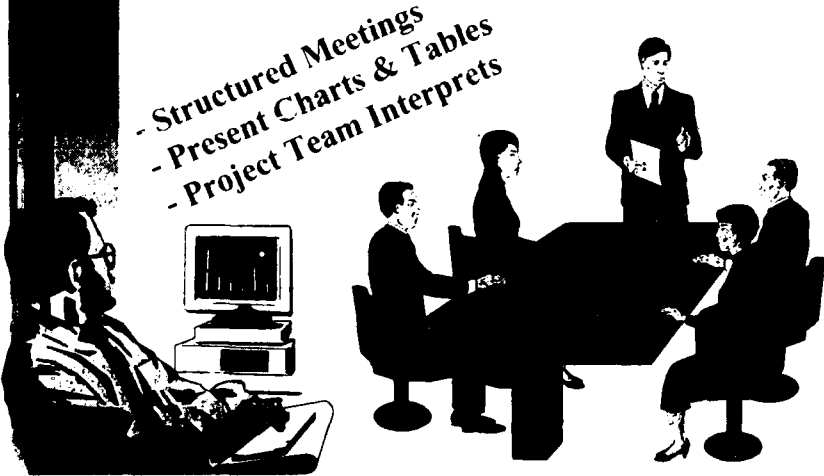
Tuesday 17 June

(T201a-5) S-7

<u>Object</u>	<u>Purpose</u>	<u>Quality Focus</u>	<u>Viewpoint</u>
Delivered Product	Understanding	Reliability and its causes	SW Development Team
<u>Quality Focus</u> Number of Failures • By Severity • By Detection group • Number of Faults • By Module		<u>Variation Factors</u> Level of Reviewing	
<u>Baseline Hypothesis (estimates)</u> Distribution of Failures By Severity • Minor 60% • Major 30% • Fatal 10%		<u>Impact Of Variation Factors</u> The higher the level of reviewing, the less failures, and the less faults slip through implementation phase	
Schlumberger		Customized Establishment of Measurement Programs Retail Petroleum Systems 1996	

Feedback Sessions

- Structured Meetings
- Present Charts & Tables
- Project Team Interprets



Schlumberger

European SEPG 1997 - Measurement Symposium
 Schlumberger Retail Petroleum Systems

Criticality of Feedback Sessions

- **Most important success factor**
- **Frequent feedback sessions**
 - every 6-8 weeks
 - 15-20 analysis slides
- **Learning and Process Improvement**
- **Regular feedback decreases risks**
- **Supports motivation and continuation**
- **Continuous update of GQM plan**

Schlumberger

European SEPG 1997 - Measurement Symposium

Schlumberger Retail Petroleum Systems

Future directions of GQM method

- **Improving GQM method**
 - Better documentation
 - Linking Product and Process goals
 - Tool development
- **On-line feedback tools**
 - Support during decision making
 - Individual feedback
 - Support during feedback sessions
 - Level-4 tool support
- **More focus on interpretation process**

Schlumberger

European SEPG 1997 - Measurement Symposium

Schlumberger Retail Petroleum Systems

Conclusions

- **GQM is powerful method**

- Goal driven
- Easy to use, but structures measurement
- De-facto standard for SPI-driven measurement

- **Enhancement of GQM method**

- Efficiency driven by industry
- Correctness/completeness driven by academics
- Value/profit driven by EU
- Product driven in Esprit PROFES

- **SPI needs GQM measurement**

Schlumberger

European SEPG 1997 - Measurement Symposium

Schlumberger Retail Petroleum Systems

Improvement by goal-oriented measurement

- Bringing the Goal/Question/Metric approach up to Level 5 -

Rini van Solingen and Egon Berghout

ABSTRACT

Metrics-programmes give powerful support to quality improvement of both software products and development processes. A well-known and popular software measurement approach is the Goal/Question/Metric method (GQM). This article presents a research road-map intended to enhance GQM. The main motive behind that presentation is that GQM will probably become the de-facto standard for software measurement. Nevertheless, many suggestions can be made to improve GQM. The Capability Maturity Model (CMM), which is intended to provide a framework to assess an organisation's software development practice, is applied in this article to assess GQM. Our research objective is, of course, to bring GQM to CMM's highest level 5.

1. INTRODUCTION

Many companies invest significant resources in software development. Because of the increasing size, complexity, quality needs and market demands for software, problems arise that are specific for software development. Examples are planning difficulties, unknown or bad product quality, projects that are never ended, milestones that are reached months or years too late, or developers who are working mostly unstructured, under high stress. Those problems (generally known as the 'software crisis'¹) are being tackled by 'Software Process Improvement' (SPI).

Currently, many software development practitioners and researchers are involved in software process improvement. Several models, methods and techniques are available, divided between two major streams.²

- Top-down approaches, such as CMM,³ SPICE⁴ and BOOTSTRAP.⁵ These are mainly based on the assessment and benchmarking of the entire software development effort according to predefined characteristics.
- Bottom-up approaches, such as GQM,⁶ QIP⁷ and AML,⁸ which mainly apply measurement to parts of the development process without predefined characteristics as their basic source of information.

The *top-down* improvement stream applies a normative model that is assumed to be the best way of developing software. By assessing an organisation, using this model, it becomes possible to identify the 'maturity' of that organisation, and propose relevant improvements.⁹ The *bottom-up* improvement stream is based on the assumption that it is impossible to define such a normative model for software development: the field of software engineering is still immature.¹ Therefore, empirical knowledge is gathered (software development is being measured), to improve understanding of software development within a specific context. Both streams are applied successfully in practice.

Software measurement is an important aid to software process improvement.¹⁰ 'Measurement' is the process by which numbers or symbols are assigned to attributes of entities in the real world in such a way as to describe them according to clearly defined rules.¹¹ Bottom-up improvement approaches always include some-kind of measurement framework. An example of such a

framework is the GQM method.^{6&12} It is a widely accepted method that is often applied in industry. GQM is goal-oriented, which makes it especially popular in goal-driven business environments.

Many people expect GQM to become the de-facto standard for software measurement in the context of Software Process Improvement. However, application in practice still is fraught with problems, such as:

- GQM cannot cope with high-level corporate goals;^{13&14}
- Application of GQM requires expert involvement, especially during the first year;¹⁵
- Support from automated tools and consultants is not widely available;^{15&16}
- Introduction processes are unknown;^{14&15}
- Literature does not sufficiently describe cost and benefit calculations from practice (at least not properly validated ones).^{15&17&18}

2. GOAL/QUESTION/METRIC PARADIGM

The GQM method originates with Professor Victor Basili, of the University of Maryland. Being involved in the Software Engineering Laboratory (SEL) at NASA, he developed a paradigm to study software development on its quality-relevant characteristics. The GQM-paradigm will be introduced shortly.

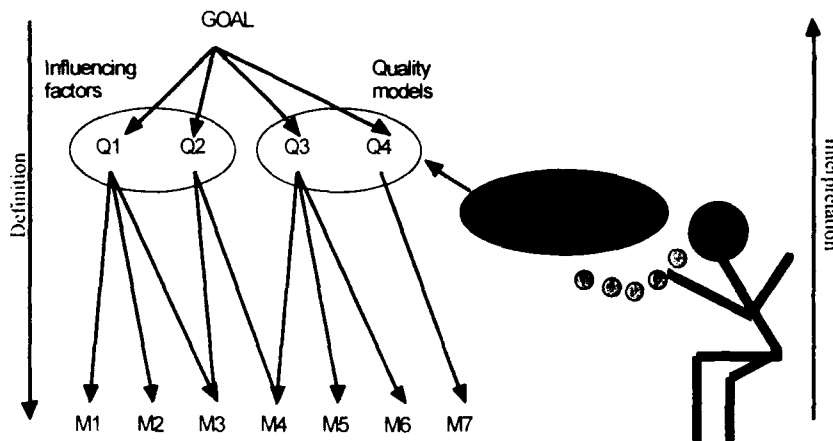


Figure 1: The Goal/Question/Metric paradigm⁶

GQM represents a systematic approach to tailoring and integrating goals with: models of the software processes, software products, and with particular quality perspectives of interest. GQM focuses on the specific needs of the software project and of the development organisation. Measurement goals are defined on the basis of high-level corporate goals, and refined into metrics. In other words, GQM defines a certain goal, refines this goal into questions, and defines metrics that must provide the information to answer these questions. The GQM paradigm provides a method for top-down metric *definition* and bottom-up data *interpretation* (Figure 1).

Application of GQM divides measurement into two parts.¹⁹ First, there is the definition process, during which goals, questions, metrics and extra procedures are defined. Second, there is the interpretation process, during which the collected data is analysed, improvements are identified, and experiences are described. Both are outlined briefly in the next paragraphs. This sub-division

does not include data-collection which precedes the interpretation process. Data collection is a separate activity that will not be discussed in this paper.

The '**definition process**' for software measurement is well described in the relevant literature. Many books and papers are available on selecting the particular metrics for a specific problem. GQM is mainly used as a technique to define metrics, even though metric definition and interpretation will influence one another.

The definition process for GQM measurement consists of the following activities:

- *Prestudy.* The prestudy examines and characterises the application context and project, in order to make current problems and goals explicit. The prestudy is an important preparation for measurement.
- *Measurement Goal selection.* During this activity informal improvement goals are described, refined, and ranked. Priorities are assigned and it is decided which goals will be used and transformed into GQM goals.
- *GQM planning.* GQM planning is the actual design of the measurements. The GQM paradigm is applied for defining a detailed tree of goal, questions and metrics. Interviews are held with project members to retrieve the necessary information.
- *Measurement planning.* Measurement planning is done to develop data collection procedures and introduce automated tools for data analysis. During this activity the initial GQM-plan is made operational for practice.

Based on the actual measurements an analysis can be performed that aims at answering the established questions, and reaching the identified goals. This process is referred to as the '**interpretation process**'. Research indicates that interpreting the measurement data is a learning process within a software team and of crucial importance to the success of the measurement programme.²⁰ Such an interpretation process typically comprises the answering of measurement questions and goals, and the identification of improvement actions. These improvement actions may refer to the software development process, but also to the improvement programme itself.

3. THE CAPABILITY MATURITY MODEL

The Capability Maturity Model (CMM) was developed by the Software Engineering Institute at Carnegie Mellon University.^{3&9} The CMM is an effective top-down framework to evaluate organisations according to the 'maturity' of their software processes. Currently the CMM is enhanced by means of other approaches based on the CMM framework, such as the European CMM called: 'BOOTSTRAP'⁵, but also a CMM version 2.0 is on its way. The CMM is used in this article to analyse the maturity of GQM and is briefly described in this section.

The CMM helps organisations improve their software development processes by suggesting a learning path ranging from chaotic to disciplined. Each maturity level adds particular capabilities that are necessary to develop software in an orderly manner. Since capabilities depend on each other, the CMM provides an improvement path, by introducing the capabilities (Key Process Area - KPA) one by one. The maturity framework can also be used to evaluate organisations and indicate the most relevant areas for improvement. To this purpose, the CMM uses assessments that position organisations on one of CMM's five maturity levels (Figure 2). The general characteristics of these maturity levels are described below.

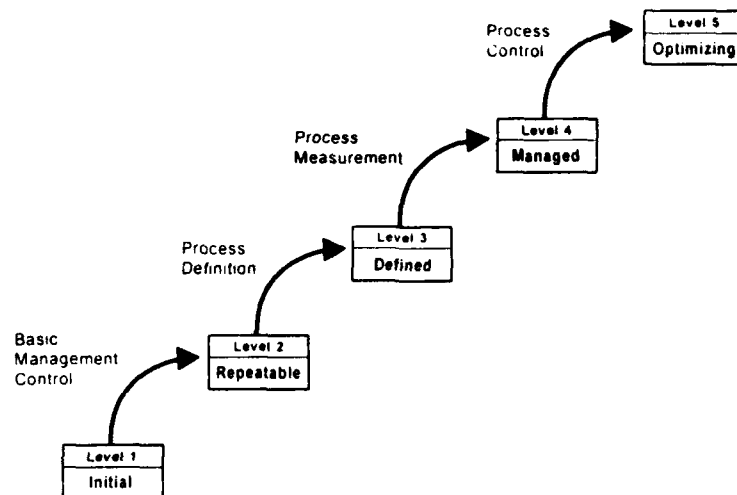


Figure 2: CMM maturity levels⁹

- Level 1: Initial process.** At level 1 the software process is characterised as ad hoc and sometimes chaotic. The inputs to the process are ill-defined, the outputs are expected, but the transition from inputs to outputs is a black box process. Success depends on individual effort and heroics.⁹ In our opinion this means the 'Just Do It' approach.
- Level 2: Repeatable process.** In the repeatable process, inputs, outputs, and constraints are identified. Project management processes are established to track costs, schedule, and functionality. It is able to repeat earlier successes with similar applications. However, still it is not clear how outputs are produced.⁹ In our opinion this means that a certain level of 'Understanding' is reached. This also reflects the first improvement areas indicated by Basili, who suggests to start 'understanding' focused measurement goals.⁶
- Level 3: Defined process.** At level 3 the organisation has defined a standard software process in which both management and engineering activities are documented, standardised, and integrated. The organisation has defined and documented the development process. The intermediate deliverables of development are well-defined and the process is decomposed into sub-activities necessary to construct the software.⁹ In our opinion this means that not only the process is understood, but it has also been 'Described'.
- Level 4: Managed process.** The organisation has initiated detailed process and product measurement and analysis. Both the software process and products are quantitatively understood and controlled. At this level, the most significant quality improvements begin.⁹ In our opinion this means that it becomes possible to 'Control' a process.
- Level 5: Optimising process.** An 'optimising process' is considered as the ultimate level of process maturity. The organisation has established a foundation for continuous improvement and optimisation of the development process. Continuous process change and improvement is enabled by quantitative feedback from the process and by introducing innovative ideas and technologies.⁹ In our opinion this is the 'mature' stage that everybody strives for. Again this is in line with Basili's ideas who suggests improvement oriented measurement goals, whenever control has been achieved.⁶

4. USING THE CMM MODEL TO EVALUATE THE GQM METHOD

The CMM is used to characterise the current status of the GQM method in this article. As with the evaluation of software development practices, the CMM will identify improvement areas but this time for a methodology, namely GQM. In this section the current state of GQM is assessed.

This current state of the GQM method is illustrated in Figure 3, by ordering available techniques and tools by CMM's maturity levels, split into the two areas: *definition* and *interpretation*.

	Level 1 INITIAL <i>'Doing'</i>	Level 2 REPEATABLE <i>'Understanding'</i>	Level 3 DEFINED <i>'Describing'</i>	Level 4 MANAGED <i>'Controlling'</i>	Level 5 OPTIMISED <i>'Improving'</i>
Measurement Definition	GQM-Paradigm	- Guidelines for definition - GQM Process Steps - Goal Template - Abstraction Sheets - Separate GQM-team - Spreadsheets	- Goal Definition - Dissemination - GQM Training - GQM Tutorials - Automated Tools - Hour recording - Defect Tracking - Planning & Tracking - Static Analysers		
Measurement Interpretation	GQM-Paradigm	- Feedback Sessions - Spreadsheets - Presentation Tools	- Planned Feedback - Database Tools - GQM Tools: - GQM aspect - Cefriel tool		

Figure 3: Current maturity levels of GQM

4.1 Level 1

The initial level contains the general idea of GQM is reflected in the GQM Paradigm. For both, *definition* and *interpretation*, the framework refines goals into questions and metrics.

4.2 Level 2

At the repeatable level several techniques are available. First of all a step-wise approach is available by which measurement is introduced and measurement programmes are defined. The GQM process steps are based upon the Plan/Do/Check/Act cycle and are tailored into the GQM process model. This model contains six steps: Characterisation, Goal Definition, GQM planning, Execution, Analysis, and Packaging.¹⁵ Several guidelines for the *definition* process are available. Those guidelines are not necessarily specific to GQM measurement because general software measurement literature is also of help.^{11&21&22}

In the TAME project a 'Go2! Template' was developed which supports a uniform way to describe a measurement goal.²³ The 'Abstraction Sheet' concept supports GQM planning and interviews by structuring the main aspects of measurement on a single sheet.¹⁵ Abstraction sheets support checking on consistency and completeness of a GQM plan. Also the concept of 'hypotheses' is introduced to support learning from measurement. Installation of a separate GQM-team guarantees the continuation of the measurement practices.⁷ Notice that one of the main reasons several measurement programmes fail in practice, is because of the lack of such a GQM-team.

The *interpretation* process is supported at the repeatable level by so-called 'Feedback Sessions'.¹⁶ Feedback sessions are meetings during which data are analysed by the development team with help from the GQM plan. These sessions are important to keep interest and motivation of the

development team, and have proved in practice to be *the* key success factor of GQM measurement.¹⁶ Support from automated tools for the interpretation process comes mainly from spreadsheets and other statistical representation tools.

4.3 Level 3

At the defined level, structured brainstorming techniques can be applied for the *definition* of measurement goals. An example of such a technique, which is GQM specific, is the 'Seven Questions' technique by which the definition of GQM goals from high level corporate goals is supported by answering questions.²⁴ Frequent dissemination of measurement results is institutionalised at level 3, for instance through papers, presentations or WWW-pages. Training of employees on the GQM concepts²⁵ becomes important to guarantee that everybody understands the underlying concepts. Typical examples of support from automated tools at the defined level are general purpose database tools, for instance, to track effort expenditure, to capture defect data, and to monitor conformance to planning. Automated metric collection may also involve static analysers that calculate specific metrics, for example, from source code.

Interpretation at the defined level is also mainly supported by feedback sessions, but because of the more sophisticated nature of the measurement programme, feedback sessions are now better planned. Support from automated tools still comes from spreadsheet and presentation tools. However, integration with database application might become feasible. Also some commercial tools²⁶ might become of interest because analysis of measurement data is already described before.

4.4 Level 4 and Level 5

Techniques of the GQM approach that provide support during the managed and optimised levels are currently not available.

5. MISSING ELEMENTS IN THE GQM METHOD

In the foregoing paragraph we have presented the current techniques for GQM measurement within the framework of the CMM. In order to bring GQM up to level 5, various additional techniques and tools should become available. In Figure 4 we present a similar table, but this one illustrates the missing elements of the GQM approach.

	Level 1 INITIAL <i>'Doing'</i>	Level 2 REPTBL <i>'Underst.'</i>	Level 3 DEFINED <i>'Describing'</i>	Level 4 MANAGED <i>'Controlling'</i>	Level 5 OPTIMISED <i>'Improving'</i>
Measurement Definition			• Link to Software Development • Detailed GQM procedures • GQM Handbook • Document Templates • GQM Inspections	• Effectiveness of measurement • Relation between process and product • Measurement of GQM activities	• Fully Integrated Tool Environment
Measurement Interpretation		• Guidelines for Feedback	• Data representation guidelines • Feedback Handbook • Fully automated feedback	• On-line Feedback • Decision Support • Cost/benefit analysis of measurement programme	• Fully Integrated Tool Environment • Cost/benefit optimisation of measurement programme

Figure 4: Missing elements in the maturity levels of the current GQM

5.1 Missing elements at Level 1

In the CMM, the first level is 'for free'. Therefore, at the initial level everything that is needed is currently available. Adopt the QQM paradigm and 'Just Do It'.

5.2 Missing elements at Level 2

For QQM measurement at the repeatable level there are insufficient guidelines regarding the *interpretation* process. Questions which are still unanswered are, for example, with what frequency should feedback sessions be held, how much information should be presented in a single session, and what amount of user involvement is advisable?

5.3 Missing elements at Level 3

In order to perform QQM measurement at the defined level, there is a need for procedures on how to do the *definition* process. The 'how?' question is still not fully answered. A QQM-handbook that describes the practical application of QQM is not available either, nor are templates for QQM documents. A second missing element in QQM is the link to software development. Besides the involvement of software developers, there is no software specificity in QQM. At a level 3 maturity of QQM, such a link should exist. Another missing element is some kind of inspection or audit by which the measurement practices themselves are evaluated.

The *interpretation* process is hardly defined at all. Something like a feedback handbook is not available, nor are guidelines on data presentation (some ideas are described in literature^{21&27}). At the defined level there is a need too for fully automated presentation tools, since many measurement projects need to be supported. Those automated presentation tools can be applied because of the defined nature in which the QQM measurements are carried out: feedback is planned, many questions are stable, and metrics are collected in a fixed way.

5.4 Missing elements at Level 4

The *definition* process at level 4 requires quantitative evaluation of the QQM measurements themselves. The effectiveness of measurement programmes should be clarified. The relation between software development and software product (quality, cost, and cycle time) should be established. Real 'control' needs to identify what the effects of specific measurement topics are, in order to select the most appropriate ones. The Esprit project PROFES²⁸ (Product Focused Improvement of Embedded Software processes) develops such techniques currently.

'Managing' the *interpretation* process, definitely needs on-line feedback. Measurement representations should always be available and up-to-date. Support during project decision making must be provided. Examples of suggestions would be, 'which module should we inspect?', or 'how many test effort is required?'. Tools that support such on-line measurement presentation are not available (yet).

5.5 Missing elements at Level 5

Looking ahead to level 5 is still difficult. The current QQM method is somewhere between level 2 and level 3. Therefore it is obviously easier to make suggestions for levels 3 and 4, than for level 5. Level 5 probably requires a fully integrated development environment in which software development, measurement *definition*, data-collection, and measurement *interpretation* are supported in an integrated fashion. All activities are expected to be optimised automatically based on the mature nature of the method. Exact specifications will be clearer on the way to level 5.

6. SCHLUMBERGER'S SOLUTIONS TO THE MISSING ELEMENTS OF GQM

In 1994, software measurement was introduced in the Research and Development department of Schlumberger Retail Petroleum Systems (RPS), by applying the GQM paradigm. This introduction was associated with a European Research project¹⁵ that had the objective to lift GQM to level 2. The fact that important elements were not yet described in the GQM method was immediately apparent. How this dilemma was (and still is) being resolved is described in this section.

Because of the commercial pressures on the Schlumberger industrial business, all improvements were focused on 'efficiency'. All the additions to GQM addressed the question of how to increase the positive benefits of GQM measurement by using less effort?

Schlumberger RPS developed several additions and improvements to GQM:

- Software development methodology is integrated with the GQM method (section 6.1).
- Detailed processes for definition are described (section 6.2).
- Guidelines for feedback sessions are defined (section 6.3).
- Tool support for Level 3 GQM measurement has been developed (section 6.4).

6.1 Integrating Software development models with GQM

To ensure correctness and completeness of the defined set of metrics, GQM identifies metric *definition* from two different perspectives:²⁴

- Metrics definition by members of the project team, using GQM modelling techniques;
- Metrics definition based on models of software processes and products.

By modelling both perspectives, two sets of metrics are identified that can be cross-checked on consistency and completeness, identifying subjects that were missed or badly defined. Both perspectives are illustrated in Figure 5.

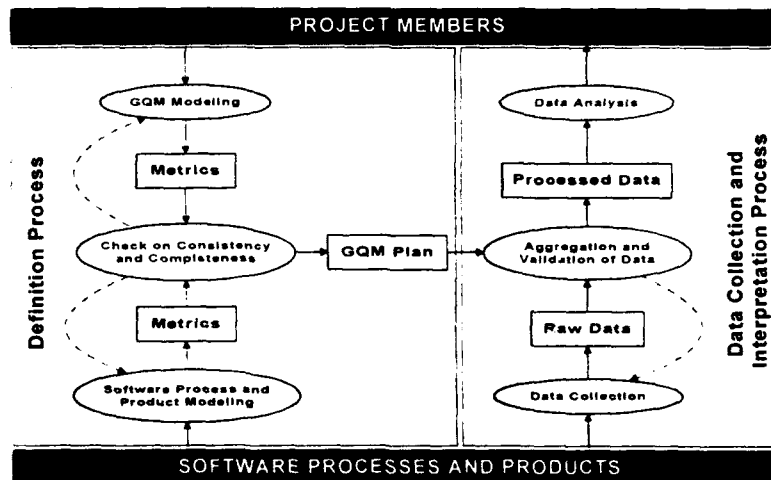


Figure 5: Applying process and product models to metric definition

6.2 Detailed definition process

During the *definition* phase of the GQM plan, a seven step procedure²⁴ is followed which results in a formal and documented definition of the measurement programme. The procedure will not be presented in detail, but the seven basic steps are illustrated in Figure 6.

6.3 Guidelines for feedback sessions

The Schlumberger RPS GQM team explored learning theories which were also applicable to the feedback processes of software measurement. Based on that research many guidelines were documented to improve the feedback process. Guidelines were identified for the project team, the GQM team, the feedback sessions, and for general management. More details have been published.^{20&29}

Those guidelines indicate why particularly the feedback process goes wrong so often. The GQM-Team should, for instance, have sufficient knowledge of the development process and possess particular personal skills to encourage the measurement programme. Another crucial point is the influence that the developers experience regarding the *interpretation* of the measurement data, or subsequent improvement actions. Learning theory also, again, illustrates that quality-focused measurement programmes are unsuitable to evaluate individual developers. This would violate almost every condition of the feedback process.

6.4 Tool support for Level 3 measurement

Schlumberger RPS developed their own software measurement tools, mainly focused on support during feedback sessions. Application of standard tools such as Microsoft Access, Excel and Powerpoint offered good starting points. Because of the object linking functionality in these tools it is possible to provide measurement analyses which are updated when new data is entered in the underlying databases. Support from such tools reduces the preparation time for feedback sessions from 3 days to approximately 2 hours. Currently no commercial GQM data analysis tools exist. To perform correct data analysis, three types of tools are needed.²⁹

- data *processing* tools, that group, order and calculate information as defined in the GQM plan.
 - data *representation* tools, that present information in bar charts, scatter-plots or tables.
 - data *displaying* tools, that present those charts on screen, in documents or in hand-outs.
- These three types of tools are not available yet, but are clearly needed for the interpretation process. Special requirement to such tools is that they are linked to each other.

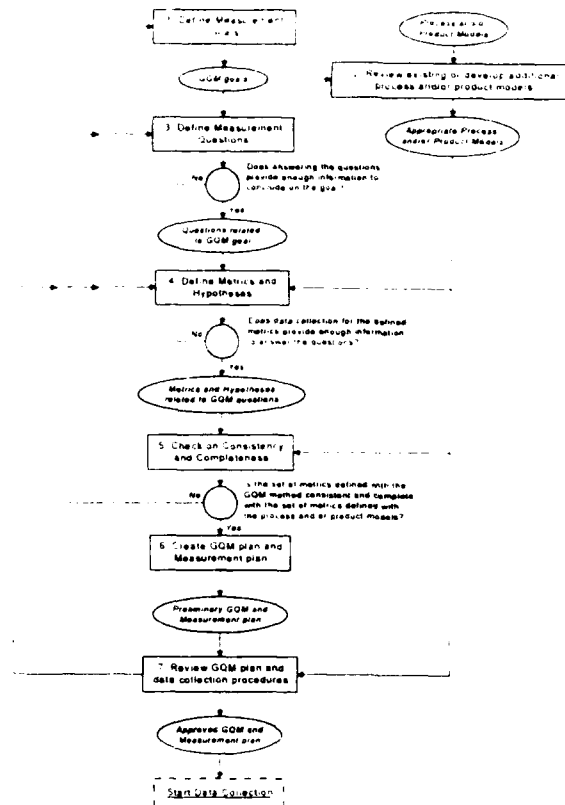


Figure 6: GQM definition steps

7. CONCLUSIONS

GQM is a powerful method that supports data collection, adjusted to a specific problem or organisation. GQM can evolve into *the* de-facto standard for software measurement. However, the current documentation on GQM is still incomplete. In this article the CMM is used to assess the GQM approach and to identify suggestions for its enhancement. This assessment indicates that the current documentation of GQM is somewhere between CMM level 2 and 3. Some support for 'defined' GQM measurement is available, but important elements are still missing.

In this paper many suggestions are described for improvements to bring GQM to level 5. Schlumberger RPS, that has applied GQM since 1994, already added a number of improvements to the method. The fact that those improvements come from industry reflect why they all aim to increase the efficiency of both software development and the quality improvement programme. Development of the GQM method is currently also done in international research settings. The Esprit project PROFES develops techniques aimed at bringing GQM to level 4.

Summarising, it is recommended to apply goal-oriented measurement in practice. Especially industrial organisations reported positive experiences with GQM, since both method and organisation are goal-driven. Schlumberger RPS will continue to apply and improve the GQM method.

8. ACKNOWLEDGEMENTS

The following persons contributed both directly and indirectly to this paper and we would like to acknowledge them: Andreas Birk, Erik Kooiman, Frank van Latum, Hans Leliveld, Ronald Loosekoot, Markku Oivo, Erik Rodenbach, the CEMP Consortium, and the PROFES Consortium.

9. ABOUT THE AUTHORS

Rini van Solingen received his M.Sc. in Technical Informatics from Delft University of Technology in 1995. He participated in the ESSI/CEMP project for Schlumberger RPS, during his Master of Science project. He then continued to work for Schlumberger RPS, as a quality engineer. In parallel he performs Ph.D. research at the department of Information and Technology at Eindhoven University of Technology. He is responsible for the software measurement projects in Schlumberger RPS and also advisor for several software measurement projects in other industrial organisations.

Egon Berghout is an assistant professor of Information Management at Delft University of Technology. He has been involved in software measurement for several years, especially through participation in the Schlumberger RPS measurement programmes. His research focuses on efficiency and effectiveness problems regarding information system development.

CONTACT ADDRESSES

Rini van Solingen, Schlumberger RPS, Industrieweg 5, 5531 AD Bladel, The Netherlands.
vansolingen@bladel.rps.slb.com

Egon Berghout, Delft University of Technology, Information Systems Department, P.O. Box 356,
2600 AJ Delft, The Netherlands, e.w.berghout@twi.tudelft.nl

REFERENCES

- ¹ W.W. Gibbs, Software's Chronic Crisis, *Scientific American*, September 1994, pp. 72-81
- ² M. Thomas, F. McGarry, 'Top-Down versus Bottom-Up Process Improvement', *IEEE Software*, July 1994 pp. 12-13
- ³ M.C. Paulk, C.V. Weber, B. Curtis, M.B. Chrissis, *The Capability Maturity Model: Guidelines for Improving the Software Process*, Addison-Wesley, 1995
- ⁴ A. Dorling, SPICE, Software Process Improvement and Capability Determination, *Software Quality Journal* 2 (1993), 209-224
- ⁵ P. Kuvaja, J. Simola, L. Krzanik, A. Bicego, G. Koch, S. Saukkonen, The BOO1STRAP approach, Blackwell Business, Oxford UK and Cambridge MA, 1994
- ⁶ V.R. Basili, G. Caldiera, H.D. Rombach, 'GQM Paradigm', *Encyclopaedia of Software Engineering*, volume 1, John Wiley & Sons, 1994, pp. 469-476
- ⁷ V.R. Basili, G. Caldiera, H.D. Rombach, *Experience Factory*, *Encyclopedia of Software Engineering*, volume 1, pages 469-476, John Wiley & Sons, 1994
- ⁸ AMI, A quantitative approach to software management, Published by South Bank University for the AMI Consortium, ESPRIT 5494, 1992
- ⁹ W.S. Humphrey, *Managing the software process*, SEI series in software engineering, Addison-Wesley, 1989
- ¹⁰ M. Oivo, *Quantitative management of software production using object-oriented models*, VTT Publications, Oulu, 1994
- ¹¹ Fenton, N.E. and Pfleeger, S.L., *Software Metrics, a rigorous and practical approach*, Thomson Computer Press, 1996
- ¹² V.R. Basili, D.M. Weiss, 'A methodology for collecting valid software engineering data', *IEEE Transactions on Software Engineering*, SE-10(6), 1984, pp. 728-738
- ¹³ R. Bache, M. Neil, 'Introducing metrics into industry: a perspective on GQM', In: *Software Quality Assurance and Measurement: A worldwide perspective*, Edited by N. Fenton, R. Whitty, Y. Itzuka, Thomson Computer Press, 1995
- ¹⁴ C. de Zeeuw, *improving the Process: The design and implementation of a software measurement information system in Schlumberger RPS*, Schlumberger RPS & Delft University of Technology, 1994
- ¹⁵ CEMP-consortium, *Customised Establishment of Measurement Programmes*, Final report of the ESSI/CEMP project, 1995
- ¹⁶ B. Hoisl, M. Oivo, G. Rube, F. van Latum, No Improvement without Feedback: Experiences from Goal-Oriented Measurement at Schlumberger, Proceedings 5th European Workshop on Software Process Technology, Nancy, October 1996, Lecture Notes in Computer Science Vol. 1149, 167-182
- ¹⁷ R. van Solingen, *Goal-oriented software measurement in practice: Introducing software measurement in Schlumberger RPS*, Schlumberger RPS and Delft University of Technology, 1995
- ¹⁸ R. Kusters, R. van Solingen, J. Trickens, 'User-perceptions of embedded software quality', In: Proceedings of the STEP workshop, IEEE 1997
- ¹⁹ R. van Solingen, E. van Veenendaal, 'Improvement by GQM measurement', In: *Software Quality from a Business Perspective*, by J. Trienekens and E. van Veenendaal, Kluwer, 1997
- ²⁰ E. Kooiman, R. van Solingen, E.W. Berghout, 'Rapid Interactive feedback', Proceedings of the 7th ESCOM conference, 1996
- ²¹ R.B. Grady, *Practical Software Metrics for Project Management and Process Improvement*, Prentice Hall, 1992
- ²² P. Goodman, *Practical implementation of software metrics*, London, McGraw-Hill, 1993
- ²³ V.R. Basili, H.D. Rombach, 'The FAME Project: Towards Improvement Oriented Software Environments', *IEEE Transactions on Software Engineering*, Vol. 14, No. 6, June 1988, pp. 758-773
- ²⁴ Solingen, R. van, Latum, F. van, Oivo, M., Berghout, E., 'Application of software measurement at Schlumberger RPS: Towards enhancing GQM', Proceedings of the 6th ESCOM conference, May 1995
- ²⁵ A. Birk, R. Kempkens, Introduction to goal-oriented measurement, Tutorial Package, & Participation of project teams in measurement programs, Tutorial Package, ESPRIT Project 9096 "Perfect", 1995
- ²⁶ Two public tools for GQM measurement are the CEFRIEL tool, and GQM-aspect tool
- ²⁷ D.P. Youll, *Making software development visible: Effective project control*, Wiley, Chichester, 1990
- ²⁸ For information look on WWW-page <http://www.ele.vt.fi/profes/>
- ²⁹ E. Kooiman, *Towards a Feedback Theory*, Schlumberger RPS & Delft University of Technology, 1996/9



Software Metrics - Real World Experiences

Company Background

- 5th largest UK Life company
- Established 1810
- Funds under management £ 20B
- 3300 Staff
 - 2600 in two main Bristol sites
 - 200 in London with small IT support
 - Local branches
- 500 IT Staff
- 3 Software Metrics Staff



Software Metrics - Real World Experiences

Introduction

Reasons For Starting A Metrics Program

- Outsourcing
- Productivity Reporting
- Expenditure Analysis
- Input to Business Decision Processes



Software Metrics - Real World Experiences

Establishing the Metrics Program

Senior Management Support

- 'Quick Wins'

Management Benefits

- Statistics
- Estimating
- Management Reporting

Management Commitment



Software Metrics - Real World Experiences

Identify Software Metrics to Collect

Establish Metrics Characteristics

- Minimum Essential Data Collected
- Purpose Defined and Communicated to Those Affected
- Collected Once
- Collected at Appropriate Time By Appropriate Person
- Accuracy verifiable
- Used and Reported in a Consistent Way
- Held for an Agreed Period
- Destroyed When No Longer Appropriate

Tuesday 17 June

(T201a-6) S-2



Software Metrics - Real World Experiences

Identify Software Metrics to Collect

Establish Metrics Characteristics

- Validity Reviewed When Changed or, At Least Annually
- New Data Collection or Reports to be Approved
- Minimal Retrospective Data Capture



Software Metrics - Real World Experiences

Function Points

Establish Internal Guidelines

CheckPoint and MK II Project Estimation



Software Metrics - Real World Experiences

Work Effort

Establish Work Effort Metrics

Work Effort Measures

- 'Bums on Seats'
- Project Time
- Person Day



Software Metrics - Real World Experiences

Spreading the Message

Metrics Collection Book

Roadshows

SLFPUG

Function Points Clinic

Function Points Counter Recruitment

SUN LIFE

Software Metrics - Real World Experiences



METRICS CONVENTIONS



SUN LIFE

- ★ 1449 Working Hours Per Year
- ★ 207 Productive Days
- ★ 222 Office Days
- ★ 9 Days Communications
- ★ 6 Days Divisional Initiatives

SUN LIFE

Software Metrics - Real World Experiences



METRICS CONVENTIONS



SUN LIFE

- ★ **All Resources** contributing 5% or more to a project are included in Effort and Cost Figures
- ★ **Productivity is**
Function Points per Staff Month
- ★ **Production Cost is**
£'s per Function Points



Software Metrics - Real World Experiences

Function Points at the Crossroads

Number Of Function Points Counters

Options



Software Metrics - Real World Experiences

Where Are We Now ??

Function Points

Software Metrics

Estimating

Conclusion

Sharing Data



SUN LIFE

SUN LIFE ASSURANCE

SOFTWARE METRICS - REAL WORLD EXPERIENCES

John Holt
Software Metrics Analyst

Introduction.

Why did we start our metrics program ? Our impetus came mainly from the business but our recent migration from our CDC Cyber environment to an IBM platform also focused business attention on our IT department.

Business areas have been under the productivity spotlight for some years now. There have been all sorts of initiatives for measuring their productivity and time utilisation. Productivity measurement and metrics has traditionally been difficult to collect for I.T departments due to the problem of quantifying a largely abstract instead of mechanical process. Also, IT is now being seen as an area that needs to be more integrated with overall business processes and corporation objectives.

However, if IT is to be involved in setting business objectives then it must accept that it will be measured as a business area. This means that software metrics must be put in place to measure expenditure, effectiveness and productivity.

Outsourcing has become a cost effective way for companies to continue to have the benefits of their IT departments but few of the problems. This was one of the results of the 'return to core business' strategy that is still going on today. Those IT departments that have avoided outsourcing have been those that can prove to their company board members that it is as cost effective, if not more so, to retain the IT function within the organisation as it is to outsource it.

Software metrics have the capability to significantly influence business decisions. Sun Life decided to centralise our metrics function into one area and expand the number and quality of metrics we collected. So the software metrics program was established.

Establishing the Metrics Program

The most important ingredient for a software metrics program is senior management support. Often, management establish a metrics program because it is fashionable. The program never really becomes established. Management wants quick wins and a magic bullet that will solve all their problems by delivering productivity statistics and

management reporting metrics the instant the program is set up. The reality is that any metrics program needs to be carefully thought out. Many statistics need to be collected over a long period, typically 18 months, before they can be used meaningfully. This doesn't lend itself easily to the provision of 'quick wins'.

Sun Life senior management realised not only the need for a metrics program but also the benefits they could get from it in terms of management statistics, estimating and reporting. The ability to use software metrics to do estimating, based on Sun Life's own data, was one of the key incentives to start the metrics program. By using our own data, we could develop a technique that would be exclusive to Sun Life and would take into account all those things that make one company's IT department different from the rest. Also a good estimation technique would enable us to give to the business a better idea of how much their proposed IT solutions were going to cost, enabling them to establish a more accurate business case.

Another benefit from strong senior management support is that the staff on the project teams then realise that management are serious about metrics and they will continue to be collected. This positive attitude was particularly important in senior management meetings when the doubters stood up to have their say about metrics. It took time to win them over, especially when, on occasions, there was insufficient metrics data. Some concerns about metrics appear as if they'll be with us forever. One lesson we've learnt is that selling metrics is an ongoing task. Any doubts expressed need to be neutralised quickly if the whole program is not to be discredited.

Identify Software Metrics to Collect.

The next issue was what metrics to collect? It's easy to collect metrics but the objective was to collect metrics that would provide the areas who wanted them with relevant and meaningful analyses. Before we looked at the metrics we should collect, we realised that if we went out and measured everything, we'd end up with an awful lot of data but no idea of how useful it was. Our manager sat down and compiled a list of characteristics against which each metric should be examined.

This was:

- **Minimum Essential Data Collected.**
We were careful not to overburden the metrics providers by asking them for too much data.
- **Purpose Defined and Communicated to Those Affected.**
We wanted the project teams to understand why we were collecting the metric.
- **Collected Once**
We were anxious not to keep asking the teams for the same data. So we made sure that our metrics collection forms didn't duplicate metrics requests. We also tried to eliminate from the forms data which we could either source from elsewhere or derive from existing data.
- **Collected at Appropriate Time by Appropriate Person.**

We targeted our metrics collection forms at people we knew had or should have the data we required. For example, project leaders should have cost and effort estimates and actuals.

- **Accuracy Verifiable.**
We ensured, as far as possible that the data we were being given was accurate. This could be difficult, especially with estimates. As the project scope changes in the early stages, the estimate obviously changes. We were always aware that we might not have the latest estimates.
- **Used and Reported in a Consistent Way.**
For the maximum benefits to come from metrics reporting, the charts and graphs produced must be meaningful and understandable. We spent some time establishing the format and presentation of our analyses, bearing in mind that some of our analyses would be for board level senior management. We were anxious that the right message should be conveyed.
- **Held for an Agreed Period.**
We intended to roll up metrics when they reached a stage where analyses on a detailed level is no longer required. Practically, this hasn't happened yet. Some of our long term analyses still require access to detailed level data.
- **Destroyed When No Longer Appropriate.**
- **Validity Reviewed When Changed or, At Least, Annually.**
We perceived that some data will no longer be relevant. For example, data that was collected relating to our old Cyber data system would be of minimal use now. Likewise, data relating to our Cobol systems will be irrelevant at some time in the future.
- **New Data Collection or Reports To Be Approved.**
As we continue to develop our reporting processes, we try and circulate draft copies of our reports and charts to as many people as possible. Their comments help us refine and improve our reports until we reach a stage where they are approved for inclusion in our various management reporting procedures.
- **Minimal Retrospective Data Capture.**
Benchmarking exercises are the main problem here. For our first exercise we had to go back to the project leaders for retrospective data capture as the data just wasn't available. For the second benchmark we were considerably better but we still found small pockets of data we didn't have. We've just completed an overhaul of our metrics collection system and have included all the metrics we need to have for next years benchmark plus some others we think will become useful in time.

The basic building blocks of any software metrics program are system size and work effort. However these are measured - Lines of Code, Function Points, Days, Weeks, Hours - is irrelevant. Just combining these two figures will give a crude productivity measure - a metrics 'quick win'. We decided lines of code was not an option for us. If we had thought about it, the sheer mix of development languages in our organisation would have meant that any results we would have obtained would have been useless.

Function Points

Our next step was to try and get some idea of the size of our systems. We employed a team of outside consultants who offered us a package which included measurement in Function Points, of some of our current systems and a training course in the Function Points technique. The consultants happened to use Symons MK II. A group of our project staff were trained in that technique. An internal user group was set up and, for approximately a year, things went reasonably well. It does take a while for a Function Points program to become established and this was no exception.

Although we used the rules established by the MK II methodology, we needed to establish our own internal guidelines, based on the MK II rules. This was because generic rules are fine as a road map but they can't be expected to cover, in detail every variance of system, platform and development methodology in an organisation.

As a corporate goal Sun Life determined that it wanted to be a world class company. In order to achieve this, it's necessary to know what the other world class companies are doing and also to learn from them. Sun Life's Senior IT manager went on a fact-finding mission to the United States for this purpose. Whilst there, he was shown an estimating tool called CheckPoint from SPR. As one of the most desirable spin-offs from the metrics program is project estimation, the chance to buy an automated estimation tool that would take in Function Points and produce an estimate, based on an already well established base of data is a major selling point.

At this time, we were busy using and promoting the MK II Function Points methodology. Whatever the virtues or otherwise of the MK II method it doesn't, currently, have an automated estimating tool that will take in MK II Function Points and give reliable estimates. There are estimating tools in the marketplace that will take in MK II Function Points but, to date, they have not had any official endorsement from either the originator of the MKII methodology or UKSMA, the MK II standards body. Productivity statistics from MK II are sketchy and are not so well established as Albrecht so estimating from empirical data can be difficult.

The major drawback of CheckPoint was that it would only accept Albrecht Function Points as input.

We installed CheckPoint. This meant that we abandoned system sizing using MK II and switched to Albrecht. This caused concern amongst the MK II Function Points counters who were against changing the Function Points methodology. But, being able to input Function Points directly into CheckPoint was too important an advantage.

Work Effort

How many days in a work year ? How many days in a work month or week. Even how many hours in a day ?

The answer is: it depends. It depends on what industry sector you're working in, what company you work for and, even, what department you're in. But, if you accept that one of the most important components of productivity metrics is work effort then, clearly, establishing an answer to this question is essential.

We had to come up with a definition of work effort which we could not only use in our software metrics but one that would also be relatively easy to collect.

We currently do not have a person based time recording system such as PMW or MS Project installed throughout Sun Life but we feel this is essential if accurate work effort figures are to be used as a management reporting metric. At the moment, our work effort data is derived from several sources: project team returns, management reports etc. Introducing a time recording system involves support, not only from senior management but also from the lowest level in the project team as it involves a considerable organisational cultural change. So we've adopted the introduction of such a system as an important objective.

Everyone from senior management to our finances area could see the benefit in a work effort metric but it needed to be defined. The first problem, as was previously referred to, was defining a work day. Here's a sample of alternative ways of doing this:

- 'Bums on Seats'
As long as the person is seated at their desk, the total time is counted. So, if a person starts at 8:00 and finishes at 5:00, that's 8 hours, assuming an hour break for lunch. If they come in at 10:00 and leave at 4:00, that's a 5 hour day.
- Project Time
The person clocks on and off the project. If a person is working on multiple projects then they will record their time to each project as they expend effort on it. This also means they record the exact times they've spent on lunches, tea breaks, cigarette breaks, 'comfort breaks', nose blowing.
- Person Day
A person is assumed to work a standard day, whatever their true hours. If the standard person day is 7 hours then that's what's recorded. If they work 10, 11 or 6 hours, it's irrelevant. Within the 7 hour day, their time would be split amongst however many projects they're currently assigned.

We decided on the standard person day. The first method was too crude to deliver good productivity figures on a project by project basis. The second method is unworkable unless there's a person based time recording system in place. So we were left with option three.

One disadvantage with both option one and three is unrecorded overtime. We do ask the project team for this figure but it is rarely available.

I've seen in one of my previous organisations a situation develop in a project where the project leader would move times between phases to try and balance the effort out. So, if he was running out of time in the analysis phase, he'd allocate time from the following phase, hoping that he'd make the time back up. The other method of

manipulating work effort estimates is to overestimate the total project. I've seen situations where the project leader has given an estimate that's about 50% above his realistic estimate, knowing that the business area would cut it back.

We adopted a 6 hour standard person day and 261 office days per person per year. From this, we deducted:

- 32 Days for Holidays and Bank Holidays
- 4 Days for Sickness
- 3 Days for Training
- 9 Days for Communication exercises, Company Briefings, non-project team meetings etc.
- 6 Days Divisional Initiatives
Extra non-project work but benefiting IT as a whole e.g., Function Points counting, Development Life Cycle reviews, TickIt Auditing.

This leaves 207 productive person days devoted 100% exclusively to project work. We still do not know:

- The amount of unrecorded overtime in a project.
- The 'non productive time' which occurs in every organisation e.g., tea breaks, social interaction etc.

Unrecorded overtime is a difficult metric to collect. I've heard industry experts suggest that this figure could be as high as 30% of the total project effort so it's clearly a metric we'd like to gather to determine the effect on project delivery times.

The figures for a standard person day were only established after many meetings. It was surprising to see the strength of opinions which emerged as, it seemed, everyone had their own definition of a standard person work day. We had a group of about 6 managers who wanted to ratify these figures. Eventually, after hard lobbying from our most persuasive team members to each manager in turn, and discussing what we were trying to achieve, we obtained consensus.

We determined that productivity would be expressed as Function Points per Person Month and Production Cost as £ per Function Point.

We produced a handout to reinforce the message:

METRICS CONVENTIONS	
★ 1449	Working Hours Per Year
★ 207	Productive Days
★ 222	Office Days
★ 9	Days Communications
★ 6	Days Divisional Initiatives
★	All Resources contributing 5% or more to a project are included in Effort and Cost Figures
★	Productivity is Function Points per Staff Month
★	Production Cost is £ per Function Point

Project Life Cycle Metrics

It was essential to integrate metrics into Sun Life's System Development Life Cycle. We made the provision of estimates a project deliverable for the Initiation, Systems Design and Implementation phases of the life cycle. By including these tasks in the Life Cycle, we were able to enforce at least two fundamental aspects of the Metrics program - Function Points and Estimating. The project manager was required to produce at least one estimate for each of the three phases of the Life Cycle and one of these had to be an estimate derived from the CheckPoint estimating software package. In order to get an estimate from CheckPoint, they also had to have completed a Function Points count since CheckPoint would only estimate from Function Points so Function Points also gained acceptance as part of the overall project deliverables.

The next task was to establish the software metrics credibility with the project teams and convince them to record the data for us.

Any group of individuals have an inbuilt resistance to being measured, especially when it hasn't been part of the culture of the organisation. They may think that management want to monitor them. Another objection to collecting the metrics was that it's an extra overhead - time that could be better spent actually getting on with the project. This is understandable. Project teams are formed, generally with one objective - to design, develop and deliver a project that will satisfy a business need. Any incidental tasks that are perceived as sidetracking or hindering them from achieving this objective are bound to be looked on as less important.

We tried to make the project data collection as easy as we could. We developed a project data collection book. We designed a series of forms similar in style to a spreadsheet. Each form corresponded with different stages of the project life cycle because some data that was required in one stage might not be required for all stages. The forms were structured so that the instructions for completion were on one side of an A4 sheet with the boxes on the other side.

The final edition of the book was quite large and many project leaders found difficulty in completing it, mainly due to time constraints. Recently, taking input from the project leaders, we've reworked the book and managed to condense it. We've also made more use of colour which has made the whole book easier to understand.

Spreading the Message

We realised that the actual collection of the metrics by the project teams would be a problem. We knew we had the support of senior management so we decided to set up a roadshow to explain the reasons why we were collecting the metrics and the benefits we would gain. We obtained a portable notice board and divided it up into the 3 areas of metrics we specifically wanted to address:

1. Estimating
2. Function Points
3. Project Metrics

Each of us then prepared a short presentation. We targeted project teams and took our roadshow to each of the IT areas. We usually tried to synchronise our arrival with team activities such as briefings so that we could add our presentations to the end while the whole team was still present. Often, this wasn't possible so we set up in a convenient meeting room and encouraged team members to walk in and look around. We made ourselves available to answer any questions. Sometimes, the teams wanted formal presentations. At the end of the series, we'd covered every IT area in Sun Life, in Bristol and London.

With the Metrics Rules and Data Collection book produced and the road shows over, we began to receive projects for Function Point counting. Since the project couldn't progress without a Function Points count and CheckPoint estimate, we were under a lot of pressure to turn the counts over as fast as possible. There was only one problem. I was the only one in Sun Life who had substantial previous Albrecht experience although some others had been through a training course the year before. Clearly, it was time to utilise this base of trained but inexperienced personnel.

A Sun Life Function Points User Group was established to spread the Function Points message. As we had most of the people who had been on the MK II and Albrecht training courses in the user group, we also started a Function Points clinic. The idea behind the clinic was that people who had an interest in actually counting Function Points would attend the clinic. Counts would be handed out and any counting problems would be discussed within the clinic. The attendees would complete 4 counts or attend 8 clinic meetings whichever came first and then half would split to form another clinic and new members would join both clinics. By arithmetic progression, we hoped to eventually cover all the project teams with people who had knowledge of the Function Points counting process.

Some of the project managers of our counters were a bit suspicious of the process. They didn't entirely trust Function Points. They'd heard that Function Points would give any figure you wanted, provided you manipulated the processes sufficiently. We stressed that we were counting according to an industry recognised standard, that we had the manuals for them to look at, if they needed, and that their resident expert had ten years experience. This satisfied most of them and they were prepared to let their team members take part.

Recruiting enough counters was and still is, a concern. Counters are taken from the project teams on a part-time basis so we are asking them to do counts in their spare time on the project teams. We had allocated 6 days out of the year for Divisional Initiatives which included tasks like these. Normally, this time was rarely used so the project managers were happy to treat this time as additional project development time. So, some of our counters came under pressure to concentrate on project work as their main priority. This caused problems for us because we were then unable to give accurate estimates to other project leaders of the likely delivery time of a count. It was also slightly embarrassing as we were supposed to be promoting the importance of accurate project estimating while we weren't able to give project leaders any idea of when their counts would be ready! Where we knew this was happening we'd try and

resolve it either directly with the project managers concerned or with the project manager's senior manager.

Sun Life's Efficiency Initiative.

In the middle of last year, we had a company-wide initiative launched called Energy. This was in response to the same market pressures that are affecting every other company where the drive is on to produce more with less. As a result of this, we found that our data collection and metrics analysis services were more in demand because, if management wish to improve efficiency, they first have to establish a base efficiency measure. The initiative also had the effect of reducing the number of counters as more of their time was needed on the project teams. A number of our counters either removed themselves from the counting process or stated that they were unavailable for counts in the medium term.

Function Points at the Crossroads

All our initiatives last year, together with strong management support persuaded the project teams that Function Points, estimating and software metrics in general were the way of the future. Now, we found ourselves in a position where the project teams were more than willing to send us their projects for counting and estimation but we had few counters to actually do the counts. We therefore quickly accumulated a backlog. We needed to address this situation quickly, otherwise we would find that the project teams were likely to begin to lose commitment to the process. Recruiting new Function Points counters was difficult. Function Point counting is not a glamorous skill like Java or designing Web pages. We overcame the problem by again drawing on the support we have from our senior management. They backed us when we approached the project teams to recruit counters. However, this was not a long term solution.

Training was another issue. I could have designed and delivered a training course but I was too busy counting projects to have the time to do it. We considered outside training courses to be too expensive.

We found ourselves in a situation where we were considering two options for the long term future of Function Points counting:

1. Continue to draw our counters from the project teams.
2. Bring counting inside the metrics area.

If we kept recruiting our counters from the project teams, we could eventually have accumulated too many counters. In this case we would have had a problem with trying to maintain counting consistency. This would be because each counter wasn't doing enough counts during the year to keep their counting experience current.

If we brought the counting process inside the metrics area we would:

1. Have greater control over consistency.

2. Would keep skills current.
3. Would quickly develop experts.

But, we could then give the impression that Function Points counting was an elitist task and project teams submitting their projects for counting would have the impression of putting them into a dark cloud with no control or knowledge of the internal processes involved as well as no commitment to the results.

We decided, on balance, that recruiting from the project teams would be preferable to keeping the process centralised. We would overcome the consistency question with distribution of the IFPUG Counting Practices manual supplemented with our own local guidelines which would deal with Sun Life specific counting issues. The solutions to these local issues would be based on the IFPUG manual.

Benchmarking

Although the collection of software metrics is an essential aid to understanding the IT processes, metrics collected in-house will only be able to be used for in-house analyses. One of our corporate objectives is to be world class. Clearly, one of the first steps in achieving that goal is to try and establish where we are now. One of the ways the IT area can discover this is to submit their data to one of the benchmarking companies. We submitted our data to CSC-PEP for analysis. In return, they provided us with analysis of where we were, not only in comparison with all companies on their database but also companies within our industry sector. Our first effort last year produced mixed results for us. We established that we were good in some areas, bad in others and average in others. Within our industry sector, we were average. This is probably due to a number of reasons, the lack of maturity of the data being one of them in that some of the data we were asked for, we had to estimate. However, it did give us some meaningful data to use in order to establish a 12 month productivity target.

Management decided that we would enhance our development environment of continuous improvement by introducing new initiatives to improve our project delivery times. This meant embracing new development methodologies like RAD and JAD through to the establishment of 'hit' teams to identify and help projects that, for one reason or another, were underperforming. We also placed a greater emphasis on the quality of the software delivered. The actual figures aren't relevant because in a continuous improvement cycle, they are changing all the time and it's usually sufficient to ensure that the figures are improving.

Gathering benchmarking data is an interesting exercise. Project teams in general are reluctant to collect metrics, for reasons I've previously discussed. In rare instances we had work effort data being recorded using different bases at different stages of the project.

At the end of the process, CSC-PEP produced our internal results followed by a workshop for senior managers. It was at this stage that the project leaders realised the impact of the data they were giving us was having on their senior managers. Suddenly, it was their project under the spotlight. Consequently, I was involved in discussions for projects that were under management scrutiny for one reason or another and revising their Function Points counts. As one project leader said to me: 'It's surprising how, having your project under the management spotlight, focuses your mind!'

Where are we now ?

Function Points.

We still have a backlog of counts. This is inevitable due to the budgetary process within Sun Life. Most project budgets are approved early in the year so that's the time when most projects start up. Even at a counting rate of 50 Function Points/Hour, that still isn't fast enough to keep up with the project startup rate. In addition we have projects that start during the year which will address legislative changes and new market opportunities.

Software Metrics.

Since the last CSC-PEP benchmarking exercise, the profile of metrics is steadily increasing. Project teams are beginning to realise the importance to management and are starting to be proactive in coming to us to ask us to provide their Function Points counts and estimates. They are also being more careful with the project data returns at implementation and providing better quality data as a result.

The important part of the metrics loop we're now concentrating on is the feedback to the project teams of the metrics they've provided. We are looking at graphs and analyses that may provide:

- Productivity Rates of project teams in Sun Life's IT and individual areas.
- Error Rate Detection
- Budgeted vs. Actual Effort and Expenditure
- Accuracy of estimates
- Development life cycle analysis
- Maintenance effort per implemented project
- Team role distribution effort.
- Number of Defects per Function Point.

We're also looking at the testing process to see if we can use software metrics to improve our error detection rate, especially in the earlier stages of the project life cycle e.g., Systems Design or Functional Specification and generally assist the project teams at this stage.

Estimating.

Our CheckPoint estimating package has given us some interesting insights. For example, it's shown us that project leaders could afford to be more aggressive with their estimates as they tend to overestimate the time required to

deliver a project. The CheckPoint package, combined with the Function Points count is giving us estimates that are within +/- 10% of project actuals. Confidence among the project leaders in the package is steadily growing. Our long term aim is to make CheckPoint the primary estimating package within Sun Life.

Metrics Focus

We have devolved some of our IT function to the business areas. This has changed the emphasis of our metrics away from justifying our IT productivity and spend to the business to one of providing metrics to the business to input into their business cases and helping them in their decision making processes.

Conclusion.

Metrics started in Sun Life with lots of enthusiasm from both senior management and the project leaders. The enthusiasm waned in the project teams, late last year and early this year. This was partly because while we were establishing the software metrics initiatives we had not yet shown how they fitted in to the development environment. However, lately, there's been another wave of interest shown in metrics. I think this is due to the project teams realising that metrics are here to stay and can deliver benefits to them. This makes our task easier. We are establishing links with other companies within our umbrella organisation of AXA-UAP such as Colonia in Germany and hope to exchange metrics data with them on a regular basis.

In conclusion, I hope I've offered some insights into what to expect in your own metrics programs. Although we had problems, we managed to solve most of them by a combination of management support, team skills, enthusiasm and a belief in what we were trying to achieve. We haven't finished yet, there's still a lot of work to do but we've made an excellent start. I wish you every success in your own metrics program.



Have You Heard ?



- We are living in the 'age of information'
- Information to double every three years
- Quality of information becoming harder to assess



Chris Herbert Lloyds TSB Group European SEPG June 1997

© Lloyds Bank plc & TSB Group plc 1997 All rights reserved



The MIAMI Experience



**Practical Experiences of Introducing
and Establishing a Successful Metrics
Program or, in other words :**

Extracting Needles from Haystacks : Painlessly!

Chris Herbert
MIAMI Project Manager
Lloyds TSB Group plc

Chris Herbert Lloyds TSB Group European SEPG June 1997

© Lloyds Bank plc & TSB Group plc 1997 All rights reserved



Agenda



- Building a Foundation for Success
- Achievements and Learning Points
- Next Steps

Chris Hubbard | Lloyds TSB Group | European SEPG | June 1997

© Lloyds Bank plc & TSB Group plc 1997. All rights reserved.



Lloyds TSB Group



- Banking & Financial Services
- Established 28 December 1995
- 2,810 High Street branches
- 82,000 employees
- Group assets : £147 billion
- Sixth largest UK quoted company on a market capitalisation of £27 billion



Chris Hubbard | Lloyds TSB Group | European SEPG | June 1997

© Lloyds Bank plc & TSB Group plc 1997. All rights reserved.



Lloyds TSB Group



Information Systems :



- Software Development and Support
- 1,400 employees
- Nine UK sites



Chris Harbison | Lloyds TSB Group | European SEPO | June 1997

© Lloyds Bank plc & TSB Group plc 1997. All rights reserved



Building a Foundation for Success



Background

- Previous unsuccessful metrics initiatives
- Strong process improvement culture
- Investment in new tools and methods
- Benchmarking
- Development of Business Plan

Chris Harbison | Lloyds TSB Group | European SEPO | June 1997

© Lloyds Bank plc & TSB Group plc 1997. All rights reserved



MIAMI



Metrics Involving All

equals

Motivated Improvement

Chris Harbour Liberty TSB Group European SEPG June 1997

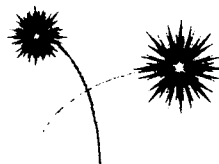
© Liberty Bank plc & TSB Group plc 1997 All rights reserved



Conception & Birth



- 3Q95
- Five corporate metrics
- Five year rolling programme
- A cohesive system
- Facilitate cultural change to measurement
- 'Healthcheck' on the business



Chris Harbour Liberty TSB Group European SEPG June 1997

© Liberty Bank plc & TSB Group plc 1997 All rights reserved



The Corporate Metrics



- Productivity (Size/Work hour)
- Delivery Rate (Size/Elapsed Week)
- Product Quality (Defects/Size)
- Process Quality (Defect removal efficiency)
- Customer Satisfaction
- Staff Satisfaction

Chris Harbert, Lloyds TSB Group, European SEPO, June 1997

© Lloyds Bank plc & TSB Group plc 1997. All rights reserved



Establishing the Project



- Gathered requirements (G-Q-M)
- Established steering committee/sponsors
- Identified stakeholders and customers
- Communicated key messages
- Advertised, Advertised, Advertised



Chris Harbert, Lloyds TSB Group, European SEPO, June 1997

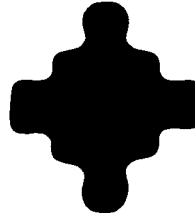
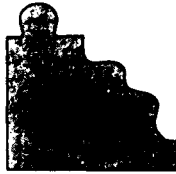
© Lloyds Bank plc & TSB Group plc 1997. All rights reserved



Metrics and Measurements : 1 **TSB**

**Expended
Effort**

**No of
Defects**



Chris Harbert, Lloyds TSB Group, European SEPG, June 1997

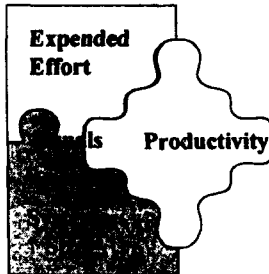
© Lloyds Bank plc & TSB Group plc 1997. All rights reserved.



Metrics and Measurements : 2 **TSB**

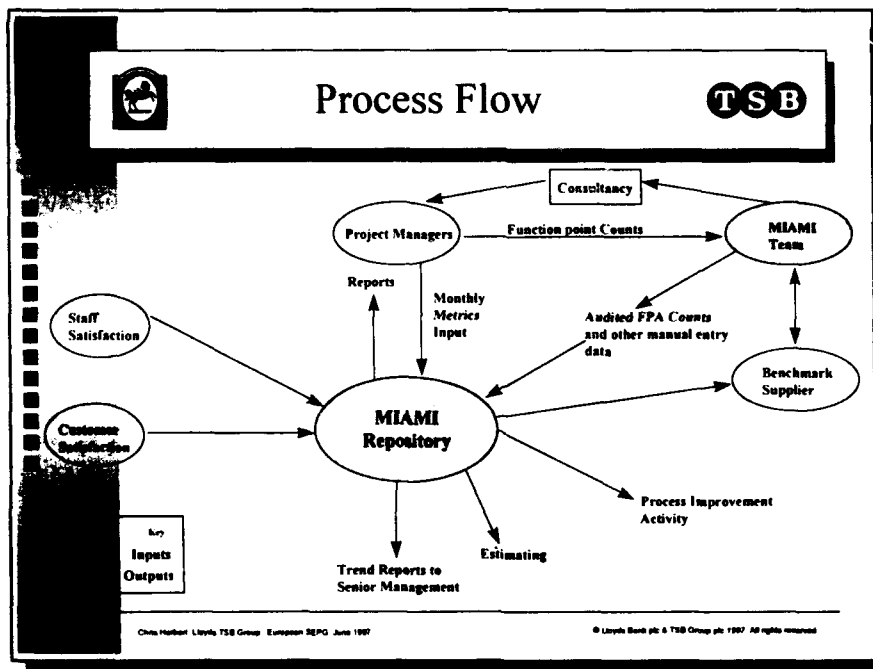
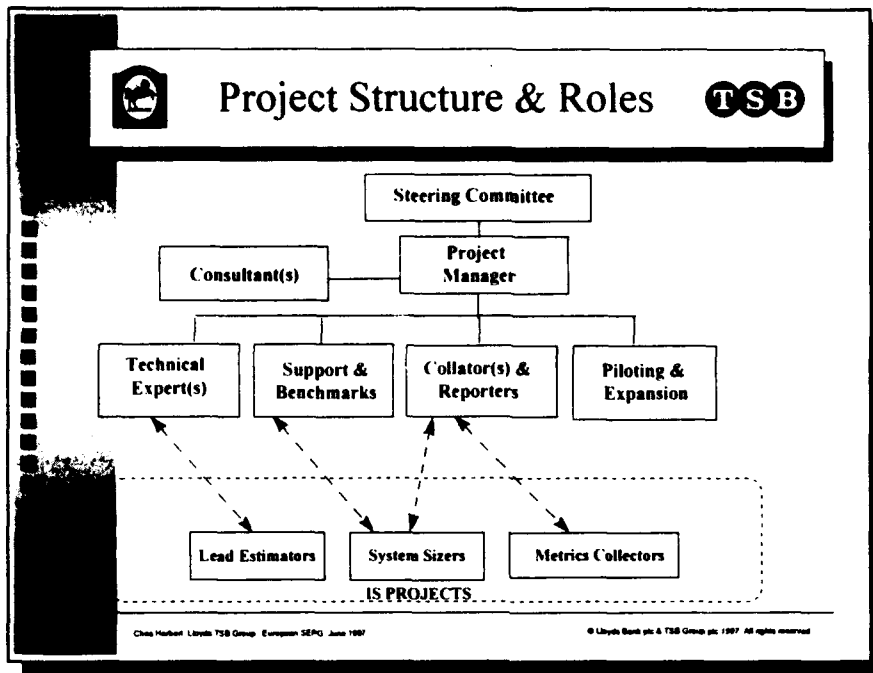
**Expended
Effort**

Productivity



Chris Harbert, Lloyds TSB Group, European SEPG, June 1997

© Lloyds Bank plc & TSB Group plc 1997. All rights reserved.





Project Strategy



- Structured approach
- Expanded project by function
 - Development
 - Support
- Piloted - refined - rolled out
- Metrics Release no. 1

Chris Harbord - Liberty TSB Group - European SEPO - June 1997

© Liberty Bank plc & TSB Group plc. 1997. All rights reserved.



Achievements to Date : 1



Corporate Level Reporting : "The Dashboard"

- 9 month rolling average
- % change over previous period
- rate of change



- **IS Senior Management Level Reporting**

- As above plus additional granularity per functional area

Project Management Level Reporting

- Standard reports to aid tracking and control



Chris Harbord - Liberty TSB Group - European SEPO - June 1997

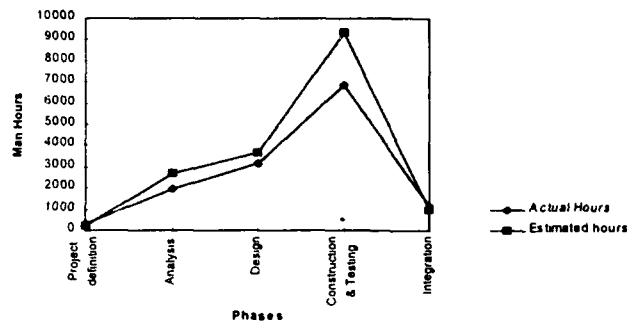
© Liberty Bank plc & TSB Group plc. 1997. All rights reserved.



Example 1

TSB

Estimated compared with Actual Effort for Project



Chris Herbert, Lloyds TSB Group, European SEPG, June 1997

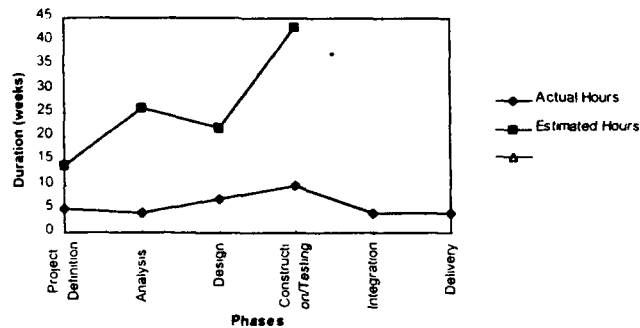
© Lloyds Bank plc & TSB Group plc 1997. All rights reserved.



Example 2

TSB

Estimates Compared with Actual Duration of Project



Chris Herbert, Lloyds TSB Group, European SEPG, June 1997

© Lloyds Bank plc & TSB Group plc 1997. All rights reserved.



Achievements to Date : 2



- 100 people trained, 140 projects captured and 150,000 fps counted
- Increased recognition as a value add service provider

Anticipated improved cost estimating capability

Chris Harbert | Lloyd's TSB Group | European SEPG | June 1997

© Lloyd's Bank plc & TSB Group plc 1997. All rights reserved



Next Steps



- Full data exploitation 
- Expansion to additional sites 
- Project Management measures

Customer Satisfaction survey



Chris Harbert | Lloyd's TSB Group | European SEPG | June 1997

© Lloyd's Bank plc & TSB Group plc 1997. All rights reserved



So, In Conclusion



Our experiences reveal at least five factors can affect our success rate :

- Commitment levels (active not passive)
- Generating customers & focusing on champions
- Keeping things simple

Providing a speedy return on investment

Automating data collection wherever possible

Chris Hetherly | Lloyds TSB Group | European SEPG | June 1997

© Lloyds Bank plc & TSB Group plc 1997. All rights reserved.



QUESTIONS ???

QUESTIONS ???

QUESTIONS ???

Chris Hetherly | Lloyds TSB Group | European SEPG | June 1997

© Lloyds Bank plc & TSB Group plc 1997. All rights reserved.

MIAMI - Practical Experiences of Introducing and Establishing a Successful Software Metrics Project (or Extracting Needles from Haystacks : Painlessly !)

Author: Chris Herbert, MIAMI Project Manager, Lloyds-TSB

Introduction

Chris Herbert has been the project manager for the Lloyds-TSB corporate software metrics initiative, labelled the MIAMI project, since its inception in 1995. This paper summarises his presentation of experiences to date with the MIAMI project.

Background

At the start of the MIAMI project the focus was on the now ex-Lloyds Bank Information Systems area. Lloyds IS had a history of metrics involvement. They were participating in external benchmarking; had experience of Function Point Analysis (FPA) and had spent some time looking at ways measurement could be used to, generally, improve management. However, it was recognised that there was no cohesive or co-ordinated approach to the collection, storage, analysis and feedback of this data. One result of this had been the suspension of the use of FPA some years previously. Any IS metrics programme therefore needed to consider measurement from four key dimensions and perspectives :-

- Corporate level metrics
- External Benchmarking
- Process Improvement (as IS were committed to process improvement using the Capability Maturity Model as the vehicle)
- Project level metrics

The culture of the organisation with respect to measurement also needed to be taken into consideration. Like many organisations, rather than being viewed as providing information of value, there was suspicion and a lack of understanding of measures. This arose partly from previous metrics initiatives that had, at least in part, failed. There had been patchy adoption that failed to deliver predicted benefits. There was also a fear of measurements being used against individuals rather than to improve the organisation's processes.

The MIAMI project was just like the vast majority of software metrics initiatives - it faced an uphill struggle!

Objectives

Terms of reference (ToR) for the MIAMI project were established and signed of by the head of IS. Included in this ToR were the following objectives:

1. To establish a cohesive system that encompasses all IS metrics requirements and determines how appropriate metrics will be collected, held, supported, analysed, fed back & displayed.
2. To be instrumental in achieving a change in culture by encouraging and communicating the value of accurate metrics.
3. To monitor the 'health' of IS and provide information that enables ISMM (the then senior management team) to shape and direct the organisation.
4. To facilitate the streamlining of certain IS processes.

Scope

The scope of the programme was initially defined based on advice from external consultants. The scope statements included the following points:

1. Encompass the whole of Information Systems whilst providing a watching brief that the Systems & Support (the umbrella organisation that included IS) quality targets and metrics reflect the IS needs and aims.
2. Comprise of a small team of central resources managing the programme whilst also collating, storing, analysing, presenting and feeding back the data. In the short term, must encourage projects to use their own local data now.
3. Be of at least five years duration with key milestones identified and delivered throughout this timeframe.
4. Interface with other relevant programmes such as the process improvement initiative.
5. Be the Facilitator / Enabler of metrics. Any measure common to multiple projects will be included within the programme, whereas local ones specific to single projects or areas will fall outside the scope.

The purpose of setting defined objectives and scope of the initiative was to show to the organisation that this was a project just like any other. While it was not intended to deliver software to the client organisation, the MIAMI project was to be managed by means of exactly the same disciplines as any other project within IS.

In addition to setting objectives and scope, the ToR also defined a small number of metrics that were to be included in the programme. This may appear counter to accepted wisdom within the metrics field where a strong link between metrics and business objectives is seen as important. We must stress that the business objectives for IS had already been defined and were embodied in the concept of working "better, quicker, faster". These objectives coupled with other internal drivers such as the process improvement work led to a set of corporate metrics being defined. The agreed six Corporate metrics are :-

No. of Function Points / Person Hour	(Productivity)
No. of Function Points / Elapsed Month	(Delivery Rate)
No. of Defects / Function Points	(Quality)
No. of Defects in development / (no. of defects in development + no. of defects in first 3 months of operation)	(Defect Removal Efficiency or Process Quality)
Information from annual questionnaire	(Staff Satisfaction)
Information from annual questionnaire and Information from Post Development Reviews	(Customer Satisfaction)

MIAMI Initiation

Investigate Requirements

The success of any metrics programme hinges on commitment from all involved, both those supplying and those receiving information. To ensure that all viewpoints were represented from the start a viewpoint analysis workshop was convened. Having identified Viewpoints and representatives of those Viewpoints an initial requirements gathering exercise was undertaken. The resulting documentation consolidated the requirements placed on MIAMI including:

- The metrics explicitly specified in the TOR
- Other metrics implied by the Business Plan
- How these are supported by the requirements of IS
- A summary of the results of the requirements gathering exercise by viewpoint

Define roles

As part of the initiation stage an organisation structure was drawn up for the project which identified the following roles. This facilitated the staffing of the team and the explanation of what was required from IS projects. It is stressed that these were roles and that one individual could fill a number of roles. The roles defined were:

Role	Responsible for
Steering Group	Overall success of the programme ensuring that the right information is provided for all aspects of IS at all levels at the right cost and at the right times. Ensures that MIAMI is correctly funded and resourced. Ensures the authorisation of the collection of data from IS. Approves plans presented by the MIAMI Project Manager. Sets tolerance limits for exception reporting.
MIAMI Project Manager	Prepares and implements the programme plans. Produces detailed plans for each phase showing the areas to be addressed and what is involved. Monitors the plans and reports exceptions according to tolerance levels set. Responsible for defining staff required and acquiring them. Delegates work to the resources involved in the project. Responsible for day-to-day running of the project.
System Sizing Expert	Ensures the collection of appropriate system sizes. Ensures the validity and correctness of the data collected. Maintains a knowledge of the current thinking in, and tools available for, system sizing. Ensures the metrics collection handbook defines the techniques and tools to be used. Ensures that the system sizers are identified on the projects concerned. Helps the system sizers with advice and guidance. Ensures system sizes are collected at appropriate times.

Metrics Expert	Ensures a clear definition of "Other Metrics". Ensures their collection. Ensures the validity and correctness of the data collected. Maintains a knowledge of the current thinking in, and tools available for, IS metrics collection. Ensures the metrics collection handbook defines the techniques and tools to be used. Ensures that the metrics collectors are identified on the projects concerned. Ensures there are appropriate mechanisms for data acquisition. Helps the metrics collectors with advice and guidance. Ensures system sizes are collected at appropriate times.
Software Developer/Maintainer	Ensures that a metrics repository is available and that it provides the functionality required by the metrics collators/reporters.
Metrics Collators/Reporters	Carry out the collection of data and the reporting back of collated information to the users of the metrics function. Discusses requirements of users and providers and communicates them to the Project Manager. Maintains the MIAMI Issues log (device that enables people outside MIAMI to make suggestions).
System Sizers	Assesses the size of the systems their projects are dealing with at the appropriate times using the appropriate techniques and tools. Requests help and advice from the System Sizing Expert. Are available for training etc. when required by changes in MIAMI circumstances.
Metrics Collectors	Collects the other data from their projects at the appropriate times using the appropriate techniques and tools. Requests help and advice from the "Other Metrics" Expert. Are available for training etc. when required by changes in MIAMI circumstances
Metrics Users	Specifies requirements for measurement information. Ensures that the information supplied is used appropriately. Available for training/education in the specification/use/meaning of metrics associated reports.
Metrics Consultant	Provides help and advice on the running of the metrics programme.
FPA Consultant	Provides help and advice on the application of Function Point Analysis (or other system sizing technique).

Produce Strategy

It was obvious that addressing all of the measurements identified in requirements gathering immediately would be too ambitious. Consequently it was decided that a phased approach would be appropriate. It was now that the benefit of a 5 year programme view became obvious. The phased approach was scheduled within the 5 year period giving both strategic and tactical level views of the MIAMI project. The following gives a sample of what was expected to be involved in each of the phases and particular issues that need to be addressed in the first phase.

Phase Objectives

MIAMI would develop along a series of cycles, probably repeating the cycles for the foreseeable future. Each cycle consists of 3 phases:

General

Phase 1 - Piloting

The aim of this phase is to establish the collection and use of a defined set of measures within a subset of projects. During this phase the collection and reporting mechanisms are defined and built and the collectors are trained in the processes they are to follow. The subset of projects will be selected as representative of those projects of greatest involvement with the measurement set selected for implementation within a particular cycle.

Phase 2 - Preliminary Rollout

This phase will take feedback from the pilot, make what amendments are indicated and roll out to other projects of the same type as those identified in the 1st phase, e.g. Development or Support.

Phase 3 - Complete Rollout

At the end of this phase all projects of a specified set within IS will be contributing to and utilising information from the measurements defined for the cycle. Any changes that have arisen from the first two phases will be incorporated.

Cycle 1

The 1st cycle had additional work in that there was no existing metrics function. It addresses the measurements specified in the MIAMI Terms of Reference plus some that address objectives implied by the business plan.

Cycle 2

This cycle (and others following) would involve a review of existing measures and further analysis of long term IS Business Goals. The resulting list of measurements would be prioritised with selection made of those to be addressed by the cycle. It was envisaged, at this stage, that the measures would include deeper granularity than those already collected.

The concept of phases operating within cycles across the whole MIAMI programme was designed to deal with the many requirements for measures and measurement based techniques that were placed on MIAMI as a result of the initial requirements gathering exercise. The structure provided an ordered, visible means of prioritising those requirements.

What also had to be addressed was the need for a technical and personnel infrastructure to support the measurement initiative together with the need for publicity, training and education. Effort was devoted to establish and equip a central metrics function which, in turn, was supported by project based "Metrics Collectors". Considerable effort was also devoted to publicising the MIAMI programme; to liaising with development and support project managers; to liaising with other functions within IS such as the Process Improvement team and to providing metrics education and training including FPA training.

The Story So Far

As with all projects, there were many day to day issues that had to be resolved but real progress was made. By the end of 1996 MIAMI was in the position of having:

- a formally agreed structure with defined roles and responsibilities;
- a formal set of procedures that define its operations;
- training materials and guidebooks to facilitate new projects joining;
- a defined set of services and reports for programme and project managers;
- incorporated Estimation into its responsibilities;
- software that supports the collection of data for Ex-Lloyds Development;
- staff committed to the project for 1997;
- most importantly, closed the loop with project and senior managers in being able to provide metrics based reports to them.

The Only Constant is Change

Having defined a strategy and a set of tactical objectives for cycles within that strategy things changed in a very significant fashion. The merger of Lloyds Bank with the TSB forming one of the largest European banks and the resultant merger of the two organisation's IS groups into one (Lloyds/TSB IS) necessitated a major, strategic re-planning exercise. With the assistance of representatives from all parts of the new organisation, this re-planning exercise was completed in a relatively short period of time and a revised strategy was signed off by the new senior management team.

Way Forward

An objective was set to expand the programme to all areas of Lloyds-TSB by the end of 1997. TSB had its own history of metrics programmes which needed to be accommodated. In addition the geographic scope was considerably extended across a number of sites countrywide. Consequently, a tactical plan was developed which set the following interim objectives:

1. It was decided that, to achieve the overall objectives stated above, MIAMI first needed to stabilise its normal operations. A plan was developed to get to that state. This included milestones such as:
 - To be producing reports for all levels of management for development;
 - To have completed a pilot on Ex-Lloyds Production;
 - To have launched the pilot on Ex-TSB production and support;
2. In addition it is planned to complete the following:
 - A pilot project for ex-TSB Sites. To end during 3rd Quarter 1997.
 - Migration to the whole of the group by 4th Quarter 1997.

Conclusions and Lessons Learned

There have been a number of lessons learned from experiences within the MIAMI Program:

Provide a Solid Foundation

- The Steering Committee has ensured high level commitment and provided authority to the program. When there was a need for arbitration because of dispute or disruption there was an escalation route to the highest levels of IS management.
- The temptation to be over ambitious was avoided by the production of the 5 Year Phased Implementation. This gave some management of expectations.
- The production of Standards and Guidelines and the ease of access to them helped in communication.
- Because a formal infrastructure was set up and approved it gave a "presence" to the program.
- The recognition that MIAMI was (and is) a series of projects and the managing of it as such enabled visible milestones to be produced and achieved.

Flexibility within Constraints

Because of the above, constraints could be identified and boundaries drawn. Thus the impact of new situations, such as:

- Larger Department;
- Software expansion;
- New Requirements;

could be easily assessed and the correct way forward established

Adequate Resources are Required

The MIAMI project has faced problems in resourcing but, for the most part, these have been overcome through the direct intervention of senior management, via the Steering Committee. In addition, the MIAMI project has used external consultants in some roles. One lesson learned in this area is that, for dependence on the externals to be avoided, clear handover plans are necessary.

Public Relations

The introduction of a metrics program can be likened to the setting up of a new department - it involves change, in the current jargon "Business Process Re-engineering" needs to take place. This needs selling.

Major points to be addressed were:

- Introducing the new disciplines
- What's in it for me? - Expanding on the advantages of the initiative
- Don't like change - overcoming fear and resistance
- How can the new department help? - explaining services

Finally it was found extremely beneficial to publicize our metrics programme successes and problems and to share these with other practitioners. We have gained at least as much as we have given in this area.



Software Engineering Institute

Process Maturity Profile of the Software Community 1996 Year End Update

May 1997

This report is a summary of the report
entitled "Process Maturity Profile of the
Software Community 1996 Year End Update
Report" prepared by the SEI.

The report is available for further
information from the SEI.

For more information, contact the SEI.



Software Engineering Institute

Outline

Introduction

Current Status

Community Trends

Organizational Trends

Summary



Introduction -1: Purpose and Source

Characterize the software process maturity of the software community

This briefing uses information from reports of Software Process Assessments (SPAs) and CMMsm Based Appraisals for Internal Process Improvement (CBA IPIs)

— Capability Maturity Model, CERT, CMR, IDEAL, Personal Software Process and PSP are service marks of Carnegie Mellon University.



Introduction -2: Data Description

SPAs and CBA IPIs conducted since 1987 through December 1996 and returned to the SEI by March 1997

- **751 assessments including 265 CBA IPIs**
- **616 organizations**
- **169 participating companies**
- **123 reassessed organizations**
- **3209 projects**

Please refer to: Terms Used in this Report on page 27



Introduction -3: Report Contents

This briefing includes three primary sections:

- **Current Status**
 - Snapshot of the software community based on the most recent assessments of reporting organizations
 - Only assessments since 1992
- **Community Trends**
 - Growth in the number of assessments performed
 - Shifts in the maturity profile over time
- **Organizational Trends**
 - Analysis of Key Process Area (KPA) satisfaction
 - Time to move up in maturity



Current Status

SPAs or CBA IPIs conducted from 1992 through December 1996

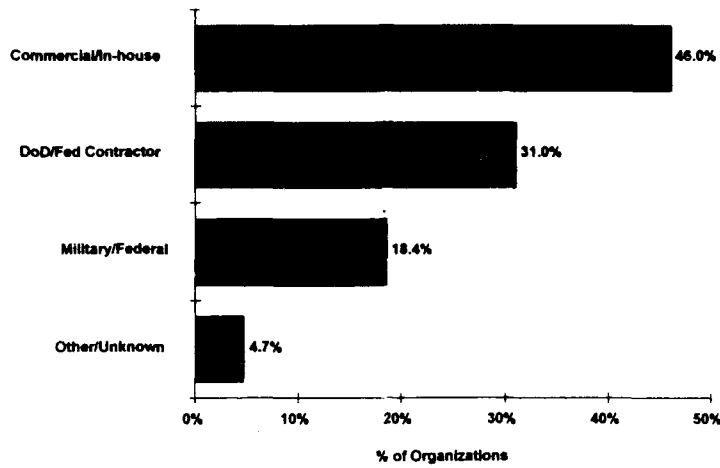
- **533 organizations**
- **153 participating companies**
- **2852 projects**
- **20.5% offshore organizations**

Please refer to: Terms Used in this Report on page 27



Carnegie Mellon University
Software Engineering Institute

Reporting Organization Types



Based on 533 organizations Note: "Other" is Research and Development, Non-Profit or Unknown (No Data Provided)

7

© 1997 by Carnegie Mellon University

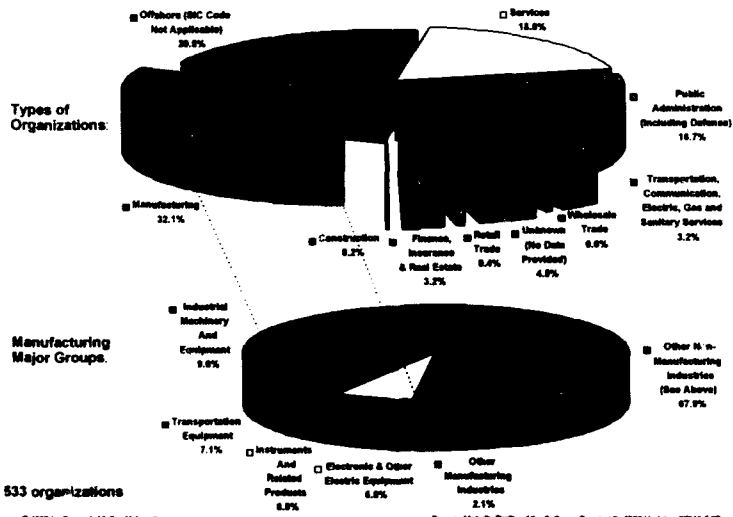
Process Maturity Profile of the Software Community 1996 Update SEMA 5.97



Carnegie Mellon University
Software Engineering Institute

Types of Organizations

Based on Primary Standard Industrial Classification (SIC) Code



Based on 533 organizations

8

© 1997 by Carnegie Mellon University

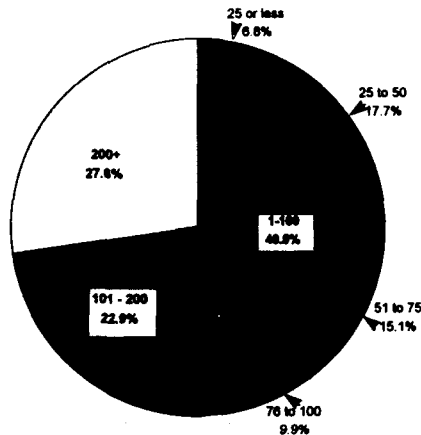
Process Maturity Profile of the Software Community 1996 Update SEMA 5.97



Carnegie Mellon University
Software Engineering Institute

Organization Size

Based on the total number of employees primarily engaged in software development/maintenance in the assessed organization



Based on 192 organizations reporting size data

9

© 1997 by Carnegie Mellon University

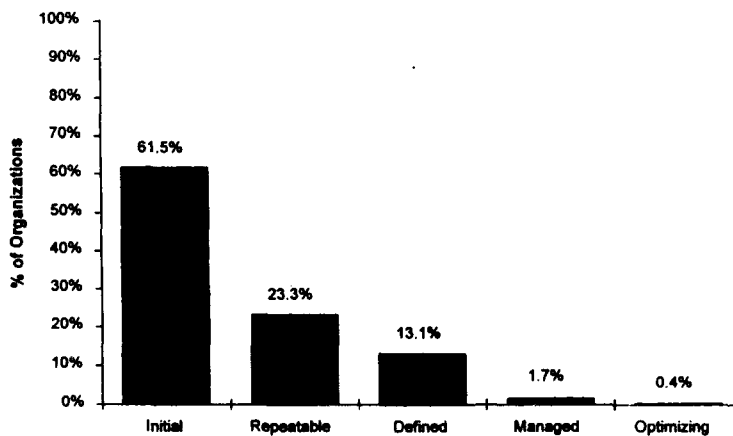
Process Maturity Profile of the Software Community 1995 Update - SP50A.5.97



Carnegie Mellon University
Software Engineering Institute

Organization Maturity Profile

April 1997



Based on most recent assessment, since 1992, of 533 organizations

10

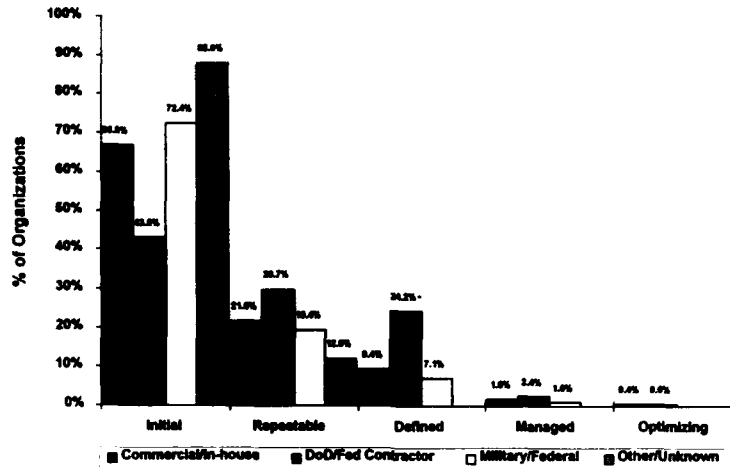
© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1995 Update - SP50A.5.97



Carnegie Mellon University
Software Engineering Institute

Maturity Profile by Organization Type



Based on most recent assessment, since 1992, of 533 organizations

11

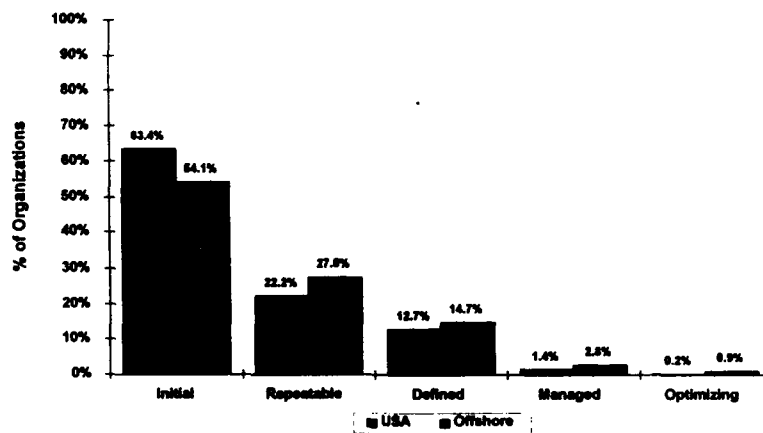
© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1998 Update - SEMMA 3.07



Carnegie Mellon University
Software Engineering Institute

USA and Offshore Organization Maturity Profiles



Based on 424 U.S. organizations and 108 offshore organizations

12

© 1997 by Carnegie Mellon University

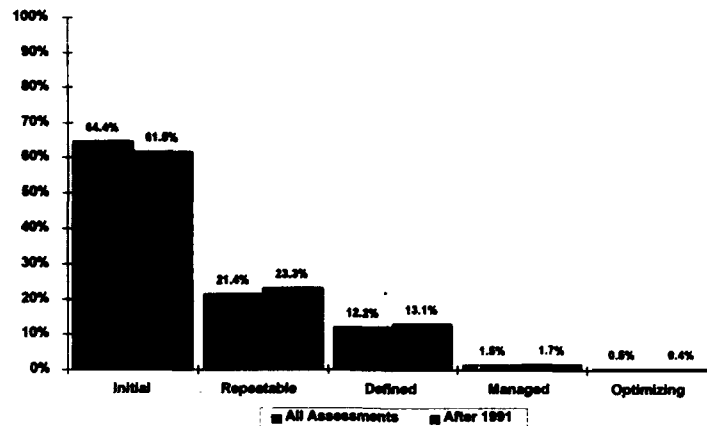
Process Maturity Profile of the Software Community 1998 Update - SEMMA 3.07



Carnegie Mellon University
Software Engineering Institute

Assessment Data Aging

A comparison of assessments reported after 1991 to all assessments reported



"All Assessments" based on 616 organizations
"After 1991" based on 533 organizations

13

© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1999 Update - SBMA.2.97



Carnegie Mellon University
Software Engineering Institute

Community Trends

SPAs or CBA IPIs conducted from 1987 through December 1996

- 751 assessments including 265 CBA IPIs
- 616 organizations
- 169 participating companies
- 123 reassessed organizations
- 3209 projects

Please refer to: Terms Used in this Report on page 27

14

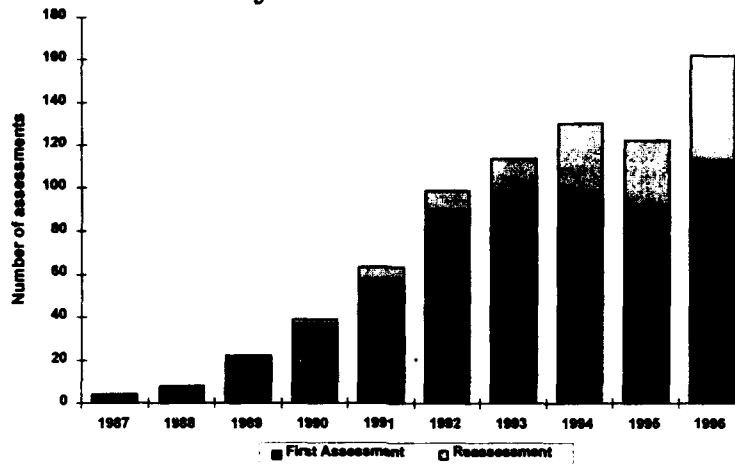
© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1999 Update - SBMA.2.97



Carnegie Mellon University
Software Engineering Institute

Number of Assessments Reported to the SEI by Year



Based on 751 assessments conducted through December 1996 and reported to the SEI by March 1997

15

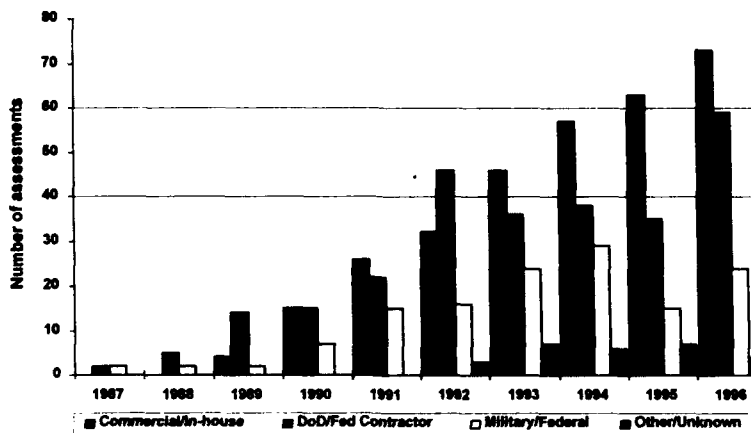
© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1996 Update - SEI-97-07



Carnegie Mellon University
Software Engineering Institute

Number of Assessments Reported by Organization Type and Year



Based on 751 assessments

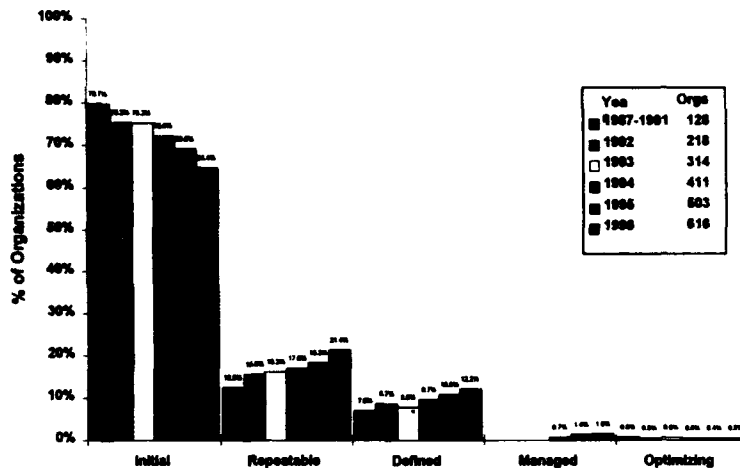
16

© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1996 Update - SEI-97-07



Trends in the Community Maturity Profile



Based on the most recent assessments of the organizations up through the year indicated

17

© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1996 Update - SEMAP.3.97



Organizational Trends

SPAs or CBA IPAs conducted through December 1996

- 192 Key Process Area (KPA) profiles
 - percent of assessments rating satisfaction of maturity level 2 and 3 KPAs
 - percent of assessments on which maturity level 2 and 3 KPAs were rated as fully satisfied
- 123 reassessed organizations including CBA IPAs

Please refer to: Terms Used in this Report on page 27

18

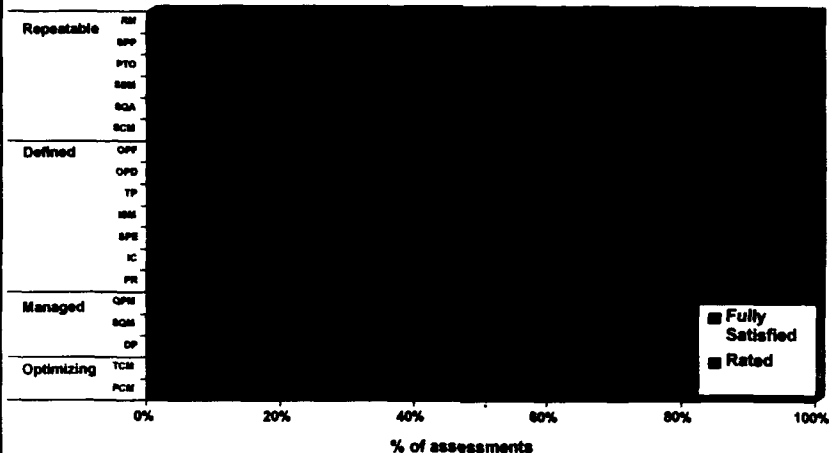
© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1996 Update - SEMAP.3.97



Carnegie Mellon University
Software Engineering Institute

Key Process Area Profiles -1 Organizations Assessed at Level 1



19

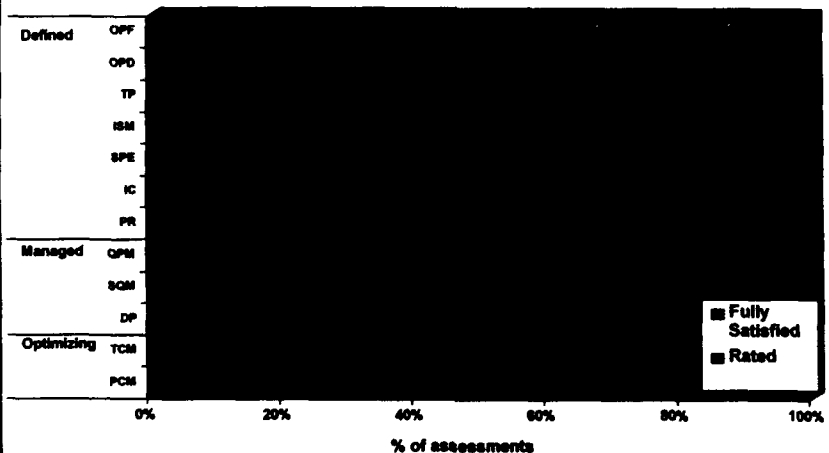
© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1999 Update - SEMA-9.97



Carnegie Mellon University
Software Engineering Institute

Key Process Area Profiles -2 Organizations Assessed at Level 2



20

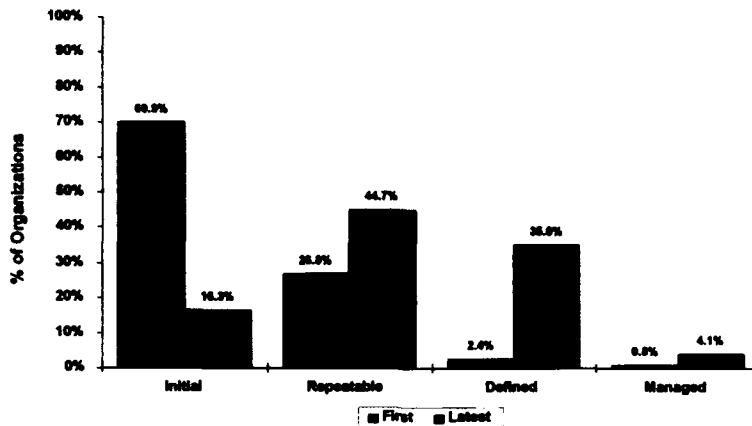
© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1999 Update - SEMA-9.97



Carnegie Mellon University
Software Engineering Institute

Maturity Level of First and Latest Assessments



Based on 123 reassessed organizations

21

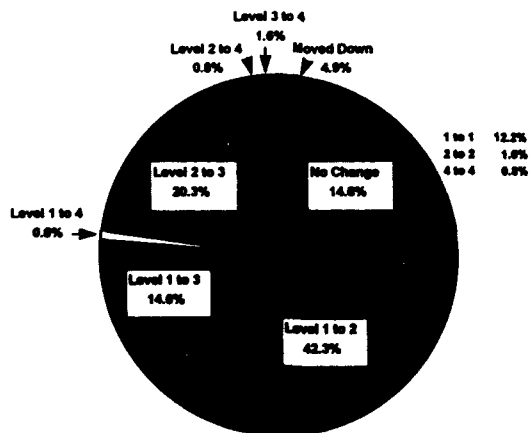
© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1995 Update - SEMMA 2.07



Carnegie Mellon University
Software Engineering Institute

Reassessments Change in Maturity Level



Based on 123 organizations accounting for 362 assessments
The organization's first and latest assessment on file was used

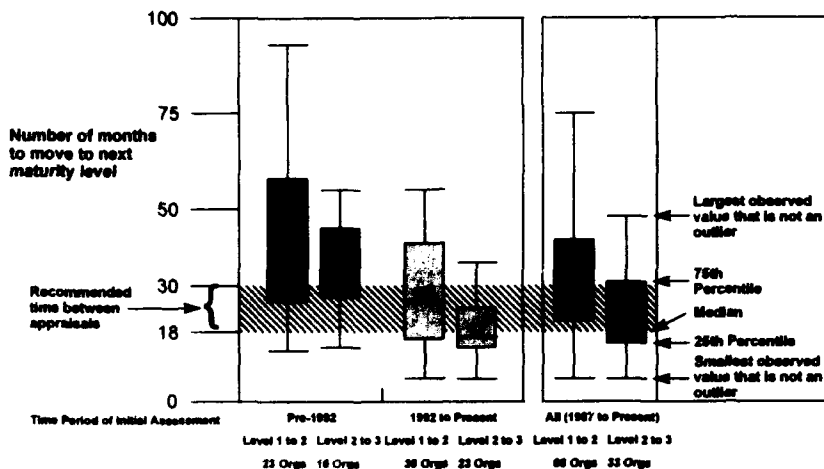
22

© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1995 Update - SEMMA 2.07



Time to Move Up



Maturity Summary -1

Number of organizations initiating software process improvement continues to increase

Increasing proportion of commercial and in-house organizations

Manufacturing organizations are conducting the most software process assessments

Nearly half of the organizations reporting size have less than 100 software personnel

Overall community profile continues to shift towards higher maturity



Maturity Summary -2

Ageing of the data produced a "jump" in the trend towards higher maturity

Trend towards higher maturity profile for offshore organizations compared to U.S. organizations

Software Quality Assurance and Software Project Planning are the least frequently satisfied KPAs among maturity level 1 organizations

Integrated Software Management, Organization Process Definition and Training Program are the least frequently satisfied KPAs among level 2 organizations



Maturity Summary -3

Higher maturity has been reached among those organizations reporting reassessments

Relatively low number of reassessments were reported compared to the number expected: approximately 22%

Based on reported reassessments, median time to move from maturity level 1 to 2 is 31 months for all organizations and 26 months for organizations that began their CMM-based SPI effort in 1992 or later.

All groupings exhibit a similar pattern for moving from maturity level 1 to 2 and level 2 to 3: level 2 to 3 tends to be faster and have less variance.



Terms Used in this Report

Organization - Appraised entity

The organization unit to which the appraisal results apply. An appraised entity may be any portion of an organization including an entire company, a selected business unit, units supporting a particular product line or service, etc..

Company - Parent of the organization

A company can be a commercial or non-commercial firm, for-profit or not for-profit business, a research and development unit, a higher education unit, a government agency, or branch of service, etc.

Offshore - An organization whose geographic location is not within the United States. The parent of the organization may or may not be based within the United States.

Assessments - The assessment methods used in this report are the Software Process Assessment (SPA) and CMMI-Based Appraisal for Internal Process Improvement (CBA IPI). However, we do request and receive other CMM for Software-based appraisals such as Interim Profile. As our sampling size of these other methods increase, they will be reported here.

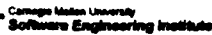


Questions and Comments Welcome

We are always interested in hearing from you regarding the maturity profile or about sending appraisal results to the SEI.

If you have questions or comments, please contact us:

E-mail: pais@sei.cmu.edu



We would appreciate it if you would take a few moments to let us know what other type of aggregated information you would like to see on the maturity profile report.

1. **How do you use the information in the maturity profile report?**
2. **What other type of aggregated information would you like to see on the maturity profile report?**
3. **Do you foresee any problem in your supplying us with the required data to create the aggregated information you would like to see?**

Please respond to:	PAIS	Include your:	Name
	Software Engineering Institute		Address
	4500 Fifth Avenue		Phone
	Pittsburgh, PA 15213		Fax
or E-Mail to:	pais@sei.cmu.edu		E-Mail

28

© 1997 by Carnegie Mellon University

Frenchspeaking Minority Profile of the Balkan Community 1995 Update - 2004.5.11



Visit our Web site for forms used to submit data and for future maturity profile reports:

Send the forms and your appraisal data to

PAIS
Software Engineering Institute
4500 Fifth Ave.
Pittsburgh, PA 15213

30

© 1997 by Cambridge Motion University

Source: Monthly Profile of the Nation's Community College Market, 1992-93, p. 22.



Carnegie Mellon University
Software Engineering Institute

Contacts for General SEI Information

SEI Customer Relations (412) 268-5800
SEI FAX number (412) 268-5758

Internet Address
customer-relations@sei.cmu.edu

Mailing Address
Customer Relations
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890

31

© 1997 by Carnegie Mellon University

Process Maturity Profile of the Software Community 1995 Update - ISMA.5.97

Tuesday 17 June

(T201a-8) S-16



Carnegie Mellon University
Software Engineering Institute

Dependence to Influence: Developing and Nurturing Effective Sponsorship

Chuck Myers

**Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213**

The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense and operated by Carnegie Mellon University.

© 1997 Carnegie Mellon University

ESEPG97-GO-1



Carnegie Mellon University
Software Engineering Institute

Agenda

- Introduction
- Change Roles
- Adopter Categories
- Developing Sponsors
- Supporting Sponsors
- Questions(!) and Answers(?)

© 1997 Carnegie Mellon University

ESEPG97-GO-2

Tuesday 17 June

(T201b) S-1



Carnegie Mellon University
Software Engineering Institute

Where Do You Work?

Military?

Other Government?

Government Contractor?

Industry?

- Finance?
- Retail?
- Industrial?

Bespoke System Supplier?

Outsourcing Organization?

Academia?

© 1997 Carnegie Mellon University

ESEPG97-GD-3



Carnegie Mellon University
Software Engineering Institute

How Long with SPI?

Just starting

< 1 year?

1 < but < 3 years?

3 < but < 5 years?

> 5 years?

© 1997 Carnegie Mellon University

ESEPG97-GD-4



Carnegie Mellon University
Software Engineering Institute

Who's Here?

- "Sponsor"**
- "Agent"**
- "Champion"**
- Participant**
- Interested/Curious**
- Other**

© 1997 Carnegie Mellon University ESEPG97-GD-4



Carnegie Mellon University
Software Engineering Institute

Assumptions

- Predominantly SEPG members**
- ~10 years experience ± a few in domain and/or management**
- ~3 years working with software process ± a couple**
- Substantial software technical experience and knowledge**
- Some experience in doing SPI work**
- Fairly high level of commitment to process focus as an enabler to better performance**
- (Lack of) sponsorship is major source of frustration**
- May be grasping at straws**
- May be searching for silver bullet**

© 1997 Carnegie Mellon University ESEPG97-GD-4



Carnegie Mellon University
Software Engineering Institute

Tutorial Objectives

When you have completed this tutorial, I hope you'll be able to do the following:

- describe change roles and change agent functions in a useful way
- develop strategies for building sponsorship starting with tutorial exercise materials
- develop approaches for supporting sponsors effectively in implementing change

© 1997 Carnegie Mellon University

ESEPG97-GD-7



Carnegie Mellon University
Software Engineering Institute

Agenda

Introduction

- ➡ Change Roles
- Adopter Categories
- Developing Sponsors
- Supporting Sponsors
- Questions(!) and Answers(?)

© 1997 Carnegie Mellon University

ESEPG97-GD-8



Carnegie Mellon University
Software Engineering Institute

Roles in Change

Champion

Target

Sponsor

Change Agent

Enabler

Note:

These roles are rarely "pure." They often evolve over time and overlap.

© 1997 Carnegie Mellon University

ESEPG97-GD-8



Carnegie Mellon University
Software Engineering Institute

Who Are Champions?

People who want the change implementation to be successful and therefore support it

Believers: They support it because they believe in it

© 1997 Carnegie Mellon University

ESEPG97-GD-10



Carnegie Mellon University
Software Engineering Institute

Who Are Targets?

People whose behavior will have to change in some way because of the change

Unknowns: Some people may support the change while others "resist" it; it is often difficult to predict who will react how in advance.

© 1997 Carnegie Mellon University

ESEPG97-GD-11



Carnegie Mellon University
Software Engineering Institute

Who Are Sponsors?

People within the organization who authorize and/or reinforce change

Authorizing Sponsor

The single individual in the organization who can

- Commit all resources required to implement a change successfully
- Enforce behavioral changes that are required.

Reinforcing Sponsors

Other managers whose support and/or reinforcement is required for successful implementation

© 1997 Carnegie Mellon University

ESEPG97-GD-12



Carnegie Mellon University
Software Engineering Institute

Who Are Enablers?

Senior management team members and members of oversight group (e.g., Management Steering Group)

Monitor working group progress and needs

"Run interference" for the working group

Serve as advocates for working group interests

Remove external obstacles to achieving the group's goals

Represent working group interests with the oversight group

© 1997 Carnegie Mellon University

ESEPG97-GD-13



Carnegie Mellon University
Software Engineering Institute

Who Are Change Agents?

People who manage change implementation details in behalf of the authorizing sponsor

Dutiful Staff Members: They support the change because it's their job.

© 1997 Carnegie Mellon University

ESEPG97-GD-14



Carnegie Mellon University
Software Engineering Institute

Change Agent Responsibilities - 1

- Facilitate change through line organizations**
- Obtain and maintain management support at all levels**
- Facilitate evaluation activities (e.g., software process appraisals)**
- Support line managers and supervisors whose work is affected by changes**
- Maintains collaborative relationships with "targets"**
- Provide consultation to development projects and management**
- Track and report progress**
- Track, monitor, and report process improvement status**

© 1997 Carnegie Mellon University

ESEPG97-GD-15



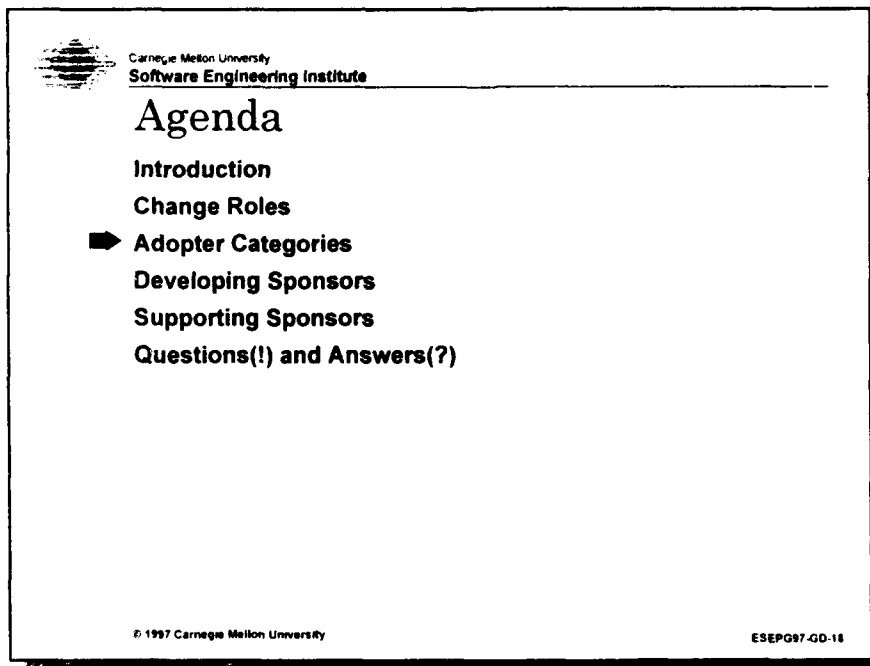
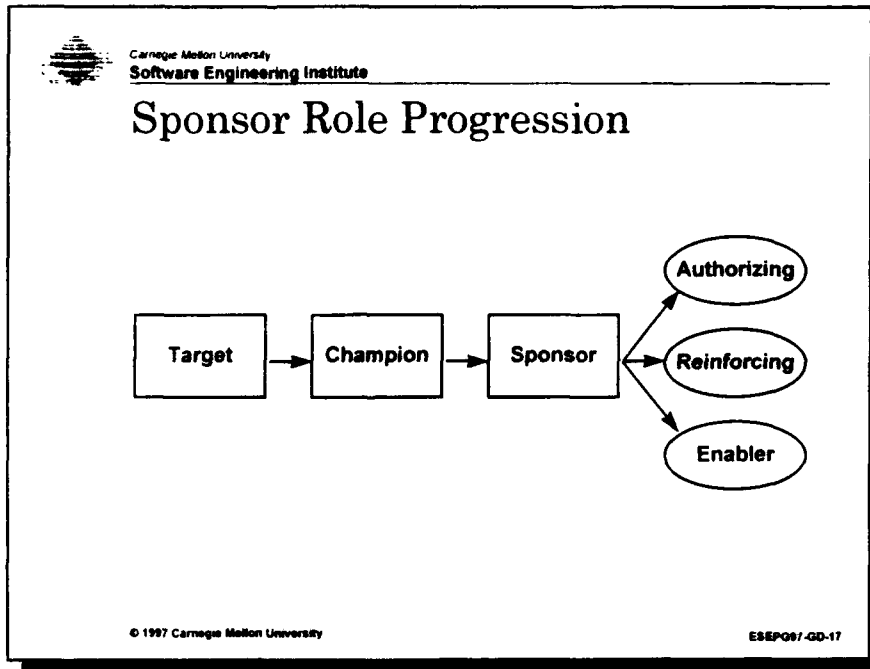
Carnegie Mellon University
Software Engineering Institute

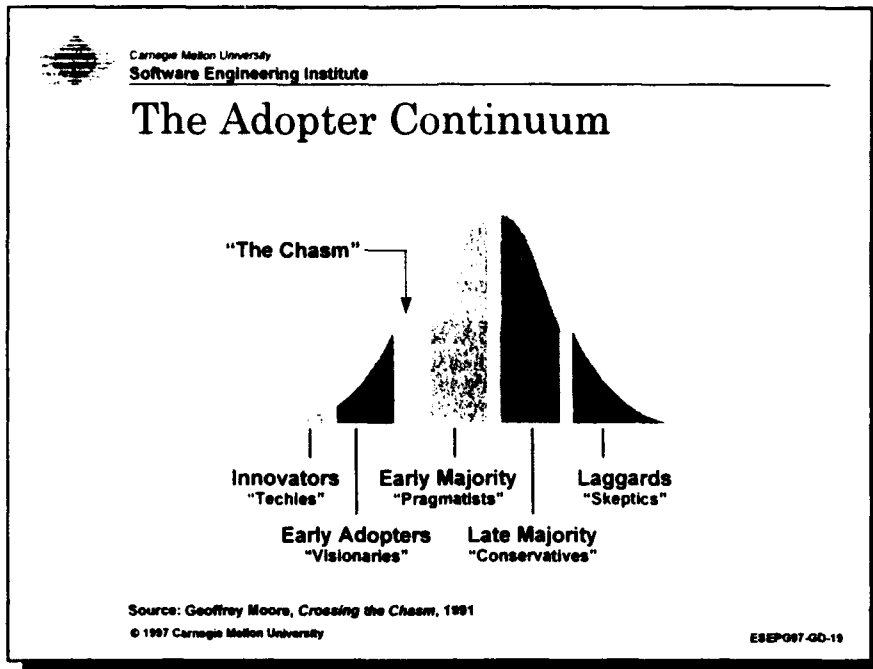
Change Agent Responsibilities - 2

- Maintain a databases and repositories**
- Serve as focal point for organizational learning**
- Arrange for training and continuing education**
- Maintain and disseminate lessons learned**

© 1997 Carnegie Mellon University

ESEPG97-GD-16





 Carnegie Mellon University
Software Engineering Institute

Innovators

- Gatekeepers for any new technology
- Appreciate technology for its own sake
- Appreciate architecture of technology
- Will spend hours trying to get technology to work
- Very forgiving of poor documentation, slow performance, incomplete functionality, etc.
- Helpful critics

© 1997 Carnegie Mellon University

ESEPG97-GD-20



Carnegie Mellon University
Software Engineering Institute

Reaching Innovators

- Focus on technical goals
- Put messages where innovators hang out
- Give them the unabashed truth
- Avoid image-type embellishments
- Let them try things out to see how they work
- Be conscious of cost/budget

© 1997 Carnegie Mellon University

ESEP097-GD-31



Carnegie Mellon University
Software Engineering Institute

Early Adopters

- Dominated by a dream or vision
- Focus on business goals
- Usually have close ties with "technie" innovators
- Match emerging technologies to strategic opportunities
- Look for breakthrough
- Thrive on high visibility, high risk projects
- Have charisma to generate buy-in for projects
- Do not have credibility with early majority

© 1997 Carnegie Mellon University

ESEP097-GD-32



Carnegie Mellon University
Software Engineering Institute

Reaching Early Adopters

- Understand the dream, or vision
- Focus on business potential (innovation)
- Maintain a project orientation
- Build strategy around "productizing"
- Schedule (and deliver) concrete deliverables early and often
- Manage expectations carefully

© 1997 Carnegie Mellon University

ESEPG97-GD-23



Carnegie Mellon University
Software Engineering Institute

Early Majority

- Do not want to be pioneers (prudent souls)
- Control majority of budget
- Want percentage improvement (incremental, measurable, predictable progress)
- Not risk averse, but want to manage it carefully
- Hard to win over, but are loyal once won

© 1997 Carnegie Mellon University

ESEPG97-GD-24



Carnegie Mellon University
Software Engineering Institute

Reaching the Early Majority

Show proof of (credible) others' success

Focus on predictability and measurability of progress and results (improvement)

Develop strategies to minimize and manage risk

Build relationships of trust based on dependability

Be sensitive to cost issues

Develop a long-term agenda

Be patient

© 1997 Carnegie Mellon University

ESEPG97-GD-25



Carnegie Mellon University
Software Engineering Institute

Late Majority

Avoid discontinuous improvement (revolution)

Adopt only to stay on par with the rest of the world

Somewhat fearful of new technologies

Like preassembled packages with everything bundled

© 1997 Carnegie Mellon University

ESEPG97-GD-26



Carnegie Mellon University
Software Engineering Institute

Reaching the Late Majority

Focus on who else is doing it

Emphasize maturity of the technology

Develop bundled packages

Think through, and present, a "whole solution"

© 1997 Carnegie Mellon University

ESEPG97-GD-27



Carnegie Mellon University
Software Engineering Institute

Laggards

"Nay sayers"

Adopt only after technology is not recognizable as separate entity

Constantly point at discrepancies between what was promised and what is

© 1997 Carnegie Mellon University

ESEPG97-GD-28

Tuesday 17 June

(T201b) S-14

Carnegie Mellon University
Software Engineering Institute

The Key Role Map

Which people need to fill a sponsor role?

- Authorizing
- Reinforcing

Who are the opinion leaders/movers and shakers?

■ Sponsors
□ Champions

© 1997 Carnegie Mellon University ESEPG87-GD-29

Carnegie Mellon University
Software Engineering Institute

Agenda

- Introduction
- Change Roles
- Adopter Categories
- ➡ Developing Sponsors
- Supporting Sponsors
- Questions(!) and Answers(?)

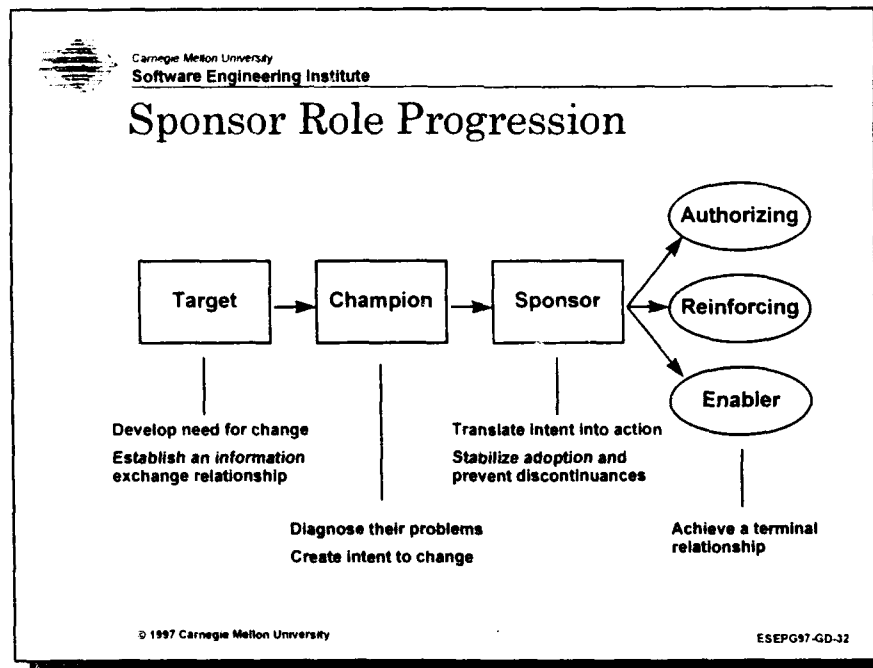
© 1997 Carnegie Mellon University ESEPG87-GD-30

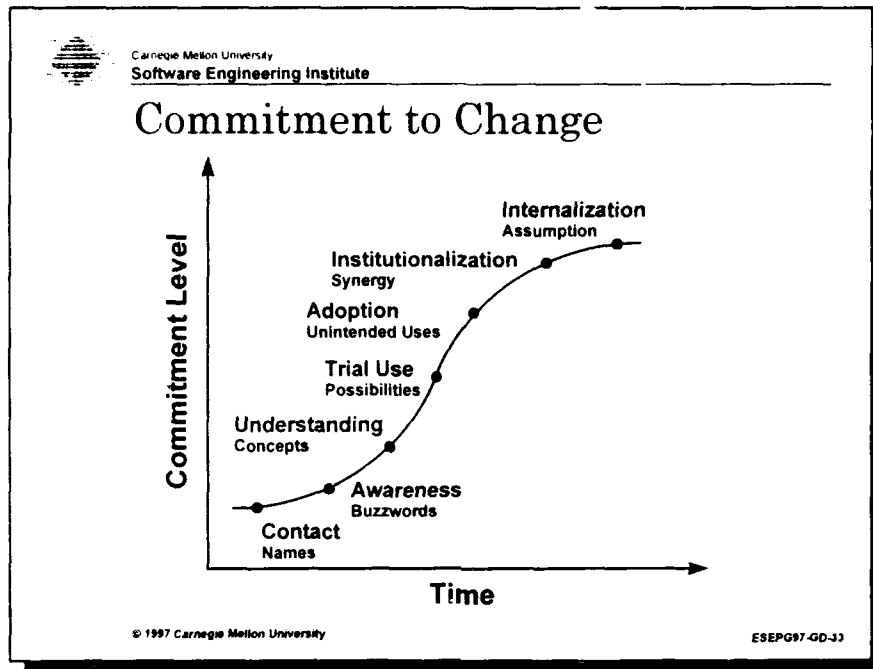
 Carnegie Mellon University
Software Engineering Institute

Sequence of Agent Efforts

- Develop need for change
- Establish an information exchange relationship
- Diagnose their problems
- Create intent to change
- Translate intent into action
- Stabilize adoption and prevent discontinuances
- Achieve a terminal relationship

Source: Everett M. Rogers, *Diffusion of Innovations*
© 1997 Carnegie Mellon University ESEPG97-GD-31





 Carnegie Mellon University
Software Engineering Institute

Develop Need for Change

Help potential sponsor become aware of the need for change

- draw attention to problems faced by that individual
- dramatize the importance of these problems
- point out new alternatives to solving existing problems

Convince them that they are capable of confronting these problems

Must carefully assess target needs

It usually works best to identify needs in a consultative manner

© 1997 Carnegie Mellon University ESEPG97-GD-34



Carnegie Mellon University
Software Engineering Institute

Establish Relationship

Develop rapport with the client (potential sponsor)

Create credibility agent competence, trustworthiness, and empathy with the client needs and problems

- clients must accept the change agent before they will accept the innovations the agent is promoting
- innovations are often judged in part on the basis of how the change agent is perceived

© 1997 Carnegie Mellon University

ESEPG97-GD-36



Carnegie Mellon University
Software Engineering Institute

Diagnose Problems

The change agent is responsible for analyzing the client's problem situation

- must view the situation empathically from the clients' perspective
- "The change agent must psychologically zip him or herself into the clients' skins, and see their situation through their eyes"

Diagnosis may involve ethical problems if not approached cautiously

© 1997 Carnegie Mellon University

ESEPG97-GD-36



Carnegie Mellon University
Software Engineering Institute

Create Intent to Change

Seek to motivate an interest in the innovation

- Why address this problem or set of problems rather than others?
- Why implement this solution rather than others?

The change must focus on resolving client problems, so that it is client-centered rather than solution-centered.

© 1997 Carnegie Mellon University

ESEPG97-GD-37



Carnegie Mellon University
Software Engineering Institute

Translate Intent to Action

Seek to influence client behavior in accordance with recommendations based on the clients' needs

Influence from near-peers in the client's interpersonal network is most important in persuasion and moving to decision

Change agent can only operate indirectly, working with opinion leaders within the client's peer networks

© 1997 Carnegie Mellon University

ESEPG97-GD-38



Carnegie Mellon University
Software Engineering Institute

Stabilize Adoption/Prevent Discontinuances

Maintain close ties with sponsor after adoption

Direct reinforcing messages (e.g., metrics and anecdotal data)

- to client
- to interpersonal network
- to management chain

Monitor sponsor's efforts for difficulties and help to resolve

Help them learn how to be a more effective sponsor

Support the sponsor in communicating effectively

© 1997 Carnegie Mellon University

ESEPG97-GD-39



Carnegie Mellon University
Software Engineering Institute

Achieve Terminal Relationship

End goal as a change agent: work yourself out of a job

Develop the sponsor's ability to be effective on their own

Shift the sponsor from reliance on agent to reliance on self

© 1997 Carnegie Mellon University

ESEPG97-GD-40



Carnegie Mellon University
Software Engineering Institute

Agenda

Introduction

Change Roles

Adopter Categories

Developing Sponsors

➡ Supporting Sponsors

Questions(!) and Answers(?)

© 1997 Carnegie Mellon University

ESEPG97-GD-41



Carnegie Mellon University
Software Engineering Institute

Who Are Sponsors?

People within the organization who authorize and/or reinforce change

Authorizing Sponsor

The single individual in the organization who can

- Commit all resources required to implement a change successfully
- Enforce behavioral changes that are required

Reinforcing Sponsors

Other managers whose support and/or reinforcement is required for successful implementation

© 1997 Carnegie Mellon University

ESEPG97-GD-42



Carnegie Mellon University
Software Engineering Institute

Sponsor Responsibilities - 1

Link improvement effort to organization's vision and mission

- provide policy oversight
- set priorities
- translate related policies
- approve plans in priority order
- integrate and build consensus among groups having different perspectives
- work with higher level management with broader concerns

© 1997 Carnegie Mellon University

ESEPG97-GD-43



Carnegie Mellon University
Software Engineering Institute

Sponsor Responsibilities - 2

Allocate resources and ensure work redistribution

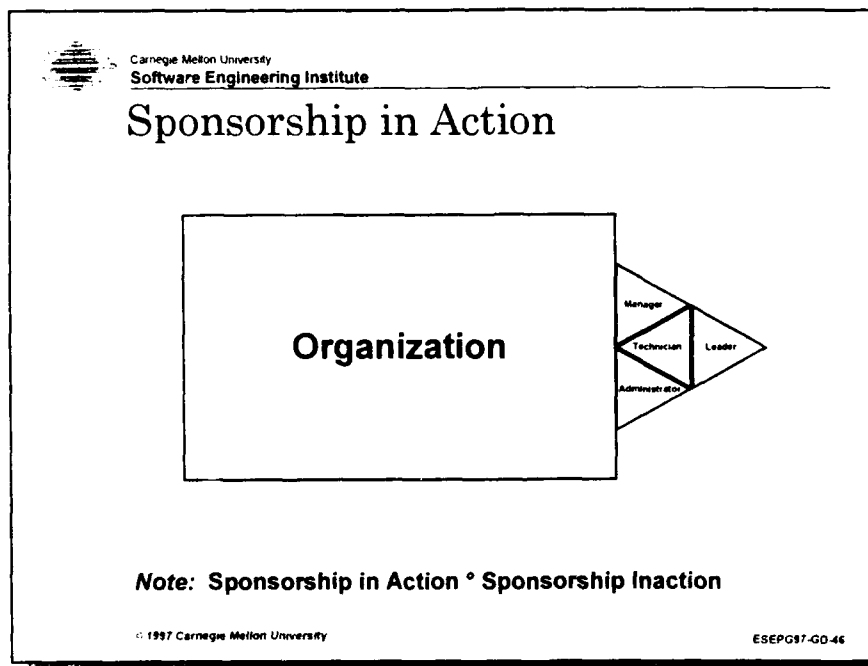
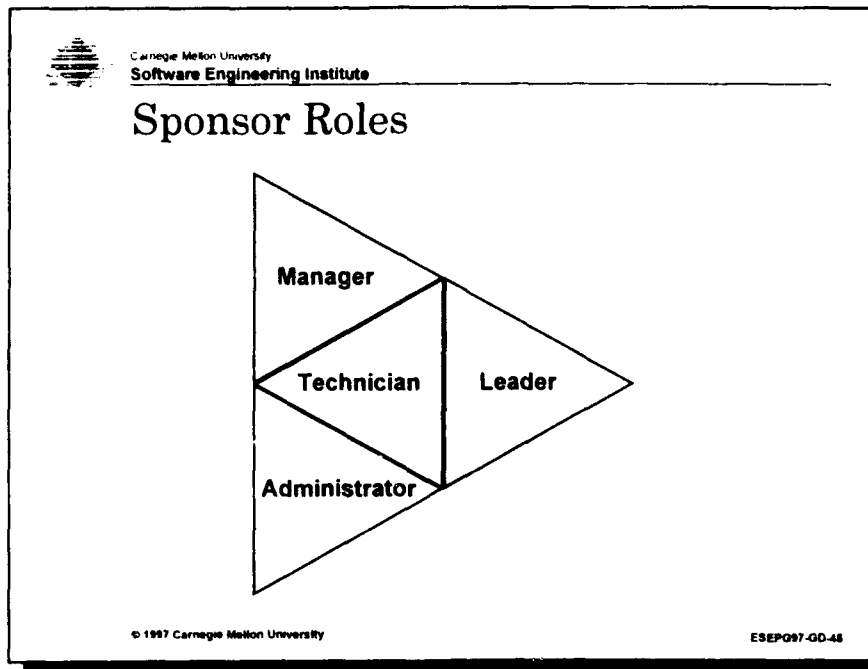
- charter working groups to prepare plans and do the work associated with
- allocate and manage resources

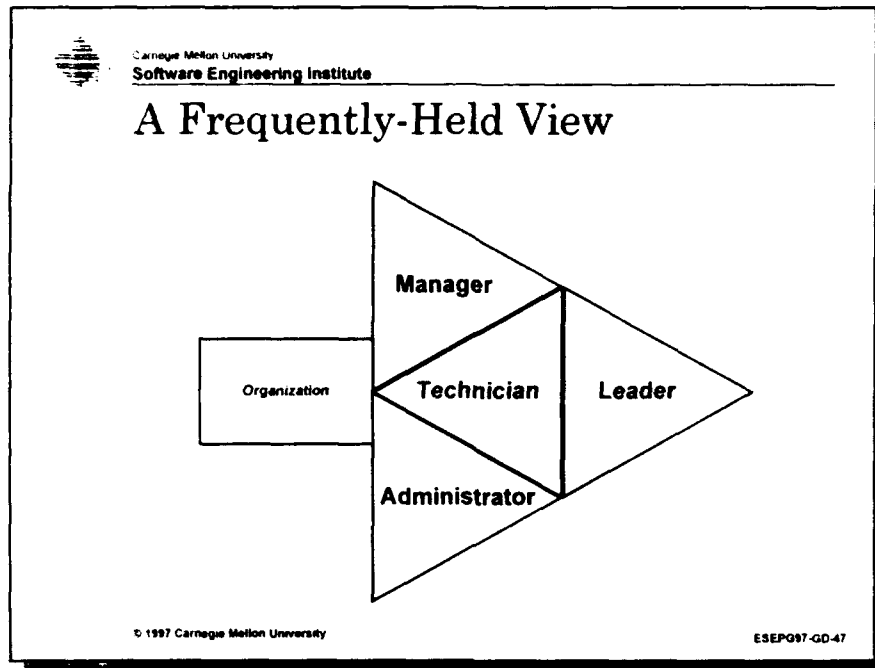
Monitor implementation results and provide mid-course corrections

- manage improvement processes
- review results
- monitor working group progress

© 1997 Carnegie Mellon University

ESEPG97-GD-44





 Carnegie Mellon University
Software Engineering Institute

Leadership Function

- Focuses on effectiveness**
- Sees that organization is doing the right things**
- Establishes strategic context**
- Clearly articulates strategic goals, needs, and direction**
- Develops and communicates vision**
- Sets priorities**
- Wins hearts and minds**
- Develops optimistic bias throughout the organization**
- Identifies cultural and other changes required for success**
- Keeps the effort in everyone's awareness**

© 1997 Carnegie Mellon University

ESEPG97-GD-48



Carnegie Mellon University
Software Engineering Institute

Managerial Function

Focuses on efficiency

Sees that the organization is doing things right

Authorizes and approves all resources required for success

Determines, models, and enforces behavioral changes

Appoints effective change agents to manage implementation details

Develops meaningful, achievable, measurable goals and objectives

Ensures implementation approach is realistic

Monitors progress

Approves mid-course corrections

© 1997 Carnegie Mellon University

ESEPG97-GD-49



Carnegie Mellon University
Software Engineering Institute

Administrative Function

Focuses on utility

Sees that the organization can do anything at all

Sees that required resources are available when needed

Establishes and implements mechanisms to support behavioral changes

Collects, summarizes, formats, and interprets data regarding progress and status

© 1997 Carnegie Mellon University

ESEPG97-GD-50



Carnegie Mellon University
Software Engineering Institute

Technical Function

Focuses on work

Sees that organization is doing its work properly

Understands technologies at an appropriate level of detail

Seeks more detailed technical input when appropriate

© 1997 Carnegie Mellon University

ESEPG97-GD-41



Carnegie Mellon University
Software Engineering Institute

Reinforcing Sponsors

Support and reinforce the authorizing sponsor's decisions, priorities, policies etc.

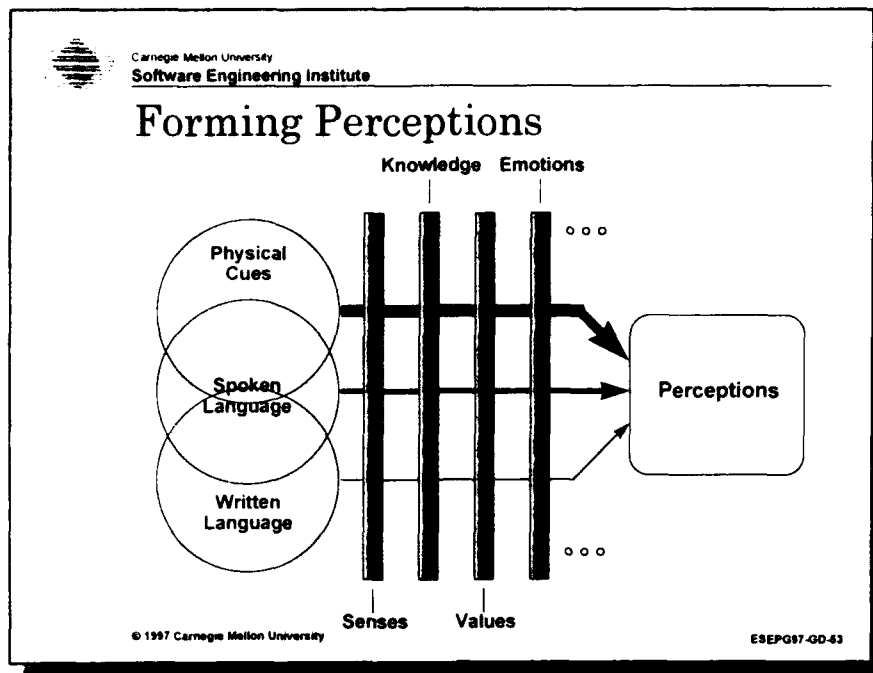
Communicate authorizing sponsor's commitment

Communicate their own commitment

Translate strategic intent to their own area of responsibility

© 1997 Carnegie Mellon University

ESEPG97-GD-52



Carnegie Mellon University
Software Engineering Institute

The Change Agent Support Role

Leader	Provide the sponsors with opportunities to show visible support
Manager	Attend to coordinating implementation specifics to ensure they are done as efficiently as possible
Administrator	Ensure that routine support functions are invisible to the people who are doing the work
Technician	Educate sponsors and give them technical input they may need to make viable decisions

© 1997 Carnegie Mellon University ESEPG97-GD-44



Carnegie Mellon University
Software Engineering Institute

Primary Leadership Elements

What leaders pay attention to, measure, and control on a regular basis

How leaders react to critical incidents and organizational crises

Observed criteria by which leaders allocate scarce resources

Deliberate role modeling, teaching, and coaching

Observed criteria by which leaders allocate rewards and status

Observed criteria by which leaders recruit, select, promote, retire, and excommunicate organizational members

Source: Edgar Schein, *Organizational Culture and Leadership*

© 1997 Carnegie Mellon University

ESEPG97-GD-66



Carnegie Mellon University
Software Engineering Institute

Secondary Leadership Elements

Organization design and structure

Organizational systems and procedures

Organizational rites and rituals

Design of physical space, facades, and buildings

Stories, legends, and myths about people and events

Formal statements of organizational philosophy, values, and creed

Source: Edgar Schein, *Organizational Culture and Leadership*

© 1997 Carnegie Mellon University

ESEPG97-GD-66



Carnegie Mellon University
Software Engineering Institute

Agenda

Introduction

Change Roles

Adopter Categories

Developing Sponsors

Supporting Sponsors

► Questions(!) and Answers(?)

© 1997 Carnegie Mellon University

ESEPG97-GD-57

Tuesday 17 June

(T201b) S-29

Exercise 1: Characterizing Your Change Effort

1. Briefly describe the change effort you will be focusing on during this tutorial.

2. What prompted your organization to take on this effort?
 -
 -
 -
 -
 -

3. What does your organization hope to accomplish by implementing this change?
 -
 -
 -
 -
 -

4. What obstacles is this effort currently encountering?
 -
 -
 -
 -
 -

Exercise 2: Adopter Continuum

Consider the people in your organization whose sponsorship is critical to the success of your change effort. Based upon your observations of them, which adopter categories do they fall into? Based upon their adopter categories, what might you do to influence them?

Influence

1. Innovator
 -
 -
 -
2. Early Adopter
 -
 -
 -
3. Early Majority
 -
 -
 -
4. Late Majority
 -
 -
 -
5. Laggard
 -
 -
 -

Exercise 3: Sequence of Agent Efforts

1. Think of one of the people you listed in the previous exercise who is not currently committed to the change your organization is implementing. How might the change agent(s) work with this individual in the following areas?

Develop the need for change

-
-
-

Establish an information exchange relationship

-
-
-

Diagnose their problems

-
-
-

Create the intent to change

-
-
-

2. Think of another individual who seems genuine in their desire to implement this change. How might change agent(s) help them translate their intent into action?

-
-
-
-
-
-

Exercise 4: Giving Support to Sponsors

1. What might the change agent(s) do to help the authorizing sponsor give visible evidence that s/he considers this change effort important and that s/he is committed to implementing it successfully? Consider support in the following areas:

What s/he pays attention to, measures, and controls on a regular basis:

-
-
-

How s/he reacts to critical incidents and organizational crises:

-
-
-

Observed criteria by which s/he allocates scarce resources:

-
-
-

Deliberate role modeling, teaching, and coaching:

-
-
-

Observed criteria by which s/he allocates rewards and status:

-
-
-

Observed criteria by which s/he recruits, selects, promotes, retires, and excommunicates organizational members:

-
-
-

2. What management concerns related to the change can the change agent(s) take charge of or provide input to for authorizing and reinforcing sponsors?

A R Concerns

☐ ☐

☐ ☐

☐ ☐

☐ ☐

3. What administrative aspects of the effort need to be overseen by the change agent(s)?

•

•

•

•

4. What sorts of technical training related to the change effort will the change agent(s) need to provide? To whom?

Training

Provided to

•

•

•

•

5. What sorts of technical input will the change agent(s) need to provide to authorizing and reinforcing sponsors?

A R Type of Input

☐ ☐

☐ ☐

☐ ☐

☐ ☐

☐ ☐



Carnegie Mellon University
Software Engineering Institute

Software Risk Management Tutorial

Audrey J. Dorofee, Ray C. Williams

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA USA 15213

The Software Engineering Institute is a federally funded research and development center sponsored by the U.S. Department of Defense and operated by Carnegie Mellon University.
© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Tutorial Objectives

- introduce a new perspective for initiating risk management activities
- provide insight and guidance into how risk management can be practiced by a customer and supplier through a team approach

2

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Risk Management*

An approach to managing that is based on *identification and control* of those areas and events in the systems engineering life cycle that have the potential for causing *unwanted change* in either the *process or product*.

*Lage, Andrew P., *Systems Engineering*, John Wiley & Sons, Inc., New York, New York, 1992.

3

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Software Risk Management

Processes, methods, and tools for managing risks in a software-intensive project

It is achieved by establishing a disciplined environment in which decisions are proactively made by

- assessing continuously what could go wrong
- determining which risks are important to deal with
- implementing strategies to deal with those risks

4

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

What is Risk?

Risks are *future* events with a probability of occurrence and a potential for loss.

Risks can be avoided, eliminated, or have their impacts lessened if they are properly managed.

A *problem* is a risk whose time has come.

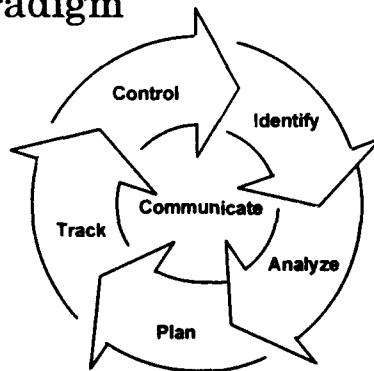
5

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

SEI Risk Management Paradigm



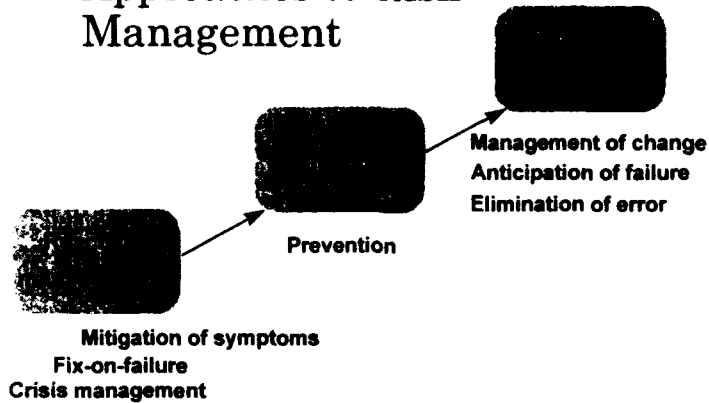
6

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Approaches to Risk Management



7

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Risk Management Overview

Software Risk Evaluation

- baseline identification and analysis of risks

Continuous Risk Management

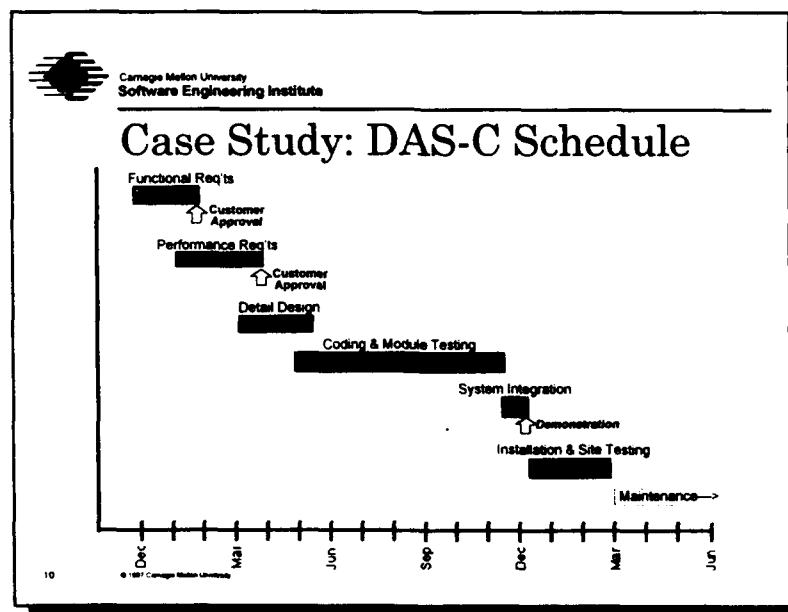
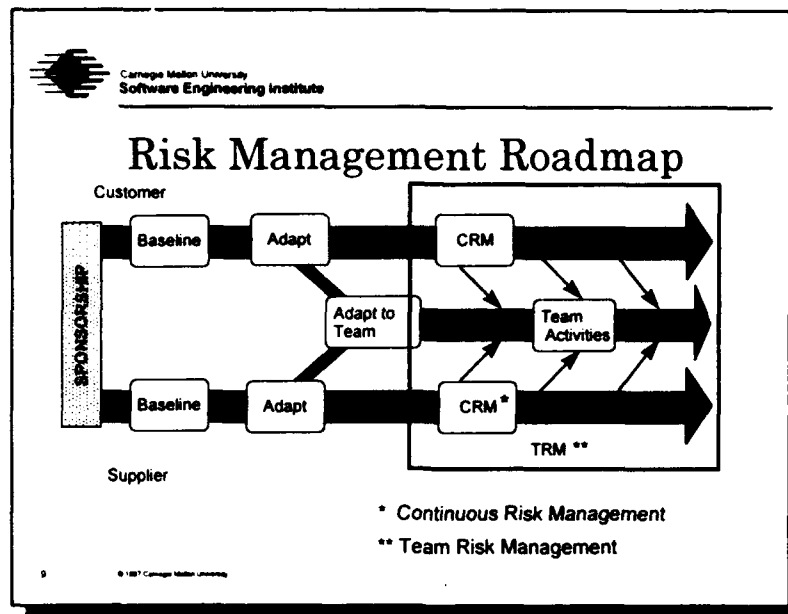
- continuous identification, analysis, planning, tracking, and control of risks within an organization

Team Risk Management

- joint management risks among organizations (e.g., team = customer and supplier)

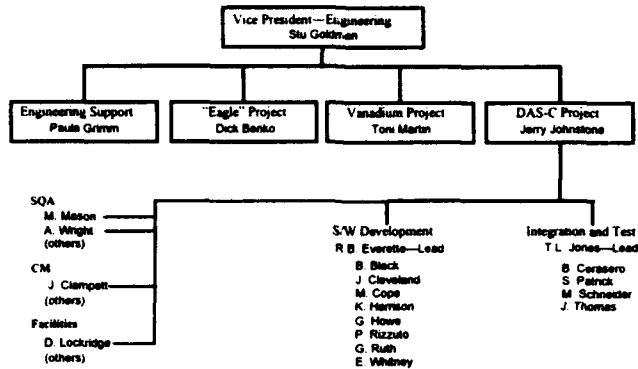
8

© 1997 Carnegie Mellon University

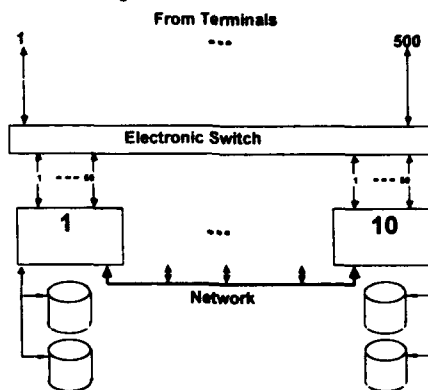




Case Study: DAS-C Organization



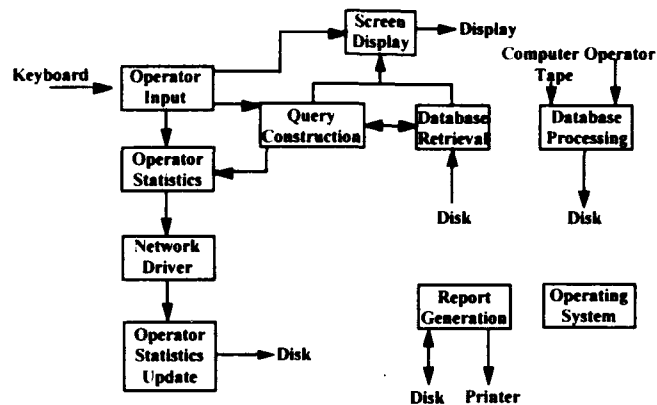
Case Study: DAS-C Hardware





Carnegie Mellon University
Software Engineering Institute

Case Study: DAS-C Software



13

© 1987 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

What is Baselineing?

A structured, repeatable process for identifying and analyzing a critical mass of risks, and planning for their mitigation within a project or program

Major components

- identification and analysis
- communication
- planning for mitigation

Based upon well-understood methods

- interviewing techniques
- brainstorming

14

© 1987 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Benefits of Effective Baselineing

Creates shared view of risks facing a project

Creates a common framework for talking about and mitigating risks

Identifies complete picture of current risks

- can track risks systematically (changes in likelihood and impact)
- can track mitigation of risks systematically

Provides motivation for *focused* project-level process improvement

Provides decision-making information to the project manager

15

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

What is a Software Risk Evaluation (SRE)?

Specific method for baselining: a structured, repeatable process to identify, analyze and plan risks within a project. It includes

- contracting and commitment
- identification and analysis
- preliminary report / mitigation strategy planning preparation
- mitigation strategy planning
- final report and closure

16

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

What does an SRE *do*?

Focuses upon risks that can affect the delivery and quality of software and system products

Provides project manager and personnel with multiple perspectives on identified risks

Creates foundation for Continuous and Team Risk Management

- **Prepares projects to conduct systematic risk identification, analysis, and mitigation planning for projects in their own organizations**

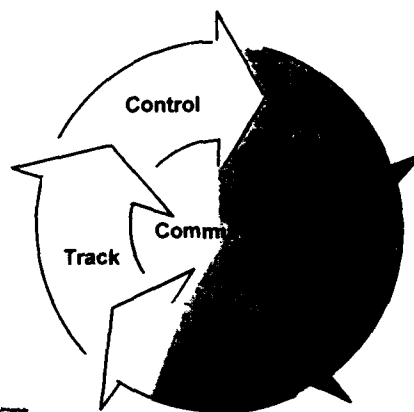
17

© 1987 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Paradigm Functions of the SRE



18

© 1987 Carnegie Mellon University



SRE: Identify and Analyze Risks

Uses Taxonomy-Based Questionnaire

- 4 or 5 group sessions with structured brainstorming interviews for identifying risks

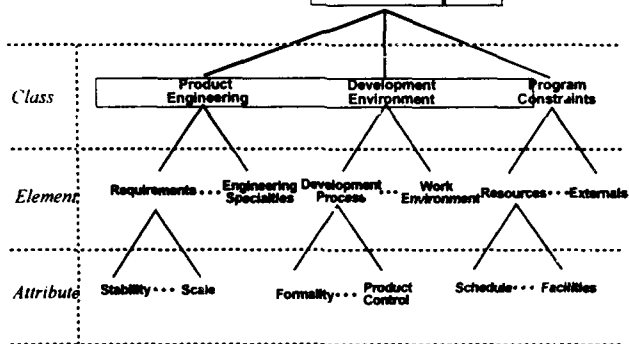
Produces diverse views of project risks

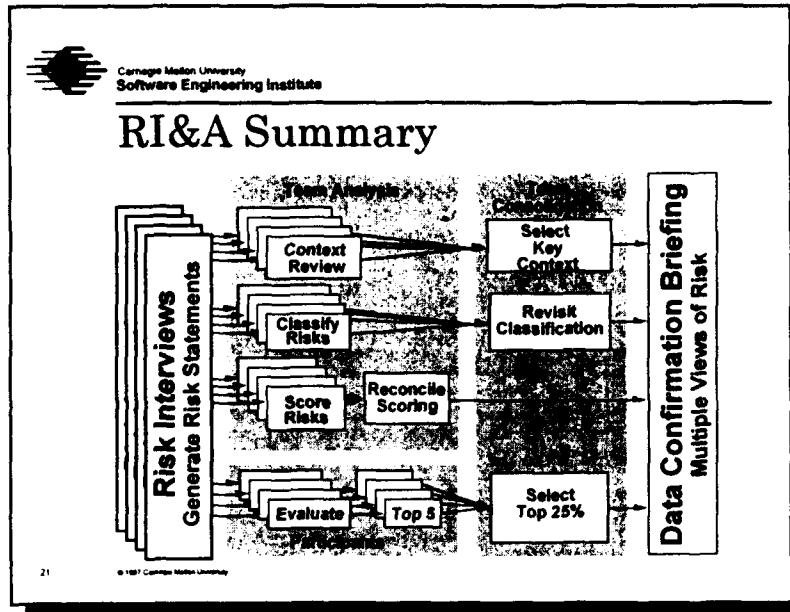
- which risks are most important risks to the program from individual's viewpoint
- expert opinion (SRE team)
 - probability, impact, of risks
 - areas of related risks



Taxonomy Structure

Software Development Risk



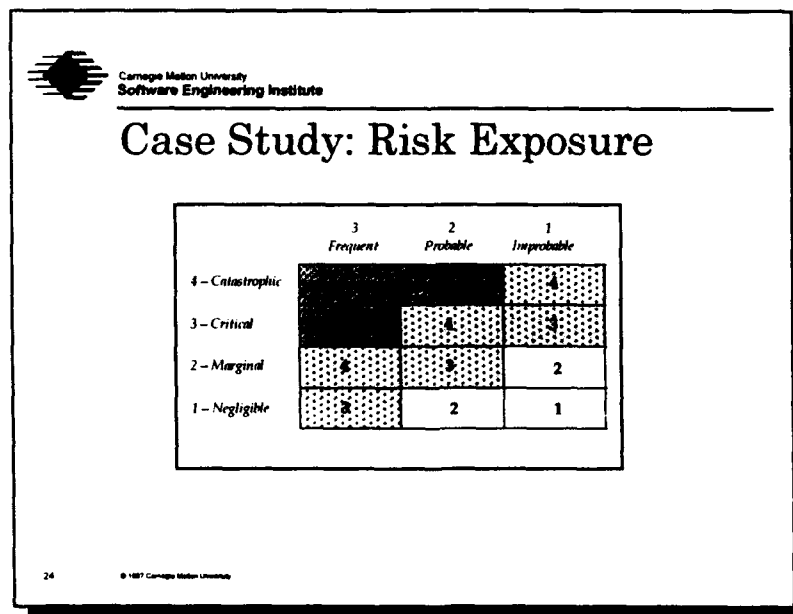
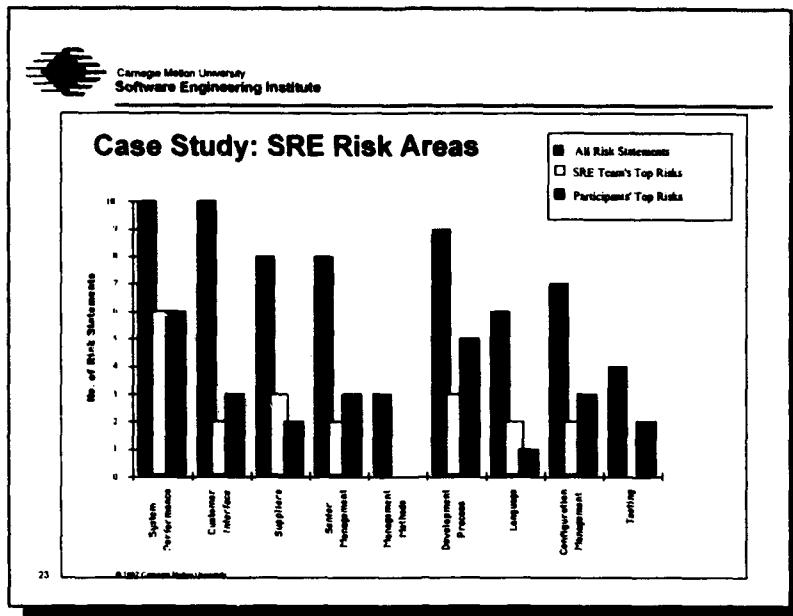


 Carnegie Mellon University
Software Engineering Institute

Case Study: Risk Areas from SRE

Risk Area	Total Risks per Area	SRE Team's Top Risks	Participant's Top Risks	Overlap
System Performance	10	6	6	4
Customer Interface	10	2	3	1
Suppliers	8	3	2	2
Senior Management	8	2	3	1
Management Methods	3	0	0	0
Development Process	9	3	5	3
Language	6	2	1	0
Configuration Mgmt	7	2	3	1
Testing	4	0	2	0
	65	20	25	12

22 © 1997 Carnegie Mellon University





Case Study: System Performance Risks

Carnegie Mellon University
Software Engineering Institute

COMPLETE DAS-C RISK LISTING (by Risk Area)

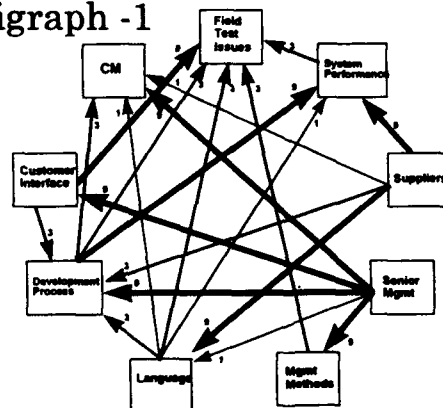
Item #	Risk Statement	Rank	Known	Test	Dev	Risk Area
41	Acceptance configuration of the system does not replicate the actual operational system configuration: unpredictable consequences and overcost in the field.	6	Yes	2		System Performance
56	We've never tried to make 10 computers work together like this: we don't know what we don't know could delay final system acceptance.	6	Yes	4		System Performance
48	Have to support 50 terminals on each computer with 6 second response time. But have only tested with 25: might have to buy more computers: network overhead, otherwise crash might be suffered.	6	Yes	1		System Performance
44	We performance analysis the test data for the system: we don't know what we don't know.	6	Yes			System Performance
57	The effect of loading on the software was considered to be "negligible" as tests were done. One computer may handle 50 operations OK but 10 computers may not be able to handle 500 operations.	6	Yes	1		System Performance
61	It would be extremely difficult (i.e. not scoped or budgeted) to build a realistic test scenario here: even for one computer: we have no way of knowing the system's performance characteristics until we're in the field.	4	Yes			System Performance
21	Procedures have been made and they indicate there are some problems (e.g. disk down): however critical developers may be unaware of these: developers may waste time on code that won't work right eventually.	4		2		System Performance
24	We've never tried to simulate so many conditions on one project: this could be unforeseen technical challenges.	4		2		System Performance
66	The time to test algorithms may result in no other paper or hardware to be for (BMT) that operations may get first tested and refuse to use the system.	3				System Performance
59	Other customer companies are reluctant to time test performance problems with Barry C and company. The computer itself might fail to meet performance needs.	3				System Performance

25

© 1987 Carnegie Mellon University

Carnegie Mellon University
Software Engineering Institute

Case Study: Interrelationship Digraph - 1



Key:
 Significant = 9
 Medium = 3
 Weak = 1

26

© 1987 Carnegie Mellon University

Carnegie Mellon University
Software Engineering Institute

Case Study: Cause & Effect Relationships

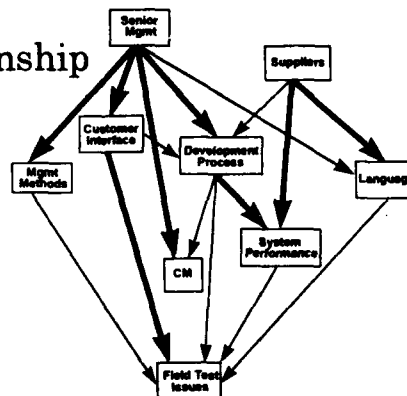
Risk Area	Top Risks/Area	Causes	Results	Totals
System Performance	6	1	3	22
Customer Interface	2	2	1	21
Suppliers	3	3	0	22
Senior Management	2	5	0	37
Management Methods	0	1	1	12
Development Process	3	3	4	33
Language	2	4	2	18
Configuration Mgmt	2	0	4	14
Testing	0	0	5	21

27

© 1997 Carnegie Mellon University

Carnegie Mellon University
Software Engineering Institute
Case Study:
Interrelationship
Digraph—2

Key:
Strong ←←
Medium ←
Weak ←



Interrelationship Hierarchy

28

© 1997 Carnegie Mellon University

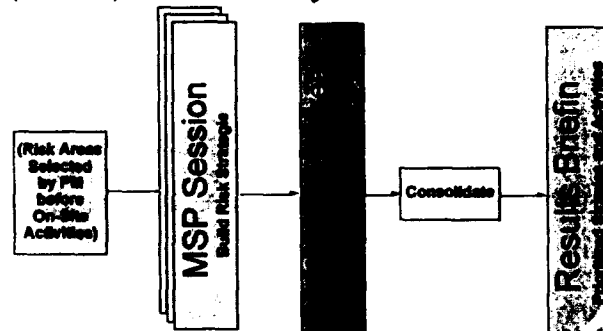
Carnegie Mellon University
Software Engineering Institute**Case Study: Senior Management Risks**

COMPLETE DAS-C RISK LISTING (by Risk Area)

Risk No.	Risk Statement	Risk (From the List)			Risk Area
		SE	Task	Point	
14	The VP is authorizing the project manager to develop new requirements; these may remain hidden, and no test cases will be developed for them.	5	Yes		Senior Management
56	Continuing to use changing business or code influences (new products) that will affect quality of the code integration, month, and schedule.	6	Yes	1	Senior Management
67	Upper management is imposing new quality assurance measures on the project that have not been tested before and for which budget was not provided; may delay progress and cause up costs in rework.	4			Senior Management
16	There is a perception that upper management is unfairly treating the project cost estimate downward to win the contract; people may run up losses to meet deadlines and performance targets.	8			Senior Management
18	VP informed using new system requirements without budget or asked the relief that is in advance the project's time of maturity.	8		1	Senior Management
22	The company doesn't have a program for maintaining and upgrading personnel skills; could hurt long term competitiveness of the company.	8			Senior Management
40	Upper management has not approved C++ training for the project; the needed training may have to come from the project budget; could will be in security.	8			Senior Management
66	Long term maintenance contract may not be a "one time" but business decisions will for reaching impacts are being made on that assumption; could lose money on the project and never make it up.	8		2	Senior Management

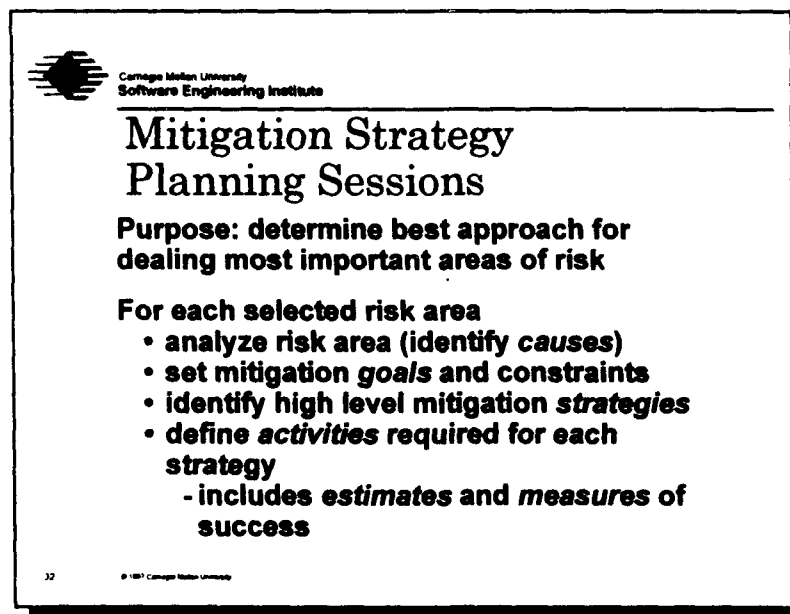
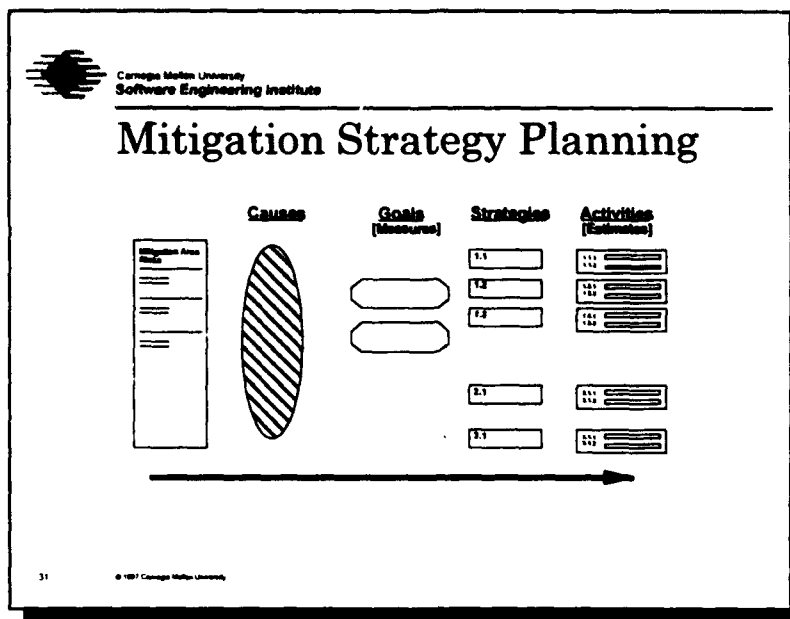
29

© 1997 Carnegie Mellon University

Carnegie Mellon University
Software Engineering Institute**Mitigation Strategy Planning
(MSP) Summary**

30

© 1997 Carnegie Mellon University





Carnegie Mellon University
Software Engineering Institute

Cross-Area Strategy Session

A Cross-area strategy session *may* be held after all risk area plans have been developed.

- **identify conflicts, commonalities, dependencies and possible sequencing.**
- **prioritize plans and actions.**
- **document overall plan.**

33

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

What is Continuous Risk Management?

Software engineering practice (processes, methods, and tools) for managing risks in a project throughout its life-cycle

- **continuously assess what could go wrong**
- **determine which risks are most important**
- **implement strategies to deal with those risks**
- **monitor success and failure of mitigation plans and significant changes in risks**

34

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Identify

Risks must be continuously identified.

Identification of risks must be integrated into existing project management mechanisms.

All project members should be able to identify risks.

35

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Analyze

Risk analysis includes risk evaluation, classification, and prioritization.

- **evaluation: probability, impact, and timeframe**
- **classification: groups of related risks into areas**
- **prioritization: ranked in order of importance**

36

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Plan

Assign responsibility so risks are not lost.

- keep
- delegate
- transfer

Determine a reasonable approach.

- research
- accept
- watch
- mitigate

Develop mitigation plans.

37

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Track and Control

The person responsible for the risk

- acquires and compiles tracking data
- reports risk and mitigation plan status

Control decisions are made by individuals, technical leads, or the program manager.

- replan
- close risk
- invoke contingency plan
- continue tracking against current plan

38

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Risk Database

Database is the simplest means of retaining and keeping risk information up to date.

Data entry forms and reports can be used as the risk information sheet, spreadsheet, and other templates.

Database enables documentation of lessons learned, trend analysis, pattern analysis to support identifying common risks (and solutions) across projects.

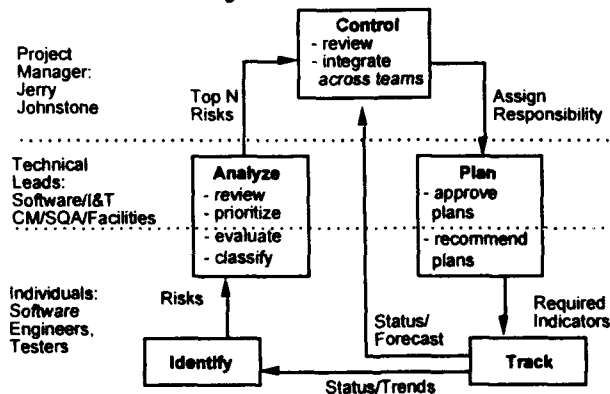
39

© 1997 Carnegie Mellon University



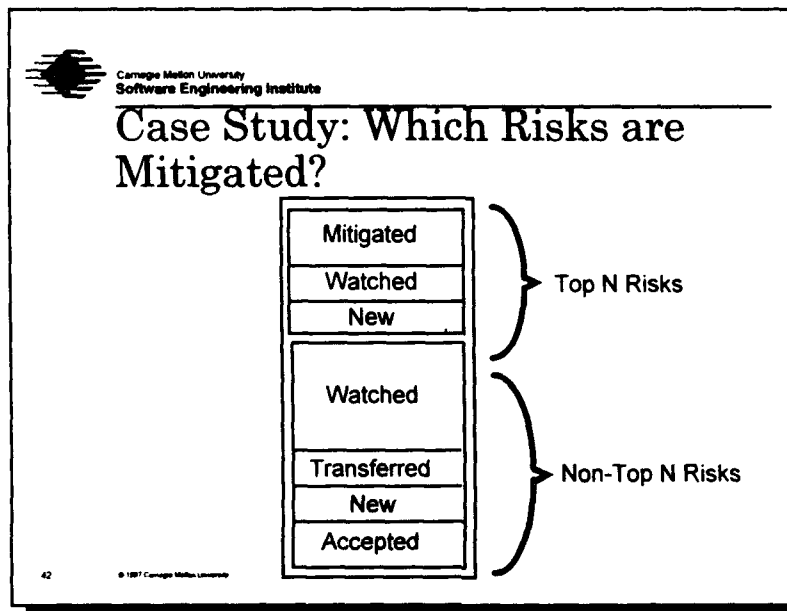
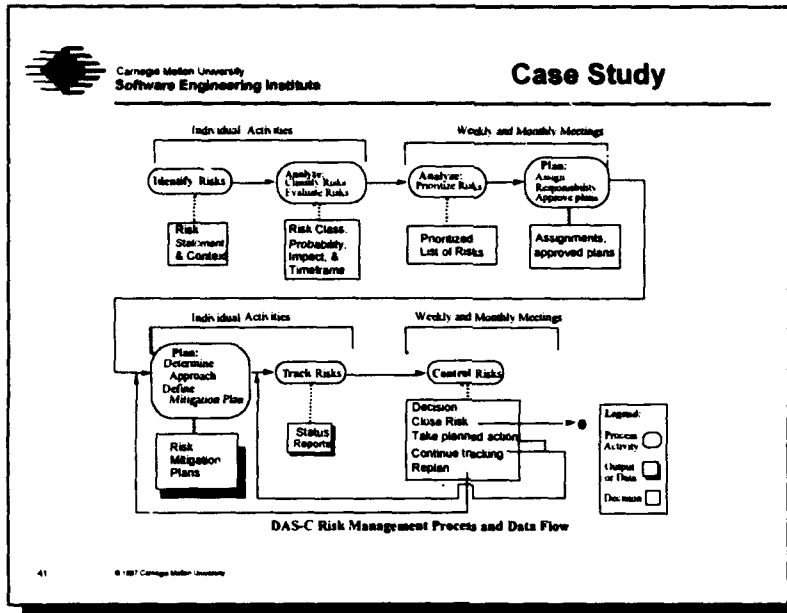
Carnegie Mellon University
Software Engineering Institute

Case Study: Communication



40

© 1997 Carnegie Mellon University



Tuesday 17 June

(T201c) S-21



Carnegie Mellon University
Software Engineering Institute

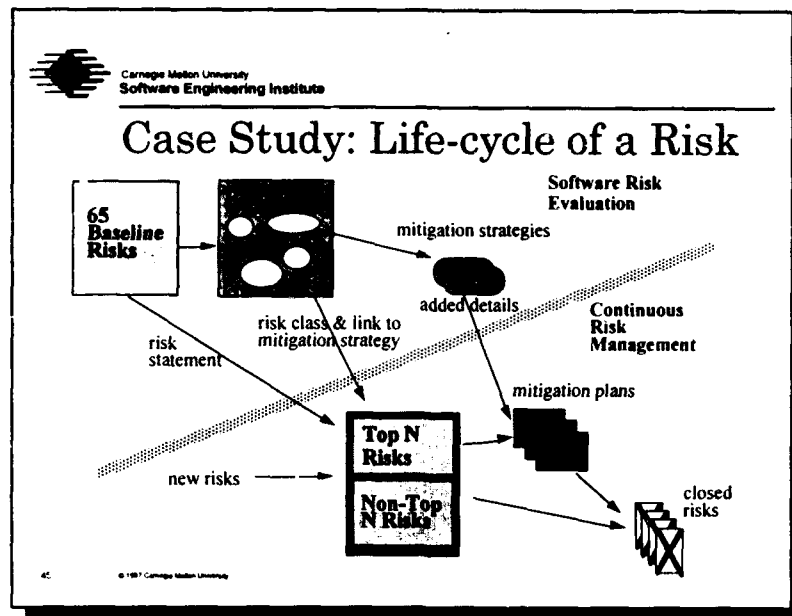
Case Study: Methods and Tools

Risk Information Sheet	Everyone uses to document new risks and add information as risks are managed
Spreadsheet Risk Tracking	Leads use to succinctly report current status information of risks
Taxonomy Classification	Everyone uses as structure for classifying DAS-C's risks
3x4x3 Attribute Evaluation (3 levels of probability, 4 for impact, 3 for timeframe)	Everyone uses to evaluate probability, impact & timeframe
Multivoting	Managers and technical leads use to prioritize risks

4) 2.10^3 Calcium Nitrate (molar)

Case Study: Risk Information Sheet

ID	4	First reviewed on Sheet	Investigator	3422
Priority	9	Statement		
Relevancy	H	There is only limited time for writing (maximum to 6 AM) at court-ordered times. Since time will be used for fixing bugs and not actual work on the case.		
Impact	M			
Comments	None			
Open	Open	Case	Assigned	TL Jones
Revised	Revised	Full Text	10.	
Investigation scheduled and planning led to an underway negotiation of settlement at the customer's facility. Now that we have a better idea of the nature of the problem, we're going to be unable to do anything more than fix the hardware. The hardware is going to be replaced with a bit of help to the. Orders don't always do code improvements to help locate bugs before set. That's not a bit of confidence that we'll be any better with this project.				
Additional Sheet one 1. Customer's equipment is fixed bugs sooner. Requires <i>refactor</i> making for two of the software engineers. R. Events to continue. Making complete by 4/6/96 2. Negotiate an increase of 30% more time in the customer's lab. J. Johnsons to begin negotiations 6/1/96. Needs to be done by 6/1/96				
Contingency Plan and 1st step Build a limited test facility here to help with finding and fixing bug fixes. Start 5/1/96 if customer refuses to provide more test facility time. Estimated cost is \$14,000 to meet a 6/1/96 need date. Overruns and use of additional people needed.				
Status	Waiting on code inspections complete. All modules are being inspected. Risk of bug introduction increased.	Status Date	4/7/96	
Customer's status	Customer seems willing to allocate an additional 20% time still trying to get 20% but it doesn't look likely Jones will try to evaluate this and see whether the contingency plan will still be needed.	5/1/96		
Approved		Checked By	Checked By	Checked By



 Carnegie Mellon University
Software Engineering Institute

Risk Management Must be Tailored

There is no one set of processes, methods, and tools to fit every project.

- tailor for organizations
- tailor for projects

Integrate into existing program management.

Improve as you learn.

Manage cultural change.

- crisis management --> open, proactive management of risk = change in culture

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

What is Team Risk Management (TRM)? -1

Expansion of Continuous Risk Management to an inter-organizational environment

- customers and suppliers (prime contractors, subcontractors, vendors)
- groups that are remote from one another, with different work cultures
- integrated product development teams (IPDTs)

TRM *depends* on Continuous Risk Management in the individual organizations

47

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

What is Team Risk Management? -2

Disciplined team-based approach to jointly managing the important risks to the overall program

At the higher team level, personnel

- determine which joint risks are most important
- implement strategies to deal with the those joint risks
- share responsibility for joint risks
- report mitigation progress to team

48

© 1997 Carnegie Mellon University

Effective and Ineffective



Carnegie Mellon University
Software Engineering Institute

TRM: Identify

Identification of risks is generally left to the individual organizations, through CRM.

Risks are generally re-worded to be suitable for an inter-organizational audience before being reported at the team level.

It is possible (and desirable) for the team to identify risks independently, using techniques tailored from CRM practices.

- team may identify another risk during Team Reviews, for example

48

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

TRM: Analyze

Team evaluation and classification of risks is based on the individual organizations' CRM processes.

The primary analytical task at the team level is prioritization.

- generating a joint list of risks that are most important to the program
- isolating risks for which it is necessary to plan and mitigate on a "common front"
- the joint list of risks is the most visible to the overall program manager

50

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Plan

To the extent possible, planning for all risks should be handled at the CRM level.

- responsibility for risks is assigned to individuals at the CRM level

In special cases, planning is done at the inter-organizational level - Joint Action Planning.

51

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Joint Action Planning

Objective

- sharing data across organizations
- effective use of expertise and knowledge from all organizations

Joint action planning involves all parties

- facilitation may be required for face-to-face use of Problem-Solving Planning method

52

© 1997 Carnegie Mellon University



Track and Control

The organization currently "owning" the risk

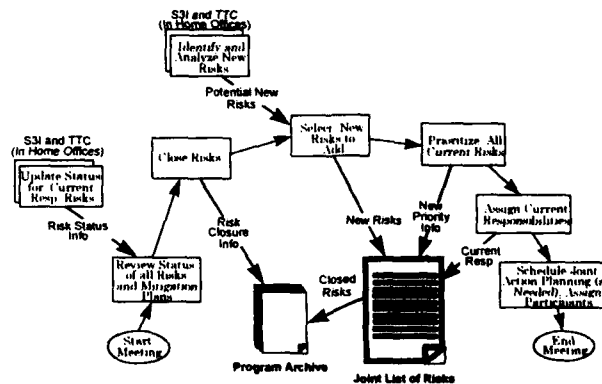
- acquires and compiles tracking data
- reports risk and mitigation plan status

Control decisions are made by the organization responsible for the risk. If it is a *joint risk*, the team will

- delegate planning to an organization or replan by a new joint action planning session
- close the risk
- invoke contingency plan
- continue tracking against current plan



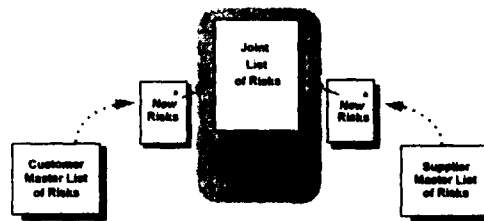
DAS-C Team Risk Management





Carnegie Mellon University
Software Engineering Institute

Team Risk Flow



* For the first Team Review, all risks are "New Risks"

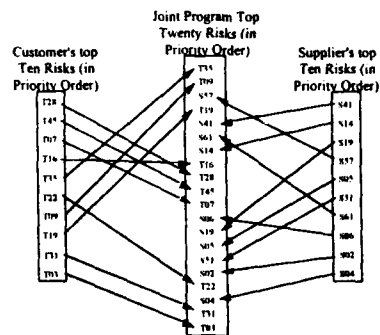
55

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Changes In Risk Priority



56

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

Team Risk Management Summary

TRM involves identify, analyze, plan, track, and control functions.

- **expansion of Continuous Risk Management to an interorganizational team**

TRM is conducted jointly with multiple organizations supporting the same program.

- **meets periodically (i.e., quarterly)**
- **each organization has own CRM process in place - CRM is the foundation for TRM**

57

© 1997 Carnegie Mellon University



Carnegie Mellon University
Software Engineering Institute

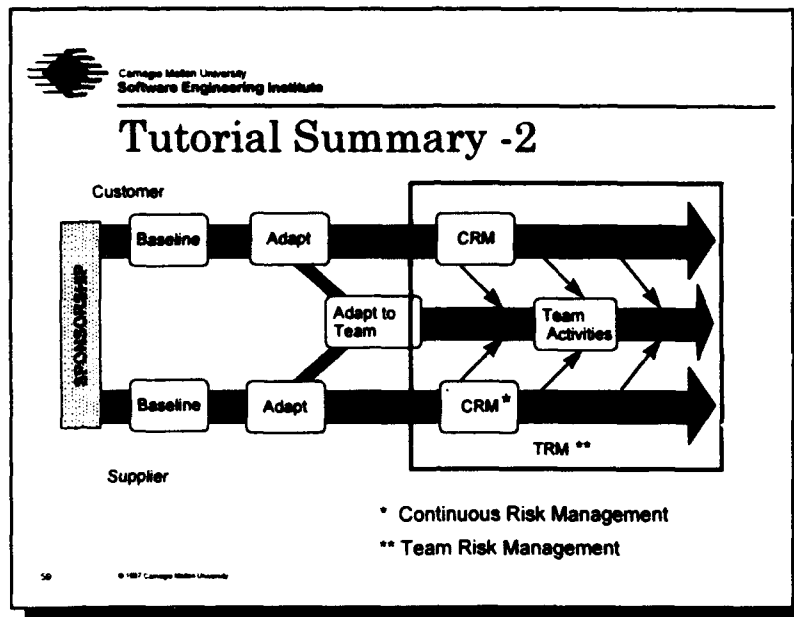
Tutorial Summary -1

Risk Management is the process of

- **continuously assessing what could go wrong**
- **determining which risks are important to deal with**
- **implementing strategies to deal with those risks**

58

© 1997 Carnegie Mellon University



 Carnegie Mellon University
Software Engineering Institute

For Additional Information

Telephone	412 / 268-5800
FAX	412 / 268-5758
Internet	customer-relations@sei.cmu.edu
World Wide Web	http://www.sei.cmu.edu/ technology/risk
U.S. mail	Customer Relations Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213-3890

60 © 1997 Carnegie Mellon University

**Software Risk Management
Tutorial Workbook and
Case Study**

**European SEPG'97
June, 1997**

© 1997 Carnegie Mellon University

Table of Contents

1. Introduction	3
2. Establishing a Baseline	9
3. Continuous Risk Management	19
4. Team Risk Management	28
 Workbook Bibliography	 36
Appendix: Complete Listing of DAS-C's Baseline Risks	37

1. Introduction

This is a workbook designed to be used in conjunction with the Software Risk Management Tutorial. It provides a case study that explores the topic of software risk management in a fictional project: the DAS-C project, in a fictional company, Shoestring Software Systems, Inc.

We will examine the following:

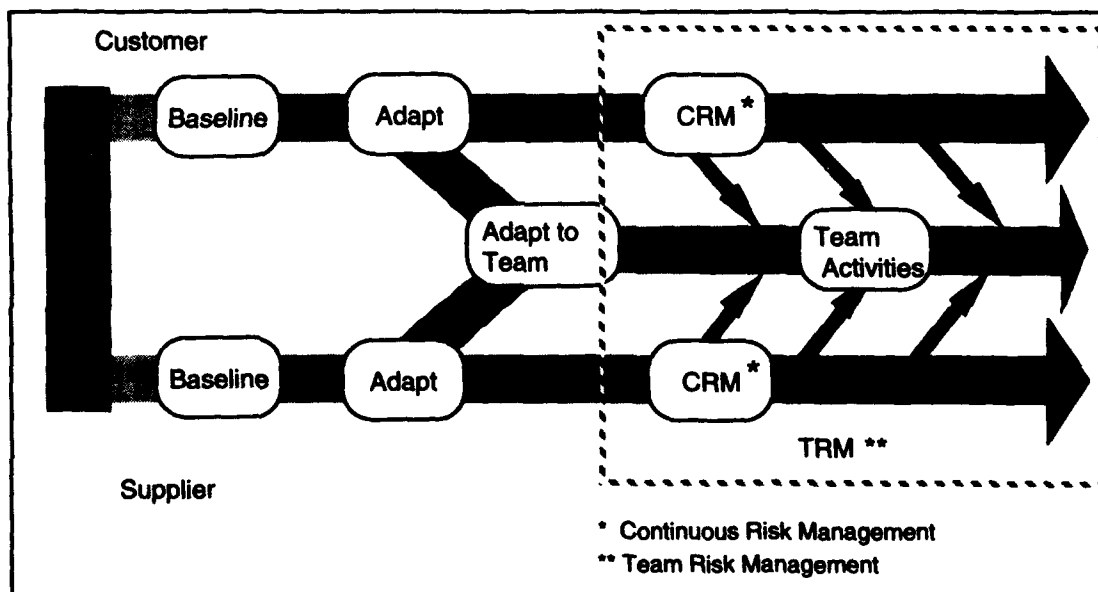
- methods for addressing different aspects of software risk
- a process for managing software risk
- the role of risk management in the larger software and systems development process

Thus, the purpose of this case study is not to design the system illustrated here, but rather to provide an example that is detailed, challenging, and varied enough to illustrate the key concepts of software risk management.

This case study is about a seemingly simple software system. Although this system does not represent a real development effort (not knowingly at least), the problems, issues, and risks that will be presented are real. They have been drawn from the experience of the course designers and from the field work of the Software Engineering Institute's Risk Program.

The following figure shows a roadmap for implementing risk management within a program. This case study deals specifically with establishing the Risk Baseline, Continuous Risk Management (CRM) and Team Risk Management (TRM).

Figure 1: Risk Management Roadmap



1.1 Case Study

The timeframe for this case study is approximately 1985. Please consider the risks and the technology available to the people in the case from that perspective (for example, a 360 MB disk-drive is a big—and expensive—disk-drive!)

Description—The Computerized Directory Assistance System (DAS-C) is a project already under contract with a high-tech engineering and development contractor, Shoestring Software Systems Inc. (S3I), a California systems integrator and custom software developer. The customer for the project is Toivolia Telephone Company (Toivolia or TTC), a Regional Telephone Operating Company located in northern Michigan.

The DAS-C is a whole new concept for Toivolia—in fact, for the whole regional telephone industry: When their operators receive a directory assistance call, they'll be able to turn to a screen, type the first 3 letters of the last name, and immediately get a screen page (or pages) to show them all the names in the company's system that fit that combination. From there they can scan the names and addresses in consultation with the caller, to quickly locate the person the caller wanted.

System Requirements:

- Directory assistance retrievals shall not exceed 3 seconds (from the time the operator initiates the query (presses enter) to the time the first screen is presented to the operator). The contract is written in such a way that if the 3-second response time is not met for 90% of up time within each month, a penalty is assessed against S3I. The penalty is taken out of the maintenance contract.
- The phone number listings are obtained from the company that prints the white pages for the area. The phone numbers are delivered on magnetic tape to the operations center. The database shall be updated and ready for use by 8 AM on the day following the day of the tape delivery.
- System downtime, defined as any period in which an operator who is in contact with a subscriber cannot access listing information, must be no more than a total of five minutes in any calendar-month period per operator.

Proof of Principle—S3I developed a patented retrieval algorithm using only the first 3 letters of a last name. The algorithm was demonstrated on a 25-operator system, which won the contract for S3I. Long-term use of the algorithm by Toivolia is contingent on S3I's receiving a maintenance contract—S3I will not sell Toivolia full rights to the algorithm.

Design—S3I proposed 10 EasyComp computers with two 360-mb removable disk drives per computer. Based upon straight-line projections from the 25-operator system, the 10 EasyComp computers will just be able to handle 500 simultaneous operators (50 operators per computer) within the performance constraints. Each computer has the complete phone listing for the area code on its disk drives. This was done to meet the data retrieval time requirement.

Electronic Switch—The terminals are connected to the computers through an electronic switch. The electronic switch provides the capability to connect any terminal to any computer. When a terminal is turned on, the switch finds the first non-busy port and connects the terminal. Ports can be manually disabled to prevent terminal connection to specific computers. Because the terminals are connected to the electronic switch through phone lines, the phone lines often take a "line-hit" and momentarily disconnect the terminal.

© 1997 Carnegie Mellon University

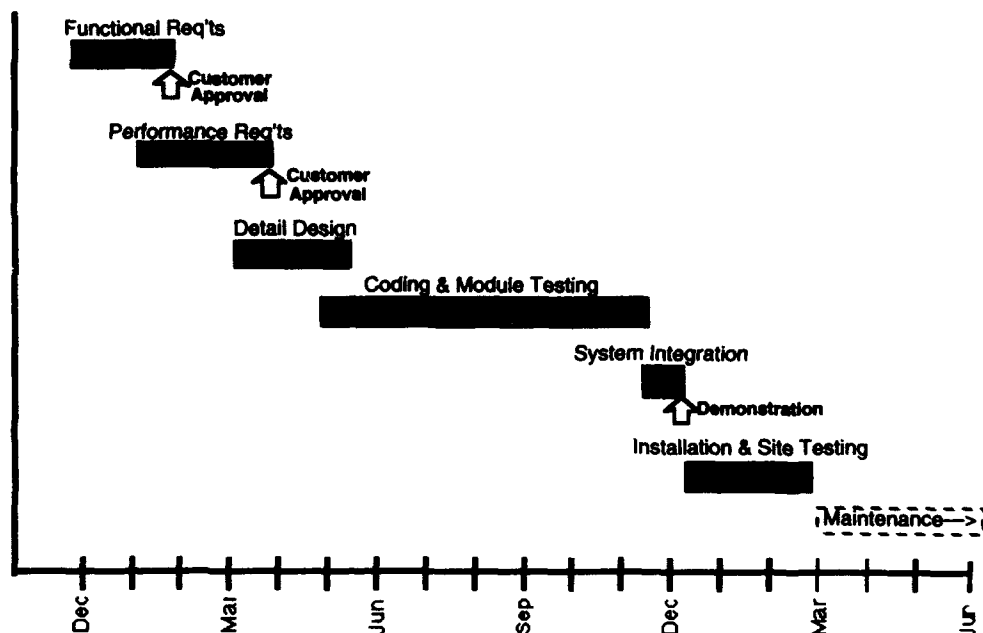
Development Approach—The project has already written (and Toivolia has approved) a “Functional Specification” which describes the general hardware and software architecture, along with a listing and brief description of all the functions the system is to perform. This Functional Specification was basically an elaboration of the proposal that won S3I the contract in the first place.

The project is currently in the process of writing a “Performance Specification” document, which will elaborate on the Functional Specification by describing all “external” characteristics of the system to be delivered, e.g., response times, security features, user interface screen layouts, and function interactions. Once Toivolia approves this document, S3I plans to proceed through design, coding, testing, and system integration without further approval from Toivolia, based on the approval of the Performance Specification (“All the rest is just design detail”). C++ is the programming language (this was a requirement in Toivolia’s original Request for Proposals).

S3I has chosen to follow a waterfall development methodology, ending with an acceptance demonstration test on a representative subset of S3I’s equipment. After shipment to Toivolia’s facilities, the full system will be installed, and final “full up” integration, testing, start-up, and user training will be performed by S3I field representatives temporarily on assignment at Toivolia.

Post-Project Strategy—After this project is in successful operation, S3I plans to aggressively sell the system to other regional telephone companies around the U.S. The S3I marketing division already has a team in place to determine features that will give the system the widest possible applicability, and the roll-out campaign is on the drawing board. This work is supposed to be covered by company overhead and should not affect the contract with Toivolia.

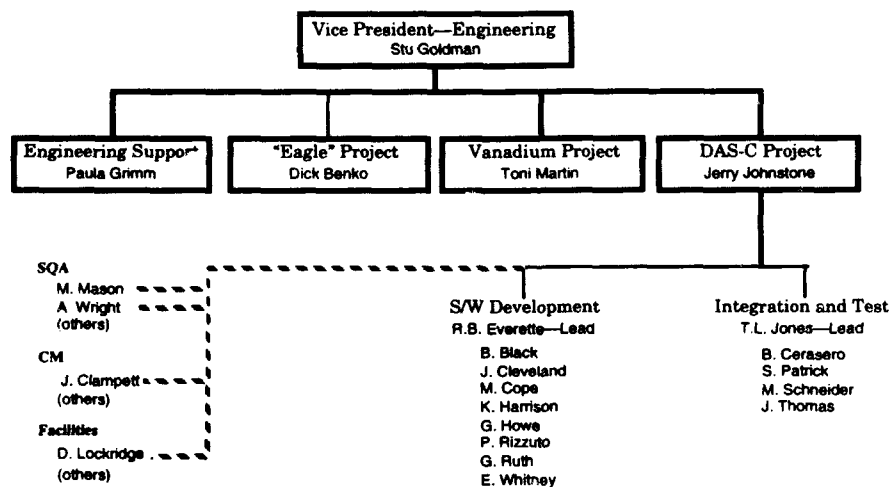
Project Manager’s Briefing (excerpts)



“This is the schedule for the project—the dark bars cover the part that is under my management, and this bar for “Maintenance,” of course, extends on into the indeterminate future.

© 1997 Carnegie Mellon University

- We've passed our first key milestone—delivery and approval of the Functional Requirements document. This milestone actually occurred about a month later than expected because of slow approval by Toivolia. However, we were able to begin our work on the Performance Requirements anyway, and I don't think we lost any time.
- Our next major milestone will be when we deliver the Performance Requirements document and it is approved by Toivolia. This document will describe the screen, report, and file layouts, security features, and other "externals" of the software, plus the final detailed hardware listing. Once this is approved everything that is of concern to Toivolia will have been addressed, and we can proceed through Detail Design (that's the internal software architecture, dataflows, and module block descriptions) and Coding and Module Testing without having to concern Toivolia further about our work.
- At the end of System Integration we'll do a one-week demonstration test of the integrated system (scaled down, of course) in our offices here, and then we ship to Toivolia. Customer representatives will just be here during the demonstration to verify that we did what our Functional and Performance Requirements documents said we would.
- We expect Installation and Site testing to take about half of the 12-week period shown; after that we'll have a couple of our programmers hang around to show the Toivolia operators how to use their new terminals and clear up any little glitches that we may have missed back here. In their spare time they can do documentation cleanup.
- Finally, of course, we'll turn the system over to our Customer Services department for long-term maintenance—under a very lucrative time and materials contract.

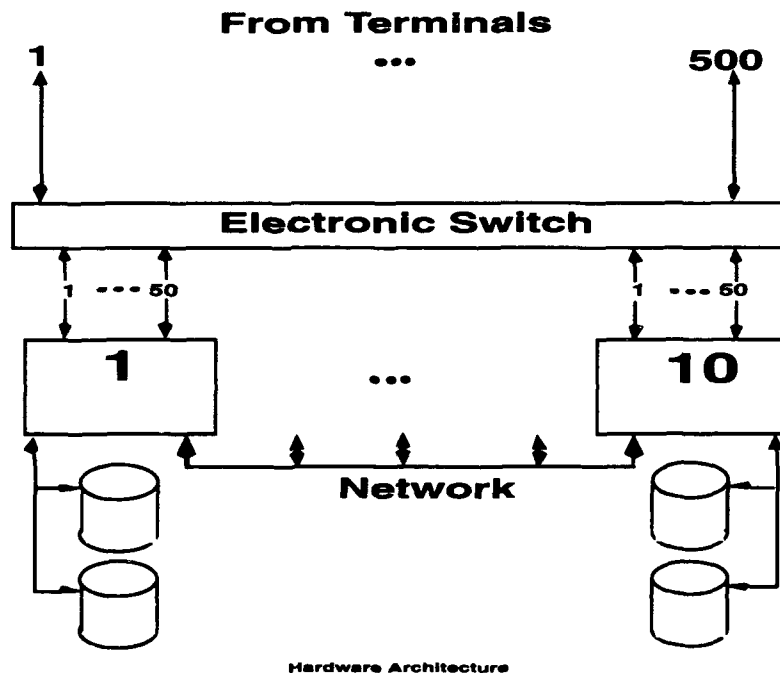


"This abbreviated organization chart is to give you an idea where the DAS-C project fits in our Engineering Division. Here I am on the right-hand side of the chart, the DAS-C project manager.

- DAS-C is one of three projects currently underway in the division. The other two are the "Eagle" project, which is a classified project for the Department of Defense, and the Vanadium project for General Electric Corporation.

© 1997 Carnegie Mellon University

- Paula Grimm heads up the Engineering Support activity, which is also at the project manager level of S3I. This department includes the division's Software Quality Assurance, Configuration Management, and Facilities activities, among others. You see a few of Paula's people named here; they're people who work for Paula, but who spend almost all their time supporting the DAS-C effort.
- The four of us report directly to Stu Goldman, the Division Manager, recently named a vice president of the company.
- My project is split in two, as you see: Software Development and Integration and Test. Everett, who leads the software effort, is almost a legend here at S3I and is the developer of the quick-as-lightning three-letter algorithm that got us this job in the first place. Eight software developers report to Everett.
- Jones is pretty new to S3I, but brings a wealth of experience about integration and test. Jones also is a tremendous authority on architecture—both software and hardware—and I'm just sorry that Jones wasn't here back when we were putting the proposal together.

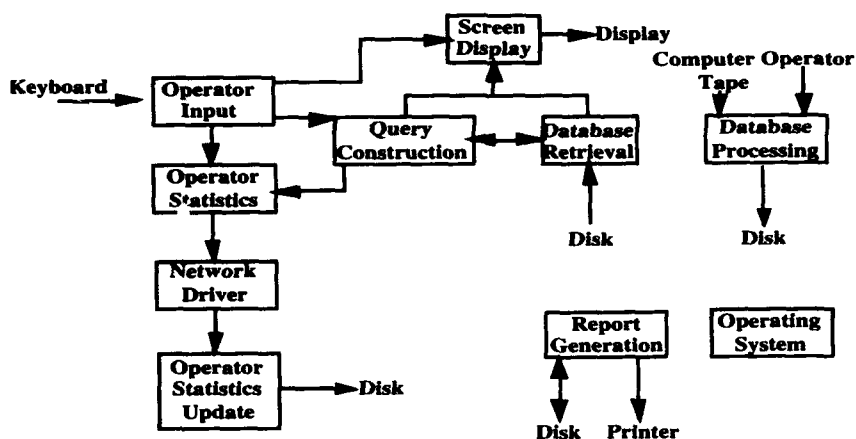


"This is the DAS-C hardware architecture.

- The system can handle up to 500 operators, and they all come in through this electronic switch. That makes the reliability of switch really important—there's only one vendor for it, but Analog Electronics has earned an outstanding reputation for reliability of their equipment.
- The ten EasyComp computers are the real heart of the system. On the diagram we've shown Number 1 and Number 10 as "typical," but each of the ten is identical, with four Megabytes of random access memory and two 360 Megabyte disk drives. The disk drives contain identical, up-to-date databases of all the system subscribers, their telephone numbers, and addresses (plus other administrative data for Toivolia Telephone to use) and the software to operate the computer.

© 1997 Carnegie Mellon University

- Behind the EasyComp computers is a network that joins all ten of them together. As each operator handles calls, statistics are accumulated and sent over the network to whichever computer is designated as the *master* computer and are logged there.



"This is the software architecture for the DAS-C system:

- The operator works through the *Keyboard* and the system sends data back to the operator by the *Display*.
- The keyboard input of the operator goes into the *Operator Input* software module, and that is used in *Query Construction* by applying the three-letter combination to the text string. This is the proprietary heart of the system. It's also the key to our maintenance arrangement with Toivolia, because it is patented, and we retain the rights to it; it can only be used on the system as long as we are the system maintainers.
- In parallel with query construction, the software compiles operator statistics in the *Operator Statistics* module, which go through the *Network Driver* to the Master Computer. The *Operator Statistics Update* module on the Master Computer updates the record for each operator by name, and records it on disk.
- The *Query Construction* module taps the information stored on the disk through the *Database Retrieval* module, which starts passing screen pages along to the *Screen Display* module.
- *Database Processing* really is fairly simple: In the early morning of each day a tape is delivered to the computer system, and all of the disk drives are updated, one at a time over the network. The ten computers will then all have the same up-to-date information—any changes that might have happened in the last 24 hours: new subscribers, or old subscribers who have left, and so forth.
- *Report Generation* is an off-line operation that accesses the disk for information, formats it, and sends it to the printer.
- Also shown as "off-line" is the EasyComp operating system—EasyOS—which we don't get involved with—we use it pretty much the way we get it."

© 1997 Carnegie Mellon University

2. Establishing a Baseline

Jerry Johnstone, DAS-C Program Managers, had a Software Risk Evaluation (SRE) performed on the DAS-C project in order to establish the project baseline set of risks. An SRE Team was used to conduct the Software Risk Evaluation; the SRE Team also provided some of their own expertise during analysis of the risk information. We will look at the results of DAS-C's Risk Identification and Analysis effort (a complete list of risks can be found in the Appendix), and discuss how to interpret them. We will then look at developing mitigation strategies for a set of related risks.

Jerry wanted to use a consistent format or structure for the risk statements, so he will keep the structure developed during the Software Risk Evaluation. The risk statements have two parts:

- Condition: something that is true or widely perceived to be true
- Consequence: something that may occur as a result of the condition

Conditions and consequences may be separated by a semi-colon or a dash. A well formed risk statement usually has only one condition, but may have one or more consequences.

2.1 Risk Identification & Analysis Results

Below are the results of the DAS-C Risk Identification and Analysis phase of the Software Risk Evaluation. In all, 65 risk statements were generated. The SRE team estimated the risk exposure for these risks, grouped them into risk areas, and prioritized them.

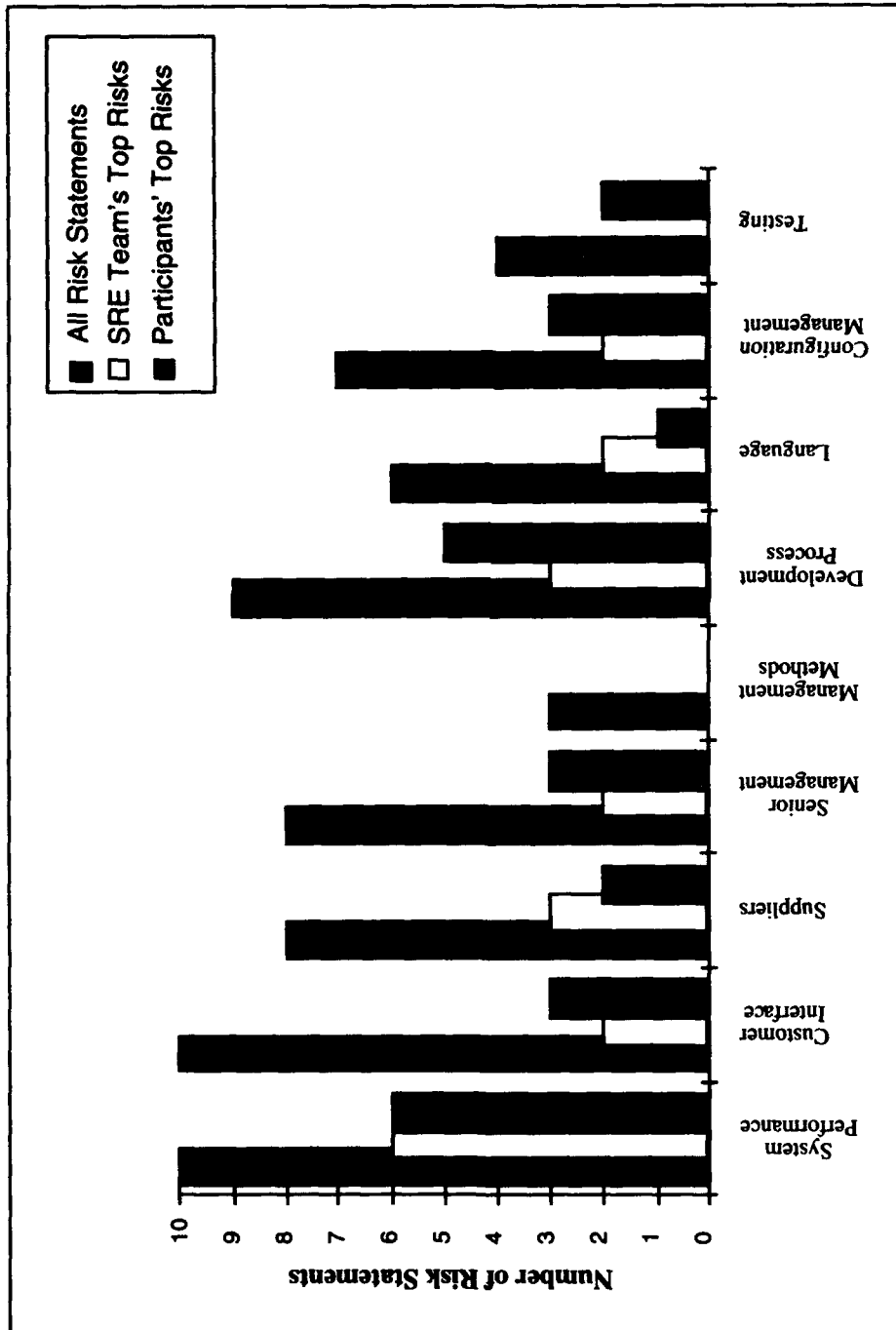
Table 1 shows the number of risks per risk area as well as how many of the risks in each area were ranked among the top risks by the SRE Team, the participants from the DAS-C project, and the overlap between their rankings. The bar chart in Figure 2 also shows the distribution of risks among all the risks areas. In addition, the team's top risks and the participants' top risks are displayed.

Table 1: Summary Data from RI&A Phase

Risk Areas	Total Risk Statements per Risk Area	SRE Team's Top Risks	Participants' Top Risks	Overlaps
System Performance	10	6	6	4
Customer Interface	10	2	3	1
Suppliers	8	3	2	2
Senior Management	8	2	3	1
Management Methods	3	0	0	0
Development Process	9	3	5	3
Language	6	2	1	0
Configuration Management	7	2	3	1
Testing	4	0	2	0
Totals	65	20	25	12

For the purposes of this tutorial, the details of how the risk statements were obtained and analyzed by the SRE team are unimportant. We will concentrate on the results of the Risk Identification and Analysis phase - the risk areas.

Figure 2: Risk Areas



2.2 Risk Areas

In this section, we will look at one of the risk areas generated during the RI&A phase of the DAS-C case study. Risk Exposure (RE) is a function of probability and impact. Impact is characterized on the Y axis of Figure 3.

Figure 3: Risk Exposure Chart

	Probability		
	3 Very Likely	2 Probable	1 Improbable
Impact	4—Catastrophic 6	5	4
3—Critical 5		4	3
2—Marginal 4		3	2
1—Negligible 3		2	1
= High = Medium = Low			

The ten risk statements that constitute the “system performance” risk area listed in Table 2. Look for the Risk Exposure (RE) value for each risk statement in the third column.

Table 2: DAS-C Risk Statements in the "System Performance" Risk Area

Risk No.	Risk Statement	Team Top			Risk Area
		Risk No.	Risk	Top 2	
41	Acceptance configuration of the system does not replicate the actual operational system configuration; unpredictable consequences and rework in the field.	6	Yes	2	System Performance
33	We've never tried to make 10 computers work together like this; we don't know what we don't know; could delay final system acceptance.	5	Yes	3	System Performance
43	Have to support 50 terminals on each computer with 3 second response time, but have only tested with 25; might have to buy more computers, network overhead, electronic switch might be affected.	5	Yes	1	System Performance
44	No performance analysis has been done for the system; we don't know what we don't know.	5	Yes		System Performance
57	The effect of loading on the network was considered to be "negligible"—no tests were done. One computer may handle 50 operators OK, but 10 computers may not be able to handle 500 operators.	5	Yes	1	System Performance
61	It would be extremely difficult (i.e., not scoped or budgeted) to build a realistic test scenario here, even for one computer; we have no way of knowing the system's performance characteristics until we're in the field.	4	Yes		System Performance
21	Prototypes have been made, and they indicate there are some problems (e.g., disk driver), however system developers may be unaware of these; developers may waste time on code that won't work right eventually.	4		2	System Performance
24	We've never tried to interconnect so many computers on one project; there could be unforeseen technical challenges.	4		2	System Performance
36	The three-letter algorithm may result in so many pages of possibilities (e.g., for "SMI") that operators may get frustrated and refuse to use the system.	3			System Performance
59	Other software companies are rumored to have had performance problems with Easy Comp computers. The computer itself might fail to meet performance needs.	3			System Performance

2.3 Interrelationship Digraph

The Interrelationship Digraph is the very important prelude to mitigation strategy planning. It is used to systematically identify and analyze cause and effect relationships among critical risks so that key drivers can become the heart of effective solutions.

- Encourages team to think in multiple directions rather than linearly
- Explores the cause and effect relationships among all the issues, including the most controversial
- Allows the key issues to emerge naturally rather than allowing the issues to be forced by a dominant or powerful team member
- Systematically surfaces the basic assumptions and reasons for disagreements
- Allows a team to identify root cause(s) even when credible data doesn't exist

Spend a few minutes reviewing the DAS-C interrelationship digraph below.

Figure 4: DAS-C Interrelationship Digraph

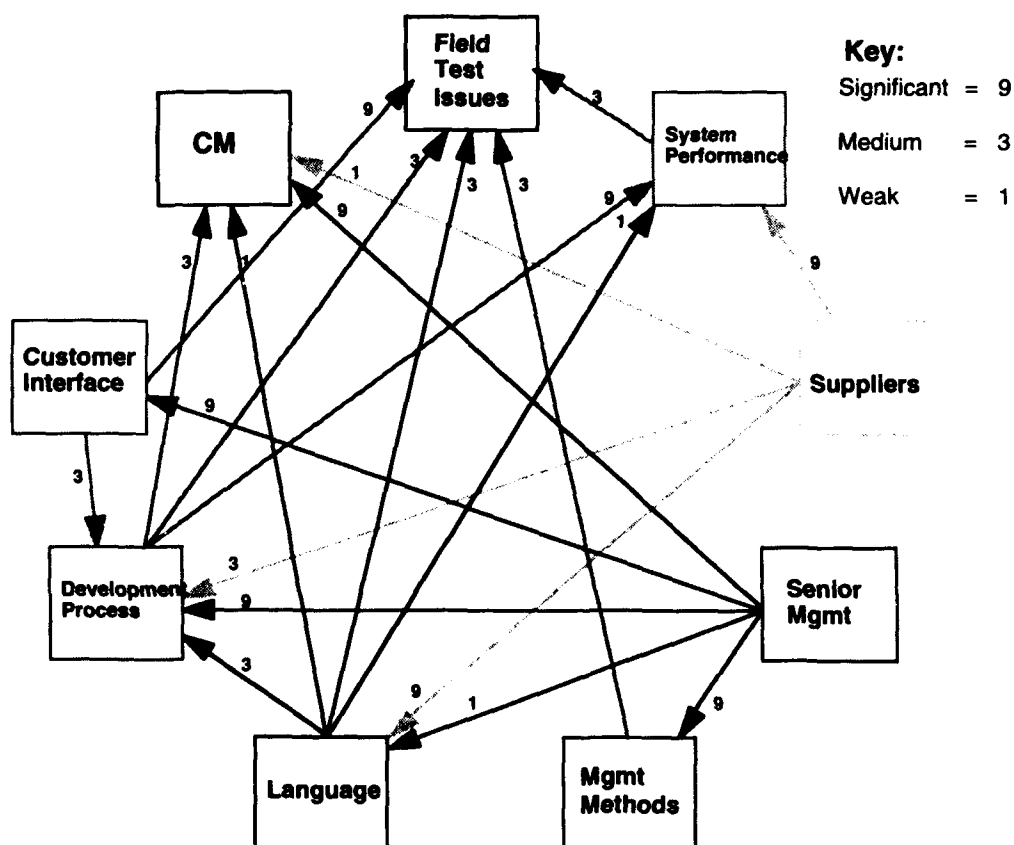


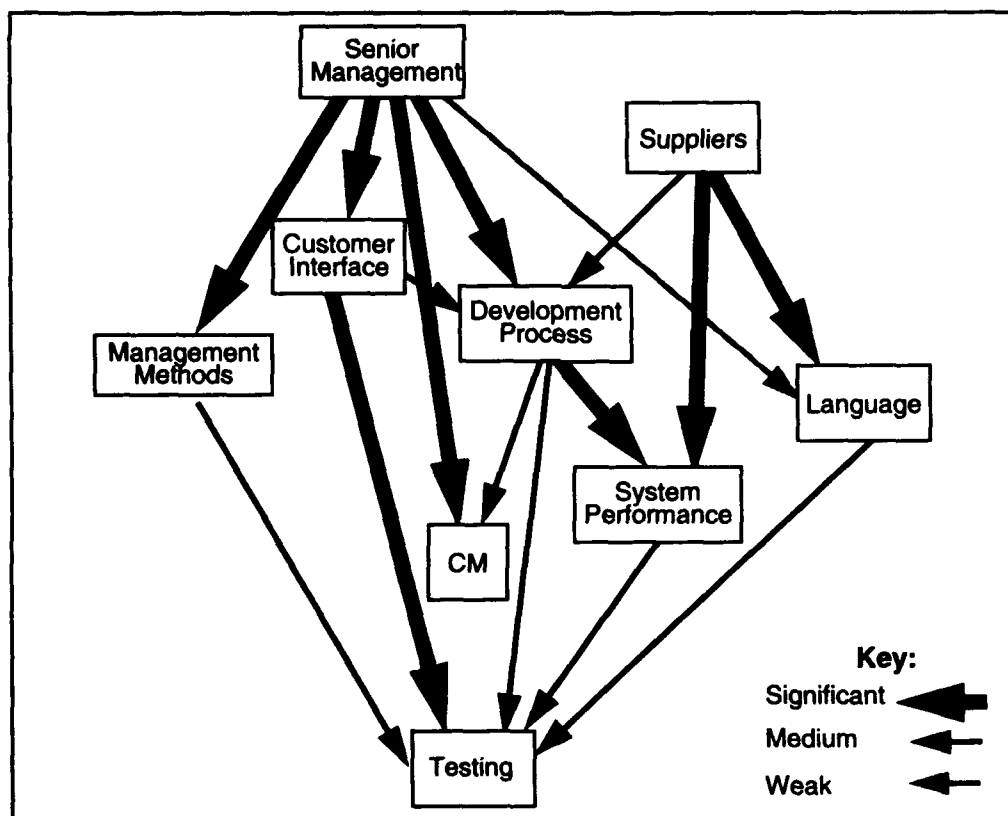
Table 3 summarizes some information about the interrelationships shown above. For example, it shows that Senior Management is a cause or driver for 5 other risk areas, but Senior Management

is not a result of any of the other areas. This fact can be used to redraw the above Interrelationship Digraph from a different perspective (see Figure 5).

Table 3: Cause and Effect Relationships among Risk Areas

Risk Area	Count of SRE Team's Top Risks in Area	Cause/Driver	Result/Rider	Total Weight
System Performance	6	1	3	22
Customer Interface	2	2	1	21
Suppliers	3	3	0	22
Senior Management	2	5	0	37
Management Methods	0	1	1	12
Development Process	3	3	4	33
Language	2	4	2	18
Configuration Management (CM)	2	0	4	14
Testing	0	0	5	21

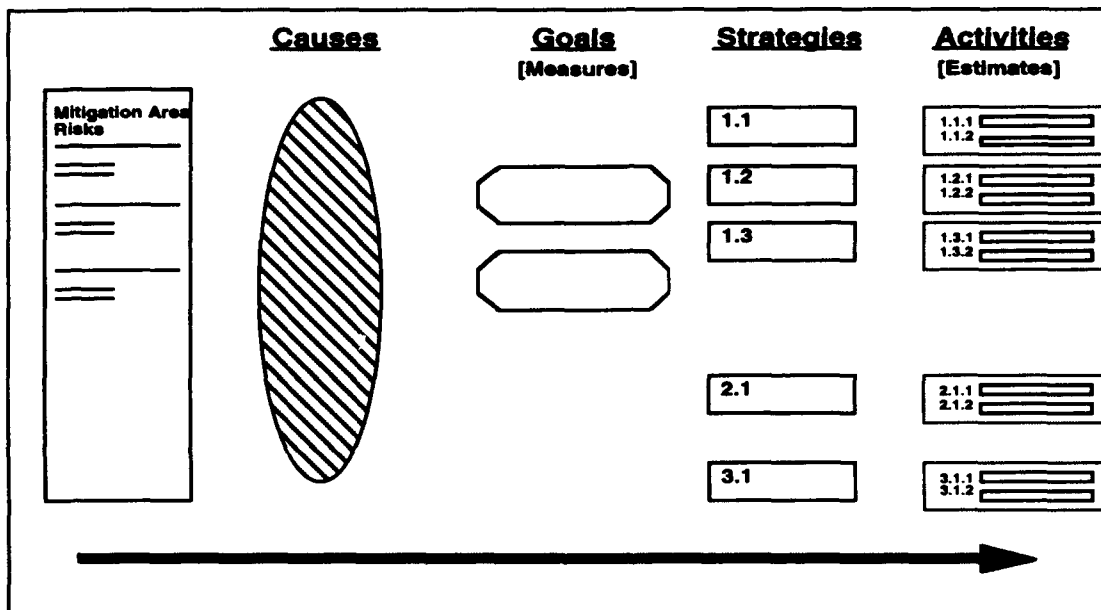
Figure 5: DAS-C Interrelationship Hierarchy: Alternative View



2.4 Mitigation Strategy Planning

The Mitigation Strategy Planning phase of the Software Risk Evaluation begins the strategy to develop a concrete plan for managing and mitigating some of the most important risks identified during the Risk Identification and Analysis phase. Figure 6 summarizes the general flow of information in mitigation Strategy Planning.

Figure 6: Mitigation Strategy Planning Information Flow



The Mitigation Strategy Planning process steps are further described in Table 5.

Table 5: Mitigation Strategy Planning Process Details

Step	Purpose	Primary Activities
Identify causes	Identify the factors that need to be addressed by the mitigation plan	- Review major risks - Identify key or root causes - Identify the key causes the mitigation plan should address
Identify mitigation goals	Describe the desired end state of the mitigation - what "success looks like"	Expand the program manager's mitigation goals (modify, delete or add goals as necessary)

Step	Purpose	Primary Activities
Identify strategies	Describe an approach to address key causes and achieve the mitigation goals - the "what to do" not the "how to do"	- Brainstorm possible strategies - Evaluate and reduce the set
Identify activities	Describe the high-level activities ("how to") required to implement each strategy	- Brainstorm activities for each strategy - Discuss dependencies and interrelationships - Select activities to implement strategies
Estimate scope of activities	Describe the scope for implementing each activity in terms of time and effort	- Develop estimates for: - people involved - person-days effort per person - calendar days or weeks to complete
Identify success measures	Describe a qualitative or quantitative means to track progress towards achieving the mitigation goal(s)	- Key measures or metrics brainstormed and discussed - Select key measures to use and decide how often to report status
Cross-area strategy planning	Assure mitigation plans for each risk area minimize duplication, conflicts, and schedule dependencies and maximize effective use of resources	- Identify conflicts and synergies among the strategies and actions developed for each risk area - Adjust mitigation strategies, activities, resources, etc. as needed

3 Continuous Risk Management

After the SRE, Jerry Johnstone, Program Manager, decided to make risk management a part of DAS-C's program management. He brought in a consultant, who helped them set up a standard risk management practice and then coached them through the next 6 weeks while everyone got up to speed. The risk management process they defined and put in place is a relatively simple one, but Jerry and his DAS-C personnel decided it was better to start simple than to get too caught up in a complicated process.

The following aspects of DAS-C's Continuous Risk Management are described:

- communication framework and distribution of risk management activities
- risk management process and data flow
- selected methods and tools
- how do the risks from the Software Risk Evaluation fit into DAS-C's Continuous Risk Management

Table 4 lists the risks in the Senior Management risk area.

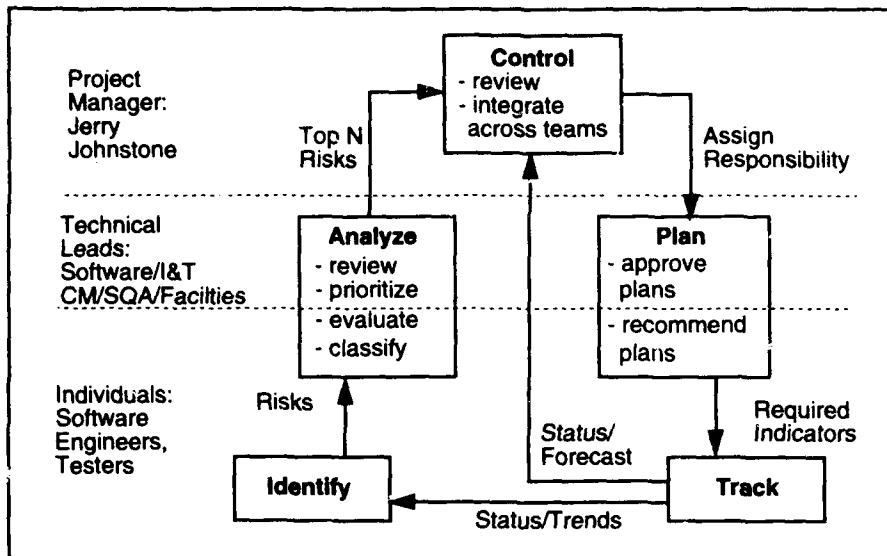
Table 4: Senior Management Risks

Risk No.	Risk Description	Impact	Time	Cost	Area
14	The VP is undercutting the project manager and introducing new requirements; these may remain hidden, and no test cases will be developed for them.	5	Yes		Senior Management
56	Requirements are changing because of outside influences (vice president); this will affect quality of the code, integration, morale, and schedule.	5	Yes	1	Senior Management
37	Upper management is imposing new quality assurance measures on this project that have not been tried before and for which budget was not provided; may delay program and drive up costs inordinately.	4			Senior Management
16	There is a perception that upper management arbitrarily revised the project cost estimate downward to win the contract; people may give up trying to meet deadlines and performance boogies.	3			Senior Management
18	VP introducing new system requirements without budget or schedule relief; this is muddying the project's lines of authority.	3		1	Senior Management
22	The company doesn't have a program for maintaining and upgrading personnel skills; could hurt long-term competitiveness of the company.	3			Senior Management
40	Upper management has not approved C++ training for project staff— the needed training may have to come from project budget; profit will be in jeopardy.	3			Senior Management
63	Long-term maintenance contract may not be a "sure thing," but business decisions with far-reaching impacts are being made on that assumption; could lose money on this project and never make it up.	3		2	Senior Management

3.1 Communication Framework

Figure 11 shows how the functions of Identify, Analyze, Plan, Track, and Control are implemented within DAS-C as well as what risk information is being communicated within DAS-C.

Figure 11: Communication Framework



Individual software engineers and testers, technical leads, and the project manager:

- identify new risks
- estimate the probability, impact, timeframe, and classification of risks
- research and recommend mitigation plans
- track risks and the progress of mitigation plans.

The technical leads (i.e., software development lead—Everette, or CM manager):

- ensure the accuracy of the probability/impact/timeframe and classification
- modify and approve recommended mitigation plans
- prioritize the risks which are managed within their teams
- report their Top N risks and issues to the project manager
- collect and report general risk management measures

The project manager:

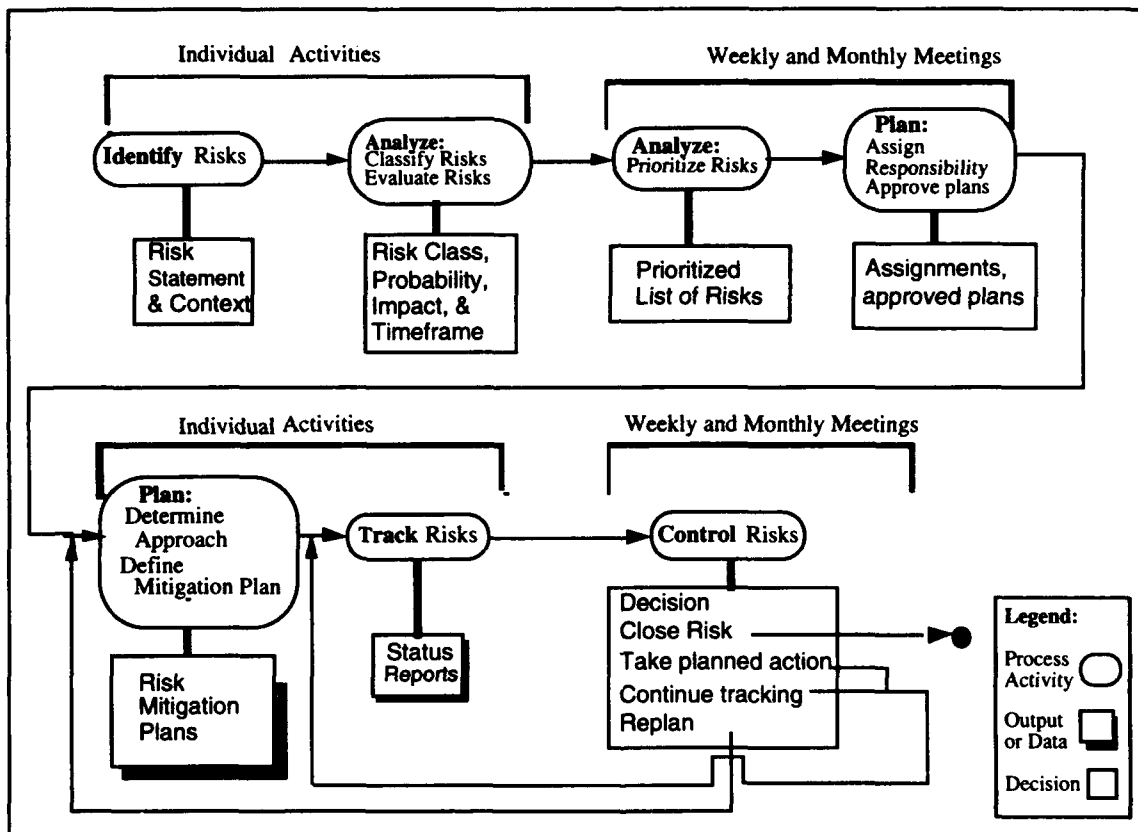
- integrates risk information from all of the technical leads
- reprioritizes all risks to determine the Top N project risks
- controls where major mitigation resources are spent
- assigns or changes the responsibility for risks and mitigation plans within the project
- handles communication external to the project
- evaluates the effectiveness of the risk management practice

© 1997 Carnegie Mellon University

3.2 Process and Data Flow

The overall process flow for DAS-C risk management is shown in the Figure 13.

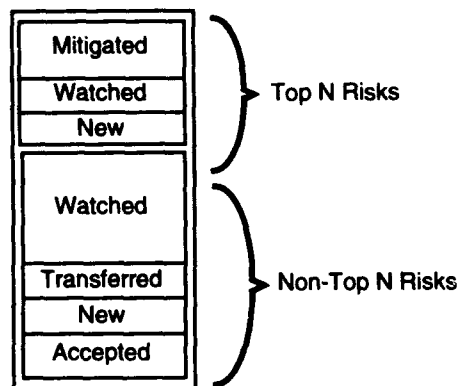
Figure 13: DAS-C Risk Management Process and Data Flow



© 1997 Carnegie Mellon University

As seen in Figure 12, the Top N risks are mitigated or watched. Non-Top N risks are watched, transferred, or accepted. New risks are statused until an approach can be determined.

Figure 12: Top N and Non-Top N Risks



3.3 Methods and Tools

Table 6 lists the methods and tools that will be used to manage risks on the DAS-C project.

Table 6: DAS-C's Risk Management Methods and Tools

Methods and Tools	Used For:
Risk Information Sheet	Everyone uses to document new risks and add information as risks are managed
Spreadsheet Risk Tracking	Leads use to succinctly report current status information of risks
Taxonomy Classification	Everyone uses as the structure for classifying DAS-C's risks
3 x 4 x 3 Attribute Evaluation 3 probability, 4 impact, and 3 timeframe levels	Everyone uses these to evaluate probability, impact, and timeframe
Multivoting	Managers and technical leads use to prioritize risks

A sample risk information sheet for one of DAS-C's risks is provided in Figure 9.

Figure 9: Risk Information Sheet

ID	4	Risk Information Sheet		Identified: 3/12/96
Priority	9	Statement There is only limited time for testing (midnight to 6 AM) at customer's site; testing time will be used for fixing bugs and not actual performance tests.		
Probability	H			
Impact	M			
Timeframe	Near	Origin Baseline	Class Field Test	Assigned To: T.L. Jones
Context Inadequate scheduling and planning led to an inadequate negotiation of test time at the customer's facility. Now that we have a better idea of the time required, we know we're going to be unable to do enough performance testing. Most of the allotted time is going to have to be used for fixing bugs. Our history indicates we tend to go into testing with a lot of bugs to fix. Coders don't always do code inspections to help locate bugs before test. There's not a lot of confidence that we'll be any better with this project.				
Mitigation Strategy 1. Increase code inspections to find bugs sooner. Requires refresher training for two of the software engineers. R. Everette to coordinate - training complete by 4/6/96. 2. Negotiate an increase of 30% more time in the customer labs - J. Johnstone to begin negotiations 4/1/96. Needs to be done by 6/1/96.				
Contingency Plan and Trigger Build a limited test facility here to help with finding and testing bug fixes. Start 5/1/96 if customer refuses to provide more test facility time. Estimated cost is \$14,000 to meet a 6/1/96 need date. Overtime and use of additional people needed.				
Status Training on code inspections complete. All modules are being inspected. Rate of bug identification increased 10% so far.		Status Date 4/7/96		
Customer seems willing to allocate an additional 20% time. Still trying to get 30% but it doesn't look likely. Jones will need to evaluate this and see whether the contingency plan will still be needed.		5/1/96		
Approval _____		Closing Date ____/____/____	Closing Rationale	

A sample spreadsheet for DAS-C is provided in Figure 10. Note that some risk areas are being statused as well as individual risks.

Figure 10: Risk Spreadsheet

6/10/96

Risk Spreadsheet

Risk ID	Priority	Risk Statement	Status Comments	Probability	Impact	Assigned To
40	1	Upper management has not approved C++ training for project staff—the needed training may have to come from project budget; profit will be in jeopardy.	Promised funding has been cancelled due to corporate changes. Meeting with new Training Manager set for tomorrow. Training set to begin next week. Contingency-use project funds for now and do training.	Very Likely	Critical	J. Johnstone
23	2	There are rumors of a possible operating system (easy OS) major upgrade; possible impact on functionality, schedule, budget.	Major upgrade is likely to be delivered in 2 months (but may slip). Working on deal with supplier to get beta test to evaluate impacts now.	Very Likely	Critical	R. Everette
101	3	"Eagle Project" manager, Benko, may need to borrow resources to meet unexpected, critical commitment; will impact our schedule and resources.	Johnstone meeting with VP Goldman and Benko to clarify exactly who and for how long and what probability really is.	Probable	Critical	J. Johnstone
3	4	Concern that the waterfall methodology that is in use is not the proper approach; may cause major problems at "big bang" integration and test time.	Incremental build methodology is now consistently being used. Most personnel have been adequately trained—test by next week. Still need to monitor for backsliding.	Probable	Marginal	R. Everette
28	5	No impact analysis of changed requirements is being done; may wind up with conflicting features, goals, and requirements.	Requirements to do impact analysis is now being strictly enforced. Everette is verifying compliance. One section is still lax. Will continue to work this.	Improbable	Marginal	R. Everette
13	6	The 3-letter algorithm that won this contract is only understood by one person; if anything happens to him, it could take years to recover.	E. Whitney is now learning the algorithm. Expected to be up to speed in 2 weeks.	Improbable	Critical	R. Everette/E. Whitney
14	7	Lack of control over introduction of new requirements from senior managers; testing is not prepared for these.	Senior managers have agreed to follow proper channels for submission of new requirements. Continue to watch.	Improbable	Marginal	J. Johnstone
64	8	There are no procedures or process in place to enforce CM; delays, time spent testing wrong system.	New procedures in place. A few false starts but working smoothly now. Recommend tracking only.	Improbable	Negligible	J. Clampett

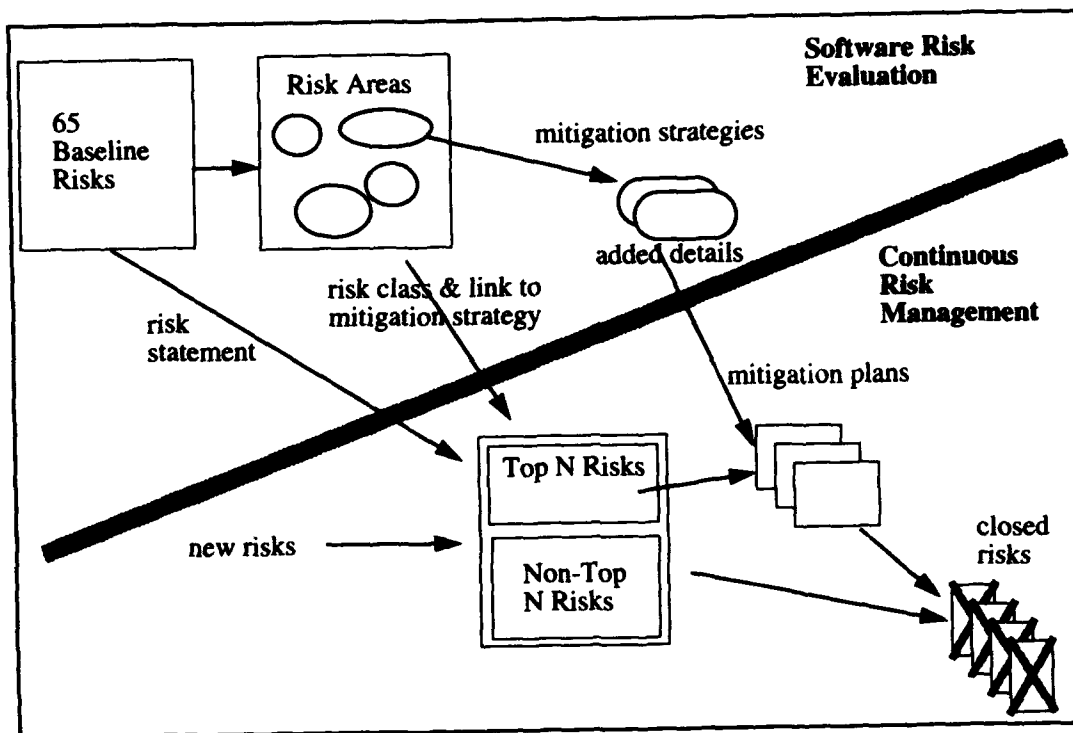
DAS-C Risk Data Base. This list shows the data elements maintained in DAS-C's risk database, which is an extension of their problem data base. Both the Risk Information Sheet and Spreadsheet are reports from the Database.

- Risk ID (unique identifier)
- Date risk was identified
- Risk statement
- Context
- Current priority
- Current probability, impact, timeframe
- Initial probability, impact, and timeframe
- Origin (who identified the risk, or Baseline if identified then)
- Class
- Who has responsibility for the risk
- Approach (watch for significant changes, accept, or mitigate)
- Mitigation strategy or actions
- Contingency plan or actions and trigger
- Periodic status summary, probability and impact at that time, and the date status was reported
- Closing approval (who has to sign off?)
- Closing date
- Closing rationale

3.4 Software Risk Evaluation and Continuous Risk Management

So, what happened to DAS-C's baseline risks and mitigation strategies? As seen in Figure 8, the mitigation strategies were fleshed out into complete plans with added details. All the areas and some individual risks were divided into the Top N risks/risk areas and the non-Top N. New risks were continuously added to the list and new mitigation plans developed and tracked. As mitigation plans succeeded, risks or risk areas were closed.

Figure 8: Life-Cycle of a Risk



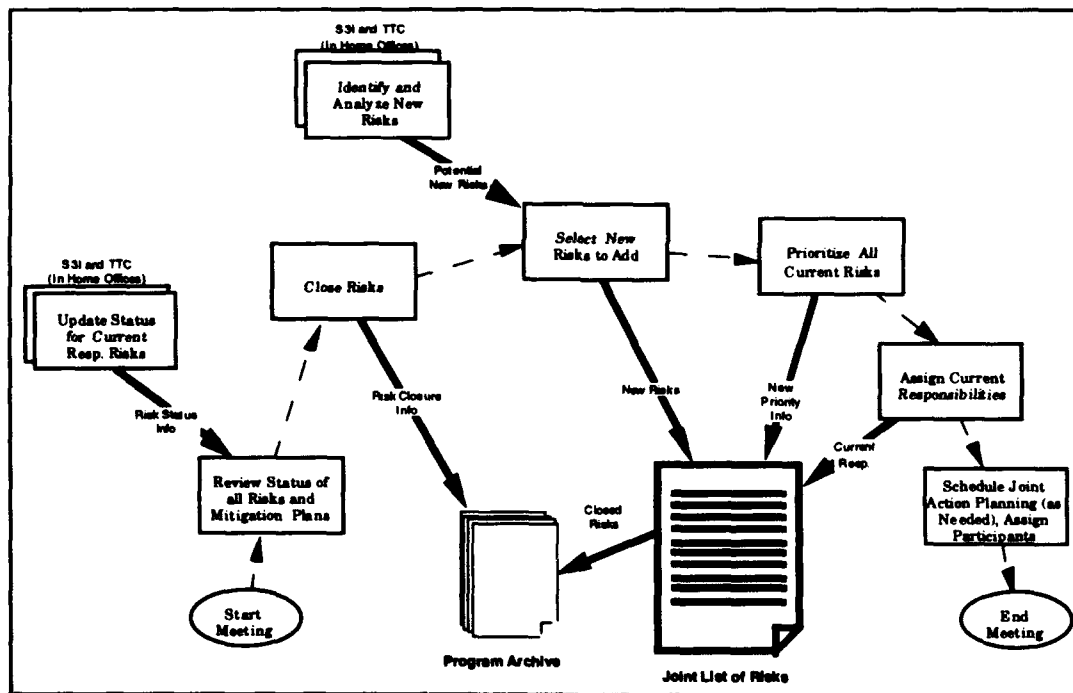
4. Team Risk Management (TRM)

Jerry Johnstone (project manager for DAS-C in Shoestring Software Systems, Inc. - S3I) and Bill Fredderson (Program Manager for DAS-C in the customer organization, Toivolia Telephone Company - TTC), with their two closest technical lieutenants, determined that they would participate in Team Risk Management at the program level.

The centerpiece of this work was the Team Review meeting, which they agreed would be held quarterly, and the first meeting was scheduled. This first meeting (and probably several subsequent ones), they agreed, would use an outside facilitator. If things went well, they planned to eventually facilitate the meeting themselves.

In the session to create a Team Review tailored to the TTC-S3I team, the participants drew up the following structure for the meeting, centering on the Joint List of Risks.

Figure 14: Team Risk Management Process Flow



In addition, they agreed upon the reporting process to be followed for sudden risks (and major problems) that might arise between quarterly meetings.

Table 7: Top Ten Risks from Toivolia Telephone Company's SRE (as captured)

Risk No.	Risk Statement
T28	S3I may not have the financial resources to fulfill their maintenance obligations if the project turns bad; they may default or go bankrupt.
T45	Toivolia Telephone has no people that are able to understand the technical details of S3I's design; we may not learn of problems until it is too late to recover.
T07	We may not be able to "sell" the S3I screen design and operator interface to our telephone operator's union. The result could be a costly strike.
T16	Upper management at S3I appears to be out of touch with the their technical staff. System performance that the S3I management promised (e.g., 3 second response) may not be possible to deliver.
T35	The Toivolia Telephone board of directors is unhappy with progress on the job; if any project milestones are missed, they may insist on cancellation.
T22	S3I's testing on the company's hardware system (midnight-6 AM) may cause crashes and disruptions of operation. Customer complaints to the regulatory commission could divert our attention from our business.
T09	S3I's fixation on "the letter of the specification" might get us the system we asked for, but not what we need. We might end up replacing it in only a few years (heads would roll).
T19	S3I has no one on staff that adequately understands telephone company operations; they may decide on an architecture that will not be capable of expansion in critical areas (and we won't learn in time).
T31	S3I has not made the development process they will be using clear to us; we have no way of determining the project risks for ourselves.
T03	The plan to buy and ship seven computers before we get the operational software may force us to buy the hardware for the system without knowing whether it will ultimately be satisfactory.

© 1997 Carnegie Mellon University

Table 8: Toivolia Telephone's Top 10 Risk Statements after Rephrasing

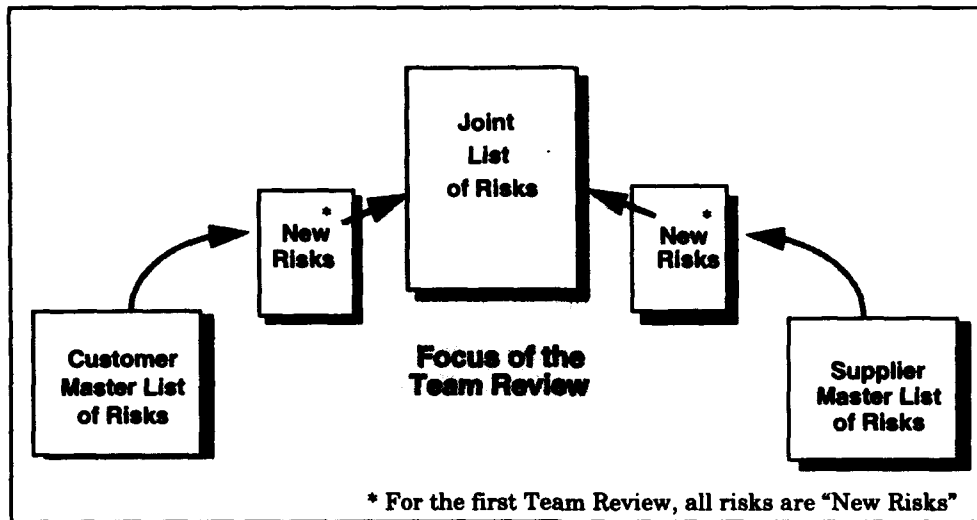
Risk No.	Risk Statement
T28	TTC does not have adequate information to determine how much S3I is at financial risk; S3I may not have the resources to see this project through.
T45	TTC has insufficient staff that can understand the technical details of S3I's design; we may not be able to review the S3I work properly.
T07	We may not be able to "sell" the S3I screen design and operator interface to our telephone operator's union. The result could be a costly strike.
T16	There is concern at TTC about details on which S3I management and technical staff appear to disagree. In particular, system performance that the S3I management promised (e.g., 3 second response) may be in question according to statements from S3I staff.
T35	The TTC board of directors is unhappy with progress on the job; if any project milestones are missed, the project could be canceled.
T22	S3I's testing on the company's hardware system (midnight-6 AM) could cause crashes and disruptions of operation. This could result in customer complaints to the regulatory commission and divert our attention from our business.
T09	S3I's insistence on "the letter of the specification" could result in our not getting the system we need. We might end up replacing it in only a few years, at enormous cost and lost opportunities.
T19	S3I has inadequate background in telephone company operations; they may decide on an architecture that will not be capable of expansion in critical areas (and we won't learn in time).
T31	S3I has not made the development process they will be using clear to us—we have no way of determining the project risks that will result from their process for ourselves. We could get blind-sided by problems that could have been predicted.
T03	The plan to buy and ship seven computers before we get the operational software may force us to buy the hardware for the system without knowing whether it will ultimately be satisfactory.

© 1997 Carnegie Mellon University

Table 9: S3I's "Top 10" Risks for the First Team Review

Risk No.	Risk Statement
S41	Acceptance configuration of the system does not replicate the actual operational system configuration; unpredictable consequences and rework in the field.
S14	There appear to be communication channels between TTC and S3I that are bypassing project management with systems requirements; this may sow confusion, result in excessive time to complete, and/or result in lack of complete system testing.
S19	Conflicts with the customer are not being resolved in a timely manner and approvals are not being received as scheduled; cost and schedule are in jeopardy.
S57	The effect of loading on the network was considered to be "negligible," but recent evidence is otherwise; system may not perform as required.
S05	Lack of technical support from Compilers-R-Us; more time spent developing work-arounds and may cause code restructuring.
S51	The C++ compiler is not living up to its advertised capabilities; poor code generation, slow performance; could result in a slowed development cycle and poor runtime performance of generated code.
S61	It would be extremely difficult (i.e., not scoped or budgeted) to build a realistic test scenario at S3I, even for one computer; we have no way of knowing the system's performance characteristics until we're in the field.
S06	There are rumors that TTC is unhappy with the Screen Display design. Not surfacing these issues could result in dissatisfaction later.
S02	We are dependent on Compilers-R-Us for support of the C++ compiler and it needs a major upgrade. This has been promised, but may not happen; we may not decide to replace the compiler until too late.
S04	There is only limited time for testing (midnight to 6 AM) at customer's site; testing time will be used for bug fixes and not actual performance tests.

Figure 15: Team Risk Flow



In the Team Review it was agreed that the prioritization criteria would be as follows:

1. Long-term benefit to TTC customers
2. Acceptance by TTC operators and the public
3. Maintenance cost

Figure 15: Changes in Risk Priority After Team Review

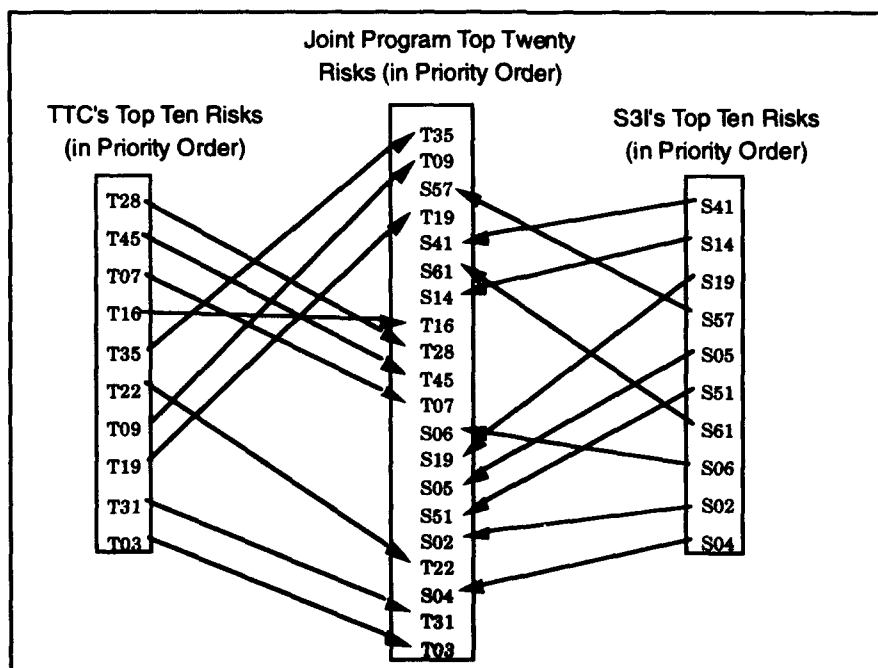


Table 10: Results of Prioritization:

Priority	Risk No.	Risk Statement	TTC	S3I	Joint
1	T35	The Toivolia Telephone board of directors is unhappy with progress on the job; if any project milestones are missed, the project could be canceled.			
2	T09	S3I's insistence on "the letter of the specification" could result in our not getting the system we need. We might end up replacing it in only a few years, at enormous cost and lost opportunities.			
3	S57	The effect of loading on the network was considered to be "negligible," but recent evidence is otherwise; system may not perform as required.			
4	T19	S3I has inadequate background in telephone company operations; they may decide on an architecture that will not be capable of expansion in critical areas (and we won't learn in time).			
5	S41	Acceptance configuration of the system does not replicate the actual operational system configuration; unpredictable consequences and rework in the field.			
6	S61	It would be extremely difficult (i.e., not scoped or budgeted) to build a realistic test scenario at S3I, even for one computer; we have no way of knowing the system's performance characteristics until we're in the field.			
7	S14	There appear to be communication channels between TTC and S3I that are bypassing project management with systems requirements; this may sow confusion, result in excessive time to complete, and/or result in lack of complete system testing.			
8	T16	There is concern at TTC about details on which S3I management and technical staff appear to disagree. In particular, system performance that the S3I management promised (e.g., 3 second response) may be in question according to statements from S3I staff.			
9	T28	TTC does not have adequate information to determine how much S3I is at financial risk; they may not have the resources to see this project through.			

Priority	Risk No.	Risk Statement	TTC	S3I	Joint
10	T45	TTC has insufficient staff that can understand the technical details of S3I's design; we may not be able to review the S3I work properly.			
11	T07	We may not be able to "sell" the S3I screen design and operator interface to our telephone operator's union. The result could be a costly strike.			
12	S06	There are rumors that TTC is unhappy with the Screen Display design. Not surfacing these issues could result in dissatisfaction later.			
13	S19	Conflicts with the customer are not being resolved in a timely manner and approvals are not being received as scheduled; cost and schedule are in jeopardy.			
14	S05	Lack of technical support from Compilers-R-Us; more time spent developing work-arounds and may cause code restructuring.			
15	S51	The C++ compiler is not living up to its advertised capabilities; poor code generation, slow performance; could result in a slowed development cycle and poor runtime performance of generated code.			
16	S02	We are dependent on Compilers-R-Us for support of the C++ compiler and it needs a major upgrade. This has been promised, but may not happen; we may not decide to replace the compiler until too late.			
17	T22	S3I's testing on the company's hardware system (midnight-6 AM) could cause crashes and disruptions of operation. This could result in customer complaints to the regulatory commission and divert our attention from our business.			
18	S04	There is only limited time for testing (midnight to 6 AM) at customer's site; testing time will be used for bug fixes and not actual performance tests.			
19	T31	S3I has not made the development process they will be using clear to us—we have no way of determining the project risks that will result from their process for ourselves. We could get blind-sided by problems that could have been predicted.			

© 1997 Carnegie Mellon University

Pri- ority	Risk No.	Risk Statement	TTC	S3I	Joint
20	T03	The plan to buy and ship seven computers before we get the operational software may force us to buy the hardware for the system without knowing whether it will ultimately be satisfactory.			

Workbook Bibliography

- [Basili84] Basili, Victor R., and David M. Weiss. "A Methodology for Collecting Valid Software Engineering Data." IEEE Transactions on Software Engineering SE-10, 6 (November 1984): 728-738
- [Boehm81] Boehm, Barry. *Software Engineering Economics*. Englewood Cliffs, NJ: Prentice-Hall, Inc., 1981.
- [Boehm89] Boehm, B. *IEEE Tutorial on Software Risk Management*. New York: IEEE Computer Society Press, 1989.
- [Brassard94] Brassard, Michael & Ritter, Diane. *The Memory Jogger™ II: A Pocket Guide of Tools for Continuous Improvement and Effective Planning*. Methuen, Ma.: GOAL/QPC, 1994.
- [Clark95] Clark, Bill. "Technical Performance Measurement in the Risk Management of Systems," Presented at the Fourth SEI Conference on Software Risk, Monterey, CA, November 6-8, 1995.
- [DSMC86] Defense Systems Management College. *Risk Management: Concepts and Guidance*. Fort Belvoir, VA: DSMC, 1986.
- [Dorofee96] Dorofee, Audrey J.; Walker, Julie A.; Alberts, Christopher J.; Higuera, Ronald P. Murphy, Richard L.; and Williams, Ray C. *Continuous Risk Management Guidebook*. Pittsburgh, PA: Software Engineering Institute, 1996.

For more information on Risk Management, contact:

Customer Relations
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890

- Phone: (412) 268-5800
- internet: customer-relations@sei.cmu.edu
- WWW: <http://www.sei.cmu.edu/>

© 1997 Carnegie Mellon University

Appendix

The following is a complete listing of the baseline DAS-C risks.

COMPLETE DAS-C RISK LISTING (by Risk Area)

Risk No.	Risk Description	Final SE	Team Top Risk?	Ind. Top 3	Risk Area
42	Requirements seem to be changing; can't be sure that the test cases cover all requirements.	4	Yes	3	Configuration Management
50	There is no formal change control process that coordinates all affected groups; test plans are not keeping up with changes.	4	Yes		Configuration Management
11	Software Quality Assurance and Configuration Management seem not to have formal, controlled plans at this time; could increase our costs and development time, we may lose or overwrite modules.	4			Configuration Management
20	Concerned about configuration management between development and field test sites; lack of CM may cause version mismatches, lost time, and rework.	4			Configuration Management
38	Configuration management is not really happening; could cause rework, delay, incorrect builds, misdirected debugging efforts (a real timebomb!).	4			Configuration Management
49	Maintenance people are not involved early in the design; the product may be unsatisfactory for long-term maintenance, and maintenance costs may turn out much higher than anticipated.	4		2	Configuration Management
64	There are no procedures or processes in place to enforce CM; delays, time spent testing the wrong system.	4		1	Configuration Management
19	Conflicts with the customer are not being resolved in a timely manner; a lot of unplanned time spent educating the customer, drag on the schedule.	5	Yes	2	Customer Interface
32	Customer has difficulty visualizing what S3I will be delivering and there is nothing in the program to satisfy this need: concerned that the customer may force rework of the system at delivery time.	4	Yes		Customer Interface
1	Customer performance in approving submittals has been unduly slow and they require more "hand-holding" than expected; performance requirement probably will not be approved on the scheduled date.	4		2	Customer Interface
6	There are rumors that the telephone company is unhappy with the Screen Display design and see it as representative of S3I work. They may cancel the project.	4			Customer Interface
30	Concern that high level of customer interaction will not cease at approval of the Performance Specification; more drag on the schedule.	4			Customer Interface
46	Toivolia accounting department wanted to do this job, and they are still trying to prove they could do it better; delay in approval cycles, have to constantly prove S3I's solution is "best."	4			Customer Interface
47	Toivolia approved the Functional Spec a month late, no reason to assume they won't be late approving Performance Spec; severe schedule impact.	4		1	Customer Interface
55	Don't have anyone on the project that has experience in internal telephone company operations; system may not hit the target for the customer and require rework.	4			Customer Interface
35	Customer interchange techniques don't appear to be working well; delays in approvals and possible difficulty in achieving final system acceptance.	3			Customer Interface
54	Tape format of subscriber information is still unknown; might have to procure some special translation routines and it could affect schedule.	3			Customer Interface
28	No impact analysis of changed requirements is being done; may wind up with conflicting features, goals and requirements.	5	Yes	3	Development Process
39	Developers are working from their own interpretation of requirements documents, not using the developed test scenarios; the system may not be properly tested and may fail final acceptance—alternatively, lots of rework.	5	Yes	2	Development Process

COMPLETE DAS-C RISK LISTING (by Risk Area)

Risk No.	Risk Statement	Risk Rating				Risk Area
		Severity	Timing	Frequency	Impact	
7	There is no interface control document required for design; may result lack of coordination of interfaces.	4	Yes	3		Development Process
3	Concern that waterfall methodology that is in use is not the proper approach; may cause major problems at "big bang" integration and test time.	4		1		Development Process
25	Development activity plans do not seem to be fully documented; this may result in poor coordination in the project.	4		1		Development Process
34	There is concern that the software development group is not reviewing integration and test plans carefully and not giving feedback; at integration and test time there may be a major confrontation between the groups.	4				Development Process
60	There is no process for defining, changing, and controlling software interfaces; may cause rework and hard-to-detect system-level bugs	4				Development Process
53	There are two competing developmental models in use—waterfall and incremental build; this may be causing confusion among the system developers.	3				Development Process
58	The past history of this company is that code and design is poorly documented; there may difficulty in maintaining what is supposed to be a "flagship" product.	3				Development Process
27	Our programmers are FORTRAN programmers; it's going to be a tough learning curve to move to C++; may cause delays, rework, hard-to-find bugs.	5	Yes			Language
31	There have been instances where programmers have been relaxing argument typing to facilitate compilation (C++ allows this); this may cause unpredictable system behavior and extensive system debugging time.	4	Yes			Language
8	There is a lack of training in C++; system developers don't know which features are "safe" to use and which should be left alone.	4				Language
29	In C++ strong typing can be turned off; that can allow mis-matches between types used in interfaces and the result can be hard-to-detect abnormal behavior.	4				Language
48	Training existing programmers in C++—even if paid for by corporate—may cause a one to two-month program delay.	4		3		Language
9	There is currently a large local demand for experienced C++ programmers; may lose the ones we have, may have to pay too much to hire from outside.	3				Language
10	Project schedule and cost estimation based on a differing application domain; may have underestimated the job and lose profit.	3				Management Methods
13	There are rumors that low performers in the project may get fired to serve as a lesson to the rest, so many people are job-hunting; we may not have everyone we need to meet our deadlines.	3				Management Methods
65	The three-letter algorithm that won this contract is only understood by one person; if anything happens to him, it could take years to recover.	3				Management Methods
14	The VP is undercutting the project manager and introducing new requirements; these may remain hidden, and no test cases will be developed for them.	5	Yes			Senior Management
56	Requirements are changing because of outside influences (vice president); this will affect quality of the code, integration, morale, and schedule.	5	Yes	1		Senior Management
37	Upper management is imposing new quality assurance measures on this project that have not been tried before and for which budget was not provided; may delay program and drive up costs inordinately.	4				Senior Management

COMPLETE DAS-C RISK LISTING (by Risk Area)

Risk No.	Risk Statement	Risk Rating			Risk Area
		Final RR	Team Top Risk?	Ind. Top 5	
16	There is a perception that upper management arbitrarily revised the project cost estimate downward to win the contract; people may give up trying to meet deadlines and performance boogys.	3			Senior Management
18	VP introducing new system requirements without budget or schedule relief; this is muddying the project's lines of authority.	3		1	Senior Management
22	The company doesn't have a program for maintaining and upgrading personnel skills; could hurt long-term competitiveness of the company.	3			Senior Management
40	Upper management has not approved C++ training for project staff—the needed training may have to come from project budget; profit will be in jeopardy.	3			Senior Management
63	Long-term maintenance contract may not be a "sure thing," but business decisions with far-reaching impacts are being made on that assumption; could lose money on this project and never make it up.	3		2	Senior Management
45	The C++ compiler has bugs; added time to develop work-arounds, aggravates lack of C++ experience of developers, may have to replace compiler, for which there is no budget.	5	Yes	1	Suppliers
5	Lack of technical support from Compilers-R-Us; more time spent developing work-arounds and may cause code restructuring.	4	Yes		Suppliers
51	The C++ compiler is not living up to its advertised capabilities; poor code generation, slow performance; could result in a slowed development cycle and poor runtime performance of generated code.	4	Yes	3	Suppliers
2	We are dependent on Compilers-R-Us for support of the C++ compiler and it needs a major upgrade. This has been promised, but may not happen; we may not decide to replace the compiler until too late.	4			Suppliers
52	The C++ compiler may not perform adequately; might have to be replaced, for which there is no budget, and schedule impact due to new learning curve.	4			Suppliers
15	The electronic switch unit is on a tight delivery schedule and may be delayed. There is no slack in the project schedule for this, and it will have a day-for-day effect on S31's deliveries.	3			Suppliers
23	There are rumors of possible operating system (EasyOS) major upgrade; possible impact on functionality, schedule, budget.	3			Supplier
26	System downtime performance depends in part on the electronic switch. The switch may not perform as well as the vendor claims, and we may have bad long-term maintenance contract losses.	3			Suppliers
41	Acceptance configuration of the system does not replicate the actual operational system configuration; unpredictable consequences and rework in the field.	6	Yes	2	System Performance
33	We've never tried to make 10 computers work together like this; we don't know what we don't know; could delay final system acceptance.	5	Yes	3	System Performance
43	Have to support 50 terminals on each computer with 3 second response time, but have only tested with 25; might have to buy more computers, network overhead, electronic switch might be affected.	5	Yes	1	System Performance
44	No performance analysis has been done for the system; we don't know what we don't know.	5	Yes		System Performance
57	The effect of loading on the network was considered to be "negligible"—no tests were done. One computer may handle 50 operators OK, but 10 computers may not be able to handle 500 operators.	5	Yes	1	System Performance
61	It would be extremely difficult (i.e., not scoped or budgeted) to build a realistic test scenario here, even for one computer; we have no way of knowing the system's performance characteristics until we're in the field.	4	Yes		System Performance

COMPLETE DAS-C RISK LISTING (by Risk Area)

Risk No.	Risk Statement	Final	Team Top	Ind.	Risk Area
		RE	Risk?	Top 3	
21	Prototypes have been made, and they indicate there are some problems (e.g., disk driver), however system developers may be unaware of these; developers may waste time on code that won't work right eventually.	4		2	System Performance
24	We've never tried to interconnect so many computers on one project; there could be unforeseen technical challenges.	4		2	System Performance
36	The three-letter algorithm may result in so many pages of possibles (e.g., for "SMI") that operators may get frustrated and refuse to use the system.	3			System Performance
59	Other software companies are rumored to have had performance problems with Easy Comp computers. The computer itself might fail to meet performance needs.	3			System Performance
4	There is only limited time for testing (midnight to 6 AM) at customer's site; testing time will be used for bug fixes and not actual performance tests.	4		2	Testing Issues
12	Testing in the field will be on back shifts—non-exclusive use of undetermined number of machines; will make it difficult to replicate some types of problems.	4			Testing Issues
62	Conditions during field startup (testing at night) may mean that our best integrators & testers will not be willing to go; troubleshooting may require excessive time.	4			Testing Issues
17	Concern that the already-scheduled testing time may not really be made available by Toivolia (three internal departments at Toivolia want the time); our field testers may be held hostage until final acceptance.	3			Testing Issues

The Personal Software Process (PSP)SM Tutorial

Watts S. Humphrey
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Sponsored by the U.S. Department of Defense

SMPSP and Personal Software Process are service marks of Carnegie Mellon University.

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 1

Tutorial Objectives

To describe the personal software process (PSP)

To show where and how the PSP can be used to improve individual software engineering performance

To show how the PSP can enable software organizations to improve their capability

To describe the status and plans for the PSP

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 2

Tutorial Agenda

PSP Description

- PSP overview
- PSP processes
- PSP planning and quality management

Break

PSP Introduction

- course and industry data
- PSP training
- PSP management

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 3

Tutorial Ground Rules

- 1. I will stick to the schedule.**
- 2. If you cannot hear me or my meaning is unclear, please let me know immediately.**
- 3. Please hold your other questions until the end of each session.**
- 4. I will allow time for questions at the end of the first session if time permits and at the end of the tutorial.**

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 4

The Software Problem - 1

The business importance of software is increasing.

- **Software is now the critical element in many products.**
- **Software cycle time often exceeds hardware cycle time.**

Poor software quality is expensive and results in

- **reduced customer satisfaction**
- **delayed shipments**
- **expensive service and enhancement**

Copyright © 1995 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 5

The Software Problem - 2

It is now generally recognized that an effective way to improve software quality is by improving the development and maintenance processes.

Organizational progress with process improvement is limited because

- **There is limited process improvement experience.**
- **Process improvement takes time.**
- **Process improvement efforts are hard to sell.**

Copyright © 1995 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 6

The Software Problem - 3

The PSP addresses these problems by

- providing convincing evidence of the benefits of process improvement
- exposing the engineers to the benefits of using effective processes in their work
- teaching the engineers effective process improvement methods

The PSP Paradigm

The PSP is based on process improvement principles.

- Practitioners establish personal process goals.
- They define the methods they will use.
- They measure their work.
- They analyze the results.
- Based on these analyses, they adjust their methods to better meet their personal goals.

The PSP Strategy

Start with the engineer's current process.

Gradually introduce new methods.

Practice these methods on module-sized programs.

The engineers then see for themselves how these methods help them.

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 9

PSP Overview - 1

The PSP is a process for individuals to use.

It applies to most structured personal tasks.

- **writing small programs or documents**
- **defining requirements or processes**
- **conducting reviews or tests, etc.**

It is introduced with module-sized exercises.

It is extendible to team development of large-scale software systems.

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 10

PSP Overview - 2

Individuals learn the PSP in 7 process steps.

They write 10 module-sized programs using these PSP steps.

- **They gather and analyze data on their work.**
- **Based on these analyses, they improve their working methods.**

The PSP exercises provide the rapid feedback needed for effective learning.

The PSP Course - 1

The PSP is best introduced with a course format.

Prerequisites

- **fluency with one programming language**
- **basic understanding of programming design**
- **mathematics through integral calculus**

Additional background is helpful.

- **statistics**
- **project management**
- **formal methods**

The PSP Course - 2

Facilities required

- personal computing capability
- spreadsheet and database support
- development environment

One semester workload

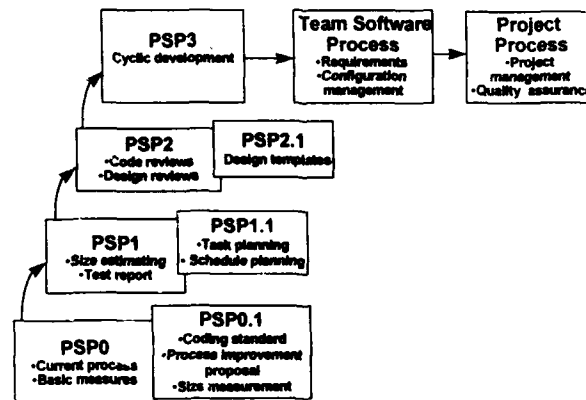
- 15+ 90-minute lectures
- laboratory time of 5+ hours per week
- study time of 5+ hours per week

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 13

The PSP Is an Evolving Process



Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 14

The PSP0 Process

With PSP0, engineers use their current design and development methods.

They gather the following data on their work:

- the time spent by phase
- the defects found in compile and test

They analyze and report these data.

The PSP0 Process Lessons

With PSP0, engineers learn to use a basic personal process.

- They gather data on their personal work.
- They learn how and why to measure the sizes of the products they produce.
- They gather baseline data on their personal processes.

The PSP1 Process

The PSP0 is augmented to include

- coding standards
- size estimating
- resource estimating
- schedule estimating
- earned value tracking
- process improvement proposal (PIP)
- test report

The PSP1 Lessons

With PSP1, engineers estimate the sizes and development times of the products they produce.

- They use their historical data to improve their estimate.
- They project the likely statistical ranges of their estimates and learn how to reduce these ranges.

The PSP2 and PSP3 Additions

PSP1 is augmented to include

- personal design and code reviews
- yield and cost of quality measures
- design completion criteria
- design templates

With PSP2, engineers see how to use the PSP for larger scale work such as

- cyclic development
- issue tracking log

PSP2 and PSP3 Lessons

With PSP2, engineers use their historical data to improve the quality of the program modules they produce.

- They measure the efficiency of their defect removal methods.
- They use various process quality measures, including yield, COQ (cost of quality), and A/FR (appraisal/failure ratio).

With PSP3, engineers learn how to adjust their personal processes for different types of work.

The Basic PSP Elements

A process script

A project plan summary form

A time recording log

A defect reporting log

A defect type standard

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 21

Time Recording Log

Enter the following information in the time recording log:

- the time when you start a project phase
- the time when you stop a project phase
- the interruption time
- the elapsed time less interruption time
- comments

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 22

Defect Recording Log

In the defect recording log, enter the following information on all defects found in reviews, compiling, and test

- the number of the defect
- the defect type
- the phase it was injected
- the phase it was removed
- the fix time
- a brief description of the defect

If the defect was injected while fixing a defect, enter that defect's number.

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 23

PSP Planning

In the PSP, engineers use their personal data to make plans.

Planning consists of

- size estimating
- resource estimating
- schedule estimating

In the PSP, lines of code (LOC) are used as the size measure

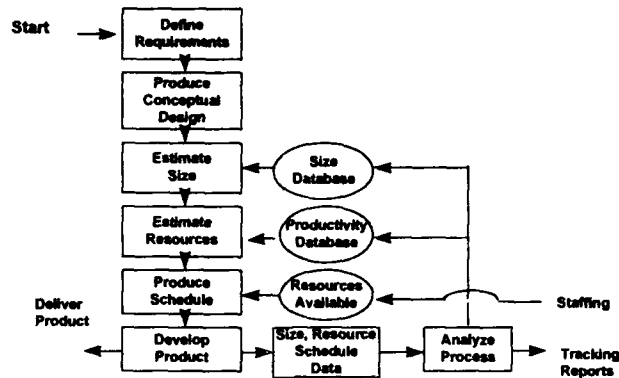
- Other measures could be used.

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 24

The Project Planning Framework



Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 26

Why Estimate Size?

To make better plans by

- better sizing the job
- dividing the job into separable elements

To assist in tracking progress

- can judge when job scope changes
- can better measure the work

Value for the PSP

- learn estimating methods
- build estimating skills

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 26

Size Estimating Principles - 1

Estimating is an uncertain process.

- No one knows how big the product will be.
- The earlier the estimate, the less is known.
- Estimates can be biased by business and other pressures.

Estimating is an intuitive learning process.

- Ability improves with experience.
- Some people will be better at estimating than others.

Size Estimating Principles - 2

The estimating objectives are to

- make consistent estimates
- understand estimate variability
- balance under and over estimates

The principal advantages of using a defined estimating method are

- You have a known practice that you can improve.
- It provides a framework for gathering data.
- By using a consistent method and historical data, your estimates will get more consistent.

Size Estimating Proxies

A proxy is a substitute.

A suitable size-estimating proxy will help visualize ill-defined products early in development.

Potential proxies are

- **functions, procedures, and methods**
- **function points**
- **objects**
- **report pages, files, and screens**

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 29

Objects as Proxies - 1

Objects make good proxies because

- **Numbers of objects correlate reasonably well with development hours.**
- **Object LOC correlate very closely with development hours.**
- **Historical object LOC data can be obtained and used.**
- **Using these historical data, object LOC can be estimated readily.**

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 30

Objects as Proxies - 2

When objects are selected as application entities, they can be visualized early in development.

Functions and procedures can often be estimated in the same way.

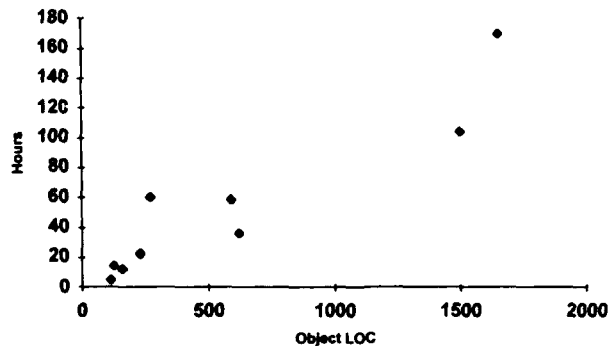
Objects, functions, procedures, and their LOC can be counted automatically.

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 31

Object LOC Correlation with Development Hours



Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 32

The PROBE Estimating Method

Start

Conceptual
Design

Identify Objects

•Number of methods
•Object Type
•Relative Size
•Reuse Categories

Calculate Added and
Modified LOC

Estimate
Program Size

Calculate
Prediction Interval

Estimate

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 33

To Make Size Estimates, You Need Several Items

Data on historical objects, divided into types

**Estimating factors for the relative sizes of each
object type**

**Regression parameters for computing new and
changed LOC from**

- estimated object LOC
- LOC added to the base
- modified LOC

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 34

C++ Object Size Ranges

Type	LOC per method				
	VS	S	M	L	VL
Calculation	2.34	5.13	11.25	24.66	54.04
Data	2.60	4.79	8.84	16.31	30.09
I/O	9.01	12.06	16.15	21.62	28.93
Logic	7.55	10.98	15.98	23.25	33.83
Set-up	3.88	5.04	6.56	8.53	11.09
Text	3.75	8.00	17.07	36.41	77.66

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 38

Estimate Program Size

Total program size consists of

- newly developed code (adjusted with the regression parameters)
- reused code from the library
- base code from prior versions, less deletions

Newly developed code consists of

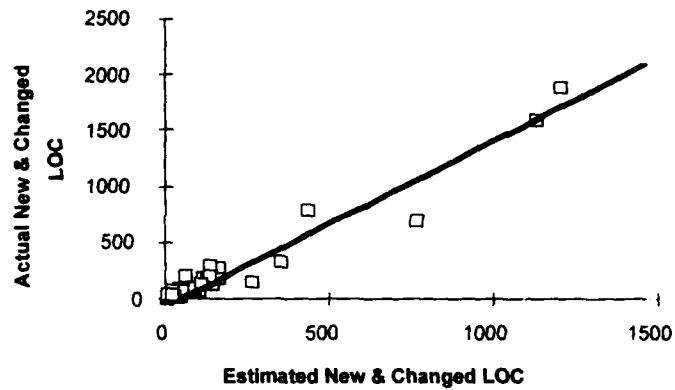
- additions to the base
- newly developed objects
- modified base LOC

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 38

Regression Calculation - Size



Copyright © 1988 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 37

The Prediction Interval

The prediction interval provides a likely range around the estimate.

- A 90% prediction interval gives the range within which 90% of the estimates will likely fall.
- It is not a forecast, only an expectation.
- It only applies if the estimate behaves like the historical data.

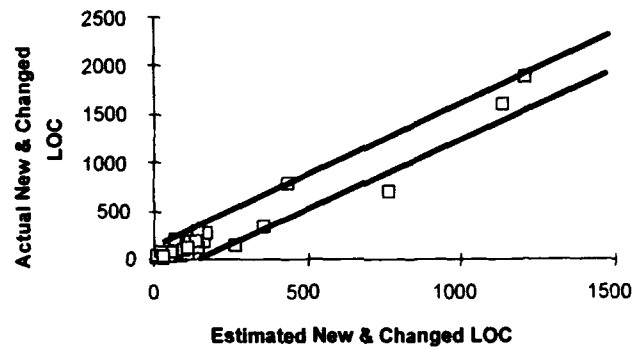
The prediction interval is calculated from the same data used to calculate the regression parameters.

Copyright © 1988 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 38

Prediction Interval - 90% - Size



Copyright © 1998 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 39

The Resource Planning Process

Start with a size estimate.

Identify available data.

Use regression when you have 3+ sets of data that correlate.

Use data for estimated LOC to actual hours where available.

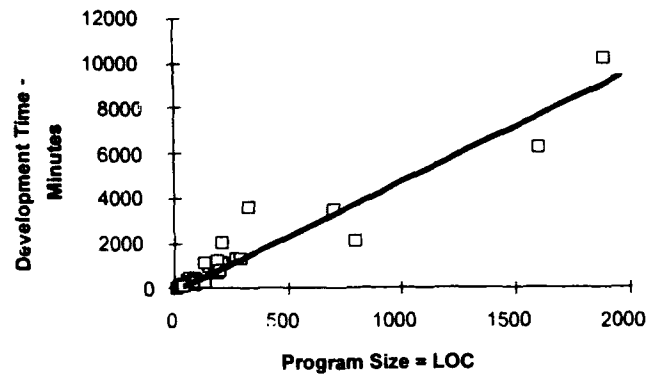
Calculate the prediction interval.

Copyright © 1998 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 40

Regression Calculation

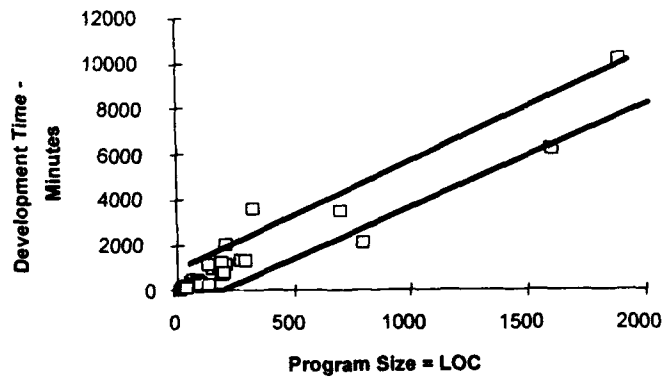


Copyright © 1996 Carnegie Mellon University

Wells S. Humphrey

PSP Tutorial - 1 41

Prediction Interval - 90%



Copyright © 1996 Carnegie Mellon University

Wells S. Humphrey

PSP Tutorial - 1 42

Schedule Estimating

To make a schedule, you need 3 things

1. the estimated direct project hours
2. a calendar of available direct hours
3. the order in which the tasks will be done

You then need to

- estimate the hours needed for each task
- spread these hours over the calendar of available hours

The PSP Quality Strategy - 1

In the PSP, defects are the basic quality measure.

Note that defects are not important to the user as long as they do not

- affect operations
- cause inconvenience
- cost time or money
- cause loss of confidence in the program's results

The PSP Quality Strategy - 2

Low defect content is an essential prerequisite to a quality software process.

- **Experienced software engineers typically inject around 100 defects per KLOC.**
- **Low defect products can be assured best at the PSP level.**

This is where the defects are injected, and this is where the engineers should

- **remove them**
- **determine their causes**
- **learn to prevent them**

Copyright © 1988 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 46

The PSP Quality Strategy - 3

If you want a quality product out of test, you must put a quality product into test.

- **Testing removes only a fraction of the defects.**
- **The more defects present in the code entering test, the more defects there will be in the code exiting test.**

To manage defects, they must be addressed where they are injected - by each software engineer.

This requires a comprehensive focus on software quality.

Copyright © 1988 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 46

The PSP Quality Strategy - 4

Data show that it is much more efficient to find defects in reviews than in testing.

- **In unit test, typically only about 2 to 4 defects are found per hour.**
- **Code reviews typically find about 10 defects per hour.**
- **Experienced reviewers can find 70% or more of the defects in a product.**
- **Unit test rarely exceeds a 50% yield.**

PSP data show that reviews find 2 to 5 times as many defects per hour as unit test.

PSP Reviews

In a personal design or code review

- **Professionals privately review their products.**
- **Their objective is to find all defects before the first compile and test.**
- **Reviews are most effective when structured and measured.**

Reviews can be used for requirements, designs, and code.

Review Yield - 1

Yield

- a measure of process quality
- the percent of defects in the product at review time that were found by the review
- a measure of the effectiveness of a process step
 - design and code reviews
 - the overall process - prior to test
 - the development process - including test

Yield for a phase or the entire process =
 $100 * (\text{defects found}) / (\text{defects found} + \text{not found})$

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 49

Review Yield - 2

Yield cannot be calculated until all defects have been found through test and product use.

Yield can be useful early in the process if all or most defects are counted.

- design and code review defects
- compile defects
- unit test defects

Using process data, control parameters can help to ensure high-yield reviews.

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 50

The Cost of Quality (COQ) - 1

Failure costs

- repair, rework, and scrap
- PSP failure costs are compile and test time.

Appraisal costs

- costs of inspecting for defects
- PSP appraisal costs are design review and code review time.

Prevention costs are finding and resolving defect causes.

The Cost of Quality (COQ) - 2

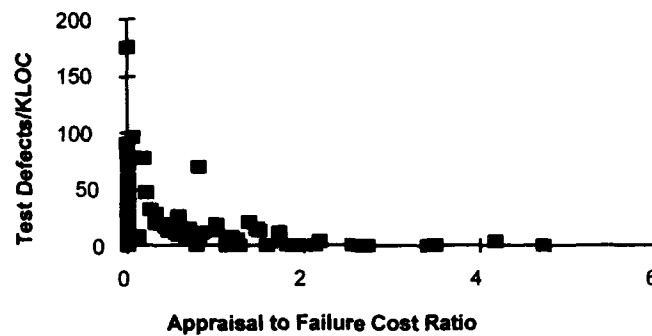
A useful PSP measure is the ratio of appraisal to failure costs (A/FR).

- $A/FR = (\text{appraisal COQ})/(\text{failure COQ})$
- A/FR measures process quality.

A/FR experience

- If measured, the A/FR of most software organizations would be near zero.
- In the PSP, A/FR should exceed 2.0.
- High A/FR is associated with low numbers of test defects and high product quality.

Test Defects vs. A/FR - Class



Copyright © 1998 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 53

Messages to Remember

1. A defined and measured process provides a repeatable basis for improvement.
2. The PROBE method provides a statistically sound framework for planning.
3. The PSP quality strategy will help engineers produce high quality products.

Copyright © 1998 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 54

PSP Tutorial - Conclusion

Watts S. Humphrey
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Sponsored by the U.S. Department of Defense

Copyright © 1990 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 88

Introducing the PSP

PSP results

- Course data
- Industrial data

The PSP and process improvement

Introducing the PSP

PSP courses

Copyright © 1990 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 88

PSP Experience

The PSP has been taught in a number of universities in the U.S., Canada, South America, Europe, and Australia.

The following data are from two sources:

- 14 students at Carnegie Mellon University**
- 104 engineers in university and industry PSP courses**

These results are typical of the results obtained from most PSP courses.

Size and Time

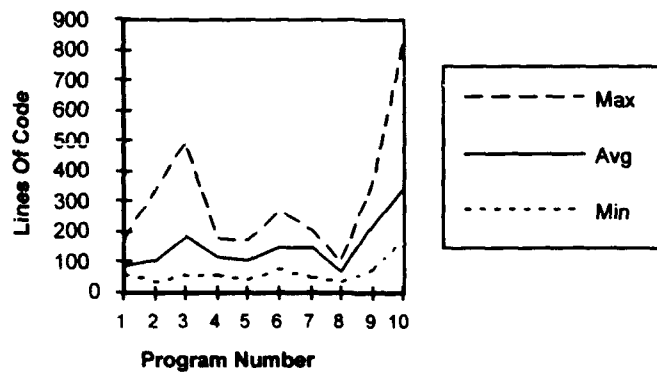
Starting with PSP0 at program 1, the engineers gather size and time data on 10 programs.

They learn how to gather and track size and time data.

They also learn to project the size and development time for new programs.

By program 10, the engineers are using the full PSP3 process.

Actual Size Range

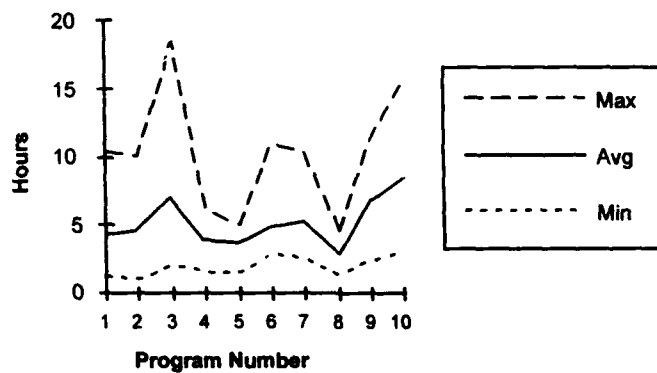


Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 88

Actual Time Range

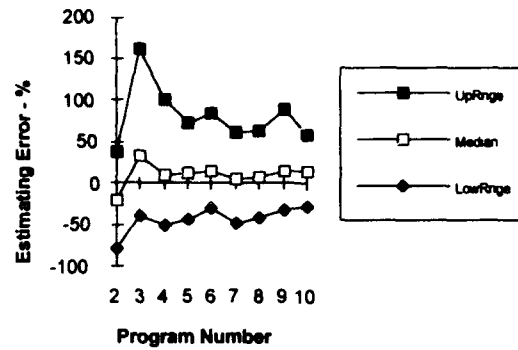


Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 89

Size Estimate Error - 104 Engineers

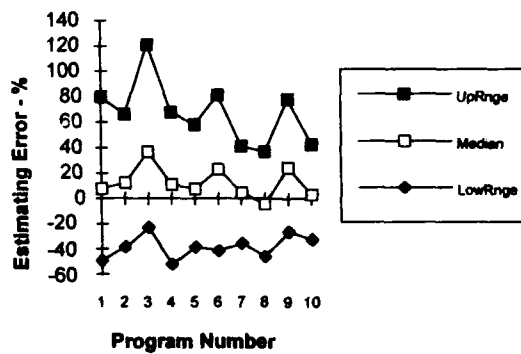


Copyright © 1998 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 61

Time Estimating Error - 104 Engineers



Copyright © 1998 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 62

Defects

With PSP, engineers gather data on every defect they find

- in compiling
- during test
- and during code reviews

They see how many defects they inject and what those defects cost.

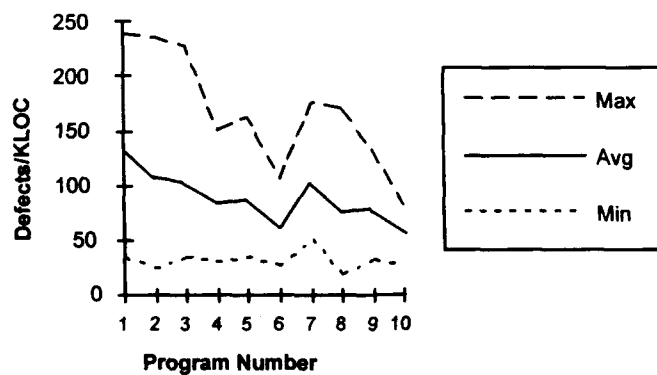
These data convince them to take more care in their work. This saves compiling and testing time.

Copyright © 1988 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 63

Total Defects - Range

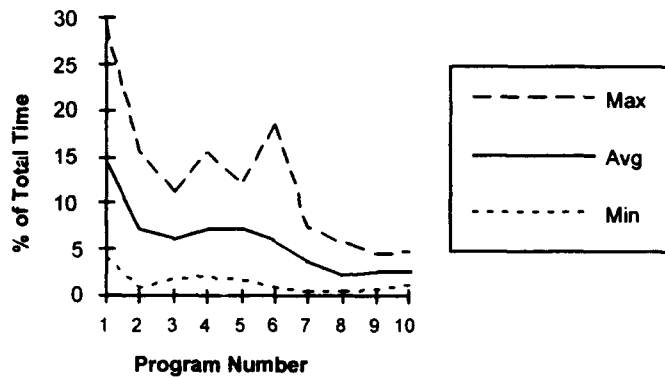


Copyright © 1988 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 64

Compile Time Range

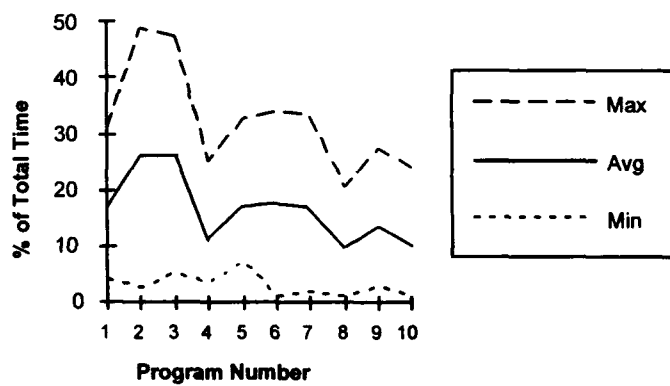


Copyright © 1988 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 68

Test Time Range



Copyright © 1988 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 69

Defect Analysis

With data on their defects, engineers can determine the effectiveness of various defect removal methods.

They find that they can remove

- 2 to 4 defects per hour during initial testing
- 3 to 5 defects per hour in design review
- 5 to 10 defects per hour in code review

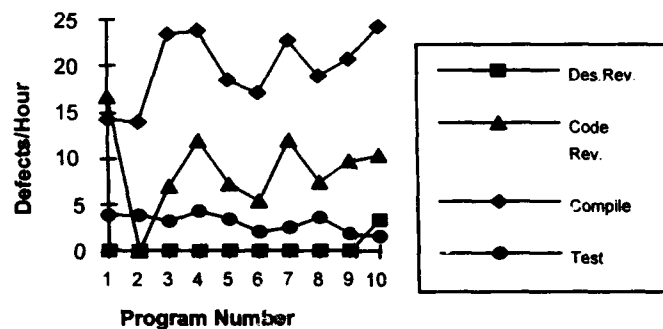
They see that reviews save them compile and test time.

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 67

Defect Removal Rates - Class

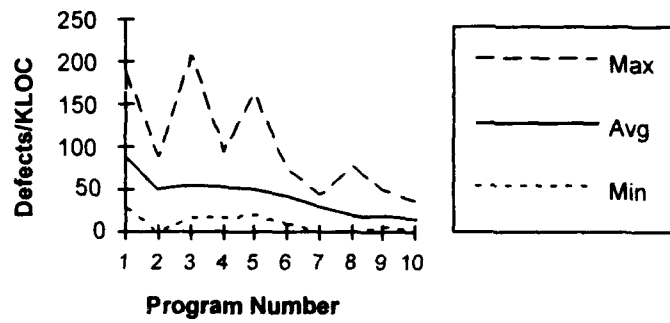


Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 68

Defects Found in Compile - Range

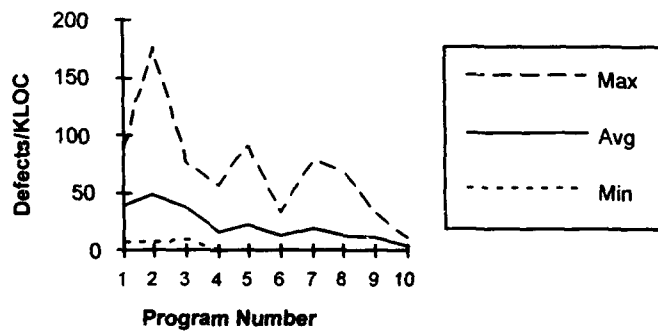


Copyright © 1988 Carnegie Mellon University

Wells S. Humphrey

PSP Tutorial - 1 68

Defects Found in Test - Range



Copyright © 1988 Carnegie Mellon University

Wells S. Humphrey

PSP Tutorial - 1 70

Cost of Quality

Cost of quality measures process efficiency.

- Fix time is percent time in compile and test.
- Appraisal time is percent time in reviews.

A/FR is the ratio of appraisal to fix time.

- It measures the effort the engineer devotes to early defect removal.
- It also indicates product quality.

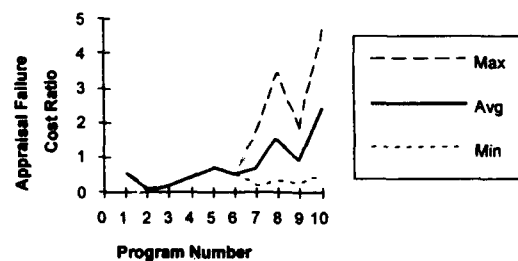
Most importantly, A/FR indicates the engineer's commitment to producing a quality product.

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 71

**Appraisal to Failure Cost
Ratio (A/FR) - Class**

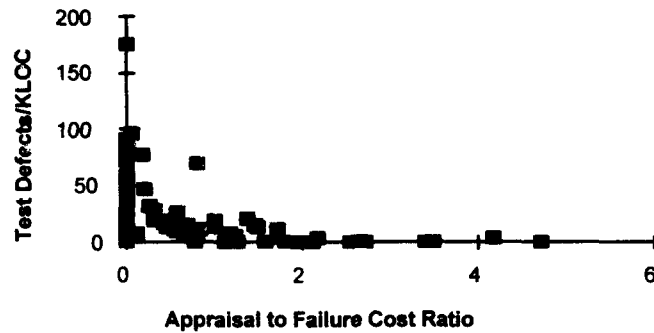


Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 72

Test Defects vs. A/FR - Class



Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 73

Productivity

Lines of code (LOC) per hour for small program modules often increases with PSP training.

- Average LOC/hour improvement is 30%.
- Median LOC/hour improvement is 12%.

The principal PSP benefit is increased quality and productivity for larger projects.

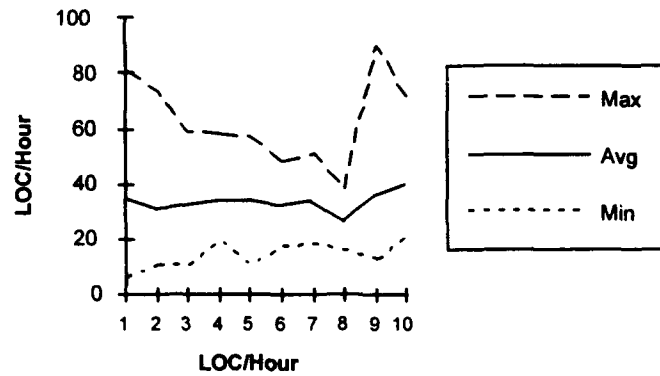
With the PSP, LOC/hour rates actually increase for larger programs.

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 74

Productivity Range

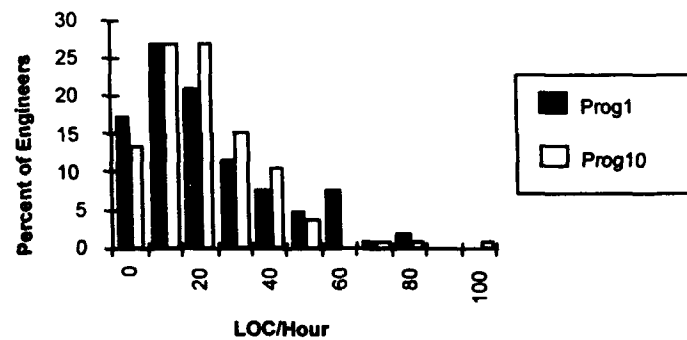


Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 78

Productivity, 104 Engineers

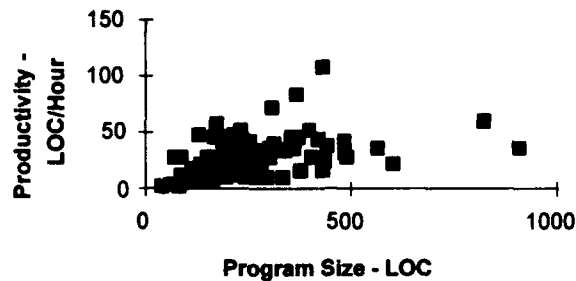


Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 79

LOC/Hour vs. Program Size - 104 Engineers, Program 10



Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 77

PSP Industrial Introduction

The PSP is being introduced in a number of software organizations.

Since few organizations have pre-PSP data, improvement comparisons are rarely possible.

The following charts show the data and experiences that are available to date.

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 78

Ford Motor Company

**Ford Motor Company, Dearborn, Michigan,
supports electronics for all divisions.**

**They have started PSP introduction with an SEI
course for several teams.**

Using PSP methods, one team

- planned a crisis project
- convinced management to accept the plan
- delivered on schedule

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 79

Motorola - 1

**The Motorola Paging Products Group, North
American Paging Subscriber Division, is in
Boynton Beach, Florida.**

**The faculty at Embry Riddle Aeronautical
University is assisting in PSP introduction.**

Two teams of engineers have been PSP-trained.

**Motorola plans to train all location engineers by
the end of 1996.**

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 80

Motorola - 2

The first PSP-trained group supports paging manufacturing.

- **Software changes are required when the product or process changes.**
- **Final software testing interrupts manufacturing.**
- **Prior to PSP, quality was a problem, and testing frequently disrupted manufacturing.**

The first release after PSP training had

- **one defect in test**
- **no defects in 5 months of manufacturing**

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 81

The AIS Corporation

Advanced Information Services (AIS) is an independent software contracting organization in Peoria, Illinois and Madras, India.

They have been working with SEI on process improvement and the PSP since 1992.

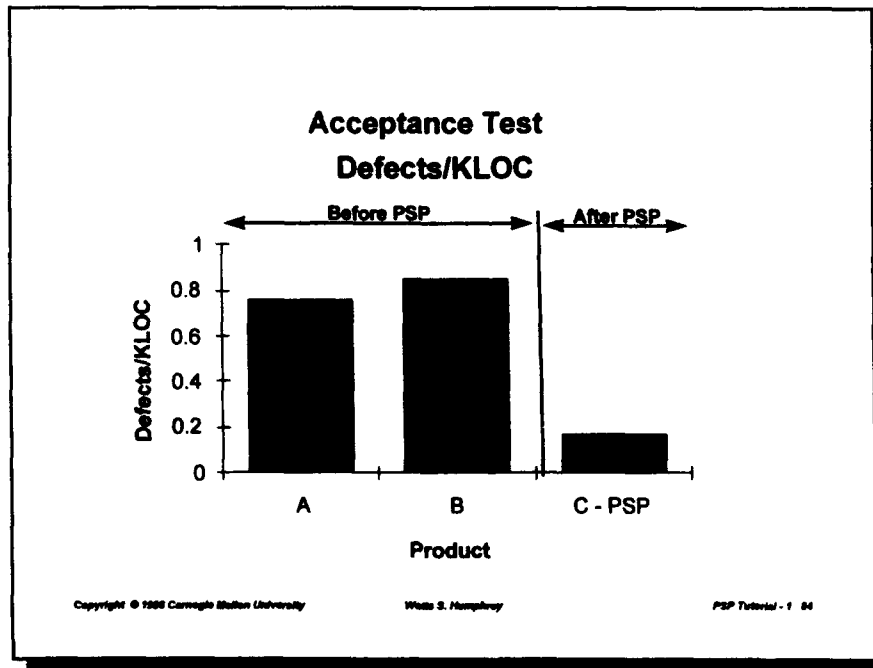
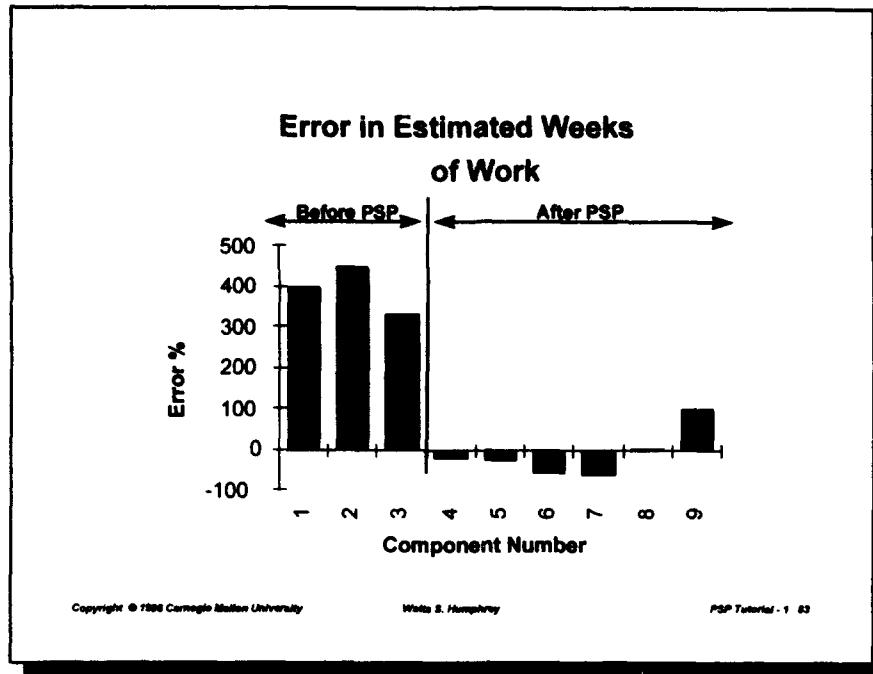
They have two SEI-trained PSP instructors.

AIS will have all engineers PSP-trained in 1996.

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 82



AIS Compile vs. Test Defects

The PSP teaches engineers that quality software is their personal responsibility.

Their objective is to remove all defects before the first compile.

When a program has many defects in compile, it will likely have many defects in test.

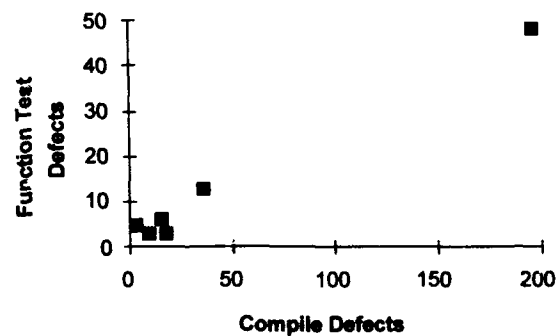
When a program has many defects in test, it will likely have many defects after test.

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 86

Compile vs. Function Test Defects

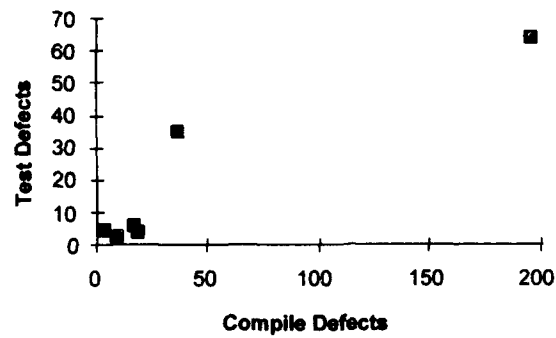


Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 86

Compile vs. Development Test Defects

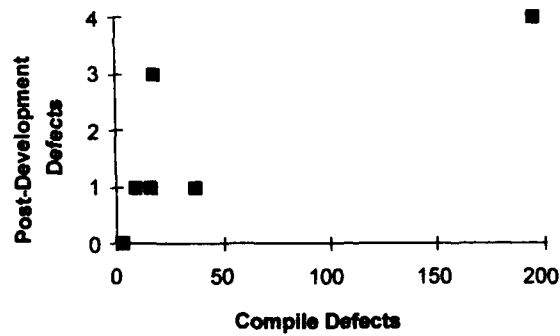


Copyright © 1988 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 87

Compile vs. Post- Development Defects



Copyright © 1988 Carnegie Mellon University

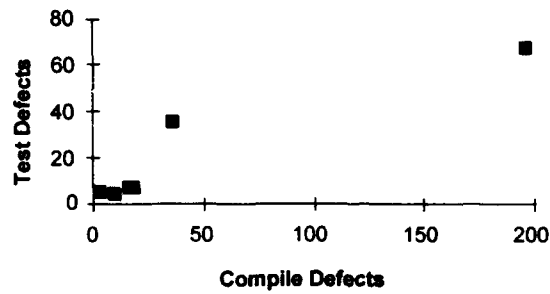
Watts S. Humphrey

PSP Tutorial - 1 88

Tuesday 17 June

(T201d) S-44

Compile vs. Development Test and Post- Development Defects

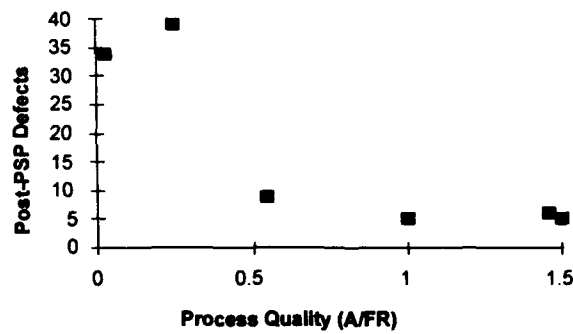


Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 89

Post-PSP Defects vs. Process Quality

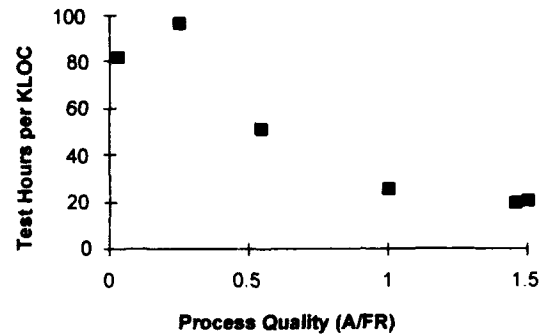


Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 90

Test Hours/KLOC vs. Process Quality



Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 91

PSP and Process Improvement

The PSP can help organizations improve their software processes in several ways.

- It helps engineers improve their performance.
- It exposes engineers and managers to the principles of all Capability Maturity Model (CMM)[™] Levels.

With PSP training, engineers are capable of using a personal CMM Level 5 process.

[™] Capability Maturity Model and CMM are service marks of Carnegie Mellon University.

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 92

Why Process Improvement Takes Time

The managers must agree to give improvement high priority.

Process definition takes engineering work.

The engineers need to be convinced to use the defined process.

To be fully effective, the engineers must gather, analyze, and use data on their work.

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 53

Process Definition Experience

Based on data from 8 process definition and modeling projects:

- **It takes about one day of engineering work for every 3 CMM activities.**
- **A documented process often has 20 to 40 pages per CMM Key Process Area (KPA).**
- **A document page can take up to 1 engineering day to complete, review, and publish.**

These times assume that the engineers understand the processes and know how to define them.

Copyright © 1996 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 54

Process Definition Effort

CMM Levels	KPAs	Activities	Engineering Days
2	6	62	132 - 240
3	7	50	150 - 297
4	2	12	43 - 84
5	3	26	65 - 129

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 86

Process Improvement Priority

Convincing managers and obtaining resources

- takes a minimum of one day
- can take forever

If the managers do not give process improvement high priority, there is no point in trying to introduce the PSP.

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 86

Getting the Processes Used

If the engineers are convinced that the processes will help them, they will use them quickly.

If the engineers are not convinced that the processes will help them, they will never use them.

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 97

PSP Supports Improvement

The PSP supports process improvement by

- teaching engineers and managers the principles of process improvement**
- providing personal experience with process improvement**

The PSP demonstrates 12 of the 18 CMM KPAs.

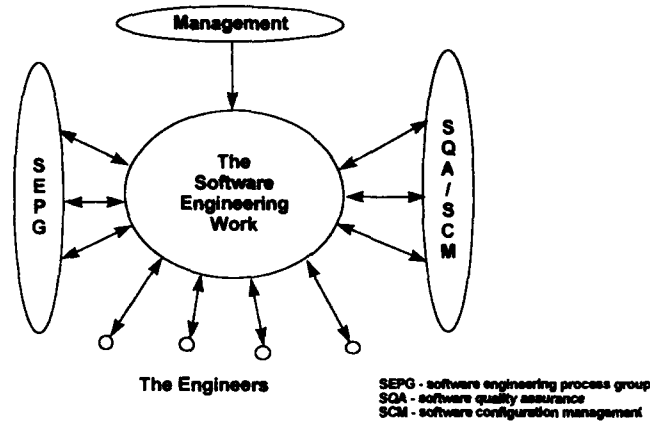
- Level 2: 2 of 6**
- Level 3: 5 of 7**
- Level 4: 2 of 2**
- Level 5: 3 of 3**

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 98

The CMM and the PSP



Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 98

The PSP and CMM Level 2

The principal constraint in getting to CMM Level 2 is getting engineers and managers to plan their work.

PSP training provides experience with

- making development plans
- gathering planning data
- tracking progress against plans

With PSP training, engineers see the benefits of project planning and tracking.

Copyright © 1996 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 100

The PSP and CMM Level 3

The principal constraints in getting to CMM Level 3 are

- defining processes**
- introducing inspections**

PSP training provides process definition skills.

- The PSP processes are definition models.**
- Engineers can adapt the PSP easily to their work.**

With PSP data, engineers can see the benefits of reviews and inspections.

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 101

The PSP and CMM Level 4

To get to CMM Level 4, organizations need to introduce detailed process measurements.

PSP training provides

- personal measurement experience**
- detailed data definitions**
- a framework for gathering and using data**

PSP-trained engineers recognize the need for and benefit of measuring their work.

Copyright © 1986 Carnegie Mellon University

Watts S. Humphrey

PSP Tutorial - 1 102

The PSP and CMM Level 5

To operate at CMM Level 5, the engineers must participate in process improvement.

PSP training provides engineers with personal experience using defined processes.

- **They see the value of defined processes.**
- **They understand process evolution.**
- **They know how to gather and use the Process Improvement Proposal (PIP).**

PSP-trained engineers recognize the need for continuous process improvement.

Improvement Summary

The PSP provides necessary process improvement skills.

The PSP convinces engineers to use defined and measured processes.

The PSP provides the data to convince managers that process improvement pays.

Introducing the PSP - 1

PSP introduction takes active management support and a substantial investment.

When properly introduced, the PSP is effective in helping engineers to

- **establish and meet commitments**
- **reduce product cycle time**
- **improve product quality**
- **define and improve processes**

Introducing the PSP - 2

PSP introduction has been successful only when

- **Qualified PSP instructors are used.**
- **The full PSP course is followed.**
- **Training is by engineering teams.**
- **Immediate management are directly involved.**
- **Senior management actively supports and tracks PSP introduction.**
- **A local support staff is used to help the engineers.**

A PSP Introduction Strategy - 1

Assume that your organization is currently working on CMM process improvement, and

- has established a Software Engineering Process Group (SEPG)**
- is at least implementing Level 2 planning, tracking, and quality assurance practices**

Produce a phased PSP introduction plan.

- Conduct pilot training for engineers and instructors.**
- Provide training to teams with team leaders.**
- Conduct executive and management training.**

Copyright © 1988 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 107

A PSP Introduction Strategy - 2

Train executives and managers before or with their engineers.

Train first-level managers with their teams.

Build a team of qualified trainers.

Copyright © 1988 Carnegie Mellon University

Walter S. Humphrey

PSP Tutorial - 1 108

SEI Courses on the PSP

Courses at SEI

- a 2-day senior management overview
- a 3-week teach-the-teachers course
- a 1-week basic PSP course
- a 1-week advanced PSP course
- a 4-day PSP instructor course

These courses can be given on-site.

Messages to Remember

- 1. The PSP shows engineers how to use process methods in their work.**
- 2. It will help them to understand and to better manage their own performance.**
- 3. PSP training will also accelerate organizational process improvement.**
- 4. PSP introduction takes concerted study and active management support.**



Carnegie Mellon University
Software Engineering Institute

A Method For Defining and Improving Software Processes

James D. Hart

**Software Process Program
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213**

Sponsored by the U.S. Department of Defense

© 1997 Carnegie Mellon University

Jim D. Hart - 1



Carnegie Mellon University
Software Engineering Institute

Who Should Attend

Anyone who is assigned responsibility for:

- **generating a process baseline for process improvement**
- **establishing process improvement objectives**
- **performing cost and time analysis for process changes**
- **piloting process changes**
- **institutionalizing best practices across the organization**

© 1997 Carnegie Mellon University

Jim D. Hart - 2



Carnegie Mellon University
Software Engineering Institute

What You Can Expect to Learn

How to distinguish a process from an activity

When a process is defined

What the attributes of a documented process are

A method for defining and improving a process

© 1997 Carnegie Mellon University

Jim D. Hart - 3



Carnegie Mellon University
Software Engineering Institute

A Method for Defining and Improving Software Processes



© 1997 Carnegie Mellon University

Jim D. Hart - 4



Carnegie Mellon University
Software Engineering Institute

Product Management Focus



Completion criteria for products are often defined in terms of schedule or cost constraints

Product specifications may come from several (many) independent sources

Work focuses on getting the products complete, not on how they are developed

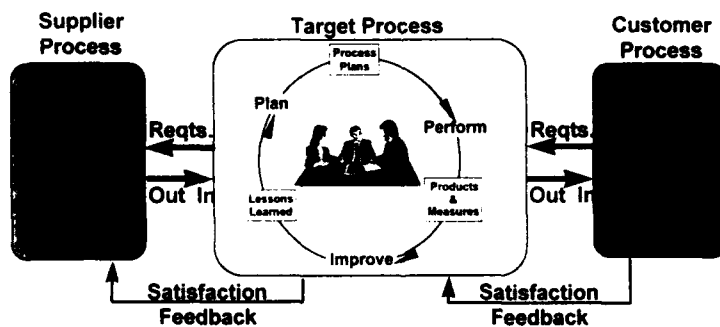
© 1997 Carnegie Mellon University

Jim D. Hart - 5



Carnegie Mellon University
Software Engineering Institute

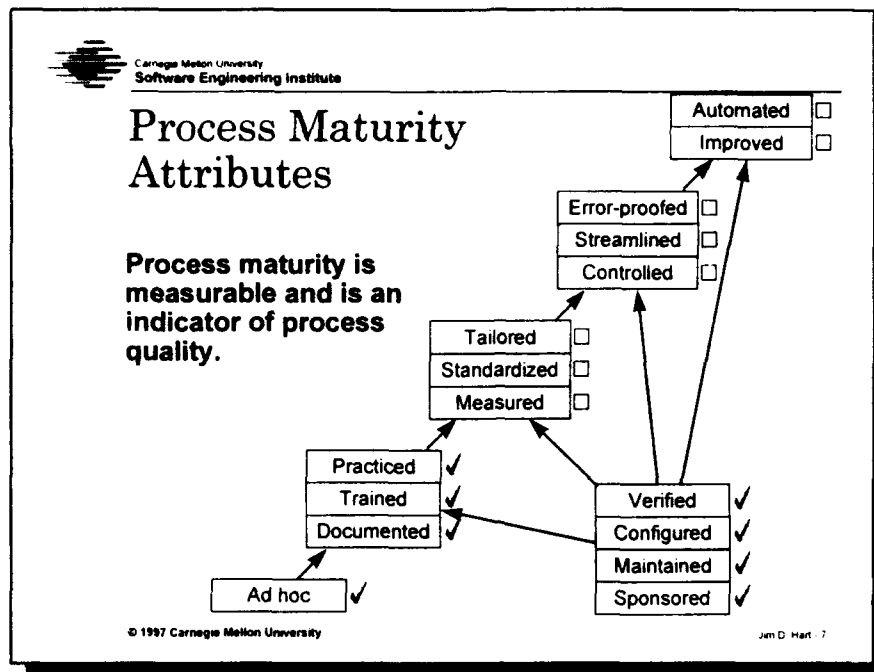
Managing the Process Chain



Process: A system of skilled people using documents, tools, and resources to plan, perform, and improve activities that produce specified products or services

© 1997 Carnegie Mellon University

Jim D. Hart - 6



Carnegie Mellon University
Software Engineering Institute

When Is a Software Process Defined?

You know a software process is defined when:

- it is documented
- training is provided
- it is practiced on a day to day basis

In other words:

Practice = Documentation = Training

Process Fidelity is the extent to which a process is defined

© 1997 Carnegie Mellon University

Jim D. Hart - 8



Carnegie Mellon University
Software Engineering Institute

Benefits of a Defined Process

Improves communication and understanding of current practices

Captures "corporate know-how" and best practices

Establishes a baseline for analysis and improvement of the process

Identifies where and how to measure the process

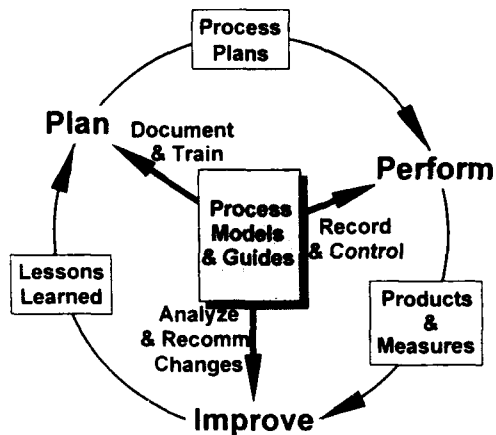
© 1997 Carnegie Mellon University

Jim D. Hart 9



Carnegie Mellon University
Software Engineering Institute

Use of Process Models



© 1997 Carnegie Mellon University

Jim D. Hart 10

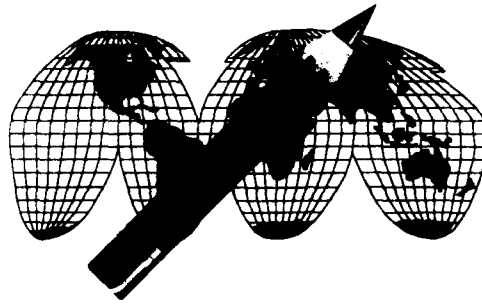


Carnegie Mellon University
Software Engineering Institute

Process 'N., Inc.

Mission: To help the world through process

Motto: "If it ain't in ink, it's not much of a process."



© 1997 Carnegie Mellon University

Jim G. Hart - 11



Carnegie Mellon University
Software Engineering Institute



Historical Company Data - 1

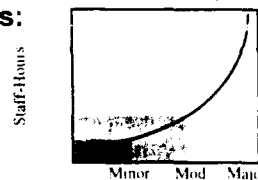
Process 'N, Inc. annually delivers:

- 35 process documents
- 500 pages (approximately 14 pages/document)

Effort to fix (ID, track, modify, re-release) post-release defects:

- major = 20 staff-hours
- moderate = 6 staff-hours
- minor = 3 staff-hours

*Process 'N, Inc.
Sensitive Information:
Do Not Divulge!!*



© 1997 Carnegie Mellon University

Jim G. Hart - 12



Carnegie Mellon University
Software Engineering Institute



Historical Company Data - 2

Estimated defects per page (on the 30 products delivered last year):

- major = 0.5
- moderate = 1.0
- minor = 5.5

*Process 'N, Inc.
Sensitive Information:
Do Not Divulge!!*

Effort we would have to had expended last year to fix ALL post-release defects (on the 30 products):

	Staff-Hours
major	$(0.5 * 500) * 20 = 5,000$
moderate	$(1.0 * 500) * 6 = 3,000$
minor	$(5.5 * 500) * 3 = 8,250$
Total	16,250

© 1997 Carnegie Mellon University

Jim D. Hart - 13



Carnegie Mellon University
Software Engineering Institute

Key Principles of a Successful Process Improvement Method

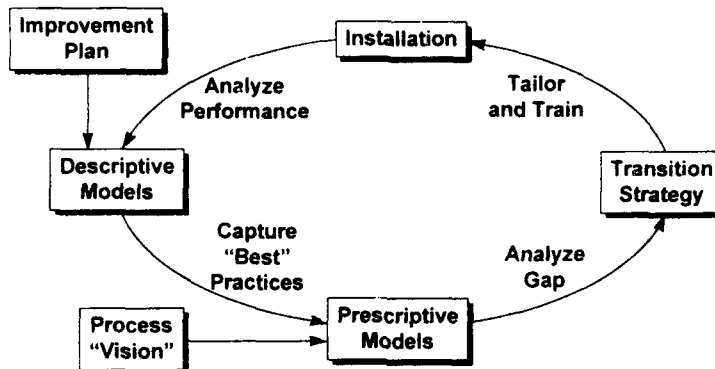
- Understand current reality before attempting to improve upon it**
- Improve processes incrementally**
- Apply process discipline to the modelling effort**
- Involve everyone in the improvement effort**
- Base all research and learning on the scientific method**

© 1997 Carnegie Mellon University

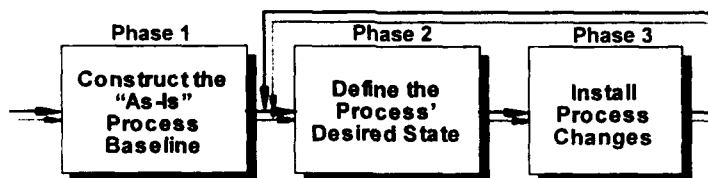
Jim D. Hart - 14



Process Improvement Cycle



Method Phases for Defining and Improving A Process



Phase 1: Capture an understanding of current reality

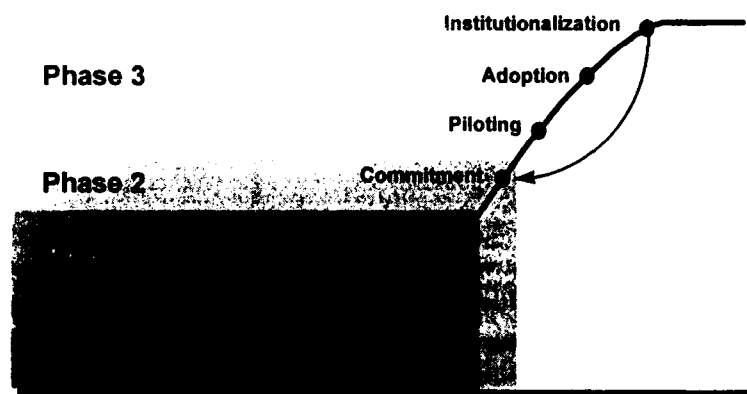
Phase 2: Create energy behind a (new) desired state

Phase 3: Make the desired state the new current reality



Carnegie Mellon University
Software Engineering Institute

Process Improvement Curve



© 1997 Carnegie Mellon University

Jim D. Hart - 17



Carnegie Mellon University
Software Engineering Institute

What You Need to Apply This Method

Approved strategic and tactical action plans

Chartered team assigned to model and improve the target process

Team members selected and agree to participate

Team members with the skills necessary to conduct the method

Business objectives (if available)

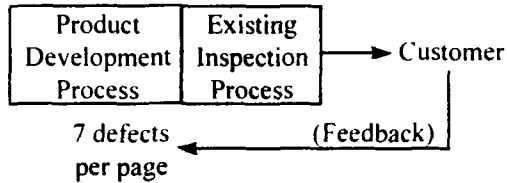
© 1997 Carnegie Mellon University

Jim D. Hart - 18

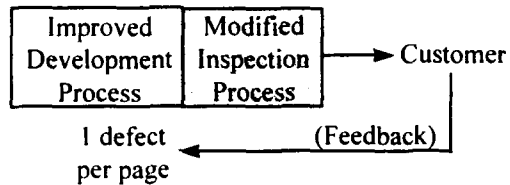


Process 'N, Inc's Business Objective

Current:



Desired:



A Method for Defining and Improving Software Processes





Carnegie Mellon University
Software Engineering Institute

Construct the
"As-Is"
Process
Baseline



Phase 1 - Current Reality

Purpose: To understand the target process as it is presently being performed within the context of the improvement effort

Desired Outcomes:

- Understanding of how the process is presently performed
- Increased sponsorship and support for improvement

What is Produced:

- "As-Is" process baseline
- List of improvement opportunities

© 1997 Carnegie Mellon University

Jim D. Hart - 21



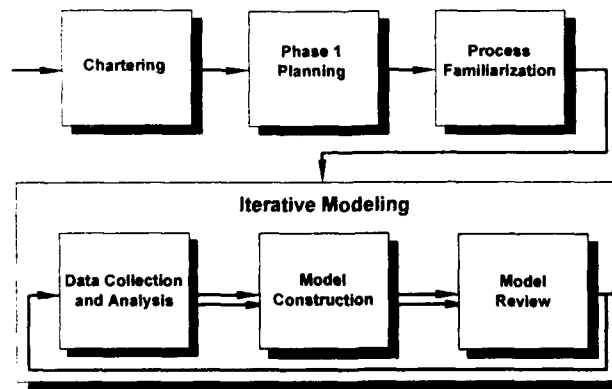
Carnegie Mellon University
Software Engineering Institute

Construct the
"As-Is"
Process
Baseline



Method Phase 1

Construct the "As-Is" Process Baseline



© 1997 Carnegie Mellon University

Jim D. Hart - 22



Carnegie Mellon University
Software Engineering Institute

Team Charter

Suggested contents of a charter:

- Customers and sponsors (by name)
- Purpose
- Linkage to business objectives
- Team scope (activities)
- Deliverables
- Team members
- Team authority and assumptions
- Team ground rules
- Signatures



What would you add, change, or delete from this list? Why?

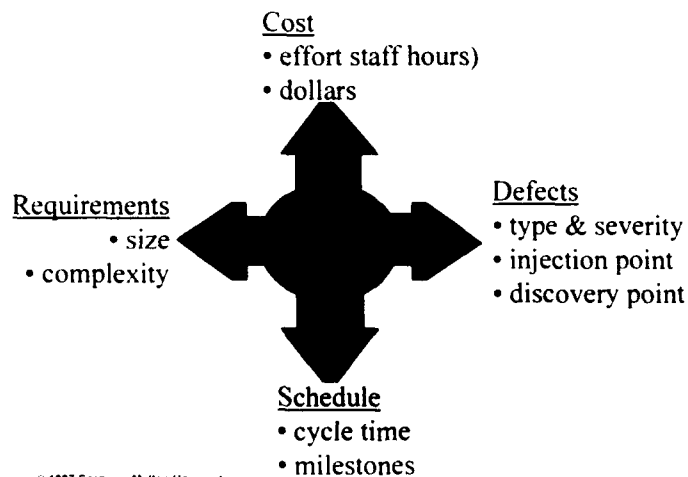
© 1997 Carnegie Mellon University

Jim D. Hart - 23



Carnegie Mellon University
Software Engineering Institute

Balancing Dynamic Forces



© 1997 Carnegie Mellon University

Jim D. Hart - 24

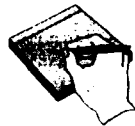


Carnegie Mellon University
Software Engineering Institute

Team Plan

Suggested contents of a team plan:

- Size and effort estimates
- Scheduled milestones
- Configuration control
- Tracking and oversight
- Risks and contingencies
- Signatures



What would you add, change, or delete from this list? Why?

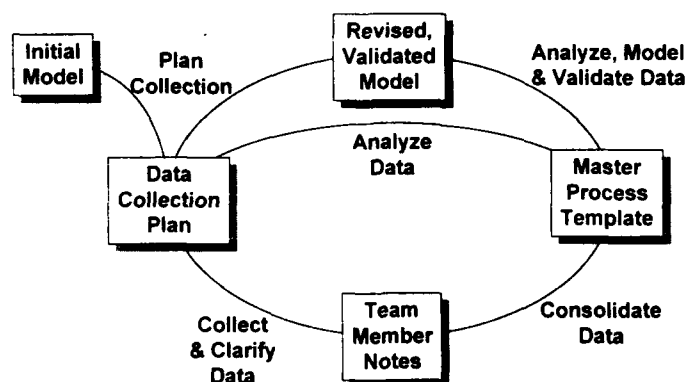
© 1997 Carnegie Mellon University

Jim D. Hart - 25



Carnegie Mellon University
Software Engineering Institute

Iterative Modeling Cycle



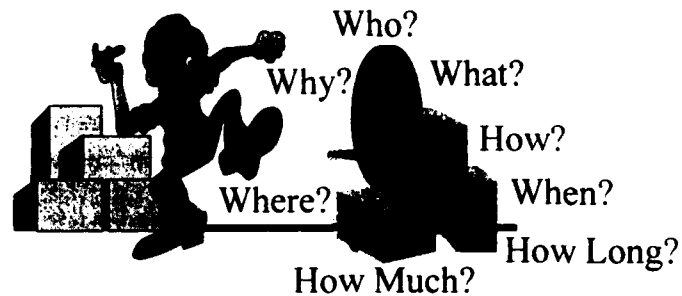
© 1997 Carnegie Mellon University

Jim D. Hart - 26



Carnegie Mellon University
Software Engineering Institute

Process Information Guidelines



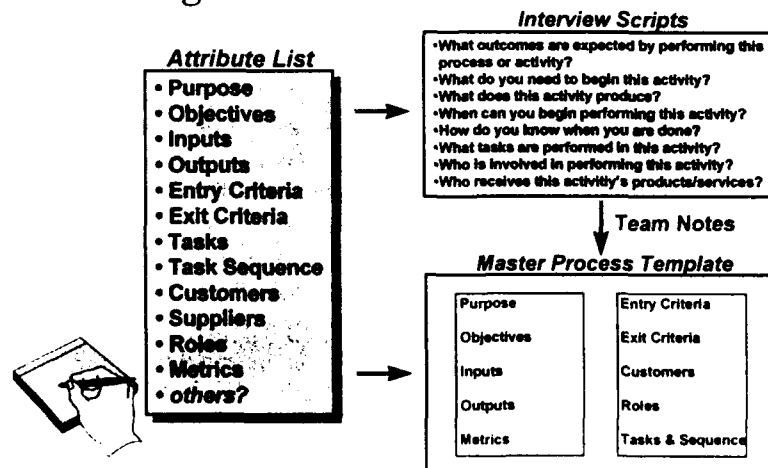
© 1997 Carnegie Mellon University

Jim D. Hart - 27



Carnegie Mellon University
Software Engineering Institute

Using Process Attributes



© 1997 Carnegie Mellon University

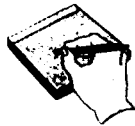
Jim D. Hart - 28



Carnegie Mellon University
Software Engineering Institute

Data Collection and Analysis

Getting the information you need to model the process can be difficult and time consuming



What kinds of cultural issues might you encounter in getting reliable information from stakeholders?

For each issue you identify, what can you do to minimize the problem?

© 1997 Carnegie Mellon University

Jim D. Hart - 29



Carnegie Mellon University
Software Engineering Institute

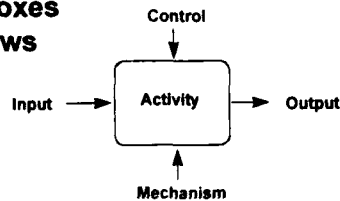
SADT Notation

Structured Analysis and Design Technique

Developed by Softech/MIT early 1970s, it is the basis for IDEF0

Constructs:

- "Activities" are boxes
- "Things" are arrows



© 1997 Carnegie Mellon University

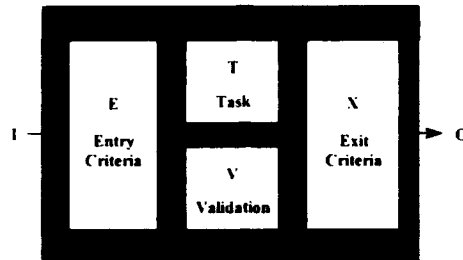
Jim D. Hart - 30



Carnegie Mellon University
Software Engineering Institute

EITVOX Notation

Based on the ETVX notation developed at IBM in the mid 1980's



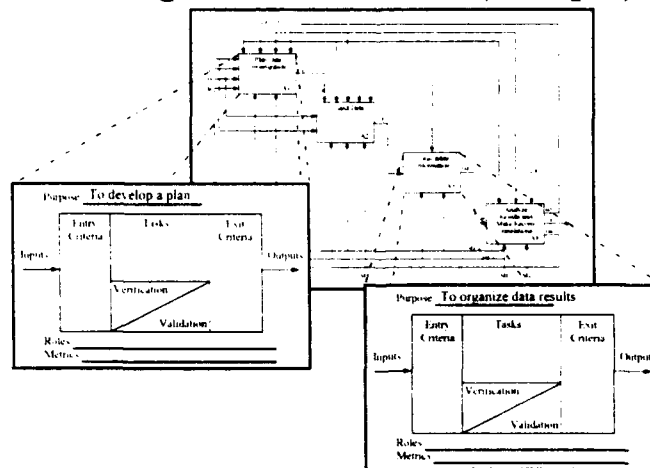
© 1997 Carnegie Mellon University

Jim D. Hart - 31



Carnegie Mellon University
Software Engineering Institute

Creating Process Models (Sample)



© 1997 Carnegie Mellon University

Jim D. Hart - 32



Carnegie Mellon University
Software Engineering Institute



Inspection Data (Last 10)

Average effort expended per inspection:

- 6 technical staff at 1.0 hour each
- 6 staff-hours (total)

Average effort to fix defects caught during inspection:

- major = 9 staff-hours
- moderate = 3 staff-hours
- minor = 2 staff-hours

© 1997 Carnegie Mellon University

Jim D. Hart - 33



Carnegie Mellon University
Software Engineering Institute



Inspection Containment - 1

Total defects contained by the inspection process (last 10 inspections):

- major = 60
- moderate = 110
- minor = 420

Defect containment, averaged per inspection

	Contained	Not Contained
major	6 (46%)	7
moderate	11 (44%)	14
minor	42 (35%)	77
Total	59 (38%)	98

© 1997 Carnegie Mellon University

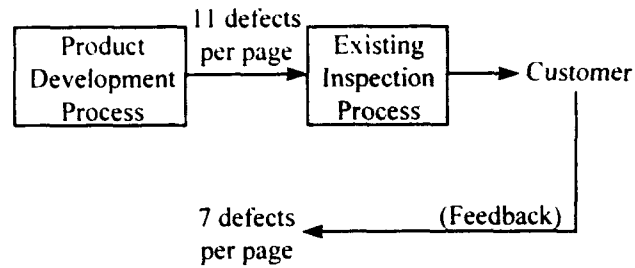
Jim D. Hart - 34



Carnegie Mellon University
Software Engineering Institute



Inspection Containment - 2



© 1997 Carnegie Mellon University

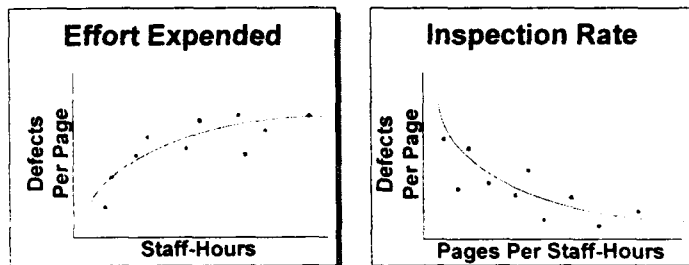
Jen D. Han 35



Carnegie Mellon University
Software Engineering Institute



Inspection Effort



© 1997 Carnegie Mellon University

Jen D. Han 36



Carnegie Mellon University
Software Engineering Institute



"As-Is" Process Baseline

"As-Is" Process Description

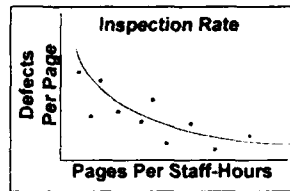


"As-Is" Performance Results

Defect containment, averaged per inspection

	Contained	Not Contained
major	6 (46%)	7
moderate	11 (44%)	14
minor	42 (38%)	77
Total	59 (38%)	98

Describes



© 1997 Carnegie Mellon University

Jim D. Hart - 37



Carnegie Mellon University
Software Engineering Institute

Common Issues in Phase 1

No measurements from the process to create a baseline

"Scope creep"

Inadequate time spent in chartering and planning

Not getting all perspectives of the process

Not seeking buy-in from the stakeholders

Affiliating with the sponsor, rather than the customers

© 1997 Carnegie Mellon University

Jim D. Hart - 38



Carnegie Mellon University
Software Engineering Institute

A Method for Defining and Improving Software Processes



© 1997 Carnegie Mellon University

Jim D. Hart 39



Carnegie Mellon University
Software Engineering Institute



Phase 2 - Desired State

Purpose: To establish a basis for making controlled and deliberate improvements to the target process

Desired Outcomes:

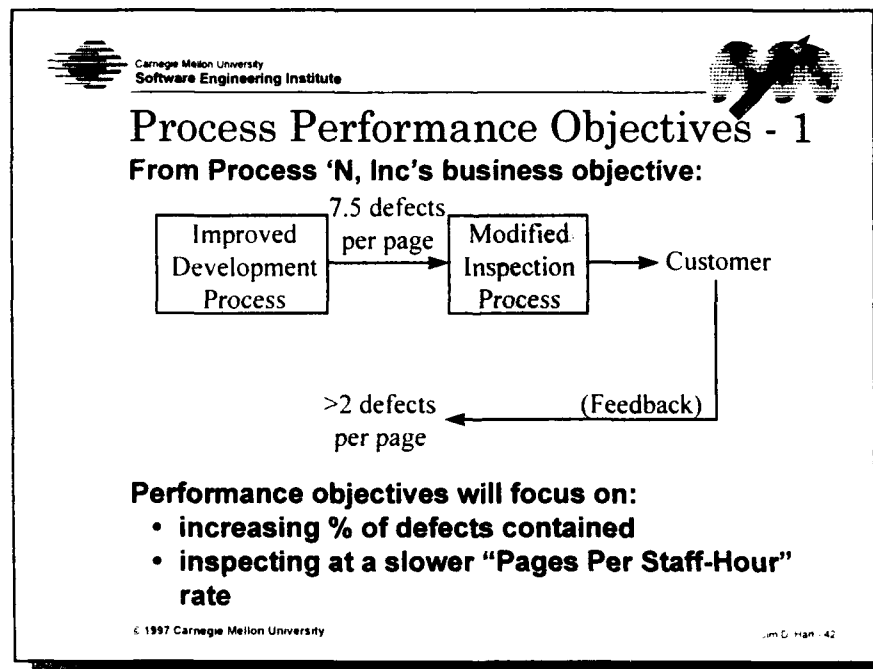
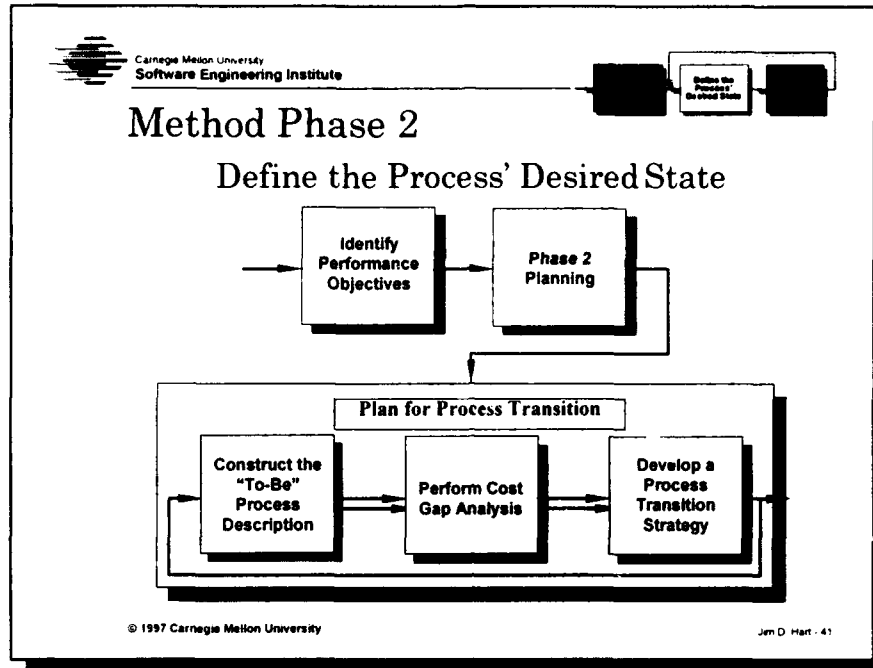
- A shared vision of the process' desired state
- Commitment to improve the target process
- A "map" of how to transition to the desired state

What is Produced:

- Quantifiable performance objectives
- "To-Be" process description
- Transition strategy

© 1997 Carnegie Mellon University

Jim D. Hart 40





Carnegie Mellon University
Software Engineering Institute



Process Performance Objectives - 2

Desired defect containment:

Defect containment, averaged per inspection

	Contained	Not Contained
major	7 (85%)	1
moderate	12 (80%)	3
minor	60 (75%)	20
Total	79 (77%)	24

Desired rate of inspection: average of less than one page per staff-hour expended

© 1997 Carnegie Mellon University

Jim D. Hart - 43



Carnegie Mellon University
Software Engineering Institute

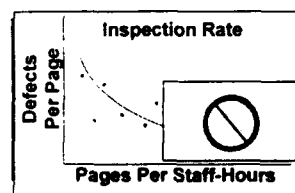


Engineering the "To-Be" Process Description

"To-Be" Performance Objectives

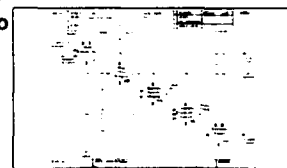
Defect containment, averaged per inspection

	Contained	Not Contained
major	7 (85%)	1
moderate	12 (80%)	3
minor	60 (75%)	20
Total	79 (77%)	24



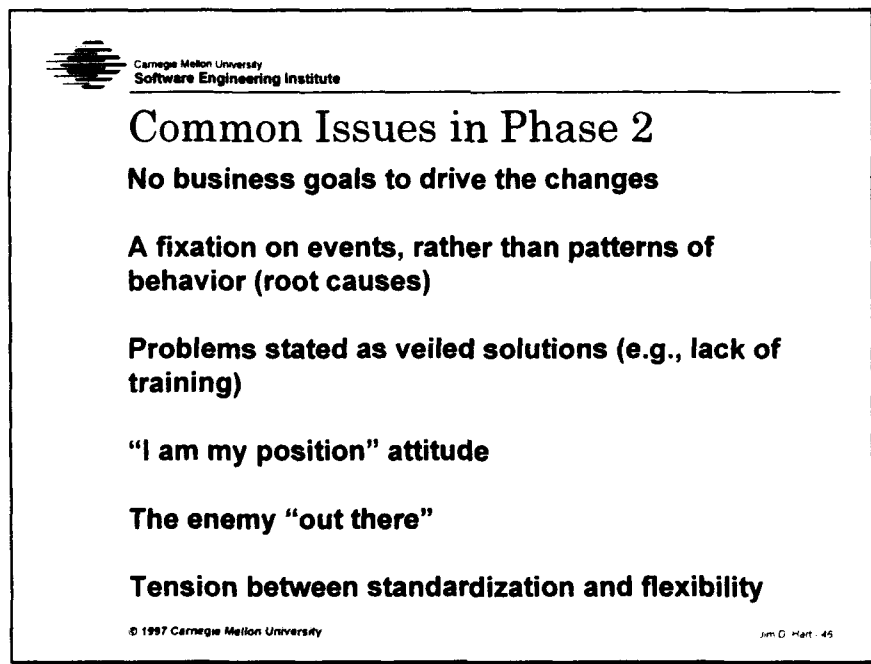
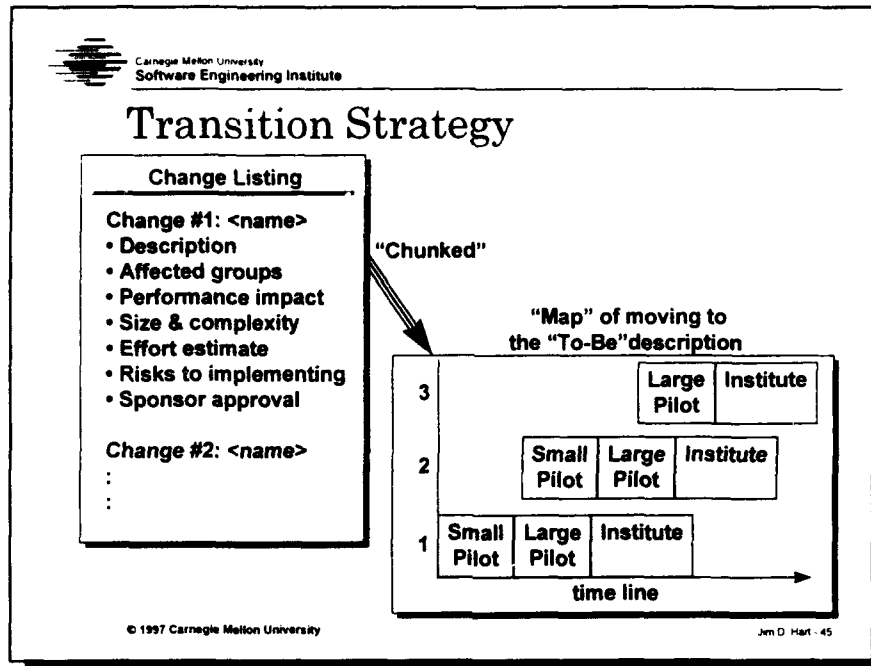
Maps To

"To-Be" Process Description



© 1997 Carnegie Mellon University

Jim D. Hart - 44





Carnegie Mellon University
Software Engineering Institute

A Method for Defining and Improving Software Processes



© 1997 Carnegie Mellon University

Jim D. Hart - 47



Carnegie Mellon University
Software Engineering Institute



Phase 3 - Installation

Purpose: To implement the identified process changes

Desired Outcomes:

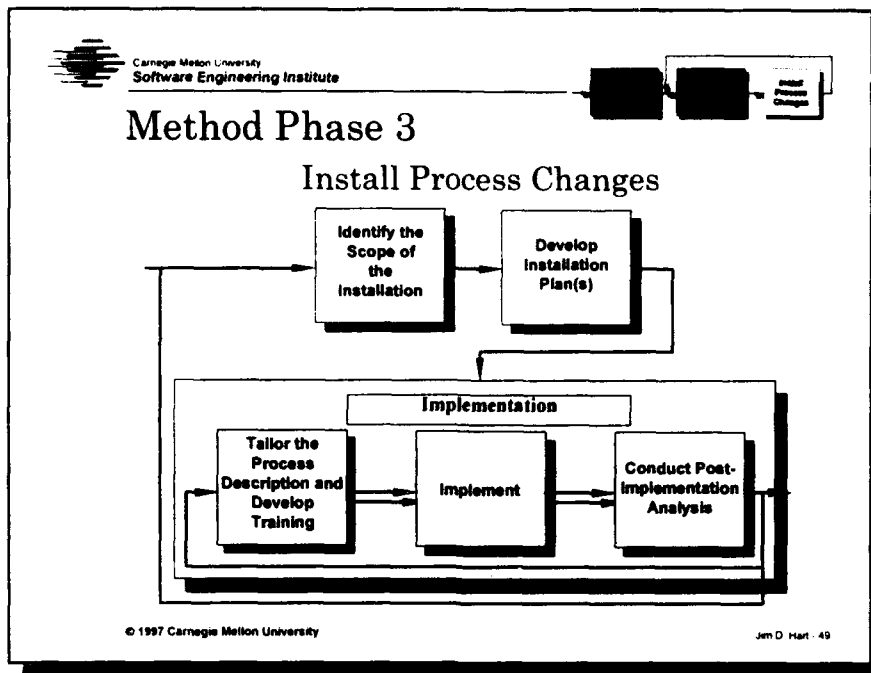
- A new "As-Is" process baseline that matches the "To-Be" description

What is Produced:

- Lessons learned
- New process performance baseline

© 1997 Carnegie Mellon University

Jim D. Hart - 48



 Carnegie Mellon University
Software Engineering Institute

Planning an Installation

Scope the installation to the right level and the right group(s)

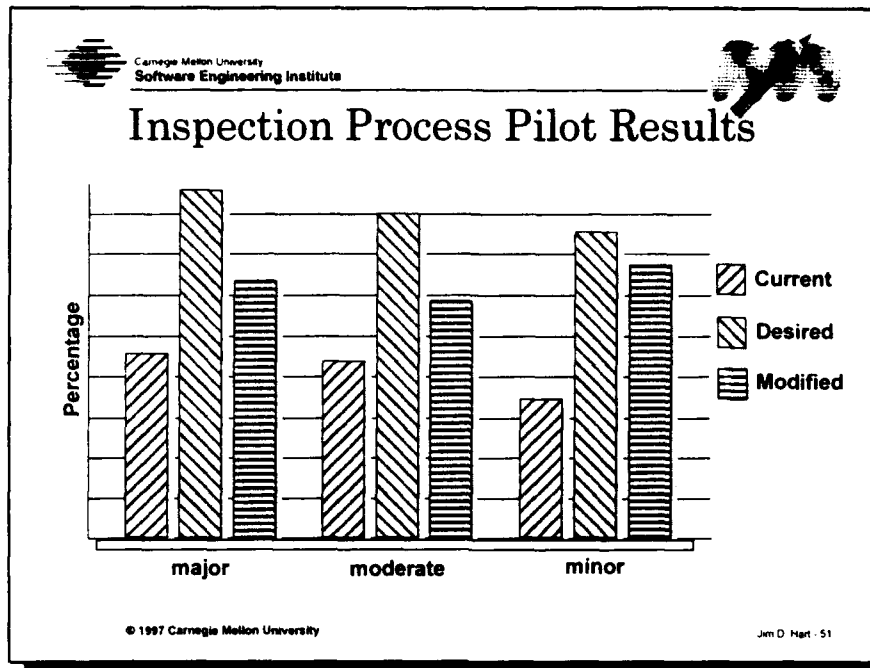
Work proactively with the group(s) who will be performing the new or modified process

Negotiate a win/win solution by making the rewards for engaging *much greater than* the risks of failure or embarrassment



What other considerations might be important for *your* organization? Why?

© 1997 Carnegie Mellon University Jen D. Hart - 50



Carnegie Mellon University
Software Engineering Institute

Lessons Learned About Inspections

Inspections are not solutions to poor quality issues. They only fix the *symptoms* in the development process.

However, inspection data can be used to help uncover *root causes* in the development process.

Significant improvement in inspection performance can be achieved with minimal effort by focusing on the process.

© 1997 Carnegie Mellon University

Jim D. Hart - 52



Carnegie Mellon University
Software Engineering Institute

Common Issues in Phase 3

Insufficient (time for) piloting.

Unable to distinguish between a process that is not working and one ill-prepared for implementation.

Skipping the Post-Implementation Analysis step.

Win/Lose mentality.

Declaring success at all costs.

Losing sight that real change is in changing behaviors.

© 1997 Carnegie Mellon University

Jim D. Hart - 53

SEPG Requirements Tutorial

Mac Craigmyle
Compita

*The Process Professional
People*

Process Professional - Requirements

© Compita Ltd 1997

- 1

Agenda

- ➔ Why Requirements Matter
- ➔ Requirements and its relations
- ➔ Requirements the process

Process Professional - Requirements

© Compita Ltd 1997

- 2

Why Improve Requirements

➡ To help get to CMM Level 2 ?

- Requirements Management
- Project Planning
- Project Tracking
- Subcontract Management
- Quality Assurance
- Configuration management

➡ Or because it really matters ?

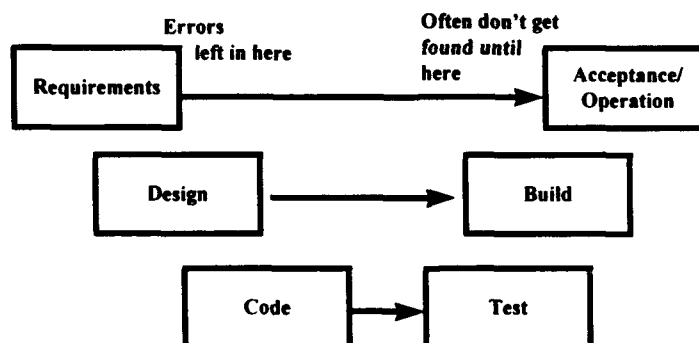
Process Professional - Requirements

© Compta Ltd 1997

- 3

The Cost to You

➡ A good development process is useless if the requirements are poor



Process Professional - Requirements

© Compta Ltd 1997

- 4

The Cost to Them

➡ The following failures had significant software requirements faults:

- Chernobyl
- Sizewell B
- Sellafield
- Challenger space shuttle
- London Underground train leaves station without its driver
- Ariane rocket
- London Ambulance System
-

(add your own favourite here)

Adapted from Computer Related Risks
P.G. Neumann 1995

Process Professional - Requirements

© Compta Ltd 1997

- 5

Your Turn

➡ Take a moment to write down what you believe the requirements process is



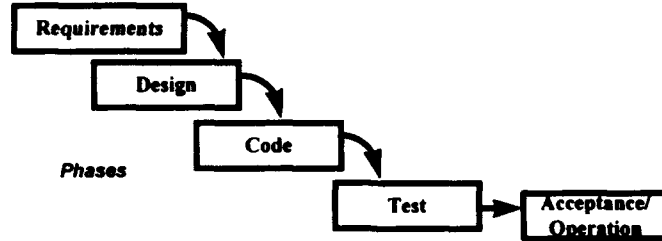
Process Professional - Requirements

© Compta Ltd 1997

- 6

Processes Vs Sequence

- ➔ Many organisations plan projects as a progression of phases or iterations.
- ➔ We need to be clear of the difference between a phase and a process
- ➔ Unfortunately we live with the legacy of the waterfall model where the phases are the processes- this can lead to confusion



Process Professional - Requirements

© Compta Ltd 1997

- 7

Requirements Phase Vs Requirements Process

- ➔ Today elements of requirements, design, and code may all occur at the same time, and still be called the requirements phase.
- ➔ As we shall see there are many ways to sequence the occurrence of the requirements process throughout the lifetime of a project. Examples include:
 - Evolutionary development
 - Stepped development
 - Incremental development
 - Prototyping ...
- ➔ In these approaches the requirements process starts and stops several times (as do the other development processes, often in parallel) as development progresses

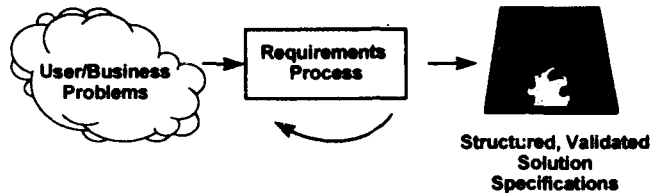
Process Professional - Requirements

© Compta Ltd 1997

- 8

Requirements Process Definition

- ➡ We will define the requirements process as the activities that develop a set of validated solution specifications for a set of problems
- ➡ These specifications (in the main) define *what* needs to be in place to solve the problem rather than *how* it will be implemented
- ➡ Each time the requirements process activates new solutions may be generated while incomplete ones are refined or discarded



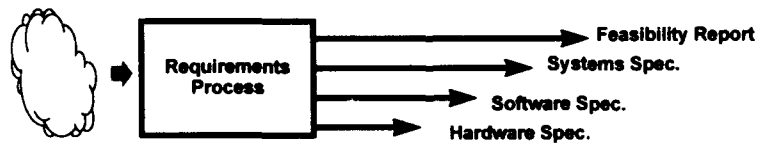
Process Professional - Requirements

© Compta Ltd 1997

- 9

Requirements Process Scope

- ➡ This definition encompasses the traditional *phases* of
 - Feasibility (trade-off, cost benefit and financial appraisal)
 - Requirements specification
 - Architectural definition (for validation of solution)
- ➡ It also relates to the creation of one or more specifications such as may be required for dual development of hardware and software



Process Professional - Requirements

© Compta Ltd 1997

- 10

The Two Dimensions of Processes

- ➡ To fully explore how to establish a robust requirements process two aspects need to be explored:
 - ◆ The behaviour of the requirements process relative to the other key processes in development
 - ◆ The internal elements of the requirements process itself
- ➡ This is the examination of the Black Box and White Box aspects of the process.
- ➡ This tutorial will examine each of these aspects in turn beginning with the the dynamic behaviour of requirements in context with the other key processes

Process Professional - Requirements

© Comptia Ltd 1997

- 11

Agenda

- ➡ Why Requirements Matter
- ➡ Requirements and its relations
- ➡ Requirements the process

Process Professional - Requirements

© Comptia Ltd 1997

- 12

Wicked Problems - Righteous Solutions

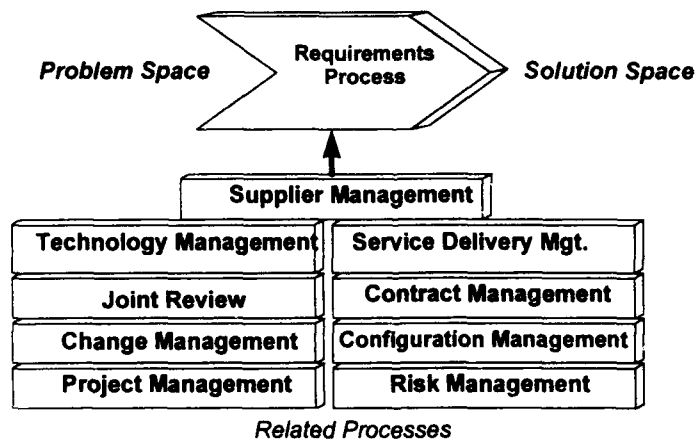
- The following objectives drive the majority of projects (in differing proportions according to the project goals)
 - Time to market
 - Cost
 - First time success (correctness of requirements)
 - Long term success
- The principal processes that assist achieving these goals are:
 - Project and Risk Management
 - Requirements
 - Change Management
 - Service Delivery Management
 - Contract and Supplier Management
 - Technology Management

Process Professional - Requirements

© Compta Ltd 1997

- 13

The Requirements Landscape



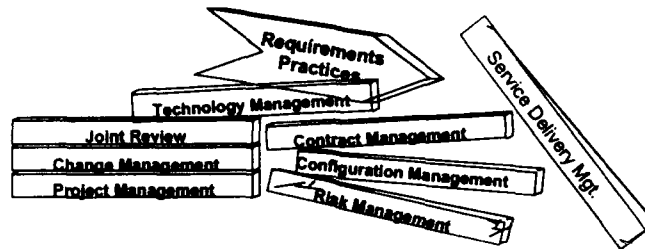
Process Professional - Requirements

© Compta Ltd 1997

- 14

Balance of Power

- ➡ If any of these supporting processes is weak or not applied then at best the requirements are sub-optimal, at worst they fail.



- ➡ A robust project must integrate these elements to the levels required by the project's goals (sadly these are often poorly defined)

Process Professional - Requirements

© Compita Ltd 1997

- 15

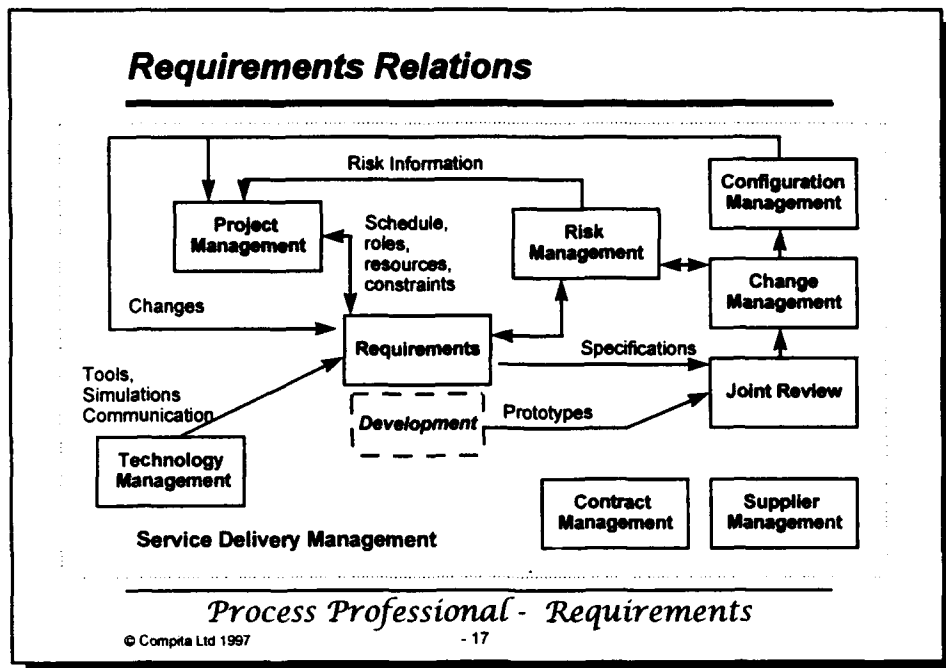
Classic Risks

- ➡ There are a number of known risks specifically related to achieving good requirements:
 - Feature creep
 - Incomplete, misunderstood and missing requirements
 - Requirements that impede rather than support end user workflow
 - Technical research swamps user needs
- ➡ Again the principle processes that assist to reduce these risks are as before:
 - Project and Risk Management
 - Requirements
 - Change Management
 - Service Delivery Management
 - Supplier and Contract Management ...

Process Professional - Requirements

© Compita Ltd 1997

- 16



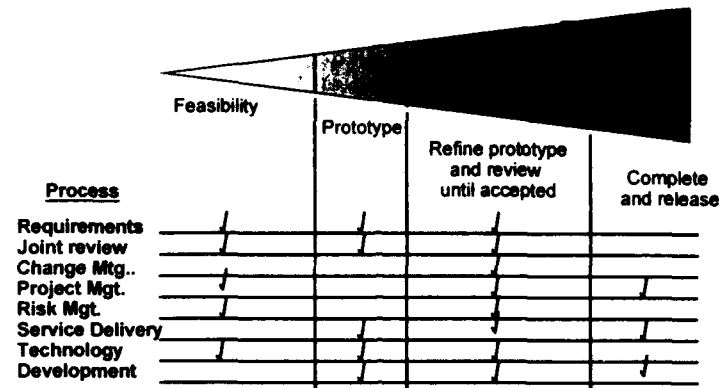
Lifecycles

- ➔ These processes can be sequenced in a number of ways to best meet a project's goals
- ➔ While there are a number of approaches we will concentrate on those that have been shown to strongly support the achievement of all four goals:
 - Time to market
 - Cost
 - First time success (correctness of requirements)
 - Long term success

Process Professional - Requirements

Evolutionary Prototyping

- ➡ Here the prototype becomes the product



Process Professional - Requirements

© Compta Ltd 1997

- 19

When To Use Evolutionary Prototyping

- ➡ If requirements are exceptionally volatile
- ➡ If the customer is reluctant to commit or sign off a requirements specification
- ➡ If the technical risks mean that development is unsure of the optimal architecture or underlying algorithms
- ➡ Risks:
 - Very difficult to estimate at start how long the project will take
 - Hard to get agreement on when the project is finished
 - Underlying structure probably makes maintenance difficult unless well controlled by application of change management, and development processes

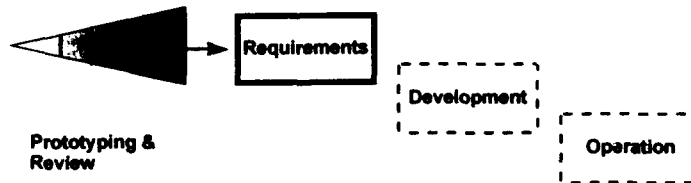
Process Professional - Requirements

© Compta Ltd 1997

- 20

Throw Away Prototyping

- ➔ Here the project is front ended by some evolutionary prototyping but followed by a more traditional waterfall



Process Professional - Requirements

© Compta Ltd 1997

- 21

When to Use Throw Away Prototyping

- ➔ Where the underlying technical components of the system are well understood (e.g. relational database) but where the Human Interface and end user workflow are not well understood
- ➔ Where fixed waterfall types of lifecycle are mandated by customer this can help reduce the requirements time
- ➔ Where there will be a long maintenance/update life and a well defined set of requirements and designs are required.
- ➔ Risks:
 - The throwaway model becomes an evolutionary one with unstructured and inappropriate architecture and languages

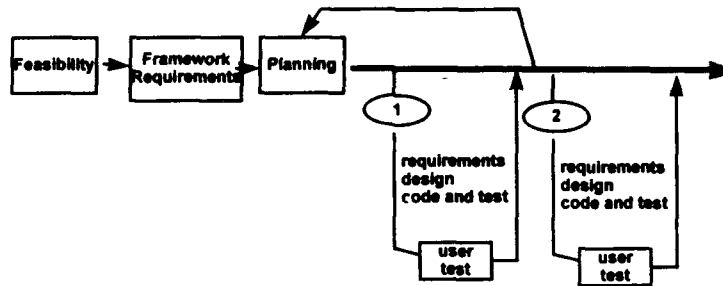
Process Professional - Requirements

© Compta Ltd 1997

- 22

Staged Delivery

- ➡ Here the known and full content of the project is delivered in stages with some adjustment following each stage



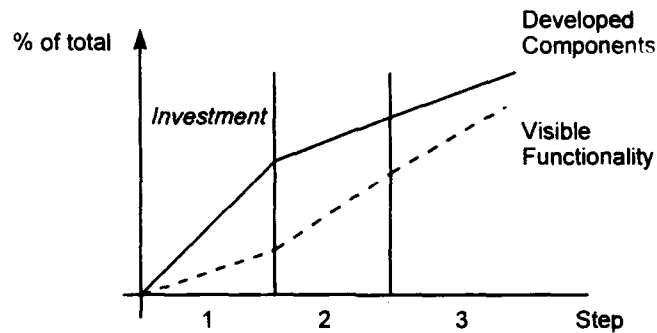
Process Professional - Requirements

© Compita Ltd 1997

- 23

Using Staged Delivery

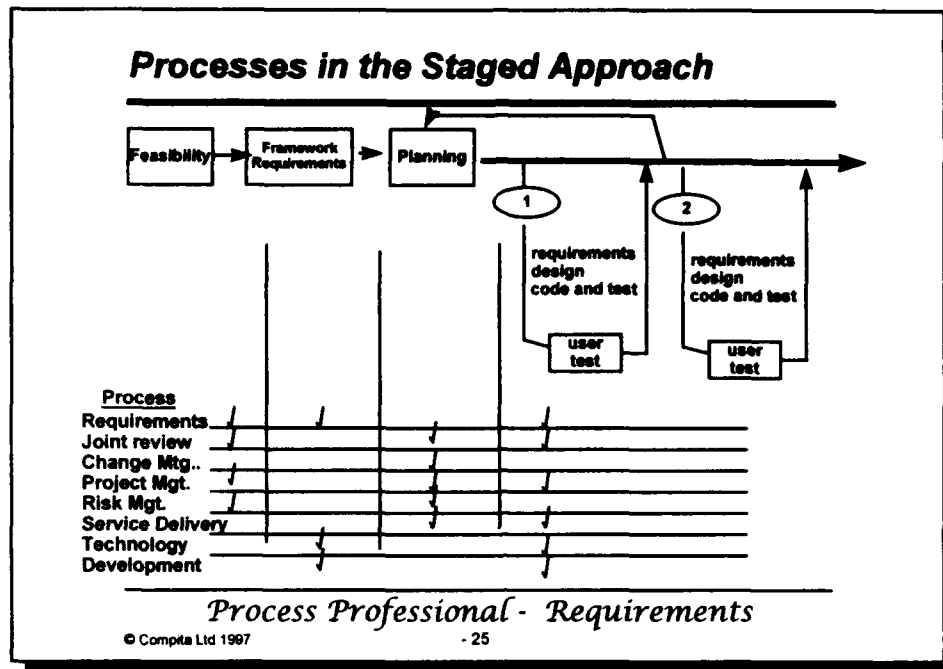
- ➡ The aim is to deliver the first 20% that provides the 80% of functionality
- ➡ First step tends to be longer than rest as it contains many of the infrastructure components



Process Professional - Requirements

© Compita Ltd 1997

- 24



When to Use Staged Delivery

- ➡ When the requirements for the complete system are reasonably stable but may change a little
- ➡ When the requirements partition well into cohesive groups of functionality
- ➡ When progress of visibility is important to the customer
- ➡ Where there will be a long maintenance/update life and a well defined set of requirements and designs are required.
- ➡ Risks:
 - Requires careful planning of both the management and technical levels
 - Can be a nightmare if system has many interdependencies.

Process Professional - Requirements

© Compta Ltd 1997

- 26

Release Based

- ➔ This modifies the staged delivery approach into a release based model by prioritising each requirement into:
 - Mandatory feature
 - Significant feature
 - Useful to have
 - Low priority
- ➔ The stages/releases are then focused on delivering the core set of mandatory features and as many of the others as the step time permits.
- ➔ Sometimes the step may be lengthened slightly to ensure the mandatory features are included.
- ➔ This requires more emphasis on the change management and joint review processes than with staged delivery

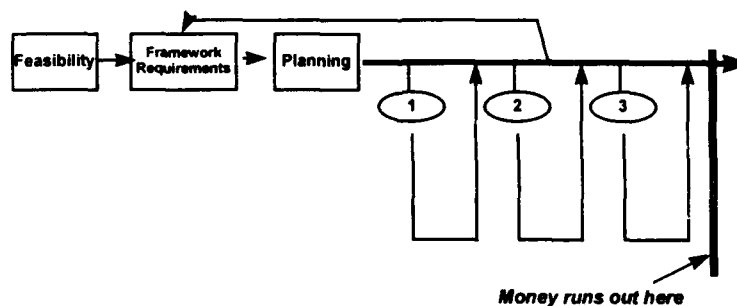
Process Professional - Requirements

© Compta Ltd 1997

- 27

Release Based Completion

- ➔ This approach can continue until
 - The money runs out
 - All the features are delivered
 - New system supersedes the current one



Process Professional - Requirements

© Compta Ltd 1997

- 28

When To Use Release Based

- ➡ Product development for multiple distribution
- ➡ Where having something ready by a fixed date is the top priority
- ➡ To control feature creep
- ➡ Risks:
 - Money runs out before first meaningful step ever completes - wasting the investment in early stages.

We recommend that some form of requirements prioritisation is *always* used

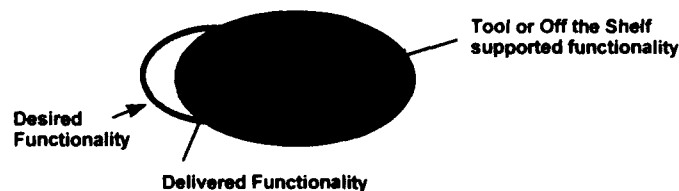
Process Professional - Requirements

© Compta Ltd 1997

- 29

Expedient Delivery

- ➡ Where cost and time are paramount or effort is limited
- ➡ Include only features that
 - can be easily created with known tools
 - lever established library components
 - can be largely achieved with off-the-shelf packages



Process Professional - Requirements

© Compta Ltd 1997

- 30

Expedient Delivery - Key Processes

- ➡ Here the process focus changes a little and the following become more emphasised
 - ◆ Supplier Evaluation
 - ◆ Technology Management
 - ◆ Joint Review

Process Professional - Requirements

© Compta Ltd 1997

- 31

When To Use Expedient Delivery

- ➡ When the fit between the desired and supported functionality is high and no mandatory features are excluded, an off-the-shelf solution is better than do it yourself
- ➡ When developer and users agree that this is the only way to get anything
- ➡ When technology risk is best deferred to a vendor
- ➡ Risks
 - ◆ Loss of control over product
 - ◆ Dependence on commercial vendors
 - ◆ Poor fit of delivered solution in real world

Process Professional - Requirements

© Compta Ltd 1997

- 32

Spiral

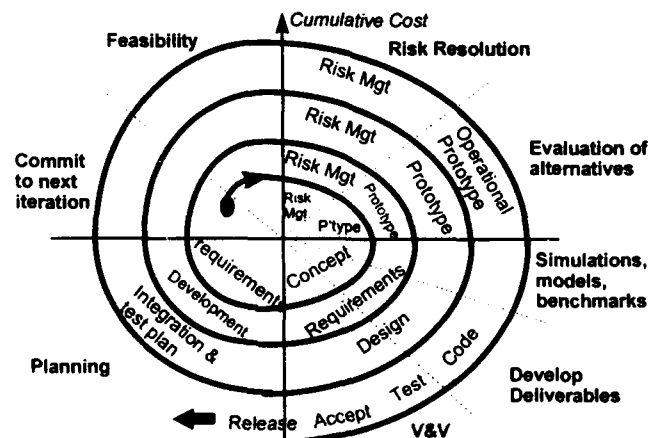
- ➡ Very sophisticated approach providing tight control
- ➡ Has a number of iterations with the following steps:
 - Feasibility of iteration
 - Risk management
 - Develop iteration deliverables and verify
 - Plan next iteration
 - Commit to next iteration - if you decide to have one
- ➡ The following illustration gives the concept - you can tailor and combine other lifecycles as required

Process Professional - Requirements

© Compta Ltd 1997

- 33

Spiral Lifecycle



Process Professional - Requirements

© Compta Ltd 1997

- 34

When to Use Spiral

- ➡ Where costs need to be contained particularly in the early stages
- ➡ As cost increases risks decrease
- ➡ Excellent all round visibility
- ➡ Excellent management of new (untried) projects and technology
- ➡ Note DSDM is a similar approach and has three iterations with four steps per iteration
- ➡ Risks:
 - Requires attentive and informed management

Process Professional - Requirements

© Compta Ltd 1997

- 35

Best Practice Toolkit

- ➡ Each approach provides a component in your toolkit
- ➡ Which is appropriate depends on the goals of the project or parts of a project
- ➡ They are not mutually exclusive and can be used in parallel for different components of the project
- ➡ You do not have to stick to the one you first thought of if things change as the project progresses

RAD is about building the most appropriate combination for the project to hand and controlling it

Process Professional - Requirements

© Compta Ltd 1997

- 36

Lifecycle Selector

Requirements not understood	X			X	X
Architecture not understood				X	X
High reliability required		X		X	X
System likely to grow significantly	X	X	X		X
Development exposed to major risks			X		X
Schedule constrained			X		
Low overhead required				X	
Requires midcourse correction capability	X				
Needs customer visibility	X				X
Needs management visibility		X	X		X
Evolutionary prototype	✓				
Staged Delivery		✓			
Released Based			✓		
Expedient Delivery				✓	
Spiral					✓

X = strong characteristic

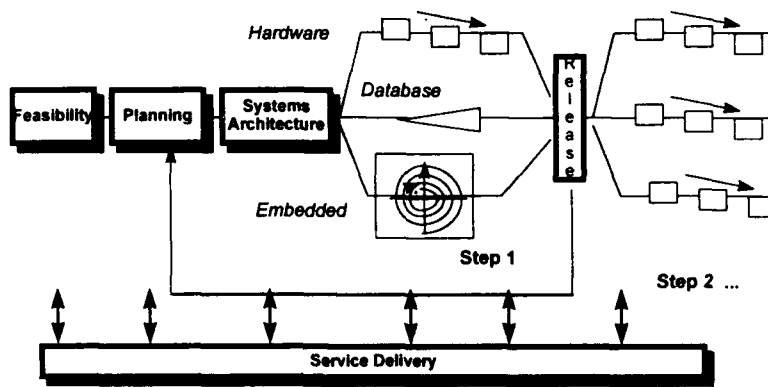
Process Professional - Requirements

© Compta Ltd 1997

- 37

Systems Engineering Example

➔ Here hardware, embedded software and database software are being developed



Process Professional - Requirements

© Compta Ltd 1997

- 38

Agenda

- ➡ Why Requirements Matter
- ➡ Requirements and its relations
- ➡ Requirements the process

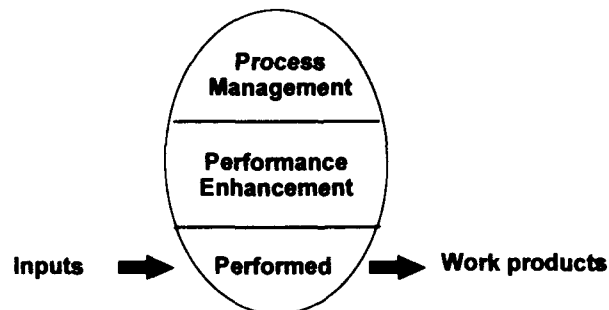
Process Professional - Requirements

© Compita Ltd 1997

- 39

15504 Process Definition

- ➡ Processes comprise three principal elements
 - Activities that transform the inputs into the work products
 - Activities that enhance the performance of the process
 - Activities that optimise the process over time



Process Professional - Requirements

© Compita Ltd 1997

- 40

What's in the Layers

- ➔ Each layer can be thought of as showing where the processes such as project and risk management, configuration management, and quality management act on the process
- ➔ Performance Enhancement
 - Performance Management
 - Quality Management
 - Work Product Control
 - Process Tailoring
 - Process Control
- ➔ Process Management
 - Process Definition
 - Technology Management
 - Human Resource Management
 - Measurement

Process Professional - Requirements

© Compta Ltd 1997

- 41

Requirements Performance I

- ➔ The work products output from requirements is the feasibility report, the requirements specification, and a definition of the acceptance criteria and tests.
- ➔ Feasibility typically explores the following:
 - The causes of the problems to be solved
 - Alternative approaches
 - Cost and benefit analysis
 - Risks, their impact and mitigation strategy
- ➔ See example Feasibility Report included in handouts.

Process Professional - Requirements

© Compta Ltd 1997

- 42

Requirements Performance II

- ➡ Define the functional and non functional requirements including :
 - Safety, Security, Human Factors (ergonomics)
 - Interface, Operations, Maintenance
 - Constraints, qualification requirements
- ➡ There are many methods and techniques associated with the description that include:
 - Use cases, playscripts and scenarios
 - Structured English
 - Data Dictionaries
 - Role Activity Diagrams
 - Flow diagrams, functional or event partitioned
 - Entity relationship models for data
 - State transition diagrams for events and logic

ISO 12207

Process Professional - Requirements

© Compta Ltd 1997

- 43

Requirements Performance III

- ➡ Whichever method or combination of techniques is appropriate they should exhibit the following characteristics:
 - Traceability to needs
 - Consistency with needs
 - Testability
- ➡ Also the definition of how the customer will ensure that the requirements have been met should be described. These are the acceptance tests or at a minimum the acceptance criteria and strategy
 - Developing acceptance tests early also influences the content and shape of the requirements document
 - It may also place development requirements on the developers to provide test applications, test data and other items.

ISO 12207

Process Professional - Requirements

© Compta Ltd 1997

- 44

Note on Traceability

- ➡ This can be hard to maintain
- ➡ Tool support for the method you use helps greatly
- ➡ Separate numbering of each requirement (and clear partitioning into individual requirements) can be used as the trace.
- ➡ Traceability matrices can then be developed to show
 - mapping of high level requirements to the low level requirement
 - requirements map to acceptance tests
 - requirements map to design elements

Process Professional - Requirements

© Compta Ltd 1997

- 45

Requirements Performance IV

- ➡ The requirements definition is no good if it cannot be translated into viable implementation
- ➡ This requires that before the requirements are seen as complete (at least for this increment) some investigation into their future feasibility needs to be made
 - Feasibility of the architectural design
 - Feasibility of the integration, operation and maintenance
- ➡ In most cases these are "tested" by the invocation of another process such as joint review or high level design.

**The handouts include a template for
a Requirements Specification**

Process Professional - Requirements

© Compta Ltd 1997

- 46

Performance Management

- ➡ This is primarily the deployment of Project and Risk Management within the development of the requirements specification.
- ➡ High level scheduling and risks are reduced by the appropriate lifecycle. Here we are looking at the more day to day use of project management and project risk management to control the definition of requirements
- ➡ Topics
 - Allocation of resource and roles
 - Allocation of time to develop specification
 - Identification and mitigation of risks in developing the requirements

Process Professional - Requirements

© Compta Ltd 1997

- 47

Allocation of Roles

- ➡ As with choosing a lifecycle there is no one correct approach to the roles and structures for all projects
- ➡ Equally we have always to work with the clay we have at our disposal
- ➡ Broadly, there are three different objectives that a requirements team may be formed to meet
 - *Problem resolution* - chiefly occupied with specific issues such as a problem during the maintenance phase
 - *Creative* - ground breaking technical innovation
 - *Focused development* - rapid development and release of understood functionality

Process Professional - Requirements

© Compta Ltd 1997

- 48

Team Structures

	Trust	Autonomy	Clarity of objective
	Corrective maintenance on live system	New products	Product upgrades
	Focus on issues	Explore alternatives	Focused tasks, clear defined roles, often marked with clear success / fail criteria
	Spiral	Evolutionary prototyping, Staged delivery, Spiral, Expedient delivery	Waterfall, Staged delivery, Spiral, Expedient delivery
	Respected, street wise, people sensitive	Cerebral, independent thinkers, self-starters, tenacious	Loyal, committed, action focused, responsive sense of urgency

Adapted from Teamwork (Larson and LaFasto 1989)

Majority of Requirements Needs

Process Professional - Requirements

© Compta Ltd 1997

- 49

Team Players

- ➔ Customer involvement and shared ownership of requirements is vital to maximise the success of a project
- ➔ There are three compositions of team players suited to the requirements phase:
 - *Function oriented team* - peer group of equal status comprising specialists in specific areas. Led by one individual.
 - *Feature team* - Cross functional team where members are empowered, accountable and balanced.
 - *Theatre team* - Headed by a director who exerts the overall vision supported by a manager who effectively "produces" the work. Other roles are negotiated by team members.

Process Professional - Requirements

© Compta Ltd 1997

- 50

Team Uses

	Fair on all types but best for Tactical	Problem Solving and Creative	Creative
	Technical Bias	New product development	Multi-Media components eg Interactive web
	• Small groups • Long term life • Single management interface	• Draws on members across functions • Builds accountability and ownership • Often used for RAD and JAD	• Dominated by strong personalities • Has strong central vision rather than fully defined aim

Process Professional - Requirements

© Compta Ltd 1997

- 51

Involving Users

- ➡ Some organisations use Business Analysts to represent the end users views:
 - Business analysts help by seeing the big picture but can sometimes not understand the operational aspects
 - End users themselves understand the operational aspects but may not really understand the wider business problems
 - Moral: try and involve both
- ➡ Joint Application Development approaches structure and ensure success with customer involvement during requirements:
 - Facilitated
 - Structured
 - Capture need, features, workflow and constraints well
 - Do not employ technical approaches (e.g.. ERD's)

Process Professional - Requirements

© Compta Ltd 1997

- 52

Meet User Half Way

- ➡ It can be forgotten that IT applications are there to serve the user who is there to serve the business
- ➡ Users are not technical specialists so they should not be expected to comprehend or sign off against technical definitions such as data flow diagrams, ERDs or state transition diagrams
- ➡ These techniques underpin the technical quality of the requirements definition and are a means to an end
- ➡ Once you have applied them to refine the requirements go back and spend the time to construct a high level specification that the end user can comprehend and agree to:
 - JAD does this
 - Developing and reviewing a user guide built from the refinement of the requirements can also be useful and keep the project progressing

Process Professional - Requirements

© Compta Ltd 1997

- 53

Managing Requirements Process Risks

- ➡ Much of the risks associated with capturing requirements can be reduced by:
 - Appropriate lifecycle
 - Appropriate methods, techniques and tools
 - Formal user involvement
 - Appropriate review mechanism (more later)
- ➡ The remaining risks are related to specific project issues and the completeness and timeliness of the specification
 - To ensure completeness consider appointing one person to ensure completeness and consistency. This person considers the wider form of the document rather than the absolute detail - the requirements architect.
 - If technology risks cannot be managed internally transfer the risk to subcontractors

Process Professional - Requirements

© Compta Ltd 1997

- 54

Quality Control

- ➡ Like risk management many of the quality issues will be managed by correct lifecycle and user involvement
- ➡ Consistency can be obtained through the use of templates and uniform methods
- ➡ The key quality approach is to have systematic reviews of the documentation and other work products such as prototypes:
 - Inspections for mandatory components
 - Structured walkthroughs for remainder
- ➡ These may require training of both end users and developers in structured review methods
 - The hard evidence abounds to prove that structured reviews of requirements leads to reduced rework, better first time success and cost control

Process Professional - Requirements

© Compta Ltd 1997

- 55

Work Product Control

- ➡ This covers all the outputs of requirements:
 - Screen details
 - Requirements definition
 - Acceptance tests
 - Tools
 - Feasibility reports
 - JAD outputs
 - Review minutes
- ➡ At a minimum a set of baselines should be maintained following each major lifecycle increment and these are controlled throughout:
 - Problem and change management
 - Configuration management and version control

Process Professional - Requirements

© Compta Ltd 1997

- 56

Process Tailoring

- ➡ This is the modification of any standard approaches to meet the needs of the project.
- ➡ For example on a project that affects a large number of end users you might set up a focus group to include invited members from other companies
- ➡ Such changes from established practice should be reviewed and explicit

Process Professional - Requirements

© Compta Ltd 1997

- 57

Process Control

- ➡ This is the use of measures to keep closed loop control on the performance of requirements as the process progresses
- ➡ These measures should relate to the goals of the requirements process for the project
- ➡ For example:
 - ◆ Early inspections could classify the type of errors found. If the majority were errors of omission further requirements work should stop and the process should be backtracked and adjusted to prevent the problem re-occurring
 - ◆ A ratio of mandatory against total requirements could trigger a refinement of the mandatory classification if it exceeds 80 % (perhaps two classifications would emerge such as High Priority Medium High Priority)

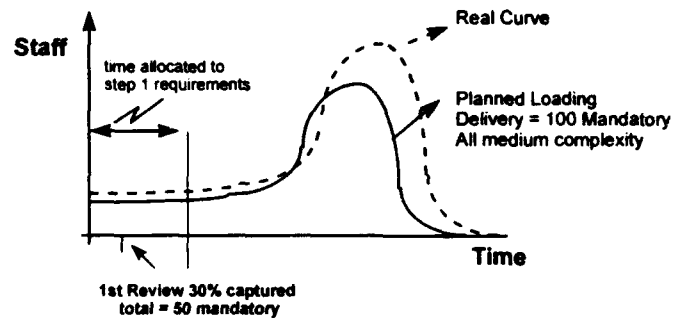
Process Professional - Requirements

© Compta Ltd 1997

- 58

Purpose of Requirements Process Control

- ➔ The sooner an indication of the true nature of the remaining phases of a project is available the more chance there is of changing the process to assure success is delivered



Process Professional - Requirements

© Compta Ltd 1997

- 59

Process Management

- ➔ This is an organisational rather than project set of good practice
- *Process Definition* - documented, validated, owned
 - *Technology Management* - correct technical infrastructure
 - *Human Resource* - available trained staff
 - *Measured* - process measures to assist continuous improvement

Process Professional - Requirements

© Compta Ltd 1997

- 60

Feasibility Study Report Example

Submitted to: SEPG Requirements Tutorial

**Submitted by: Mac Craigmyle
on behalf of Compita Limited
Scottish Software Partner Centre
Station Road
South Queensferry
EH30 9TG**

Scope

- Identifies the project, customer, and other stakeholders
- States intended readership
- Records assumptions and constraints (such as time and cost for the study)
- Details any exclusions

Objectives

- Statement of the subject of the study and specific objectives to be explored
- Statement of customers known areas of concern (e.g. we want x but think y is an issue here)
- Diagram (optional) to position proposed component within the context of a larger system. This should illustrate any major interfaces.

List of High Level Requirements for Study

- Specific software or hardware platforms required for the system to function. Include version details
- Any supporting software or hardware for the system

Business and Quality Requirements for the Study

- This can simply refer to contract or highlight key points as required by the state the main project is at.
- Description or list of purchaser supplied hardware or software and any proofing test it may be subject to. Characteristics such as liability maintenance and ownership of product or Intellectual Property Rights.
- Purchaser responsibility such as support and assistance.
- Specific business goals and quality requirements.

Findings and Recommendations

Management Summary and Conclusion

- Details of the major findings and the report's final recommendation

Risks and Scope of Recommendations

- Details on any risks associated with the recommendations and any limits to be considered when reading the recommendations

Cost Benefit Analysis of Alternatives

- Quantified details on each alternative, its cost and the cost for each associated benefit.
- Selection criteria for final recommendation.

Investigation Details

- Description of how investigation was performed
- Description or reference to other documentation about items developed during investigation (e.g. prototypes, simulations)
- Results from tests and documentation references

Abbreviations

References

Change History

Details the current and historic status of the document and any relevant authorisations.

Requirements Template Example

Submitted to: Amsterdam SEPG Tutorial

Submitted by: Mac Craigmyle
on behalf of Compita Limited
Scottish Software Partner Centre
Station Road
South Queensferry
EH30 9TG

Date:

Scope and Objectives

To illustrate and define a standard format for requirements specifications.

This template attempts to cover all of the topics that might appear in a requirements specification. The intent of this is not that any one requirements specification may contain all of the defined sections, but rather to provide a framework that all requirements specifications can follow. Where a particular section would normally be present but is not required in a particular specification, then the heading could be included in the specification with an appropriate comment in the text following e.g. Not Applicable.

Template

Scope

- Identifies the project, customer, and other stakeholders
- States intended readership
- Records assumptions and constraints
- Change management details
- When this is a subsystem of a wider system these details are given
- References to associated documents (e.g. proposals, contracts, project numbers).

Component or System Description

A brief introduction to the component or system architecture that is covered by the specification. This section should be brief, since it is included only to help the reader quickly understand what is being specified.

A context diagram should be included to assist in positioning the proposed component or system. All key interfaces should be illustrated.

Architecture Overview

Overview of architectural or relevant part of high-level design. This should only be included when the purchaser has specifically requested a particular system architecture e.g. client-server, or the purchaser has made it a requirement to define part or all of the system architecture as part of the contract.

Purchaser Requirements

This section shall include those requirements that directly affect the purchaser's use of the component or system. It is divided into two sections, functionality and characteristics:

- the *functionality* of the component or system describes what it can do, e.g. print a report. All functions must be stated so they can be tested.
- the *characteristics* of the system provide those attributes of the system by which its quality is described and evaluated. All characteristics will be quantified and testable

Each paragraph (or group of paragraphs) should contain a reference tracing where the requirement comes from. Each sentence or paragraph should be numbered; wherever possible only one requirement should be defined per numbered item.

Each paragraph (or group of paragraphs) should indicate its importance, for example by classifying it as one of:

- *Mandatory* - Absolutely essential feature; product will be cancelled if not included.
- *Required* - Individual features are not essential, but together they affect the viability of the product.
- *Desired* - Nice-to-have feature; one or more of these features could be omitted without affecting the product viability.

Human Interface

This section shall define the required menu structures, screen/window designs, report layouts and other interfaces to operators and/or supervisors. At this stage, the requirements may be broad or relate to existing standards or products.

Reference may be made to other specifications and standards.

Data Types

This section shall include a description of all of the system or application data types that are available to the purchaser, either by the use of application development tools or by the use of forms, displays, reports and printouts

Control Structures

This section shall detail the control structures for the system or application.

Application Development Environment

This section shall specify the components of the system that are available to the *customer* for use in developing applications. It should contain at a minimum the data types and languages or application generators that are available.

Hardware

This section shall detail any hardware requirements that exist because of a perceived purchaser need.

Software

This section shall detail any software requirements that exist due to purchaser needs. If the purchaser is providing product to be interfaced to or incorporated into the system or component being defined, then this should be clearly stated in the requirements, and all assumptions and requirements documented. These requirements may include some of the following:

- Operating System
- Database
- Communications
- Interfaces.

Purchaser-Related Characteristics

In most instances the customer will have specified few, if any of the following characteristics. They should therefore be included either if they are a specific strength or capability of the proposed system, or alternatively to put specific limits on certain characteristics from the very beginning to avoid open-ended debates at acceptance test. If some of these characteristics are not specified in this section, they may be required to be specified under the company requirements section e.g. many characteristics have a direct bearing on the company's support costs once the system is in use.

Pre-operational

- Packaging
- Installation
- Configuration

Functionality

- Suitability
- Accuracy
- Interoperability
- Compliance - standards
- Security

Reliability

- Maturity
- Fault tolerance
- Recoverability

Usability

- Understandability
- Learnability
- Operability

Efficiency

- Time behaviour
- Resource behaviour

Maintainability

- Analysability
- Changeability
- Stability
- Testability

Portability

- Adaptability
- Installability
- Conformance
- Replaceability

Documentation

This section shall detail the requirements for the documentation that must be available to the purchaser for the component or system.

Company Requirements

This section defines the requirements that are necessary to ensure that the system or component meets the developer's business goals as opposed to the purchaser's needs. Any conflicts in these needs must be resolved, either by obtaining a concession from the purchaser or foregoing the development organisation's need.

In specifications which are distributed outside Disks, this section may be omitted and placed in a separate document.

Business Requirements

Cost

This section could discuss the costs associated with the system being specified if not detailed elsewhere. It could cross-reference the project plan for details and include only a summary here.

These costs should include all development costs and possibly projected support costs. If possible this section should also discuss any flexibility in the costs, and any cut-off costs, beyond which the development would be stopped because it would no longer be cost-effective to finish the system.

Make/Buy

This section should discuss the criteria to decide whether this system or component (or part thereof) would be more cost effectively bought-in or sub-contracted rather than developed in-house e.g. commodity application, lack of experience, lack of resources etc.

Relationship to future products

This section should cover the requirements placed on the system or component that relate to its relationship to other products that are not yet developed e.g. to ensure compatibility with future products and systems.

Scheduled ship date

This section should reference the plan or detail the projected shipment dates of the system, including any interim releases or staged shipments that might be planned. This section should also document any constraints or dependencies associated with these shipment dates.

Support considerations

This section should discuss any special or unusual support considerations that this system or component might require e.g. first time a JAVA system will be shipped to the field.

Company Hardware Requirements

Hardware Functionality

This section should cover the required capabilities of the hardware that are required by the company, but are not necessarily required or relevant to the purchaser e.g. requirement for the hardware to support multiple operating systems, or must support ethernet.

Hardware Characteristics

This section should cover the required characteristics of the hardware that are visible to the company, but are not necessarily visible or relevant to the purchaser. At a minimum this should include any requirements for diagnosis of the hardware.

Company Software Requirements

Software Functionality

This section should cover the required capabilities of the software that are required by the company, but are not necessarily required or relevant to the purchaser e.g. databases, operating systems, communications (remote access), diagnostics.

Software Characteristics

This section should cover the required characteristics of the software that are visible to the company, but are not necessarily visible or relevant to the purchaser e.g. reusability of code, packaging.

Acceptance Criteria

This section should detail the outline acceptance criteria that will be used as the basis for developing the validation (acceptance) test plan.

For requirements involving specific contracts, it may be replaced by a reference to the acceptance criteria contained in the contract.

Glossary

A glossary of terms and definitions used in the requirements specification that might not be known to the readership or open to misinterpretation. If a standard glossary is available this might be referenced in the reference section and included with the specification to any readers or reviewers of the specification.

Appendix A Traceability

Rev no.	Requirement	Design	Implementation	Test	Re test
A	1.2.4	5.4.3	1.5.7.8	345	345c

Process Improvement Action Planning

(A Step By Step Tutorial)

**John D. Vu
Associate Technical Fellow
Software Engineering
Research & Technology
The Boeing Company**

The Boeing Company

John D. Vu



Agenda

Introduction

After the Assessment

Step by step in the Establishing Phase

Step by step in the Acting Phase

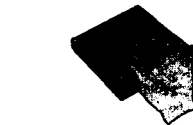
Step by step in the Leveraging Phase

Lessons Learned

Discussion

The Boeing Company

John D. Vu



Objectives

To provide guidance to an organization that has completed an assessment and needs a plan of action.

To provide step by step guidance on how to establish an action plan successfully.

To discuss lessons learned from software process improvement.



The Boeing Company

John D. Vu

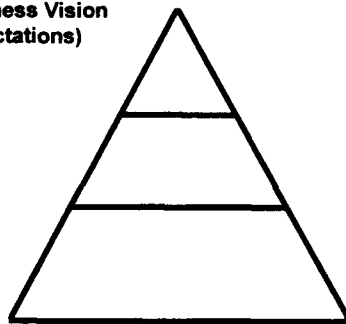


3

Process Improvement Approach

Improvement Direction

Strategic Business Vision
(Goals, Expectations)



Process Improvement
(Data, Activities)

Improvement Actions



The Boeing Company

John D. Vu



4

Five Principles Of Process Improvement

- 1) Improvement direction must start at the top.
- 2) Everyone must be involved in the improvement process.
- 3) Effective improvement requires knowledge of current process.
- 4) Improvement is continuous.
- 5) Improvement requires investment.



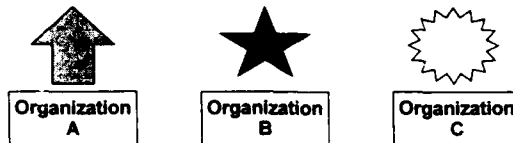
The Boeing Company

John D. Va



5

Implementation Differences



There is not one right way to implement process improvement.

Process improvement does not mean the same to all organizations.

Some organizations have different implementations of process improvement that may be considered better.

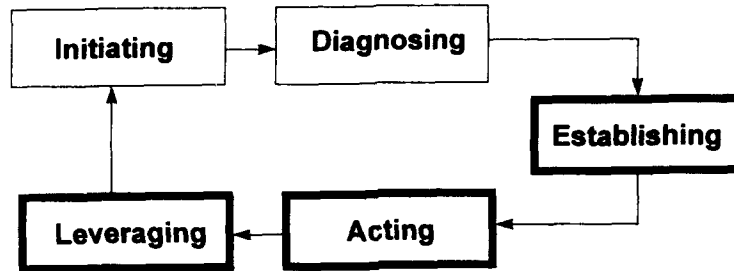
The Boeing Company

John D. Va



6

The IDEAL Model sm

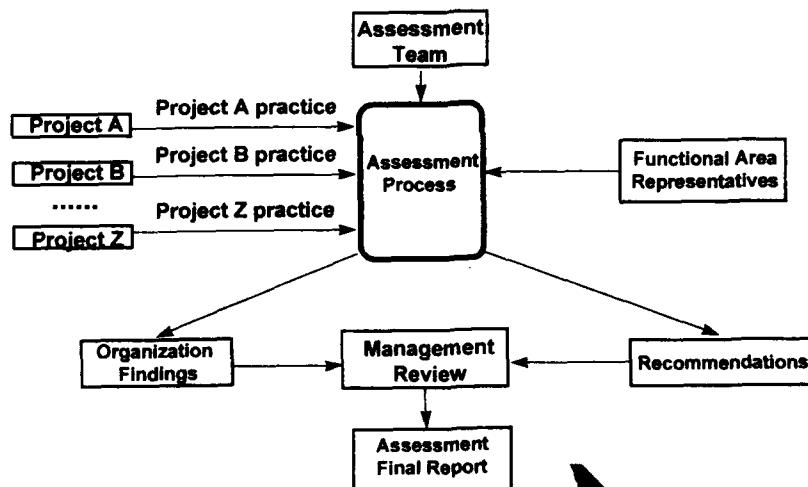


The Boeing Company

John D. Vu

7

Diagnosing Phase



The Boeing Company

John D. Vu

8

After the Assessment

Many organizations:

- Stalled after an assessment
- Do not have an action plan
- Failed to implement any improvement task
- Failed to realize the benefit of process improvement



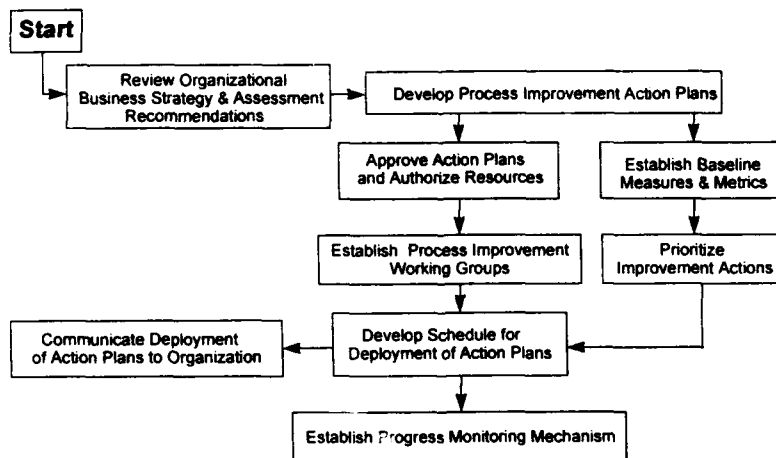
The Boeing Company

John D. Vu



9

Step by Step in "Establishing" Phase



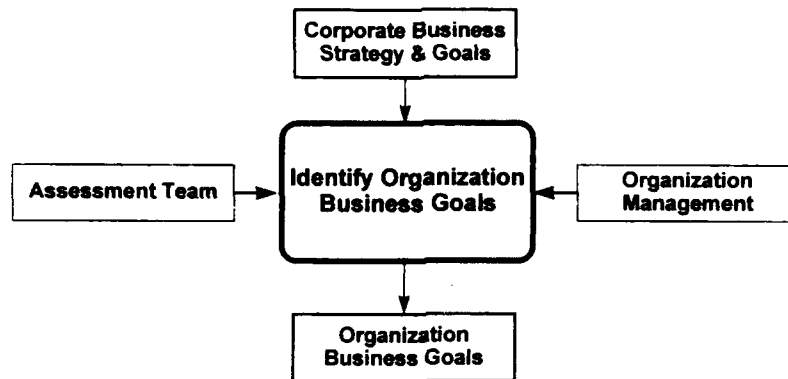
The Boeing Company

John D. Vu



10

Review Business Strategy



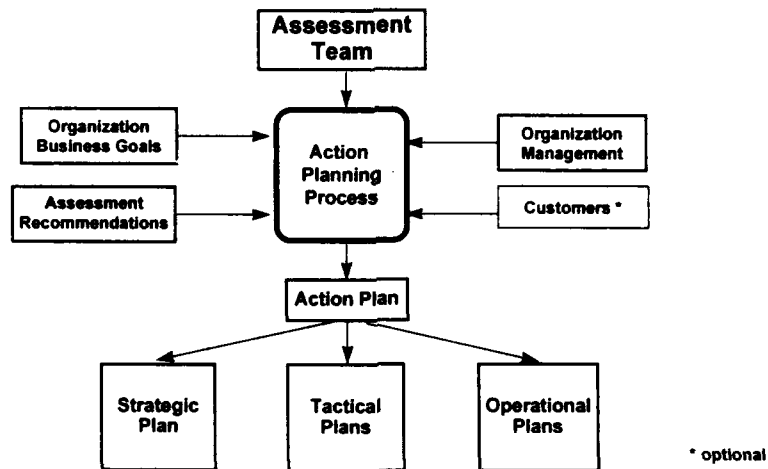
Capability Maturity Level should not be the only business goal

The Boeing Company

John D. Vu

11

Develop Process Improvement Action Plan



The Boeing Company

John D. Vu

12

Action Plan - Overview

1) Strategic plan (3-5 year plan):

- Defines goals & objectives for process improvement
- Defines the infrastructure for process improvement
- Establishes the master schedule

2) Tactical plan (1 year plan):

- Identifies the action items to be deployed in a Key Process Area
- Establishes schedule for a Key Process Area
- Documents the lessons learned for a Key Process Area

3) Operational plan (1- 3 month plan):

- Describes the new practice or technology to be developed
- Identifies deliverable(s)
- Identifies education & training requirements
- Identifies measurement(s)
- Establishes the project schedule

The Boeing Company

John D. Vu

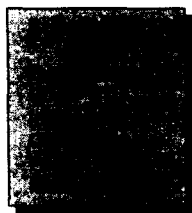


13

Action Plan Templates

An Action Plan is a formal, written response to the assessment and the "Roadmap" for improvement.

An Action Plan is a set of plans consisting of a high level strategic plan, a tactical plan for each KPA, and an operational plan for each action task.



Tactical Plan # 1

- Purpose
- Goals
- Description
- List of tasks
- Schedule
- Measurement & Metrics
- Lesson learned

Operational Plan # 1

- Scope
- Description
- Deliverable
- Training
- Schedule
- Metrics

The Boeing Company

John D. Vu



14

Example Of A Typical Strategic Plan

Table of Content

- 1.0 Organization commitment
- 2.0 Organization responsibilities
- 3.0 Scope
- 4.0 Vision
- 5.0 Mission
- 6.0 Values
- 7.0 Purpose
- 8.0 Goals & Objectives
- 9.0 Approach
- 10.0 Charter
- 11.0 SEPG members list
- 12.0 Other related documents (Policies / directions)

The Boeing Company

John D. Yu



15

Example Of A Typical Strategic Plan - 1

1.0 Organization Commitment:

The ABC organization formally commits to continuous process improvement using the Software Engineering Institute's (SEI) Capability Maturity Model (CMM) and all relevant key process areas. Process improvements shall be verified through a formal assessment process (i.e. CBA/IPI) and the review of improvement data by management.

2.0 Responsibilities:

The responsibility for administering the process improvement resides with the ABC organization's Software Engineering Process Group (SEPG), which is hereby empowered to ensure that the commitments above are implemented. The SEPG also is responsible for establishing the long term goals for process improvement in alignment with the company business goals and objectives.

The Boeing Company

John D. Yu



16

Example Of A Typical Strategic Plan - 2

3.0 Scope:

The following business / applications / units / programs / projects have been subject to an assessment and will be held accountable for implementing process improvement according to this plan

4.0 Vision:

To be a process managed organization that is highly valued by our customers

5.0 Mission:

To become a process managed organization by continuously improving all of our processes to satisfy our customers

The Boeing Company

John D. Vu



17

Example Of A Typical Strategic Plan - 3

6.0 Values:

People: We will improve our people skills and knowledge, treat them fairly, and pro-actively recognize their contribution to the company

Teamwork: We build trust and teamwork with open, candid communication throughout the organization. We will share technologies, best practices, and team with our suppliers and customers

Performance: We encourage high expectations and strive to be the best support and services organization to our customers

Innovation: We accept change as the rule, not the exception, and drive it by encouraging creativity and striving for technical leadership

7.0 Purpose:

We are making these improvements to strengthen our position in a global marketplace that increasingly values quality ... "

The Boeing Company

John D. Vu



18

Example Of A Typical Strategic Plan - 4

8.0 Goals & Objectives:

- Improve productivity by 35%
- Reduce cycle time by 25%
- Decrease service cost by 10%
- Reduce software defect by 40%
- Increase pre-release defect detection by 20%
- Increase test coverage by 45% by the end of next fiscal year
- Increase our organization's software capability maturity to level 3

9.0 Approach:

We intend to accomplish this incrementally, with an average of two specific major improvement efforts per fiscal year.

We will spend approximately 5% of our annual budget on improvement efforts.

The Boeing Company _____

John D. Vu



19

Example Of A Typical Strategic Plan - 5

10.0 Charter of the SEPG

(Insert the charter of the SEPG)

11.0 SEPG Membership List

- 11.1 Full time
- 11.2 Part time
- 11.3 Team member selection criteria
- 11.4 Rotation schedule

12.0 Related Documents

- 12.1 Company Policies/Directions
- 12.2 Other initiatives related to process improvement
- 12.3 Other SEPGs documents

The Boeing Company _____

John D. Vu



20

Example of a Typical SPI Infrastructure



Steering Committee
(Sponsor, Managers, Customers)

Set policy
Provide direction
Provide resources
Promote improvement



Software Engineering Process Group
(Project Managers, Senior Engineers)

Coordinate SPI activities
Track progress
Collect "Best practices"
Measure improvement



Process Improvement Working Groups
(Engineers, Practitioners, etc.)

Implement action tasks
Measure changes
Report progress

The Boeing Company

John D. Vu



21

Example of a Typical Master Schedule

Priority	Tactical plan	Year # 1	Year # 2	Year # 3	Year # 4	Year # 5
1	Plan # 1	▼	▼			
2	Plan # 2		▼	▼		
3	Plan # 3		▼	▼	▼	
4	Plan # 4			▼	▼	
5	Plan # 5			▼	▼	▼

The strategic master schedule is rolled up from a tactical schedule

The Boeing Company

John D. Vu



22

Example Of A Typical Tactical Plan

Table of Content

- 1.0 Purpose
- 2.0 Goals
- 3.0 Measurements & Metrics
- 4.0 List of Tasks
- 5.0 Status & Monitoring Mechanism
- 6.0 Roles & Responsibilities
- 7.0 Deployment Approach
- 8.0 Schedule
- 9.0 Lessons Learned



The Boeing Company

John D. Vu



23

Example Of A Typical Tactical Plan - 1

1.0 Purpose:

To establish a SQA process to review and inspect software products and processes to ensure compliance

2.0 Goals & Objectives:

Goals: (Insert the SQA goals from CMM)

Objectives: Reduce non-compliance issues 25% by 199X

3.0 Measurements & Metrics:

Number of SQA review

Number of defects identified during SQA review

Number of action items resulting from SQA review etc.

The Boeing Company

John D. Vu



24

Example Of A Typical Tactical Plan - 2

4.0 List of tasks:

- 1) Define a SQA review process
- 2) Define a SQA checklist
- 3) Revise current SQA procedure in OSSP
- 4) Provide training for new SQA personnel
- 5) Define a SQA plan template
- 6) Define criteria to conduct a quality probe audit

5.0 Status & Reporting Mechanism:

These tasks will be tracked monthly (status on milestone or deployment phase completion) by organizational management and SEPG.

6.0 Roles & Responsibilities:

The process owner of this plan is XXXX, who is responsible for the deployment of all activities within this plan. He (she) will coordinate with the deployment projects for status, coordinate the deployment activities, and recommend them for institutionalization across the organization

The Boeing Company

John D. Vu

25

Example Of A Typical Tactical Plan - 3

7.0 Deployment approach:

All activities within this plan shall follow the deployment process
(Define, Pilot, Refine, Institutionalize)

8.0 Schedule:

Tactical Schedule & Milestones - rolled up from operational schedule

9.0 Lessons learned:

Will be added at completion of plan
(Compilation of lessons learned in deployment project)

The Boeing Company

John D. Vu

26

Example Of A Typical Tactical Plan Schedule

Priority	Action tasks	Month # 1	Month # 2	Month # 3	Month # 3	Month # 4	Month # n
1	Task # 1	▼	_____	▼			
2	Task # 2	▼	_____	▼			
3	Task # 3		▼	_____	▼		
4	Task # 4			▼	_____	▼	
5	Task # 5			▼	_____	▼	
6	Task # 6			▼	_____	▼	
7	Task # 7				▼	_____	▼

A tactical master schedule is rolled up from an operational schedule

The Boeing Company _____

John D. Vu



27

Example Of A Typical Operational Plan

Table of Content

- 1.0 Statement Of Work / Requirements
- 2.0 Goals & Objectives
- 3.0 Task Breakdown Structures
- 4.0 Deliverables
- 5.0 Measurements & Metrics
- 6.0 Status reporting
- 7.0 Deployment approach
- 8.0 Schedule
- 9.0 Project Lessons Learned

Operation plan is a project plan of an action task

The Boeing Company _____

John D. Vu



28

Example Of A Typical Operational Plan - 1

1.0 Statement Of Work:

To establish a process to review and inspect software development to reduce product defects

2.0 Goals & Objectives:

Reduce post-released defect 20% in a year

3.0 Task Breakdown Structure:

Define a software review process (est. 40 hrs)
 Pilot review process in 3 projects (est. 120 hrs)
 Revise review process based on pilot results (est. 40 hrs)
 Provide training on new review process (est. 120 hrs)
 Institutionalize review process across organization (est. 3 mo)

4.0 Task Deliverables:

Guideline to conduct software review
 Software review checklist
 Training material for software review

The Boeing Company

John D. Vu



29

Example Of A Typical Operational Plan - 2

5.0 Measurements & Metrics

Number of defects / action items identified during reviews

6.0 Status & Reporting Mechanism

These milestones will be tracked monthly by project manager ...

7.0 Deployment approach

All activities within this plan shall follow the deployment phases (Define, Pilot, Refine, Institutionalize) each phase shall not last more than 3 months

8.0 Schedule:

Schedule & Milestones chart

9.0 Lessons learned:

Will be added at completion of project

The Boeing Company

John D. Vu



30

How Realistic Is One Year?

Constraints:

- Availability of resources for process improvement
- Availability of the right skills.
 - Process engineers are scarce
 - Good planners / estimators are scarce
 - External consultants are expensive
- Training

There are many "low hanging fruit" or instant solutions to be implemented in the plan.

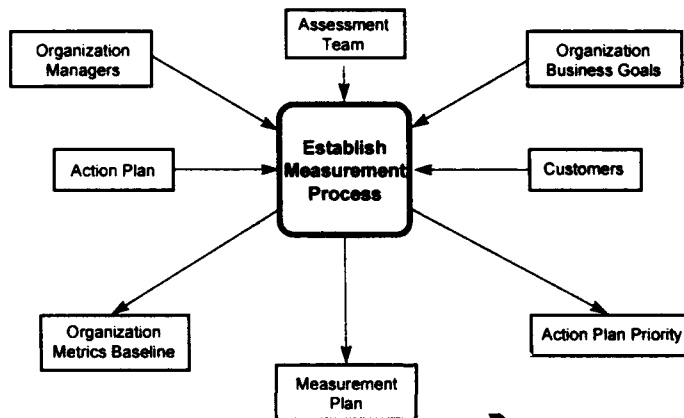
*Plans should be aggressive but achievable.
Get started and keep the pressure on.*

The Boeing Company

John D. Vu

31

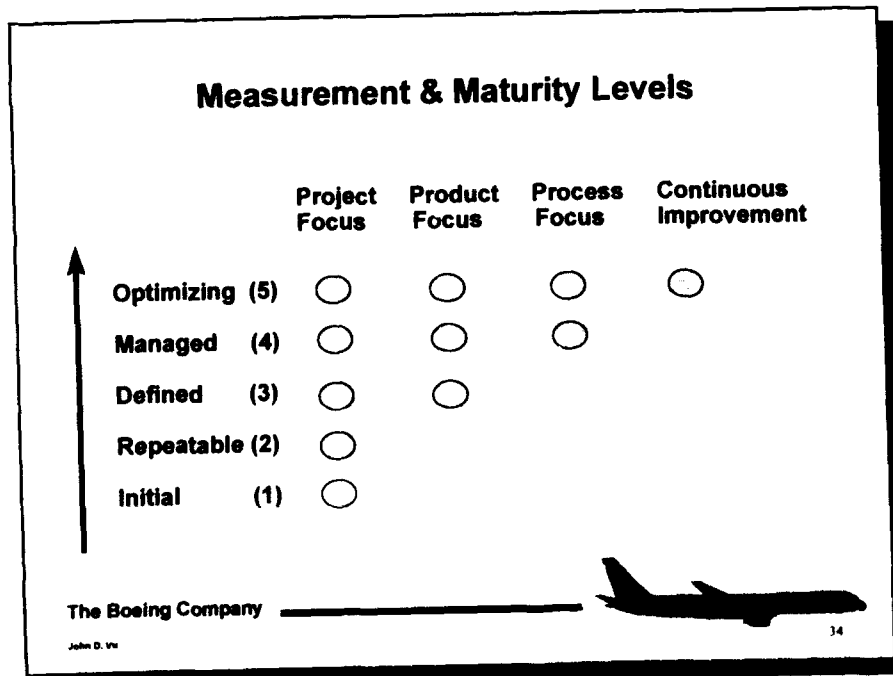
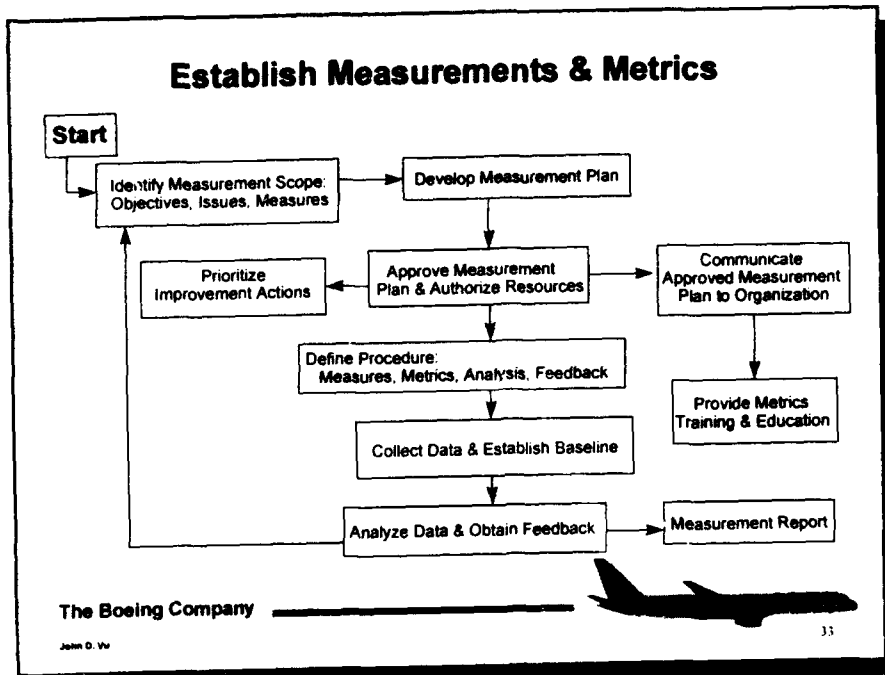
Establish Baseline Measurement



The Boeing Company

John D. Vu

32



Example Of A Typical Metric Definitions

Defect rate = Defects / product size

Cycle time = Elapse time to complete a process

Efficiency = Product size / work effort

Cost performance = Total project cost / product size

Schedule = Total on time schedule / Total schedule



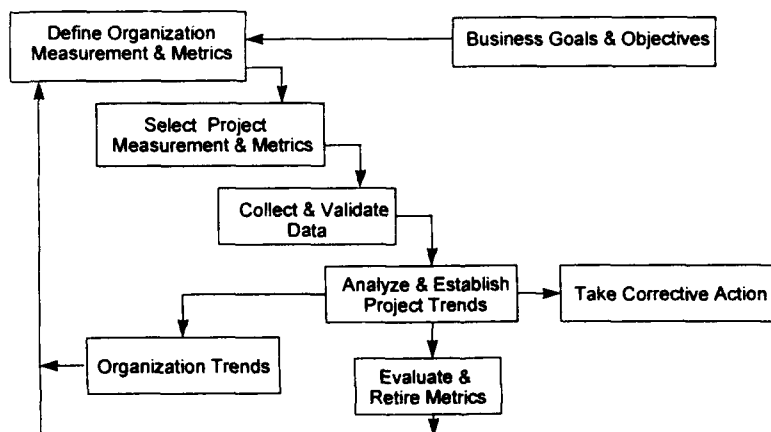
The Boeing Company

John D. Vu



35

Metrics Life Cycle



The Boeing Company

John D. Vu



36

Prioritize Improvement Action

Select improvement task where the managers and users can be expected to cooperate in the improvement effort.

Balances projects in the Tactical Plan so that at least half:

- Have the potential to realize significant measurable improvements
- Involve most practitioners in organization
- Involve collaboration between different parts of the organization

Customer participation is essential.



The Boeing Company

John D. Vu



37

A Common Mistake

Task: Develop a well-defined, standardized process for X.

Notion: A detailed process, when followed, will result in quality product.

Problem: At level 1:

- There is little time to adhere to a defined process.
- A well defined process is too complex and provides too much information to be absorbed in a chaotic environment
- Standardizing without piloting violates deployment rules and results in a sub-optimized solution

Result: The standardized process is often ignored.

The Boeing Company

John D. Vu



38

Lessons Learned

Always use a pre-defined template for an action plan.

Translating assessment recommendations into a set of small tasks, with each of the tasks not taking more than 3 months to complete.

Use Work Breakdown Structure to breakdown tasks.

Action planning activities should NOT take more than 1- 2 weeks.

Establish Measurement & Metric baseline ASAP.

Focus on action.

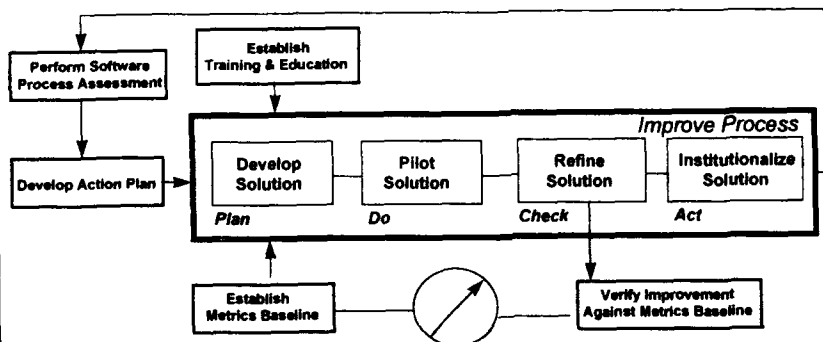
Action plan should be updated / revised when necessary.

The Boeing Company

John D. Vu

39

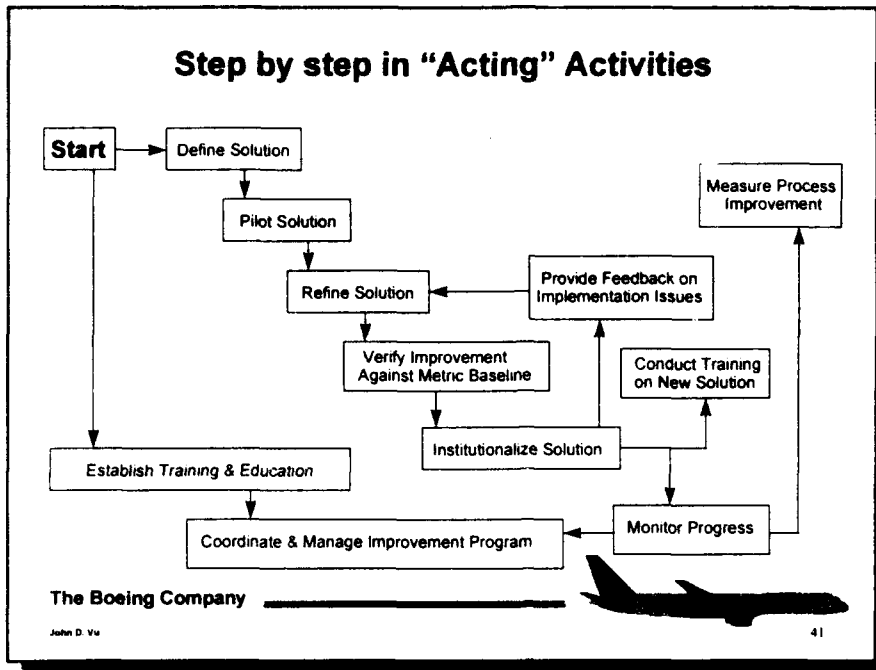
The "Acting" Phase



The Boeing Company

John D. Vu

40



Deployment Rules

- All tasks must follow deployment phase (Define, Pilot, Refine, Institutionalize).**
- Each phase must be completed within 3 months.**
- All tasks must have deliverable(s).**
- All tasks must be implemented as a project with project plan (Operational plan).**
- All tasks must have roles and responsibilities clearly defined in the plan.**
- All tasks are subject to review and monitoring.**
- No more than 2 to 3 tasks should be deployed at the same time.**

The Boeing Company 

John D. Vu 42

Some Questions To Consider

- 1) What problem is this improvement task intended to solve?
- 2) How does this improvement task solve the problem?
- 3) What are the measurable benefits of this improvement task?
- 4) What are the costs of developing this improvement task?
- 5) How similar or different is this task from the current process?
- 6) Is this the right time to implement this task? Can it be used now?
- 7) What are the possible consequences of using the result of this task?

The Boeing Company _____

John D. Vu



43

Common Errors

Selecting a task no one is interested in.

Selecting a desired solution instead of a process.

Selecting a process in transition.

Selecting a system to study, instead of a process.

Selecting a task not based on assessment findings.

Lessons learned:

The First 3 months: Form working group or process action team

The Next 3 months: Decide what to do

The 3 months after: Argue on deliverables

The last 3 months: Cancel project

The Boeing Company _____

John D. Vu



44

Improper Expectations

Time frame

- Don't expect "instant" improvement
- Expect solid measurable results in 2 - 3 years, not sooner
- Some expectation can still be met by "low hanging fruit"

Effort

- Realize the improvement working team needs time for learning
- Don't overload the improvement working team
- Everybody must participate in the improvement working team

Bad assumptions

- Our existing improvement efforts are the right ones & have priority since they are already started
- Improvement is easy once we have a action plan

The Boeing Company

John D. Vu



45

Institutionalization

"The building of infrastructure and culture that support methods, practices, and procedures so that they are the on-going way of doing business, even after those who originally defined them are gone".

"That's the way we do thing around here".

A Process must be:

Defined
Documented
Practiced
Measured
Verified
Maintained
Continuously Improved

The Boeing Company

John D. Vu



46

How To Document Your Processes ?

- Document only processes that help solve key problem**
- Resist the temptation to document everything**
- Only document high level overview processes ("WHAT" not "HOW")**
- Large documentation effort overwhelms organization**
- Limit document to a few pages**
- Increasing level of details only when needed**
- Processes must be used and measured**
- Promote the evolution of useful processes**

The Boeing Company

John D. Vu



47

Lessons Learned - 1

Not all deployments are successful but valuable lessons can be learned:

- **Learn by doing - start with something simple to gain momentum**
- **Do not take on too much too soon - start with a few improvements**
- **Practice skills on yourself first**
- **Don't do everything yourself - share your work but learned how to do it**
- **Do not develop an "ivory tower" with a selected elite improvement team. The team must consist of people from projects since everybody must participate and "buy-in"**

The Boeing Company

John D. Vu



48

Lessons Learned - 2

Software process improvement provides measurable return on investment - only when measured

Typical return on investment is between 5:1 to 8:1

Improvement means different things to different organizations

**What are your business goals ?
How do you measure progress ?
Have you looked at the bottom line ?**

Software process improvement is a long-term effort:

**2 to 4 years to go from level 1 to 2
2 to 3 years to go from level 2 to 3**

The Boeing Company



John D. Vu

49

Lessons Learned - 3

Most process improvement efforts failed because:

- **Lack of Commitment from management**
- **Lack of skills to do process improvement**
- **Lack of understanding of the CMM**
- **Form "committee" instead of "working group"**
- **Lack of monitoring mechanisms and metrics**



The Boeing Company

John D. Vu

50

To Ensure Success - 1

- 1) Keep solution simple, small steps at a time**
- 2) Focus on assessment findings**
- 3) Process Improvement is the goal, not the maturity level**
- 4) Management must support improvement**
- 5) Manage process improvement as a project**



The Boeing Company

John D. Vu



51

To Ensure Success - 2

During process improvement managers must:

- **Meet regularly with working groups**
- **Insist that they follow deployment rules**
- **Insist that they follow process improvement phases (Define, Pilot, Refine, Institutionalize)**
- **Track progress and report status**
- **Not hurry trying to automate the improved process**

Communicate, Communicate, Communicate



The Boeing Company

John D. Vu



52

To Ensure Success - 3

For every tool introduced:

- 1) The problem must be understood
- 2) A solution must be developed (Method / Process)
- 3) The solution must be practiced
- 4) The solution must be measured, and then
- 5) Opportunities must be identified for automation

(Jumping to step 5 will delay steps 1,2,3,4)

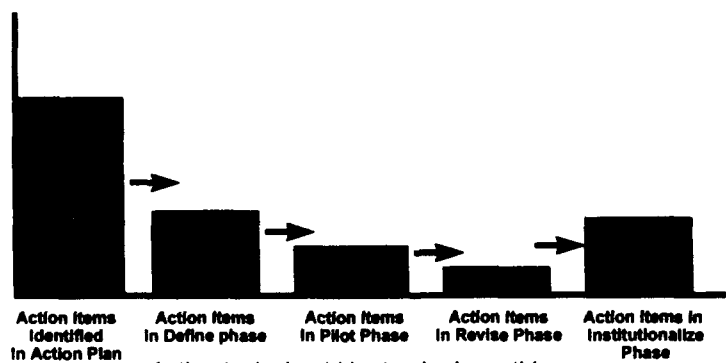


The Boeing Company

John D. Ve

53

Track & Monitor Improvement Actions



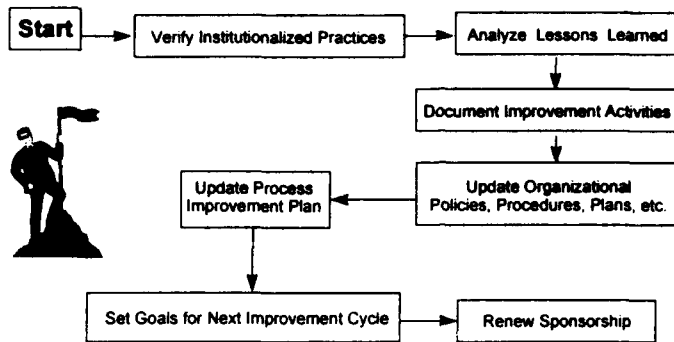
Action task should be tracked monthly

The Boeing Company

John D. Ve

54

Step by step in "Leveraging" Phase



The Boeing Company

John D. Vu

55

Lessons Learned

Improvement data needed to be analyzed for improvement benefit and help set goals for next improvement cycle.

Organization documents need to be updated when new practices / process has been institutionalized.

Lessons learned need to be documented in tactical plan and shared across organization.

Nothing more convincing than having the right data.

Celebrate success and share lessons learned.

Communicate, communicate, communicate.

The Boeing Company

John D. Vu

56

Cost of Implementation Failure *

Each time an improvement effort fails to achieve its stated objectives, it incurs both short-term and long-term costs

	Short Term	Long Term
Direct	Wasted resources: • Money • Time • People Business goal is not achieved	Business strategies not accomplished
Indirect	• Morale suffers • Job security threatened	• Lower confidence in leadership • Resistance to change increase • Next change more likely to fail

* Adapted from IMA material

The Boeing Company

John D. Vu



57

Time Required



Initiating: Plan for process improvement efforts (1 to 2 weeks)

Diagnosing: Identify current software processes maturity, strengths and weaknesses, and establish baseline for process improvement (1 to 2 weeks)

Establishing: Translate assessment findings and recommendations into an action plan for process improvement (1 to 2 weeks)

Acting: Execute, monitor, and evaluate improvement activities as identified in action plan (12 to 16 months prior to re-assessment)

Leveraging: Revise plan, document lessons learned, measure improvements, renew sponsorship, institutionalize practices (2 to 4 weeks)

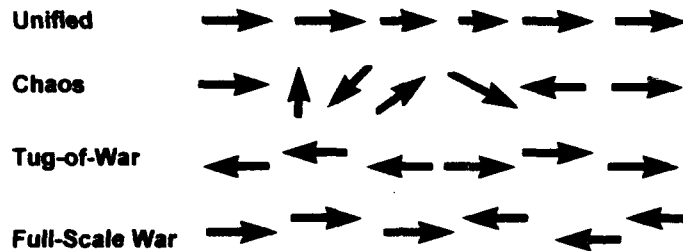
The Boeing Company

John D. Vu



58

Is Your Organization Ready To Change ?



If your organization is not ready to change, do not assess

The Boeing Company

John D. Ve



59

Key Success Factors

- Management Commitment
- Resources for Process Improvement
- Ability, Skills, Knowledge
- Measurements and Metrics
- Monitoring Mechanism
- Training (formal and informal)
- Customer Participation

The Boeing Company

John D. Ve



60

Questions & Answers

The Boeing Company

John D. Va



61

Conclusion

Make it happen!

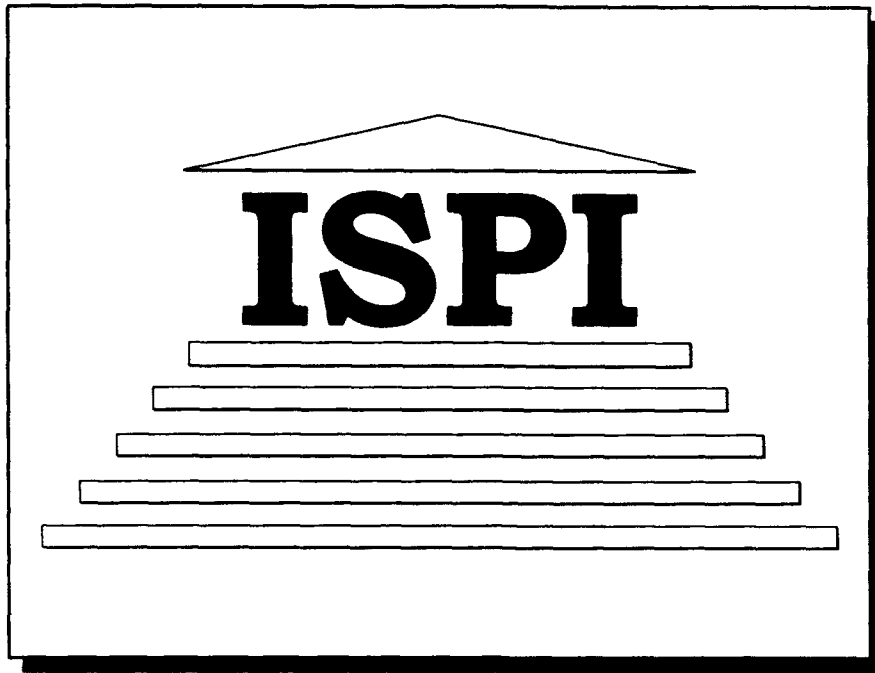


The Boeing Company

John D. Va



62





Level 2 Project Management

**European
Software Engineering
Process Group
Conference**

Amsterdam - June 17 1997

**Tim Kasse & Peter Leeson
Institute for Software Process Improvement Inc.**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 2



Agenda

- Who is ISPI?
- Introduction
- Accepting the Project
- Planning the Project
- Managing the Project
- Closing the Project
- Conclusions

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 3



ISPI Background

Institute for Software Process Improvement Inc. (ISPI)

- Founded in 1991 by Tim Kasse and Jeff Perdue
- Incorporated in 1996

Spin-off of the Software Engineering Institute's Process Program

ISPI is an international, full service, process improvement consulting company, assisting organizations in implementing process improvements that support their Business Objectives

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 4



ISPI Background - 2

ISPI's process improvement consulting services include:

- Process improvement implementation support
- Action planning guidance and support
- Process improvement related training
- Assessments and Evaluations
- Process improvement awareness and expectation setting

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 5



ISPI's Ongoing Relationship with the SEI

- Ten members of the ISPI team are authorized to conduct CBA IPis
- ISPI has a CRADA to teach the SEI's Introduction to the CMM course world wide
- ISPI is one of two vendors to have a CRADA with the SEI to teach the Software Capability Evaluation course
- Six members of the ISPI team have completed SCE Lead Evaluator training at the SEI
- ISPI works with and advises the SEI on improvements to the CMM through Jeff Perdue, who is a CMM Advisory Board member
- Jeff Perdue is a part-time Visiting Scientist at the SEI to help develop CMM version 2.0

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 6



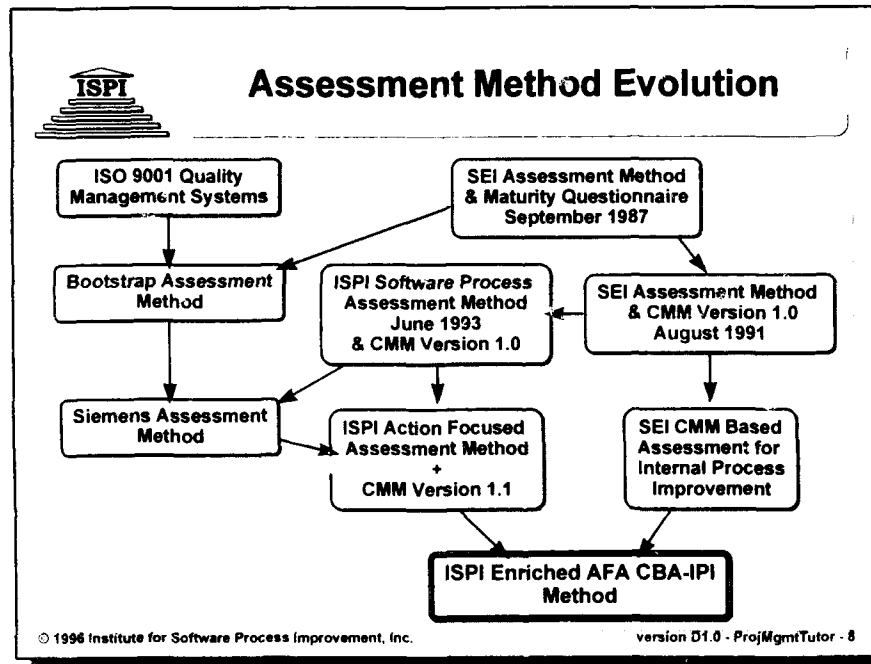
ISPI Clients

- AEG
- Alcatel Alsthom
- American Management Systems
- Bell South
- Carnegie Mellon Research Institute
- Center for Information Management
- Citibank International
- Digital Equipment Corporation
- Eastman Kodak
- EDS UK
- Ericsson
- Hewlett Packard
- Iberdrola
- Iberia Airlines
- IBM Spain
- ING Groep

- Israel Aircraft Industries
- Liberty Mutual
- Martin Marietta
- Microtec Research Inc.
- Motorola
- Naval Air Warfare Center (China Lake)
- Naval Air Warfare Center (Point Mugu)
- Nokia
- NYNEX
- Oerlikon Aerospace
- SEMATECH
- Siemens AG
- Software Engineering Institute
- Software Technology Inc.
- Sterling Software
- Thomson CSF

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 7





Agenda

- Who is ISPI?
- **Introduction**
- Accepting the Project
- Planning the Project
- Managing the Project
- Closing the Project
- Conclusions

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 9



What is a Project?

A set of related activities intended to accomplish specific objectives for a given customer

A one-shot, time limited, goal-directed, major undertaking, requiring the commitment of varied skills and resources.

> Project Management Institute

Examples:

- Write a Requirements Specification
- Develop and install a complete Traffic Control System for Alcatraz

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 10



What is project management?

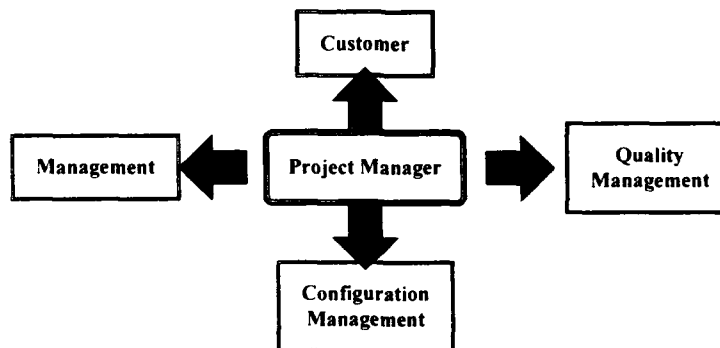
Establishing and maintaining an environment that gets the work done.

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 11

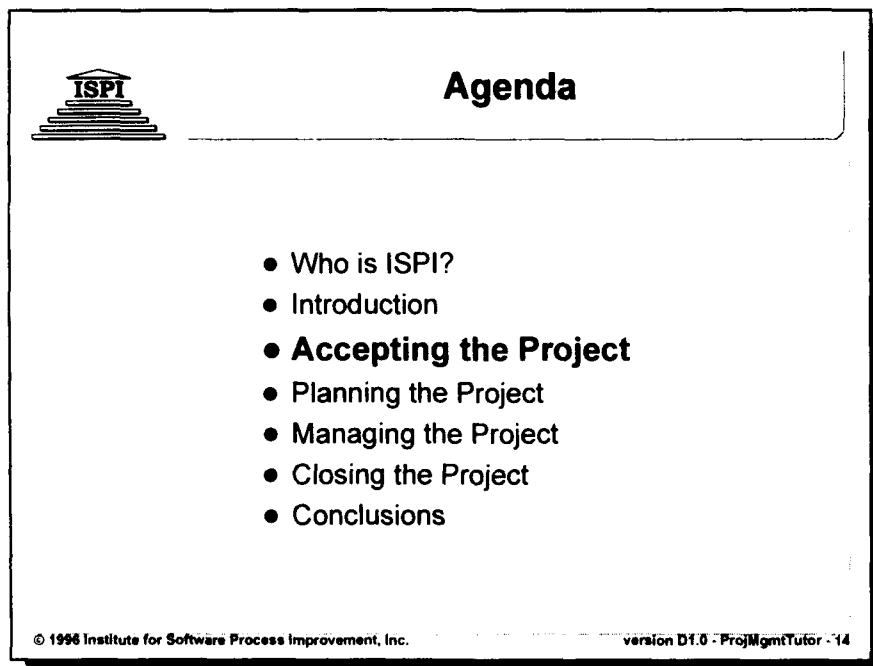
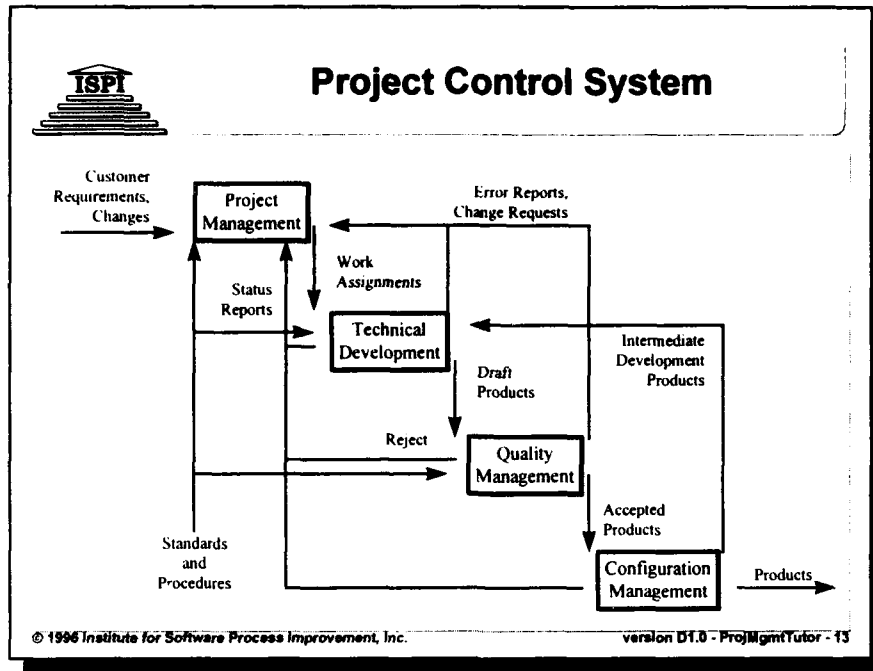


Project Interfaces



© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 12





Accepting the Project

Defining the Customer

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 15



Who are the Customers?

- Internal
 - Marketing
 - Product Manager
 - Systems
 - Testing
 - Sales
 - Another development unit in a prime contractor role
- External
 - Customer/Purchaser
 - End user
- Usually there are multiple “customers”
- Sometimes it is unclear who the customer is

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 16



Accepting the Project

Defining the Requirements

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 17



Project Requirements Functions

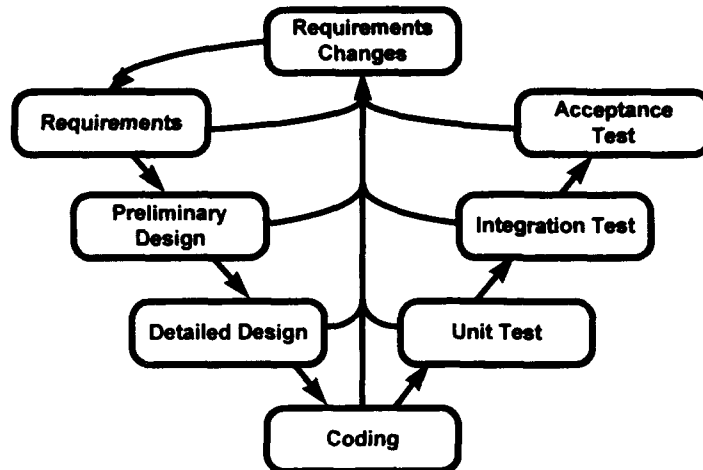
- Elicitation, analysis, and specification of requirements
- For requirements changes
 - Perform an impact analysis to resolve issues before the requirements changes are approved for implementation and scheduling
 - Change the following to remain consistent:
 - > Affected software plans
 - > Work products
 - > Project activities
- Verification and Validation
 - Verification - Are we doing the right job? Did we understand the requirements correctly?
 - Validation - Are we doing the job right? Does the product correspond to the requirements?

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 18



Requirements in the SW Development Life Cycle



© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 19



Purpose of Requirements Engineering

Produce requirements specifications with sufficient detail and clarity:

- So that the developed system meets the customer's needs
- So that it can be verified that the system meets the customer's needs

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 20



Systems and Software Requirements

- 1

First and foremost -- Requirements are Requirements

- That is, all requirements are indistinguishable (as to whether they are hardware, software, etc.) until systems analysis and design have been performed
- They should describe behavior and not implementation
- Most requirements can be satisfied with hardware, software, and/or peopleware -- this is a design decision

Software requirements are system requirements which have been allocated to software

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 21



Systems and Software Requirements

- 2

Often, the allocation -- which is a design decision -- will produce additional requirements, called derived requirements

- Interface requirements
- Cross-checking requirements - i.e., using software to check hardware functions
- Design decisions may require additional supporting requirements
- "Build versus Buy" -- choosing to use third-party software or standard components

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 22



Requirements Information Supplied by the Customer

- Systems Objectives and Overview
- Functionality
- Functional constraints
- Design constraints
- Data and communication protocols
- Project Management
- Priority
- Quality Assurance

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 23



Requirements Information Produced by Analysis

- System Model
- Feasibility
- Cost Benefit Analysis
- Risk

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 24



Information Supplied by the Organization or Project

Quality Factors

- Maintainability
- Reliability
- Portability
- Safety
- Usability

Performance requirements

Hardware requirements

Resource requirements

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 25



Accepting the Project

Defining the System

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 26



Systems Objectives

Reason for the system

- Why is it being built?
 - for Research and Development
 - for a specific customer
 - to meet existing market needs/requirements
 - to meet anticipated market needs/requirements
 - to drive the market
 - to upgrade an existing system to newer technology
- What is the problem it is trying to solve?
 - a known problem
 - an anticipated problem (e.g., loss of maintenance/production support for existing technology)

Describe the problems to be solved, not the solution

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 27



Systems Overview

The overview describes the interactions between the system and its environment

- Other systems with which it must interface
- Protocols it must use to interface with them
- Who its users are and in what manner they will use it (e.g., a single system may have several users it must support in different ways after it has been delivered: end users, maintainers, administrators, operators, auditors/reviewers...)
- Expected change in its environment over its lifetime

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 28



Functionality

Statements describing:

- Functional structure
- Externally observable behavior
- Internal behavior needed to support the external behavior (Derived Requirements)

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 29



Functional Behaviors

- Modes and controls
- Start-up/shut-down
- Normal mode operation
- Failure mode operation
- Recovery and fall-back
- Built-in tests

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 30



Functional Constraints

- Performance
- Efficiency
- Response times
- Capacities
- Safety
- Security
- Quality issues
- Reliability

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 31



Design Constraints

- Development standards
- Libraries
- Operating Environment:
 - System compatibility
 - Interaction with existing systems (hardware and/or software)
 - Hardware
 - Operating system
- Only include in requirements when essential to customer satisfaction

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 32



Accepting the Project

Defining the Project

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 33



Feasibility

Does a viable effective solution exist?

Types of feasibility

- **Technical** - software can be developed with current technology
- **Operational** - can the software be developed within environmental and/or performance constraints
- **Economic** - considers developmental costs, operational costs, and benefits
- **People** - availability of people with the necessary skills

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 34



Cost/Benefit Analysis

Analyze both quantifiable and non-quantifiable costs and benefits

For each alternative, the estimates include:

- Development costs
- Operating costs
- Estimated benefits

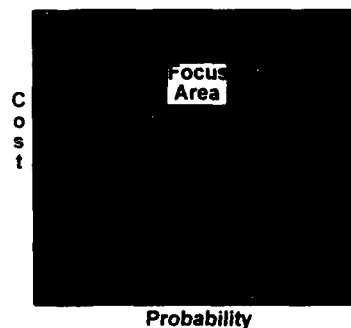
© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 35



Risk

- Potential risks should be identified and assessed
 - Cost, schedule, technical, dependency on outside factors
 - Assessed for level of importance and likelihood of occurrence
- Managing risk
 - Focus on higher importance and most likelihood of occurrence
 - Reduce the likelihood of occurrence, or reduce importance (e.g., with redundancy, fall-back, back-up)
 - Create contingency plans to minimize the damage



© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 36



Agenda

- Who is ISPI?
- Introduction
- Accepting the Project
- **Planning the Project**
- Managing the Project
- Closing the Project
- Conclusions

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 37



Planning the Project

Project Roles that need to be filled

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 38



Project Manager Roles

Task

- Goals
- Objectives
- Checkpoints
- Activities
- Relationships
- Time estimates
- Schedule

Relationship

- Direct
- Reinforce
- Inform
- Vitalize
- Empower
- Risk

Alan Randolph and Barry Posner,
Effective Project Planning and Management

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 39



Role of the Customer

The Customer is the Individual or Organization responsible for accepting the product and authorizing payment to the developing Organization.

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 40



Role of Customer Representative

- **Advantage of an internal customer representative:**
 - Can defend the interest of many customers (e.g., generic developments)
 - Necessary in an organization that has not learned to interact positively with the customer
- **Define and Approve Requirements**
- **Provide Support and Information**
- **Request Changes**
- **Accept the End Product**
- **Provide Feedback**
- **Participate in Project Reviews**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 41



Role of Quality Manager

Ensure the required level of quality is built into all product items by identifying and managing the risks to the project objectives:

- **Ensuring the requirements specification contains quality requirements**
- **Planning the quality activities**
- **Involvement with all phases and activities of project**
- **Measuring conformance to standards and procedures**
- **Report and seek resolution of issues at the appropriate level**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 42



Role of Configuration Management

Establish and maintain the integrity of the products of the software project throughout the project's software life cycle:

- **Identify Configuration Items**
- **Define Baselines**
- **Manage or control changes to Baselines and Configuration Items**
- **Provide status information about baselined configuration items**
- **Audit the existing system against the requirements**
- **Ensure Requirements Traceability**
- **Ensure the reliability of a milestone (in conjunction with quality manager)**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 43



Role of Management

Define long term strategic interests, and ensure that the projects being developed conform to them:

- **Provide policies**
- **Monitor overall status**
 - **Schedule**
 - **Costs**
 - **Risk**
- **Provide resources as appropriate**
- **Approve at critical points**
- **Ensure correct decisions are made**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 44



Project Factors

The project manager is responsible for planning and managing:

- Scope
- Resources
- Time
- Cost
- Risk
- Quality

© 1995 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 45

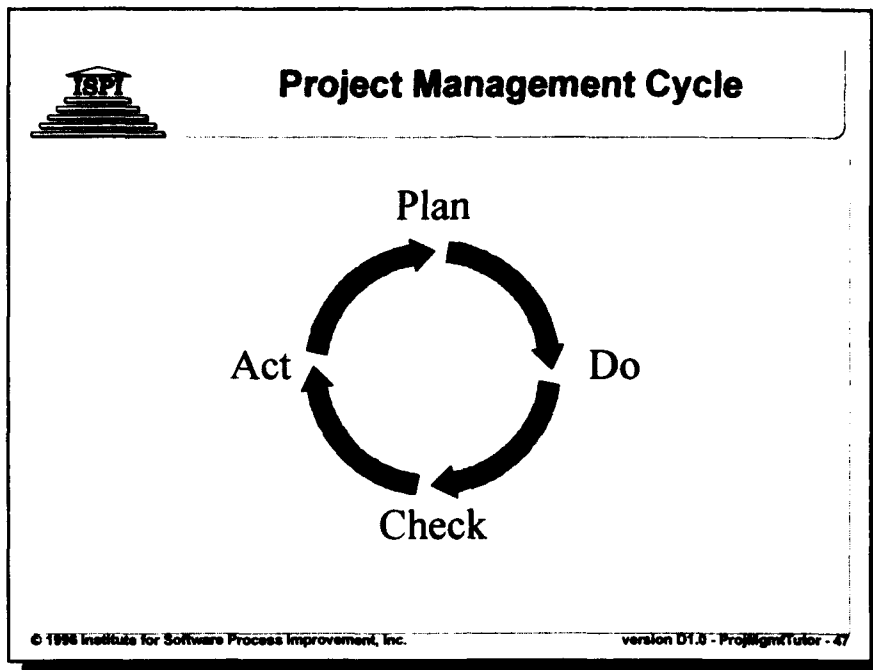


Planning the Project

The Project Management Lifecycle

© 1995 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 46



The slide is titled 'Why Plan?' and lists four reasons for planning. It includes the ISPI logo in the top left corner. The reasons are listed as bullet points. Below the list is a quote from Harold Kurzner. The slide also contains a copyright notice and a version number at the bottom.

ISPI

Why Plan?

- To eliminate or reduce redundancy
- To improve efficiency of the operation
- To obtain a better understanding of the objectives
- To provide a basis for monitoring and controlling work

– Harold Kurzner: Project Management: a Systems Approach to Planning, Scheduling, and Controlling

© 1998 Institute for Software Process Improvement, Inc. version D1.0 - ProjMgmtTutor - 48



Elements of Planning

Planning a project considers the following:

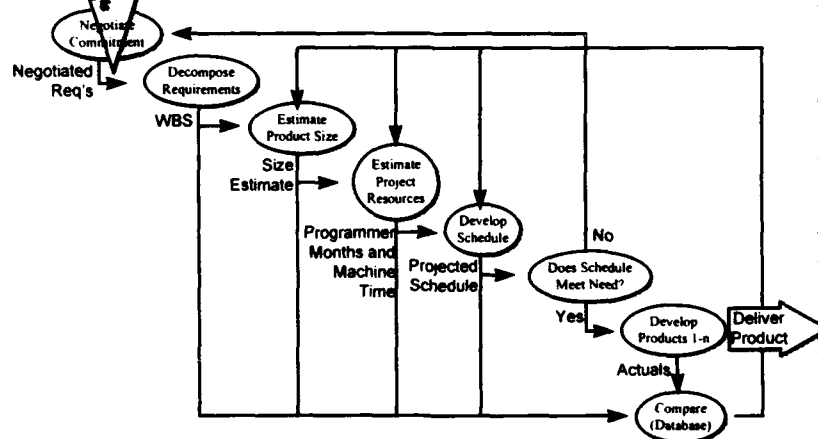
- Goals
- Work
- Resources
- Estimates
- Schedule
- Risk
- Budget

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 48



Software Development Planning Cycle



© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 50



Iterative Nature of Planning



© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 51



Planning the Project

Choosing a Software Development Lifecycle

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 52



Commonly Accepted Life Cycles

- Waterfall Model
- Overlapping Waterfall Model
- V-Model
- Spiral Model
- Evolutionary Model

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 53

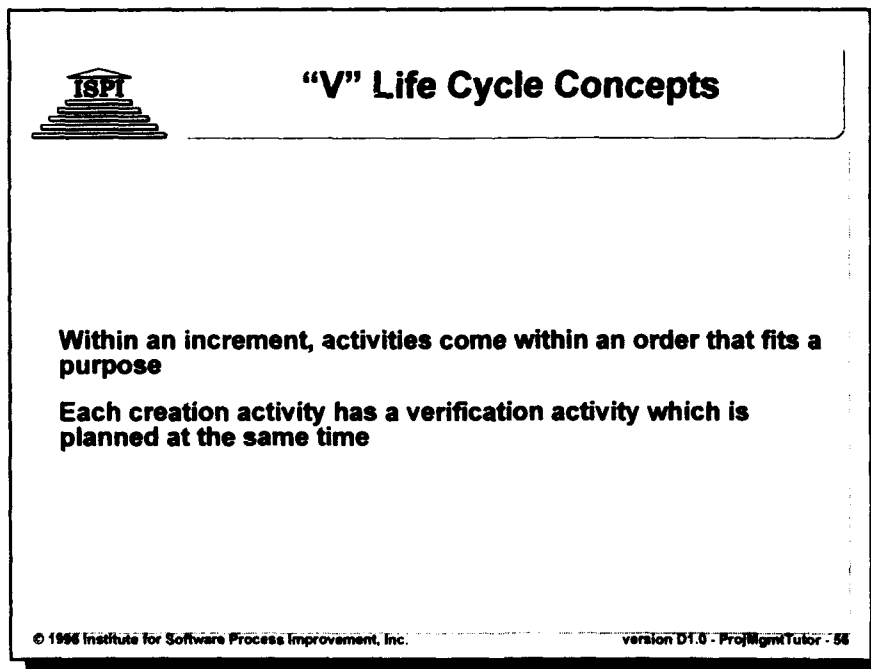
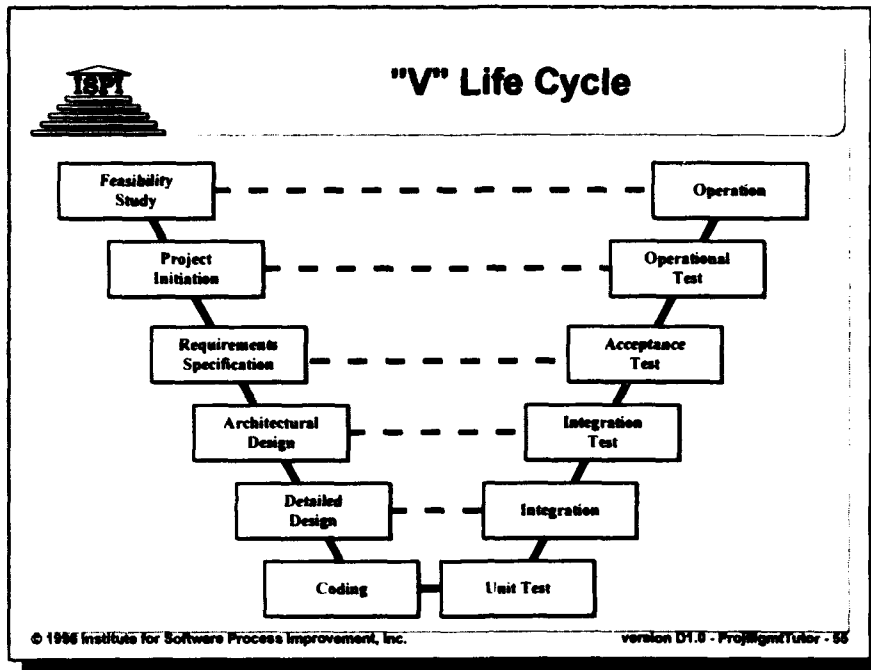


The "V" Life Cycle

**A sample traditional approach to Software
Development Lifecycles**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 54





Incremental Deliveries

Structuring the Product into a Logic Based Effort

A series of incremental internal product deliveries should be planned from the beginning

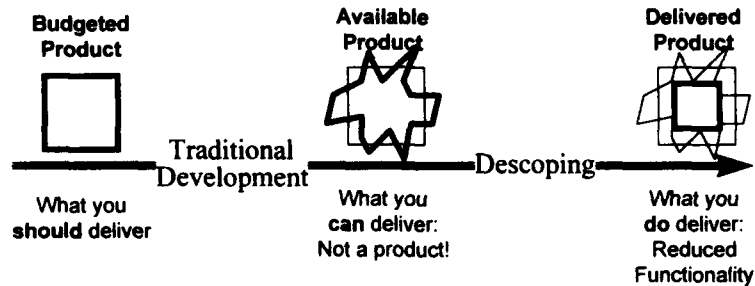
- Always have a working system
- Descoping plan as part of the project plan

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 57



Rationale for Incremental Development

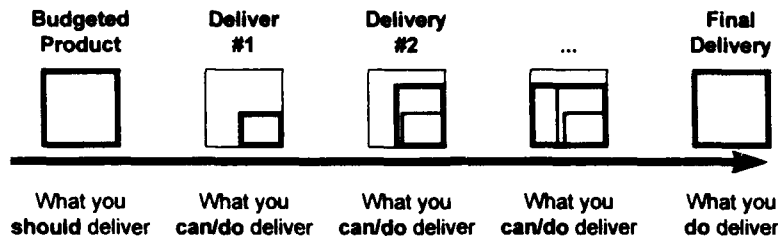


© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 58



Rationale for Incremental Development - 2



© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 55

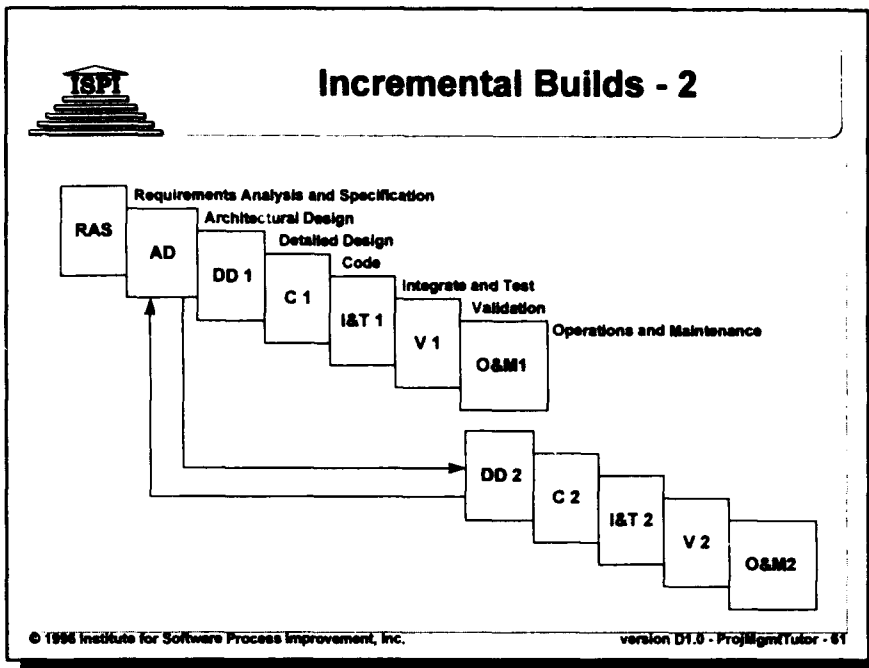


Incremental Builds - 1

- Strategy to deliver the functionality of a product through a planned set of deliveries, each containing increased functionality and capability
- Achieved by splitting the development cycle into a number of more manageable units, once the complete architectural design has been defined
 - Each unit consists of detailed design, coding, integration and testing, and validation phases for a defined set of functions
- In all cases, each deliverable should be usable, and provide a subset of the final required capabilities
- Increments should not exceed three to six months

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 56



ISPI

"V" and Incremental

Combination of the two:

- The "V" Life Cycle determines the roles and responsibilities (e.g., designers are to prepare test data)
- The Incremental Model allows to plan how the development of increments interact and prepare the planning and scheduling of the project
- Every individual delivery within the incremental development should be considered as a separate sub-project and go through the appropriate phases of the "V" life cycle

© 1996 Institute for Software Process Improvement, Inc. version D1.0 - ProjMgmtTutor - 62



Agenda

- Who is ISPI?
- Introduction
- Accepting the Project
- Planning the Project
- **Managing the Project**
- Closing the Project
- Conclusions

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 83



Project Management

- Tracking Actuals to Estimates
- Taking Corrective Action
- Historical Data Base
- Using Reviews for Project Control
- Configuration Management
- Quality Management

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 84



Project Tracking and Oversight

- The software development plan must be used as the basis for tracking the software activities and communicating status
- Actual results of all software projects should be tracked routinely against their planned estimates
- Corrective actions must be taken when actual results deviate significantly from estimates
- Changes to planned commitments must be understood and agreed to by all affected groups and individuals

[from SEI-93-TR-25,
"Key Practices of the Capability Maturity Model, Version 1.1"]

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 65



Tracking Actuals to Estimates - 1

- Size
- Effort and Cost
- Schedule
- Staffing Needs
- Computer Resources
- Software Risks
- Team Capability Measurement
 - Experience

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 66



Tracking Actuals to Estimates - 2

Technical Activity Actual to Estimates

- Software elements designed
- Software elements coded
- Software elements unit tested
- Software elements integrated
- Systems testing
- Test plans complete
- Test specifications complete
- Test suites developed
- Milestone reviews held

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 67



Tracking Actuals to Estimates - 3

- Yes, collect it all... but that is not practical is it?
- Key data:
 - Size Estimate and Measurement
 - > per Subsystem
 - > including a complexity factor
 - Effort Estimate and Measurement
 - > per Subsystem
 - > per Task
 - Elapsed Time Estimate and Measurement
 - > per Subsystem
 - Team Capability Estimate
 - Risks Planned and Tracked
 - Test plans, results, defects

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 68



Size Estimate and Measurement

- By Subsystem
 - Size can be estimated in SLOC, Function Points, Modules, Features
 - A choice needs to be made from
 - > the easier to measure (SLOC)
 - > the easier to estimate (Features)
 - A possible mid-point can be reached by allowing the estimates to progress from one point to the other as the project advances
- Including a complexity factor
 - This is a simple factor considering the probable or measure complexity of the modules or sub-systems.
 - E.g.: Range from A (Very Simple batch report of data) to D (Complex, highly computational, real-time interactive processing)

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 69



Effort and Time Estimate and Measurement

- per Subsystem
 - Include all the intermediate estimating rounds - some people are better at estimating than others, you should be able to recognize them later
- per Task
 - The measurement should be made per elementary task throughout the whole lifecycle.
 - This information will not only allow to refine the estimating process in the future, it will allow to determine the places where the process needs to be refined

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 70



Team Capability Estimate

- Based on the tasks that are being performed, we need to be able to determine the best people and the capability level of the people involved
- E.g.:
 - A: Has practical experience and theoretical knowledge of the tasks to be done
 - B: Has participated in similar projects
 - C: Has theoretical knowledge of the kind of problem
 - D: Has no preliminary experience or knowledge in this line
- Process has to be simple to apply and understand

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 71



Collection Process -1

- Process needs to be quick and easy
- Data has to be collected
 - at Estimation
 - Daily
 - Weekly
 - at Milestones
 - at Project Completion

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 72



Collection Process - 2

Quick and Easy

- The collection of data should not require any extra effort on behalf of the developers or the project leader:
 - If the collection of data requires time and effort during the development process, it will not be performed on a regular basis

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 73



Taking Corrective Action - 1

When actuals tracked against estimates indicate that established thresholds have been exceeded, corrective action must be taken, for example:

- Human resources reallocated
- Computer resources increased or decreased
- Estimates redone
- Features and Functionality reviewed
- Schedule modified
- Contingency plans for risk abatement implemented

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 74



Taking Corrective Action - 2

Significant corrective actions may require replanning

- **Software Development Plan must be revised**
- **All new software project commitments and changes to commitments must be documented**
- **All changes to commitments must be communicated to the members of the software development project and other groups that are supporting the software project (e.g., SQA, SCM, Test, Documentation, Data)**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 75



Updating the Historical Database

All data used in tracking actuals to estimates needs to be included in the historical data base

- **Presumptions and Assumptions**
- **Constraints**
- **Number of passes at estimating**
- **Basis of estimating**
- **Different results attained**
- **Identification of estimator**

This information needs to be stored immediately before biases based on actuals impact it

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 76



Reviews as a Management Control Tool

During any project, management requires a means of assessing and measuring progress

- Are we ahead or behind where we expected to be?
- Will we complete the work as planned?
- Do we require more computer or people resources in order to meet the planned schedule?
- Is the required functionality being implemented?
- What risks are we taking based on the project information that we have today?

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 77



Using Reviews to "test" or measure Software Quality - 1

"True" progress can not be measured by counting the completion of tasks unless there is a reliable way of measuring the quality of the work performed and knowing that it would not have to be redone or changed later (Rework!)

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 78



Using Reviews to "test" or measure Software Quality - 2

Reviews can be used to examine the quality of the work products throughout the software lifecycle that cannot be computer tested and improve those work products

Informal reviews

- Consider alternative implementations
- Exchange techniques and style variations
- Educate the participants
- Point out efficiency and readability problems or modularity problems

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 78



Using Reviews to "test" or measure Software Quality - 3

Formal reviews

- Verify that the software element(s) satisfies both its specification and preceding intermediate work products
- Verify that the software element(s) conforms to applicable standards
- Identify real or potential deviations from standards and specifications
- Collect engineering data (i.e., defect and effort data)

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 80



Management Reviews

A Management Review is a formal evaluation of a project level plan or project status relative to that plan by a designated review team.

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 81



Objectives of Management Reviews

- **Tracking progress according to plan, based on an evaluation of the product development status**
- **Changing project direction or identify the need for alternative planning**
- **Maintaining control of the project through adequate allocation and reallocation of resources**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 82



3 Types of Management Reviews -1

Project Manager Review

- Project Manager and development staff attend
- Representatives from Systems, SQA, SCM, Test are also in attendance
- Technical, cost, staffing, and schedule performance are reviewed against the software development plan
- Software risks are identified, prioritized, and contingency plans are discussed
- Necessary commitment changes are acknowledged, documented, and approved
- Planning documents are updated as necessary

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 83



3 Types of Management Reviews - 2

Milestone Reviews

- Formal milestone dates are documented in the software development plan, tracked and reviewed
- Customer's or End Users may be involved with the formal milestone reviews
- Typical milestone review content includes:
 - Technical progress
 - Commitments
 - Plans
 - Engineering activities
- Project and software risks are addressed
- Action items are recorded, assigned and reviewed
- All decisions are documented with corresponding actions and resolutions

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 84



3 Types of Management Reviews - 3

Senior Management Oversight Reviews

- Technical, cost, staffing, and schedule performance are reviewed against the software development plan
- Project's role in strategic plan is reviewed
- Software Risks are reviewed from the project and organizational point of view
- Quality of resulting life-cycle work products is reviewed
- Project process efficiency and effectiveness is reported
- Necessary commitment changes are acknowledged and approved
- Conflicts not resolved at lower levels are discussed and resolved

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 88



Reviews - A Measure of the Work Product Quality

It is necessary for each Project Leader to know at all times

- What is the project's progress against what was planned and committed?
- What is the quality of the work that has been performed?

Reviews on the software life-cycle work products indicate the quality of the evolving system each step of the way increasing dramatically the probability of the highest quality product being delivered

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 88



The Need for Configuration Management

Inadequate CM can be disastrous:

- The latest version of source code cannot be found
- A difficult bug that was fixed at great expense suddenly reappears
- A developed and tested feature is mysteriously missing
- A fully tested program suddenly does not work
- The wrong version of the code was tested
- No one knows which modules were delivered to the customer
- "Undocumented features" suddenly appear

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 87



The Purpose of CM

The purpose of CM is to establish and maintain the integrity of project work products throughout the project's life cycle
— from SEI-93-TR-25, "Key Practices of the Capability Maturity Model, Version 1.1"

CM manages change in order to answer the following questions:

- What is my current configuration?
- What is my status?
- How do I control changes to my configuration?
- What changes have been made to my product?
- Do anyone else's changes affect my work or product?

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 88



Expected Results of CM

- Control is maintained for all work products (including data and documentation), regardless of medium
- Change requests and problem reports undergo a specified review process
- Only authorized changes are implemented
- A history of changes and modification is maintained
- Interfaces are identified, documented and controlled
- Subcontractor impacts to the project configuration are managed
- All deliveries are of a known quality:
 - all configuration items have been properly identified and documented
 - the configuration items have been controlled by change management
 - the delivery configuration has been properly approved

© 1998 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 89



CM Principles - 1

CM is...

- The backbone of the development process. It helps ensure:
 - Product quality
 - Process improvement
- A process that manages product evolution throughout its life cycle
- A process to create a verifiable history of the product as it matures
- A means of communication:
 - for project members
 - for customers

© 1998 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 90



CM Principles - 2

CM involves...

- **Identifying the configuration of the product (i.e., selected work products and their descriptions) at given points in time**
- **Systematically controlling changes to the configuration**
- **Maintaining the integrity and traceability of the configuration throughout the life cycle**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 81



CM Principles - 3

CM provides visibility into the status of the evolving product:

- **What changes were made to the product?**
- **Who made the changes?**
- **When were the changes made?**
- **Why were the changes made?**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 92



CM Principles - 4

CM provides a common point of integration for all planning, oversight, and implementation activities

CM impacts all data and processes and spans all areas of the life cycle

CM needs to be applied to all developments:

- Software
- Hardware
- Documentation
- Validation
- ...



CCB

The Configuration Control Board (Change Control Board):

- Has the authority for *establishing* and *managing* the project's baselines
- Ensures that every change request is properly *considered* and *coordinated*
- Ensures that every release is built from *baselined components* according to approved component build lists

If an existing system-level CCB is not taking software into account, it needs to be expanded to include software configuration / change control



Baselines

A baseline is an approved snapshot of the system at points in the development lifecycle corresponding to the release plan and the version strategy

- Record of a contract
- Serves as the basis for further development
- Can be changed only through an agreed upon change procedure

A baseline could be

- A specification (i.e., requirements specification, design specification)
- A product that has been formally reviewed and agreed upon
- A partial system



Key CM Activities: Configuration Identification

Identifying the structure of the system

Identifying all related life cycle work products

Providing a unique identifier for each of those work products

Supporting traceability between the software and all other related products



Key CM Activities: Change Control

Who can initiate the change request?

The individuals, group, or groups who are responsible for evaluating, accepting, and tracking the change proposals for the various baselined products

The "Change Impact" analysis expected for each requested change

How the change history should be kept

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 87



Key CM Activities: Status Accounting

Maintaining a continuous record of the status and history of all baselined items and proposed changes to them

Providing traceability of all changes to the baseline throughout the life cycle

Answers the questions

- **What changes have been made to the system?**
- **What changes remain to be implemented?**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 88



Key CM Activities: Configuration Auditing

Configuration audit verifies that the product is built according to the requirements, standards, or contractual agreement

Verifies that all products have been produced, correctly identified and described, and that all change requests have been completely processed

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 99



CM support to the Project Leader

A strong understanding and implementation of CM helps the Project Leader

- **Control changes to the requirements**
- **Allow the project members to develop at a fast pace without interference during the early stages of development**
- **Control developers "improving" the code when it is at the infamous 90% complete stage**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor -
100



CM support to the Project Leader - 2

Assists the Project Leader in producing accurate and up to date Status Reports:

- Provides status reports to the Project Leader indicating what items are undergoing the most change in terms of number of changes and frequency of changes
- Traceability provides the Project Leader with a level of confidence that what the developers are developing is what is demanded by the requirements and nothing more
- Helps ensure the integrity and consistency of the evolving system so that the code and associated documentation and specifications are synchronized

Assists the Project Leader to develop in an iterative approach thereby reducing complexity and risk

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor
101



Basic Configuration Management

Even if an organization has little or no configuration management in place, some very simple steps will add a great deal of control and project tracking information

- Formalize the use of baselines
- Uniquely identify system components
- Establish a simple change control
- Produce Status Reports on
 - Configuration items
 - Change Requests
 - Problem Reports

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor
102



Summary: The CM Functions

What is the system configuration?	Identification	The system consists of the following baseline documents and products:...
How are changes to the configuration controlled?	Control	The steps to process changes are...
What changes have been made to the system?	Status Accounting	The system configuration and related changes at this line are the combination of the following baselines, changes, pending changes:...
Does the system satisfy the requirements?	Auditing	The system as currently built differs from the baselines and approved changes as follows:...

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 103



Quality Management

Quality Management includes all activities of the overall management functions that determine the quality policy, objectives, and responsibilities, and implements them by means such as:

- Quality Control
- Quality Assurance
- Quality Management
- Quality Functions
- Quality Planning
- The Role of Quality Assurance

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 104



Quality Control and Quality Assurance - 1

Quality Control evaluates the products

- Checks the quality of the product(s)
 - Is the product within tolerance?
 - Is the product (or lifecycle work product) of an acceptable quality?
- Tools and Techniques
 - Reviews
 - Tests
 - Inspections



Quality Control and Quality Assurance - 2

Quality Assurance evaluates the process

- Checks that the process is working
 - Is the process being followed?
 - Are the QC checks being applied?
 - Are the QC checks efficient?
 - Is the process causing quality problems?
 - Is the process working for the organization?
- Tools and Techniques
 - Audits
 - Assessments



Quality Management A Management Tool - 1

Quality Management is a management tool to ensure that the officially established process is actually being implemented. More specifically it ensures:

- An appropriate development methodology and lifecycle is being used
- The projects use standards and procedures in their work
- Independent reviews and audits are conducted
- Documentation is produced to support maintenance and enhancement
- Mechanisms are in place and used to control changes
- Testing emphasizes all of the high-risk product areas



Quality Management A Management Tool - 2

- Each software task is satisfactorily completed before the succeeding one is begun
- Deviations from standards and procedures are exposed as soon as possible
- The project is auditable by external professionals
- The quality control work is itself performed against established standards
- The SQA Plan and the Software Development Plan are compatible



Software Quality Functions - 1

Software Quality Functions include;

- Planning for quality
- Designing in quality factors (e.g., maintainability, reliability)
- Establishing the use of standards and procedures
- Reviews
- Testing
- Audits
- Setting quality goals
- Providing visibility into the process and product quality for management (Reporting)
- Getting non-compliance issues resolved before the product is delivered to the customer

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 109



Software Quality Functions - 2

These quality functions may be performed by:

- Project Leaders and software developers
- Quality Manager or Quality Representative
- Organizational level SQA Group
- Systems Engineering
- Independent Test
- Documentation
- Customer

and others.....

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 110



Software Quality Plan - 1

Given a Software Development Plan, a Software Quality Plan must describe:

- What quality functions will be performed?
- Who will perform them?
- During what phase of the software lifecycle will they be performed?
- Who has approval authority?
- How will conflicts over non-compliance be resolved?

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 111



Software Quality Plan - 2

Without a Software Quality Plan against which to measure, it is difficult to determine:

- If the product quality goals are being met
- If the process being followed is effective
- If the customer requirements will be met
- If additional management support is required

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 112



The Role of Quality Assurance

- Ensures that the right quality plan is developed to match the criticality of the lifecycle work products
- Ensures that the project is developed in compliance with the defined processes
- Ensures that the processes defined are adequate for the project
- Provides feedback to the project's management about the effectiveness of the processes the developers are following
- Provides feedback to the SEPG about the usability of their processes

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor
113



The Quality Role of Project Management

The Project Leader needs to...

- plan, organize, commit, run, deliver...
 - setting up a team
 - ensure the availability of the required resources at the right time
 - plan, manage, track risks that may affect the project
- a Product...
 - corresponding to the requirements agreed to
- in Time and Budget...
 - according to the plan that was set up and agreed to
- with the defined quality characteristics
 - i.e., meeting the requirements from all points of view

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor
114



Project Leader Expectations from SQA Representatives

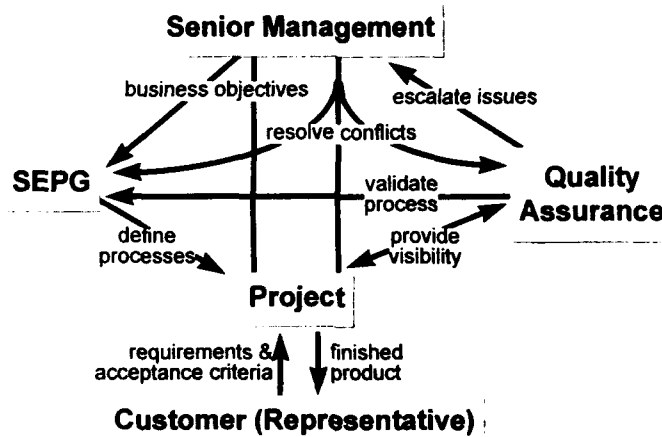
- Assistance in creating an executable and successful project plan
- Assistance in creating the project's software quality plan
- Assistance in choosing the right standards for the project's needs
- Assistance in tailoring the standards and processes for practical use by the project
- Assistance in setting up peer reviews for the software life-cycle work products
- Performing quality audits and traceability audits to ensure that the quality goals are being met and the system's integrity is maintained

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 115



Project Management Interaction



© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor - 116



Agenda

- Who is ISPI?
- Introduction
- Accepting the Project
- Planning the Project
- Managing the Project
- **Closing the Project**
- Conclusions

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor
117



Project Completion Review

- Topics
 - Events or Results
 - Team
 - Evaluation
 - > Accomplishments
 - > Improvements
- Attendees:
 - Customer (or Representative)
 - Sub-Contractors
 - Project Team
- Multiple Meetings

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor
118



Documenting the Review

- Introduction
 - Project Background
 - Primary Outcomes
 - Team Members
- Project Description
- Project History - Sequence of Events

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor -
119



Documenting the Review - 2

- Comparison of Actuals with Estimates
 - Overall Measures and Metrics
- Evaluation
 - Accomplishments Against Expected Results
- Risks, Problems, Cost, and Solutions
- Accomplishments and Recommendations

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor -
120



Evaluation

- **What:**
 - Processes
 - Tools
 - Assumptions
- **Recommendations**
 - What went well
 - What to improve
- **Root Cause Analysis**
 - Did it go well?
 - Did it go wrong?

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor -

121



Lessons Learned

- **Document Project Completion Review**
- **Highlight Lessons Learned**
- **Update Historical Database for use on next project Kick-Off**

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor -

122



Agenda

- Who is ISPI?
- Introduction
- Accepting the Project
- Planning the Project
- Managing the Project
- Closing the Project
- **Conclusions**



Project Management

Project Management is establishing and maintaining an environment that gets the work done

To effectively manage and control a project the Project Leader must:

- Be able to define the customer
- Define and manage the requirements
- Understand the system that must be built
- Establish necessary project roles
- Understand the project factors that must be managed
- Establish the Project Management Lifecycle
- Choose a Software Lifecycle



Project Tracking

- **Manage and Control the project throughout its lifecycle by:**
 - Tracking actuals to estimates
 - Taking corrective action when necessary
 - Establish and update a historical database
 - Use reviews throughout the Project Management Lifecycle
 - Establish and maintain the integrity of the evolving system through Configuration Management
 - Ensure the necessary quality functions are identified and implemented throughout the Project Management Lifecycle
 - Capture lessons learned to use in the next project Kick-Off

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor
128



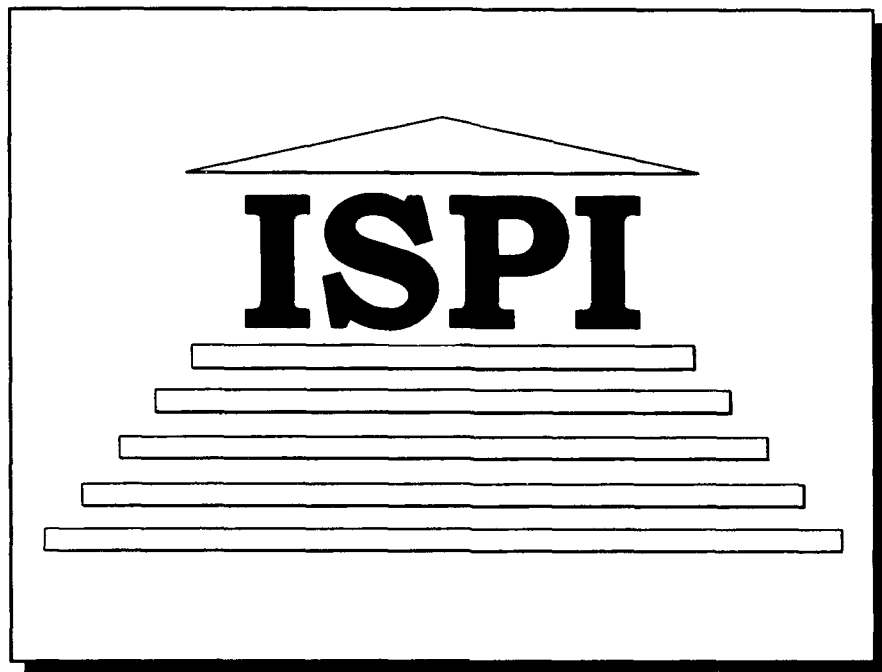
Project Support

A number of activities need to be run to support the project leader by providing additional management information and assistance:

- Configuration Management
- Quality Assurance

© 1996 Institute for Software Process Improvement, Inc.

version D1.0 - ProjMgmtTutor
128



15 N. Collinwood Drive Pittsburgh PA 15215 (USA) Tel. 00 1 412 781 1701 Fax. 00 1 412 781 0805	Klein Heiken, 101 B.2950 Kapellen (Belgique) Tel. 00 32 3 605 4875 Fax. 00 32 3 605 4876
http://www.ibp.com/plt/ispi	

The second rectangular frame contains the same ISPI logo as the first. Below the logo, a table provides contact information for two locations. The table has two columns, one for the USA office in Pittsburgh and one for the Belgium office in Kapellen. Each column lists the address, telephone number, and fax number. A separate line at the bottom of the frame provides the website URL.