CMU/SEI-95-SR-027

Carnegie Mellon University
Software Engineering Institute

SEI Program Plans: 1996-2000

Volume I: Five-Year Strategic Plan
Volume II: One-Year Plans/Proposals

January 1996

Carnegie Mellon University
Software Engineering Institute

January 1996

Colleagues:

I'm pleased that you've chosen to read the *SEI Program Plans: 1996-2000* (also known as the SEI 1&5 Year Plan). This document, which is in two volumes, presents the SEI strategy and one-year implementation plan for calendar year (CY) 1996, together with the SEI five-year program plans. Volume I describes the five-year strategic plan, and Volume II describes the one-year tactical plan.

Every year, we prepare a similar document to submit to our sponsor as a contract deliverable. This year marks the third year that we've published a version for public release. This document provides an insight into the direction of the SEI and our planned activities and outputs for the future.

This document was written in the first and second quarters of 1995 and was delivered to our sponsor (ARPA/ESC) as a contract deliverable in July 1995. As such, it was a draft plan; its execution depends primarily on approved resource allocations. The planning starts long before the Congress completes its budget authorization and appropriation. Historically, circumstances such as changing customer needs and shifting resource allocations have made it necessary to change our plans.

Volume II describes the work proposed for CY 1996 in two categories. The baseline outputs are those that were approved in a previous year and for which work continues into 1996. The add-on proposals describe new outputs that the SEI is prepared to initiate in 1996. Based on anticipated funding, ARPA recently was able to approve approximately 45% of these add-ons.

At the end of this letter is the table of approved add-on work outputs. Many received full funding; some received partial funding, and as a result, the work content has been modified. The SEI may seek customer funding for some outputs that were not approved or were only partially funded.

In reading this document, please consider opportunities in which you can work with us. As discussed in Chapter 4 of Volume I, the SEI has developed a range of relationships that provide mutual benefit to our customers in industry, government, and academia as well as the SEI. These relationships include the subscriber program, the resident affiliate program, Software Process Improvement Network (SPIN) organizations, advisory boards and working groups, and collaboration programs. I invite you to investigate which opportunities are right for you and your organization.

To contact us at the SEI, write or call:

Customer Relations
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890

Phone: 412 / 268-5800
FAX: 412 / 268-5758
Internet: customer-relations@sei.cmu.edu
World Wide Web: http://www.sei.cmu.edu

We look forward to working with you toward our common goal of improving the practice of software engineering.

Sincerely,

[illegible]

Julia Allen
Deputy Director

Special Report
CMU/SEI-95-SR-027
January 1996

SEI Program Plans: 1996-2000
Volume I: Five-Year Strategic Plan

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Table of Contents

List of Figures

Introduction to Volume I

This document, which is in two volumes, presents the Software Engineering Institute (SEI) strategy and one-year implementation plan for calendar year (CY) 1996, together with the SEI five-year program plan. It is submitted in response to the Contract Data Requirements List item A001. Volume I (this volume) describes the five-year strategic plan, and Volume II describes the one-year tactical plan.

This document is, in essence, a proposal. It describes the strategic directions and offers detailed options for the coming year. Until the proposed options are selected and budget allocations are approved by the sponsor, the SEI cannot commit to specific work or supporting schedules.

In Chapter 1 of Volume I, we set the strategic context by discussing the SEI charter, mission, vision, strategy, orientation, and customers. The SEI mission is *to provide leadership in advancing the state of the practice of software engineering to improve the quality of systems that depend on software.*

In Chapter 2, we describe the factors that determine SEI plans and set the context for their implementation in support of the SEI mission and strategy. The SEI strategy is to improve software engineering practice by maturing the skills of the software engineering practitioners who develop and maintain software, the managers who organize and lead these activities, and the infrastructure that supports these software professionals (Maturing the Profession). Our approach to improving the skills of these software engineering professionals is to mature the organizational and managerial processes through which software is acquired, developed, and maintained (Maturing the Process) and to mature the technology used to develop and maintain software (Maturing the Technology). These activities, combined with our core competency in software technology transition, form the strategy for executing the SEI mission.

In Chapter 3, we describe the SEI technical program. We concentrate our technical effort in *technical impact areas*. Within each technical area, we identify those processes, methods, and tools that are most effective in dealing with certain kinds of software engineering problems, and we work at getting more software engineers and software engineering organizations to use the best practices available for dealing with these problems. Technical impact areas currently focus on software process, risk management, disciplined engineering, and trustworthy systems.

In addition to the effort reflected in the technical impact areas, the SEI pursues activities aimed at enabling best practices to be used widely. These *impact enabling areas* deal with maturing the infrastructure of the software engineering profession and with software engineering transition strategies and methods.

The Software Process impact area has responsibility for maintaining competency in software process maturity modeling, definition, and measurement. The Risk Management impact area has increased its emphasis on acquisition as recommended by the Joint Advisory Committee (JAC) that assists the Advanced Research Projects Agency (ARPA) in guiding SEI activities. The Disciplined Engineering impact area maintains competency in methods and tools for engineering software systems. The Trustworthy Systems impact area addresses the increasing importance of trustworthy software. Its initial focus is on computer systems and their potential vulnerability to disruptive or otherwise criminal activities, which builds on the effectiveness of the SEI CERT^{SM}[1] Coordination Center in countering such activities. The Professional Infrastructure activity concentrates on establishing and improving the structures needed to support a software engineering profession, on the premise that such an infrastructure helps disseminate improved software engineering practices. The Transition Strategies and Methods activity reflects an explicit emphasis on ensuring that effective software engineering technology transition practices are used by the SEI, by our customers, and by the software engineering community.

Software technology transition is a core competence of the SEI that cuts across and influences all of the activities in the impact areas. The SEI mission requires a technology transition strategy that gives us leverage in meeting the needs of our customers. Chapter 4 provides information on the range of relationships that give our customers opportunities to participate in technology transition activities with us. In this chapter, we provide details about these relationships and about how we reach out to the software engineering community, to keep it informed about and involved in our work.

Details about the specific baseline and proposed add-on work outputs are in Volume II, and a list of all such work outputs is in Appendix A of Volume II.

1. CERT is a service mark of Carnegie Mellon University.

Volume I
Chapter 1 Table of Contents

1 Strategic Context

The purpose of this chapter is to describe the strategic context for the five-year plan and the one-year implementation of that plan.

The Software Engineering Institute (SEI) was established in 1984 by Congress as a federally funded research and development center with a broad charter to address the transition of software engineering technology. The SEI is funded by the Advanced Research Projects Agency (ARPA) through a contract with the Air Force Materiel Command/Electronic Systems Center, and through additional contracts with other sponsors, clients, and partners. These relationships determine organizational, funding, and reporting structures as well as providing a natural focus for selecting customers and activities.

As an integral component of Carnegie Mellon University (CMU), the SEI maintains a highly qualified staff and conducts its activities in a manner commensurate with that of the university. As a member of the CMU community and as an ARPA-funded organization, the SEI has access to leading edge technology and supports ARPA's commitment to satisfying the needs of the Department of Defense (DoD) and CMU's commitment to transferring improved technology to the community at large.

1.1 Charter

The SEI charter is to:

- Provide the means and leadership to bring the ablest professional minds and the most effective technology to bear on rapid improvement of the quality of operational software in software-intensive systems.
- Accelerate the reduction to practice of modern software engineering technologies.
- Promulgate the use of this technology throughout the software community.
- Foster standards of excellence for improving software engineering practice.

The SEI is funded by a combination of "basic" funds from ARPA and direct support from specific customers. The basic funding enables the SEI to engage in a combination of research, education, technology exploration, and development of transition products and services to achieve broad technology transition. (Because there are a variety of interpretations for the terms "products and services," this document uses the term "outputs" in this context.) The SEI may receive funding from federal agencies other than ARPA for specified work consistent with the charter, and the SEI is encouraged to collaborate with industry.

1.2 Mission, Vision, and Strategy

Software represents an enormous opportunity for cost-effective flexibility in military and commercial systems. Historically, software engineering organizations, both within and outside the DoD, have experienced significant difficulties in acquiring, deploying, and maintaining large-scale software systems. Acquired software often does not meet expectations, is delivered late and over budget, and is difficult to change to meet evolving needs. We believe that these problems can be avoided by bringing an engineering discipline to the way software is acquired, created, and maintained. However, the current state of the practice is far behind the state of the art, and the state of the art itself is not yet adequate to support a true engineering discipline. But insofar as the state of the practice does not reflect the state of the art, technology transition is the means of closing this gap.

Our **mission** is to provide leadership in advancing the state of the practice of software engineering to improve the quality of systems that depend on software.

We want software engineering organizations to be capable of applying best software engineering practices to produce high quality software that meets their expectations, at a competitive price and on predictable schedules. Therefore, we are committed to the evolution of software engineering from an ad-hoc, labor-intensive activity to a managed, technology-supported engineering discipline.

We **envision** ourselves as a software engineering technology transition organization, dedicated to improving software engineering practice. We see ourselves in the role of enablers, improving the practice by establishing human and technology connections that will reduce obstacles to technology transition and will encourage improved practices to spread throughout the DoD, government, and industry.

Our intent is to identify and transition those processes, methods, and tools that will help software engineering organizations make lasting improvements to their overall software engineering capabilities.

Our **strategy** for implementing this intent is to improve the state of the practice of software engineering by maturing the software engineering profession (Maturing the Profession). This strategy is based on maturing the skills of the software engineering practitioners who develop and maintain software, the managers who organize and lead these activities, and the infrastructure that supports these practitioners and managers. Our approach to improving the skills of these software professionals is to mature the organizational and managerial processes through which software is acquired, developed, and maintained (Maturing the Process) and to mature the technology used to develop and maintain software (Maturing the Technology). These activities, unified by our core competency in software technology transition, form the strategy for executing the SEI mission. (See Section 2.3 of Volume I for more information on this strategy.)

In applying this strategy, we will focus our technical activities in software engineering technology areas of critical importance to our customers. We will also address other important software engineering issues, but will not seek to establish a technology transition leadership position in those areas. To achieve a technology transition leadership position in an area of focus requires at least 25-30 people with appropriate expertise, as well as an additional cadre of specialized support. With our size constraints, we cannot expect to focus in more than five areas. A minimum of four areas appears necessary to have the broad impact prescribed in the charter.

We have chosen to call these areas of focus *software engineering technical impact areas* to emphasize that they are areas of software engineering in which we intend to have a significant impact. In the remainder of this document, they will simply be referred to as technical impact areas.

In each technical impact area, we analyze the state of relevant technology to see how it may affect the practice of software engineering. We analyze the state of the practice to uncover best practices and to understand where improvements are needed. We select high leverage technology, concepts, and technical approaches that have potential for improving the state of the practice, and we help to determine how this potential can be achieved by software engineering practitioners and managers. This technology maturation work is planned so the SEI and others can determine (1) whether the technology should be developed further, (2) how the risks of adoption can be reduced in the eyes of potential adopters, and (3) how the costs and benefits of adoption should be measured to ensure an acceptable return on investment by technology adopters.

Also in each technical impact area, we identify (1) who will benefit; (2) their strategic intent, needs, and requirements; (3) our vision, goals, and objectives; and (4) the specific outputs we will develop to achieve those goals and objectives.

Our outputs include courses, events, publications, prototype software, videotapes, and guidance and advice in the use of our outputs. These outputs are intended to help the software community improve its management practices, technical practices, and the capabilities of its personnel.

To maintain our understanding of (1) the needs of software engineering organizations, (2) the benefits of particular engineering practices, and (3) the difficulties of transitioning improvements into an organization, we work directly with specific customer organizations to understand their software engineering practices and to introduce improved practices.

1.3 Orientation

As a software engineering technology transition organization, the SEI promotes software engineering and supporting technology. Technology is our strength, and we are technology driven. However, we do not promote technology for its own sake; unless technology is successfully used by an increasingly widespread portion of the software engineering community, we will not be successful in fostering an improved state of the practice. Consequently, we are also needs driven. A need can result from an encountered problem, an opportunity enabled by innovation, or anticipation of future problems or technological advances. We help organizations understand the root causes of their software engineering problems as needs. Four considerations influence which problems we address and how we seek to ensure that improved solutions are moved into the practice:

1. The mission to advance the state of the practice of software engineering requires the SEI to have a broad impact by concentrating on those problems that are pervasive and by identifying effective means of getting people and organizations to use improved software engineering technology.
2. The SEI is in a trusted position that demands objectivity. Organizations expect the SEI to exert independent technical judgment and influence based on a broad and deep understanding of the field, and to understand and transition solutions to the root causes of problems, not simply to eliminate a symptom.
3. The SEI is a relatively small organization. More needs and problems exist than we can address, and there is more work to be done than we can expect to accomplish. We must be selective in choosing problems that are strategically important and have high-leverage potential, understand where outside expertise is available, and work within our abilities. We must work with and through others to get improved software engineering practices broadly installed in the community.
4. The SEI, by contract, is not permitted to compete in markets predictably and properly satisfied by commercial enterprise.

We are committed to be a needs-driven transition organization in this sense and have made this orientation an explicit part of our business. We will pursue technologies that offer solutions to meet real needs. To have a broad impact, we will provide solutions in the form of outputs that help organizations help themselves; we will also license and stimulate others to act as agents of change.

1.4 Customers

Customers are beneficiaries of SEI outputs. The SEI has customers in the DoD, in other federal agencies, and in industry and academia. The latter develop much of the DoD software and train software practitioners. To provide better service, we have identified three special categories of customers (**sponsors, clients,** and **partners**) with whom we collaborate in the development, maturation, and initial transition of needed outputs.

1.4.1 Sponsors, Clients, and Partners

Figure 1-1 shows some of the SEI interactions that help us perform our mission of improving software practice.

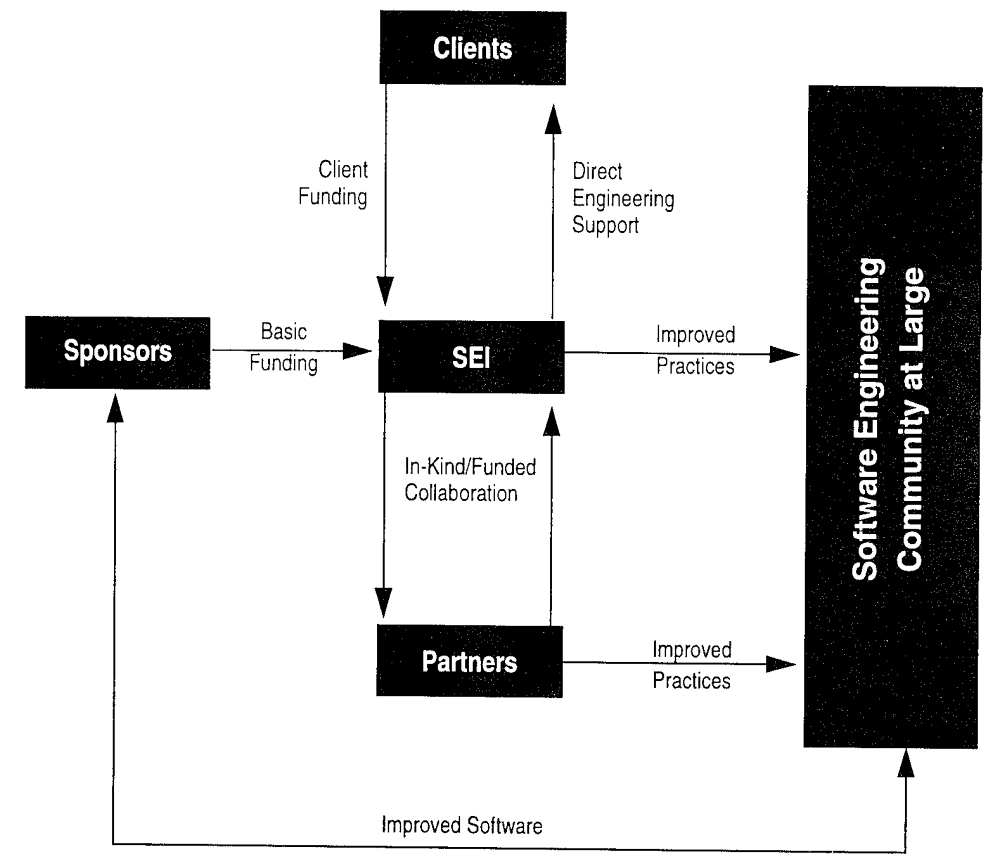


Figure 1-1: Interaction with Sponsors, Clients, and Partners

Sponsors invest funds in the development of capability. For instance, a sponsor might fund the SEI to investigate a certain technology in terms of its benefits to the software engineering community and its ability to be transitioned. Sponsorship may be tied to the condition that the sponsor be the initial transition target (customer) of a resulting output. A specific SEI activity could have multiple sponsors. We work with our sponsors to ensure that we are working on the right problems and to get their support for our approach and plan.

The DoD, through ARPA, is our major sponsor and invests funds in the SEI (basic program funding). These funds enable the SEI to understand needs, evaluate technology, and propose and test solutions. Then we develop and demonstrate outputs for our customers, as well as foster the transition of outputs to our customers. These funds also enable the SEI to develop and maintain relationships with the supporting software infrastructure in the United States.

Clients work with the SEI to address specific software engineering problems and transition issues. A significant portion of the SEI's total resources is received through technical objectives and plans (TO&P) funding from government clients, and a lesser portion is from cooperative research and development agreements (CRADAs) with industry. Whereas basic program funding enables the institute to investigate emerging ideas and technologies, client agreements provide the means for the SEI to proof-test or transition promising results into practice for specific customers. This type of interaction establishes a near-term conduit for SEI outputs to flow into the software community, and it permits the SEI to maintain insight into the nature of software practice. Through client agreements, the SEI works in the field to promote and verify improved practices in conjunction with the sponsor and to gather data that will influence future efforts.

Partners invest resources, including funds and people, to collaborate in the development, demonstration, or transition of SEI outputs. They may benefit directly or indirectly by the partnership. They also assume some risk. They contribute to the success of a specific output by providing expertise, perspective, credibility, and/or delivery capability. Organizations that send resident affiliates (that is, individuals on long-term assignment at the SEI from their home institutions) are, by definition, partners.

Partners provide us with insight into problems, assist with testing and transitioning SEI outputs, or offer a context for demonstrating solutions. The primary consideration in matching our capabilities to specific partners' needs is the credibility partners bring to the test or demonstration. They should bring specialized expertise to the SEI or be representative of a class of potential customers. They are selected based on their contribution to the success of an activity, their relative importance to an SEI sponsor, or their contribution to the sponsor. In addition, commercial vendors may provide leverage for SEI outputs by becoming transition partners who service broader markets than the SEI would be able to serve.

1.4.2 Acquisition, Development, and Post Deployment

Our customers focus on three distinct phases of the software life cycle: (1) acquisition, (2) development, and (3) post-deployment support. Each phase generates somewhat different software engineering concerns and different transition issues.

Acquisition is the phase in which requirements are defined and contracts are let for software development to meet these requirements. Concerns of acquisition organizations include policy, standards, requirements definition, cost and schedule estimation, contract and risk management, reengineering, reuse, training, and testing.

Development is the phase in which software is created that satisfies the requirements of the contracts resulting from the acquisition phase. The concerns of development organizations include requirements specification, design, coding, integration, testing, risk management, installation, training, and project management.

Post-deployment is the phase that addresses the support of the software after the system is fielded (operational). The principal concerns of post-deployment software support (PDSS) organizations are reliability, maintainability, reengineering, and the costs associated with these.

Volume I
Chapter 2 Table of Contents

2 Strategic Overview

This chapter describes the strategic factors that determine the plans of the Software Engineering Institute (SEI) and sets the context for their implementation in support of the SEI mission. Section 2.1, Situation Analysis, and Section 2.2, Effect of Major Trends on Software, provides an analysis of the current political and economic situation, and describes the major trends that are projected to significantly impact the field of software engineering and the SEI over the next five years. Section 2.3, Strategy for Improving Software Engineering Practice, describes the strategic framework which unifies the SEI's activities in support of the SEI mission over the next five years, the SEI's selected core competency, and the rationale for its selection. Section 2.4, Planning Considerations, specifies constraints considered by the SEI in defining its technical program. Section 2.5, Conclusion, summarizes the strategic context that forms the basis for the SEI technical program described in Chapter 3 of Volume I and in Volume II.

This plan reflects a commitment to effective cost control, increased leverage of resources, and focused efforts in those areas that will provide the highest payoff to SEI customers. At the same time, the plan reflects the support of the SEI to address evolving national priorities resulting from the changing world political and economic environment.

2.1 Situation Analysis

The U.S. has undergone a paradigm shift as a result of dramatic geopolitical events of the early 1990s, such as the dissolution of the Soviet Union, the Clinton Administration's national focus on strengthening the economy, and current shifting national priorities causing reduced government spending. As a result, several trends have emerged that affect the SEI technical plans significantly during this five-year planning period.

Third-world countries that are increasing their military belligerence and boldness, the changing and ill-defined regionalized military threat, and the renewed focus on national competition at the global level are influencing national priorities and emerging policies. The U.S. economy is undergoing a complex transformation; and as a result, there is a new linkage between technology and economic policy. This transformation is caused, in part, by the shrinking defense industry; the transition to just-in-time inventory and total quality practices; and the continuing belt-tightening and adaptation caused by global competition. The Department of Defense (DoD) cannot afford a unique technology base separate from the industrial sector. There is an emerging equivalence of national defense security and economic security, and the priority to create a strong integrated technology base to support both.

2.1.1 DoD Budget Reductions, Downsizing, and the Changing Role of the Military

The portion of the nation's budget dedicated to defense continues to decline. Likewise, due to the reduction in DoD-unique needs, the requirement for the DoD to control a unique technology base is declining. National defense capability will become more and more dependent on technology that is first developed and applied in the commercial sphere. Budget reductions and reductions in emerging DoD programs have caused a decrease in DoD organic capabilities and contractor bases, a growing need for increased flexibility, a concern for evolving systems, and a need to acquire systems more efficiently. Much of this will result in pressure to maintain existing systems and components for longer periods, creating more dependence on reuse and reengineering for extensive and responsive system modifications. Fewer new systems will be built, and existing systems will need to be evolved to meet new threats. Simulation will become an even more cost effective alternative for military training and system evaluation. Commercial off-the-shelf (COTS) products will play a larger role in rapid, flexible system integration for supporting military readiness in regional and global conflicts. The integration of software COTS products into systems to support military needs must become a strength of the industry supporting defense.

2.1.2 The Increase in Global Competition

International competition will intensify as world industrial and technological capability is distributed among industrialized nations. Internationalization of economic and technological activity will deepen the interdependence between national economies and lessen the line between domestic and foreign policies. The focus on international competition will reinforce the need for international standards and highlight the increase of sophistication in global technology. Hence, the SEI will be required to increase its international participation with the technical community; assist in the development of the standards required for doing business in this environment; interpret the impact of these standards on our own economic base; and advocate the formulation of a responsive national policy for effective competition in the global marketplace.

2.2 Effect of Major Trends on Software

Government agencies need to acquire, develop, and maintain software-intensive systems more efficiently as the number of systems to maintain increases and the newly acquired systems become more complex. Faced with declining budgets, this need will require improvements in the process by which organizations buy software-intensive systems. The DoD's current system acquisition model has evolved from the hardware acquisition model. The current system acquisition model assumes that requirements are well understood early in the product development life cycle, and that systems can be built as they are initially conceived.

A measurable process that supports successful management of software-intensive systems acquisition needs to be developed.

Use of products and standards emanating from the commercial world offers an attractive way for the DoD to acquire and evolve systems of high quality at minimum cost and risk. The present DoD acquisition system lacks sufficient flexibility to adequately incorporate COTS and advanced technology demonstrations, nor does it have the means to accelerate schedules sufficiently to field systems that exploit current technology. DoD system acquisition procedures are changing to encourage more frequent use of COTS products. In using COTS, industry standards will become even more important to the DoD. Hence, the SEI will focus on improving the capability to design systems and their architectures using and integrating these standards.

The increased reliance on networked systems of cooperating systems across both unbounded networks, such as the Internet, and bounded networks, such as the Defense Information System Network, require an increased sensitivity to system vulnerability from malicious attempts to disrupt and deny usage. The SEI can capitalize on its experience from the Computer Emergency Response Team to aid the DoD in building trustworthy systems. The trustworthy systems work will be readily extendible to the national systems that are providing part of the national information infrastructure.

Increased national competition and requirements for increased flexibility will require shorter system development times. This suggests that future systems will be created and configured on demand from proven concepts, architectures, and components. This situation will place greater emphasis on software architecture, reuse, and reengineering in the short term and design for reengineering and automatic program generation in the longer term. It will also require more effective software engineering practices that can be applied much earlier in the system life cycle.

Defense planners will have to deploy smaller and better-trained forces, supported by high-performance equipment adaptable to changing threats. The equipment and systems that support better training are increasingly dependent on software for their functionality. Concurrently, computing power, because of advanced semiconductor technology, continues to increase. These trends create demands for affordable, reliable, and flexible software that are more and more challenging to satisfy.

The reduction in operational funding for the military heightens the importance and cost effectiveness of computer simulation for training and system evaluation. The increased use of real-time simulation, both for defining and refining system requirements and for training, mandates software that can meet time constraints within a networked environment. It also suggests the need for vastly improved human interface technologies to provide the required realism for effective evaluation and training.

The need to respond quickly in a worldwide theater of operations suggests increased portability of command control and intelligence facilities. It requires a greater use of telecommunication technology, such as teleconferencing and telepresence, so that people with much needed skills who are located remotely can solve local battlefield problems.

Budgetary constraints and rapid changes in military strategy are creating the need for more flexible manufacturing capability. DoD acquisition may fund prototype development and defer full-scale production until the need is clearly demonstrated. The emphasis on "agile manufacturing" for both defense and commercial organizations implies that the SEI should devote more attention to processes and tools for supporting manufacturing technology transition efforts.

The experiences and successes garnered from defense-related research and development (R&D) are directly applicable to civil sector needs. The SEI expects to see no more than the same level of funding support from the DoD and, most likely, a decline. At the same time, we can expect increased need and funding support from other federal agencies as they struggle to overcome the same software-related problems that have been addressed effectively within the DoD by the SEI over the past decade.

As the DoD downsizes, the emphasis on improving the U.S. economy and infrastructure moves the nation toward a more commercially oriented R&D base, with focus on technology transition to the commercial sector. The result of this is that increasing amounts of research relevant to the DoD will be conducted by civil agencies or industry. Hence, the SEI must provide more support to these civil agencies and industry to participate in this broadened range of dual use technology developments relevant to DoD software engineering.

The SEI must respond to the changing political and economic environment and the need to revitalize the national infrastructure, particularly the nation's industrial base. We will increase our focus on efficient acquisition, development, and maintenance of software-intensive systems, taking into consideration dual-use software technology. We will support the development of enhanced software architectures; improved techniques for reuse and reengineering; real-time simulation; expanded and more flexible communications capabilities; system integration of commercial software; and software engineering processes and tools for advanced manufacturing technologies.

2.3 Strategy for Improving Software Engineering Practice

The current political and economic situation, as described in Section 2.1, establishes the importance of software to the defense and economic well being of the nation. Because software pervades nearly every aspect of society, it is vital to address effectively the issue of continuous improvement of the practice of software engineering as an essential ingredient of our national strategy. With this in mind, it is important to articulate clearly the strategic framework which supports the SEI mission to improve the state of the practice of software engineering.

Figure 2-1 represents the SEI strategic framework to improve software engineering practice in support of our national strategy. It is through this strategic framework that the mission of the SEI will be executed effectively.

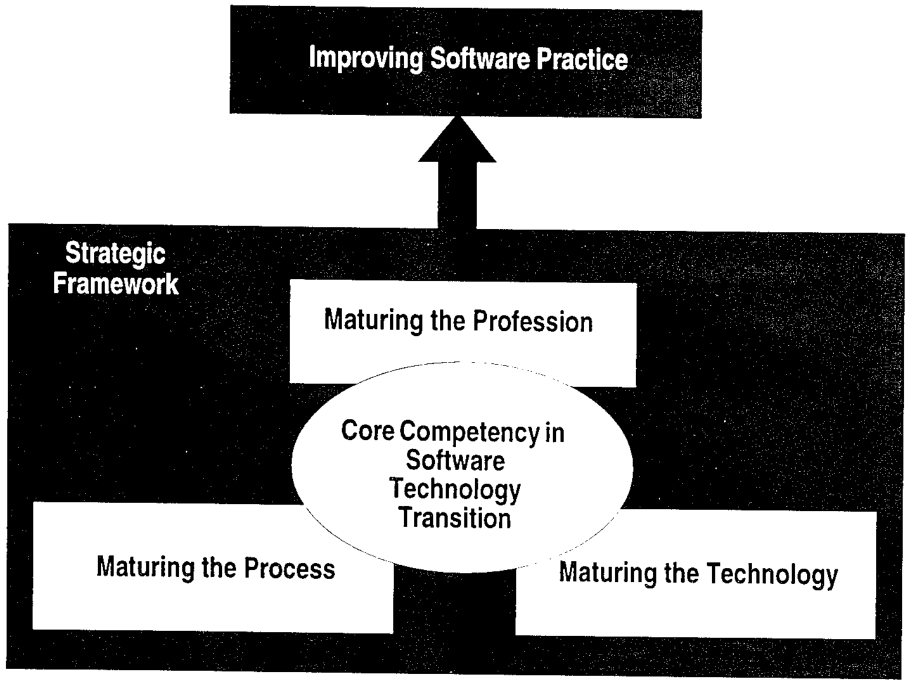


Figure 2-1: Strategic Framework for Improving Software Practice

The SEI strategy for improving software engineering practice is to transition software technology to software engineering practitioners and the supporting infrastructure. Software technology transition is central to the SEI mission. The goal is to mature the skills of the software engineering practitioners who develop and maintain software, the managers who organize and lead these activities, and the infrastructure that supports these software professionals (Maturing the Profession). Our approach to improving the skills of these software engineering professionals is to mature the organizational and managerial processes through which software is acquired, developed, and maintained (Maturing the Process) and to mature the technology used to develop and maintain software (Maturing the Technology). These activities, combined with our core competency in software technology transition, form the strategy for executing the SEI mission.

2.3.1 Maturing the Profession

Software practice is performed by people using tested, proven, and effective processes, methods, and tools that they and their predecessors have created. These people are software professionals, and it is through them that the SEI intends to improve software practice. To do this, the SEI aims to mature the skills of software engineering practitioners and managers. Our approach to improving the skills of these software engineering professionals is to mature the processes through which they develop and maintain software, the technology they use to develop and maintain software, and the infrastructure that supports the software engineering profession. Improving the skills of software professionals will result in improved effectiveness and efficiency in the state of the practice of software engineering. For details on how the SEI transitions these improvements into the profession, see Section 3.6 of Volume I and Chapter 5 of Volume II.

2.3.2 Maturing the Process,

The Capability Maturity Model^{SM} (CMM)[1], conceived as a model for judging the maturity of the organizational and managerial processes of an organization and for identifying the key practices that are required for maturing these processes, has become a standard for assessing and improving software processes. Through the CMM, the SEI has put in place an effective means for modeling, defining, and measuring the maturity of organizational and managerial processes used by software professionals. With assessment results acting as benchmarks, organizations can then use key process areas (KPAs) and best practices to improve the maturity of their processes.

The CMM is an example of a framework that facilitates understanding the underlying technology and processes. Understanding is a key criterion for bringing a process under continuous improvement. Definitional work on other frameworks that directly address the processes contributing to the maturation of software engineering profession was begun in 1995. Defining frameworks for the Systems Engineering Capability Maturity Model, the People Capability Maturity Model, and the Software Acquisition Maturity Model are all in progress. It is this work, along with continued support and refinement of the CMM, that will assist in maturing the processes that support sound software engineering practices.

2.3.3 Maturing the Technology

More mature organizational and managerial processes are necessary but not sufficient for a mature software engineering profession. SEI experience suggests that where organizational and managerial processes are at CMM level 3 or higher, the need for more mature technology becomes evident. Particularly needed are technologies that allow quality attributes such as performance, reliability, timeliness, dependability, and trustworthiness to be reliably and accu-

[1] Capability Maturity Model and CMM are service marks of Carnegie Mellon University.

rately predicted and controlled. Also needed are commonly accepted system and subsystem architectures and models that operate at different levels of abstraction.

2.3.4 Core Competence in Software Technology Transition

By software technology transition, we mean movement of the best software engineering processes, methods, and tools into broad use in the software engineering community. The SEI is a value-adding transition agent between researchers whose results improve software practice and practitioners who can apply the results to solve important and pervasive software problems. The SEI adds value by identifying relevant research results and making them understandable and applicable by practitioners, and by identifying root causes of problems faced by practitioners and making them understandable and applicable to researchers. Thus, while the computing research community aims to advance the state of the art, the SEI aims to incorporate state-of-the-art advances into the state of the practice. Through our interactions with practitioners and researchers, the SEI seeks to identify "best practices" and to promote widely their introduction into the practice of software engineering. Successful technology transition results in overall improvement in the state of software engineering practice.

To build its core competency in transition, the SEI adapts transition models from other disciplines and adapts them for application to software. These models help us approach technology transition in a systematic and effective way. We identify transition methods and transition vehicles that facilitate adopting and institutionalizing improved processes, methods, and tools. We develop transition products and services that help people help themselves to improve their practices.

Because the size of the SEI technical staff is limited, we seek leverage for our transition efforts. We gain leverage by influencing those organizations that are building the software infrastructure, such as software engineering educators and the Software Process Improvement Network (SPIN) community by providing services—primarily advice and guidance to government organizations—that aid continuous improvement efforts, and by working with transition partners who can take our products and services to the community at large.

2.4 Planning Considerations

In developing plans within the context of the changing political and economic environment, the SEI must consider several factors. The most significant of these planning factors are:

- The mission requires that the SEI facilitate the transition of appropriate software technology into practice for the mutual benefit of the DoD, civil agencies, and the industrial sector in support of national priorities and objectives.

- The SEI strength is in the area of software technology maturation and transition—broadly defined to include traditional "computer science" and the evolution and transition of engineering practice. The SEI must maintain its focus on technology to provide the stability that is vital to an R&D program, and must emphasize efforts that result in transitionable products rather than personal services.
- The small size of the SEI requires highly leveraged resources and effective means for technology transition.
- Technology transition requires knowledge of, and involvement with, technologies, user communities, and the transition process. While SEI activities focus on software technology, they have a planned effect of increasing SEI knowledge about user needs in specific domains.
- The SEI must balance technical depth with a broad understanding of software practice in its personnel.
- The SEI must maintain its position as an objective third party to function effectively as a center of excellence and an objective broker to both government and industrial sectors in software engineering technology.
- The SEI, by contract, is not permitted to compete in markets predictably and properly satisfied by commercial enterprise. Our approach to transition, which seeks to use the existing U.S. infrastructure as partners, helps us to avoid competition.

2.5 Conclusion

The importance of software to our national security, both in terms of defense and economic well being, is clear. As articulated in this chapter, the criticality of software emphasizes the importance of the SEI mission to advance the state of the practice of software engineering. The SEI is firmly committed to this mission, and has established and implemented the strategic framework described in this chapter to achieve its mission. The foundation for this strategic framework is found in our technical program, described in Chapter 3 of Volume I and in Volume II. The goal of the SEI technical program is to improve software engineering practice by maturing the process, the technology, and the profession.

In responding to the challenges discussed in this chapter, the SEI seeks to take advantage of the organizational, historical, and situational differences that distinguish it. These differences include:

- Accomplishments to date, particularly in the areas of software process modeling, real-time systems, software engineering education, computer emergency response, and software architecture.
- The SEI status as a federally funded R&D center chartered to act as an objective broker performing software engineering technology development and transition.
- The SEI association with Carnegie Mellon University and its relationship to its world-class faculty in such areas as computer science, electrical and computer engineering, and economics and business.

- The SEI sponsorship by the Advanced Projects Research Agency (ARPA), which provides us access to the ARPA software research community.
- The SEI charter to provide research and technology support throughout the federal government, enhancing our ability to support the objectives of defense conversion.

The SEI has established itself as the leader in the field of software engineering. We have developed and are implementing a strategic framework for improving the state of the practice of software engineering. In supporting this framework with a strong technical program and a well-focused core competency in software technology transition, the SEI has positioned itself to respond effectively to new requirements and technologies and to respond to the defense community and other agencies within the federal government.

Volume I
Chapter 3 Table of Contents

3 SEI Technical Program

3.1 Overview

In this chapter we describe the technical program of the Software Engineering Institute (SEI) and how it has been designed to improve software engineering practice.

3.1.1 Structure

The SEI concentrates its technical effort in *technical impact areas*. Within each technical area, we identify those processes, methods, and tools that are most effective in dealing with certain kinds of software engineering problems, and we work at getting more software engineers and software engineering organizations to use the best practices available for dealing with these problems. Technical impact areas currently focus on software process, risk management, disciplined engineering, and trustworthy systems.

In addition to the effort reflected in the technical impact areas, the SEI pursues activities aimed at enabling best practices to be used widely. These *impact enabling areas* deal with maturing the infrastructure of the software engineering profession and with software engineering integrated transition strategies and methods. The relationship of the technical impact areas and the impact enabling areas to the SEI strategic framework is shown in Figure 3-1. Within each impact area, work outputs are described in related clusters, which we call *activity areas*.

The Software Process impact area has responsibility for maintaining competency in software process maturity modeling, definition, and measurement. The Risk Management impact area has increased its emphasis on acquisition as recommended by the Joint Advisory Committee (JAC) that assists the Advanced Research Projects Agency (ARPA) in guiding SEI activities. The Disciplined Engineering impact area maintains competency in methods and tools for engineering software systems. The Trustworthy Systems impact area addresses the increasing importance of trustworthy software. Its initial focus is on computer systems and their potential vulnerability to disruptive or otherwise criminal activities, which builds on the effectiveness of the SEI CERT Coordination Center in countering such activities. The Professional Infrastructure activity concentrates on establishing and improving the structures needed to support a software engineering profession, on the premise that such an infrastructure helps disseminate improved software engineering practices. The Transition Strategies and Methods activity reflects an explicit emphasis on ensuring that effective software engineering technology transition practices are used by the SEI, by our customers, and by the software engineering community.

While all impact areas broadly support the strategic framework, software process and software risk management are principally directed toward maturing the process. Likewise, disciplined engineering and trustworthy systems are principally directed toward maturing technologies and engineering practices. The professional infrastructure and transition strategies/methods areas are principally directed toward maturing the profession and toward supporting software engineering technology transition competence, both within the SEI and in software engineering organizations.

This chapter gives an introductory overview of the impact areas and the activity areas within each group. A detailed discussion of the activity areas and proposed work is contained in Volume II. This chapter discusses the following impact areas:

Section Number	Impact Areas	Page
3.2	Software Process	I-33
3.3	Risk Management	I-47
3.4	Disciplined Engineering	I-57
3.5	Trustworthy Systems	I-67
3.6	Professional Infrastructure	I-75
3.7	Integrated Transition Strategies and Methods	I-85

3.1.2 Software Process

The objective of our software process work is to mature the organizational and managerial processes employed by software engineering organizations, i.e., we seek to ensure that software engineering organizations employ increasingly effective sets of managerial processes as they gradually improve their software engineering capability. The outputs of this impact area have been highly visible within the software engineering community worldwide. In particular, the Capability Maturity Model (CMM) for Software, introduced by the SEI to describe the maturity of organizational and managerial processes, has been widely adopted as the basis for guiding software engineering improvement efforts, and its concepts are being incorporated in U.S. and international standards. It has also been responsible for the formation of software engineering process groups (SEPGs) in a large percentage of the Department of Defense (DoD) contractor community. Due in part to software process efforts, software process improvement network (SPIN) groups have been formed in many major U.S. metropolitan areas. Key activity areas are: *process maturity modeling*, which includes updating the CMM to version 2.0, maturity modeling efforts for additional areas—Systems Engineering (the SE-CMM) and people management (the People CMM), and providing an

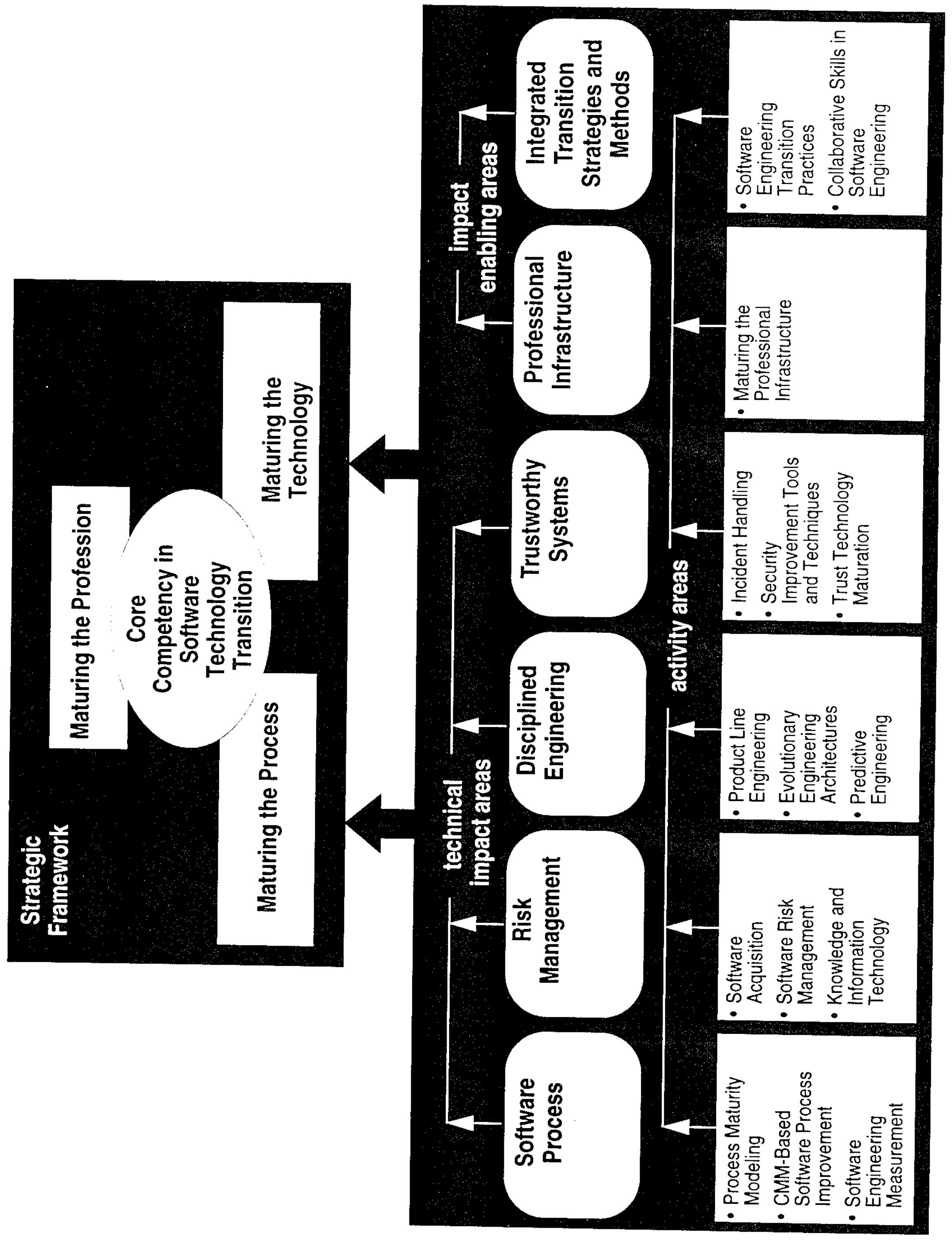

Strategic Framework
Maturing the Profession
Core Competency in Software Technology Transition
Maturing the Process
Maturing the Technology
technical impact areas
impact enabling areas
activity areas
Software Process
Risk Management
Disciplined Engineering
Trustworthy Systems
Professional Infrastructure
Integrated Transition Strategies and Methods
• Process Maturity Modeling
• CMM-Based Software Process Improvement
• Software Engineering Measurement
• Software Acquisition
• Software Risk Management
• Knowledge and Information Technology
• Product Line Engineering
• Evolutionary Engineering Architectures
• Predictive Engineering
• Incident Handling
• Security Improvement Tools and Techniques
• Trust Technology Maturation
• Maturing the Professional Infrastructure
• Software Engineering Transition Practices
• Collaborative Skills in Software Engineering

Figure 3-1: Proposed 1996 Strategic Framework, Core Competency, and Areas

integrating architecture for maturity models that may be developed at the SEI or elsewhere; *CMM-based software process improvement*, aimed at producing new and updated products for process assessments and evaluations based on the latest version of the CMM and on a common appraisal architecture; and activities focused on *software engineering measurement*, including support for project/product measures and increased support and leverage of data repositories reflecting the practice of software engineering.

3.1.3 Risk Management

The objective of our risk management work is to provide a systematic and structured process, supported by methods and tools, for identifying, analyzing, and mitigating the uncertainties encountered within a specific software engineering effort. We are focusing on risk because many of the most serious issues encountered in systems acquisition are the result of risks that remained unrecognized until they have already created serious consequences. Our work in risk management has demonstrated that structured techniques, even quite simple ones, are effective in identifying and quantifying risk. Key activity areas are: *software risk management*, in which we are working on a variety of approaches to installing risk identification and mitigation techniques into organizations; *software acquisition*, in which we are extending our risk management work to cover the special needs of organizations that acquire software-intensive systems; and work on *knowledge and information technology*, in which we are developing a repository of information about common risks, mitigation strategies, and lessons learned. The repository technology being used in this area is being shared across impact areas that are also gathering, storing, and analyzing data about the state of the practice in software engineering.

3.1.4 Disciplined Engineering

Our work in the disciplined engineering impact area is focused on maturing the technology underlying software engineering. In particular, we focus on the use of models and architectures for describing, quantifying, analyzing, and predicting system properties so engineering trade-offs can be made in a disciplined fashion. Key activity areas are: *product line engineering*, in which we focus on understanding the commonality and variability in software systems so engineers can reduce the number of distinct systems that need to be designed and maintained; *evolutionary engineering architectures*, in which we focus on the ability of systems to rapidly change their capabilities and adapt to changes in the environment while maintaining standards of product quality; and *predictive engineering*, in which we focus on increasing our ability to systematically analyze systems, to understand the consequences of choosing among engineering alternatives, and to reliably predict quality attributes of systems.

3.1.5 Trustworthy Systems

The objective of our work in trustworthy systems is to help meet the security and integrity needs of the increasing number of people who depend on the use of computer networks. We seek to foster the widespread adoption of tools and techniques that improve the security of existing systems, identifying and maturing technologies that can be used to produce software and architectures that are highly resistant to attack. Key activity areas are: *incident handling*, in which we facilitate the resolution of computer security incidents and gather information on system vulnerabilities; *security improvement tools and techniques*, in which we explore and mature components used to build systems that are predictably secure; and *trust technology maturation*, in which we seek to mature network security technologies and practices.

3.1.6 Professional Infrastructure

The objective of our work in maturing the professional infrastructure is to help promote high standards of professional practice and mechanisms that support the rapid dissemination of new knowledge. We intend to collaborate with other organizations to help ensure support for the evolution of the software engineering profession. Work in this area is being planned in 1995, but it is proposed that work in 1996 address recommended practices in software engineering, in which we will help specify recommended software engineering practices and procedures for disseminating this information and professional development, in which high quality curricula for continuing education and training are published and used by individuals and organizations to ensure their knowledge and skills are up to date.

3.1.7 Integrated Transition Strategies and Methods

The objective of our work in the transition strategies and methods area is to ensure that effective technology transition strategies are used by adopters of improved software engineering practices, and that technology vendors provide products that are more easily transitioned because they understand the transition barriers and adoption processes of their customers. The net effect will be to improve the rate of technology transition in software engineering. There are two activity areas: *software engineering transition practices*, in which work is aimed at adapting existing technology transition knowledge to the needs of the software engineering community and ensuring that this knowledge is widely exploited in the community; and *collaborative skills in software engineering*, in which we focus on identifying and enhancing the skills software engineers and managers need to collaborate effectively. We focus on collaborative skills because the development and maintenance of software systems is a group process. Activities in this area include dissemination of tools, methods, and practices that increase the ability of teams of software engineers to work together more effectively in applying software engineering practices.

3.2 Software Process

Software engineering is becoming the dominant discipline in product development organizations in just about every manufacturing and development industry. It is already the dominant discipline in many support and service industries. As a senior vice president of a major New York bank said last year, "We are a software company masquerading as a bank."

However, large numbers of software development organizations in the United States continue to have an ad hoc, crisis-driven process that often results in projects producing poor-quality systems that are chronically late and over budget. Indeed, most software organizations are at the initial level of software process maturity when appraised against the Software Engineering Institute (SEI) Capability Maturity Model^{SM} (CMM^{SM})[1] for Software (based on 435 SEI-assisted, vendor, and self assessments as of December 1994; see Figure 3-2). Results from 60 of these assessments are for non-U.S. organizations and reflect virtually the same profile.

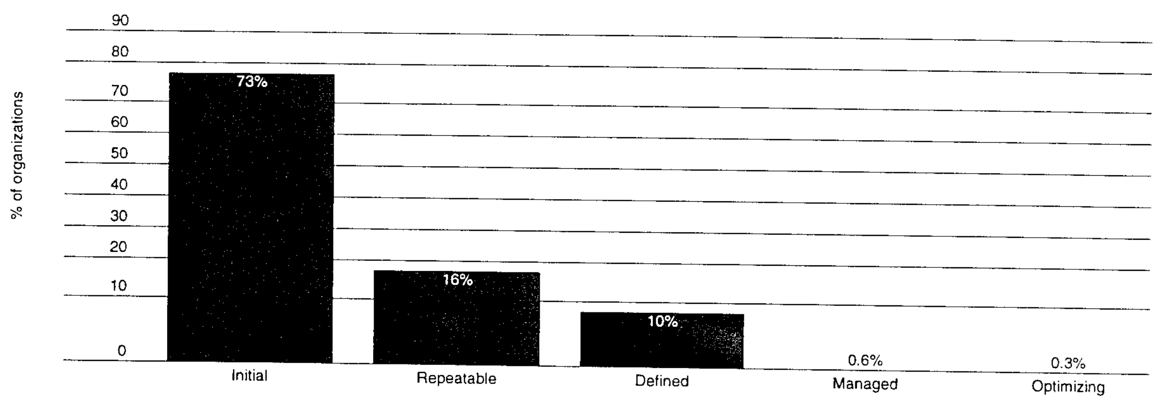


Figure 3-2: Organization Maturity Profile

On the other hand, organizations that have established long-term software process improvement (SPI) efforts report evidence that they are beginning to show movement toward higher levels of maturity, matched by business-related improvements. Figure 3-3 reflects data reported to the SEI on the maturity improvements of organizations that have been reassessed. Fully 83 percent of the 48 reporting organizations achieved capability maturity improvements between their initial and latest assessments. (Business-related results are discussed in detail in Section 3.2.1.1)

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

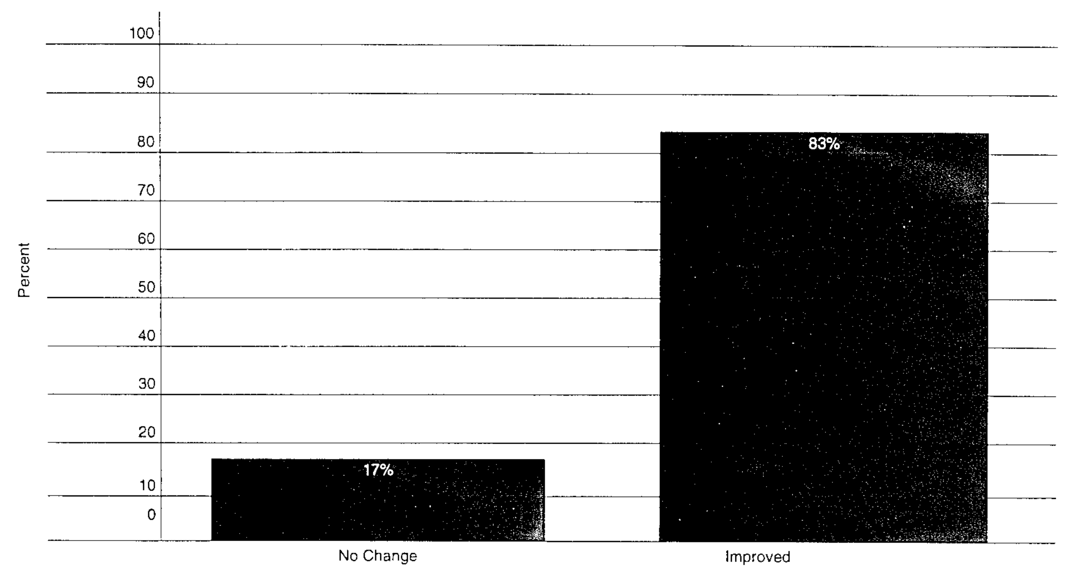


Figure 3-3: Change in Maturity Level Between Reassessments

Lower maturity, crisis-driven software environments are characterized by

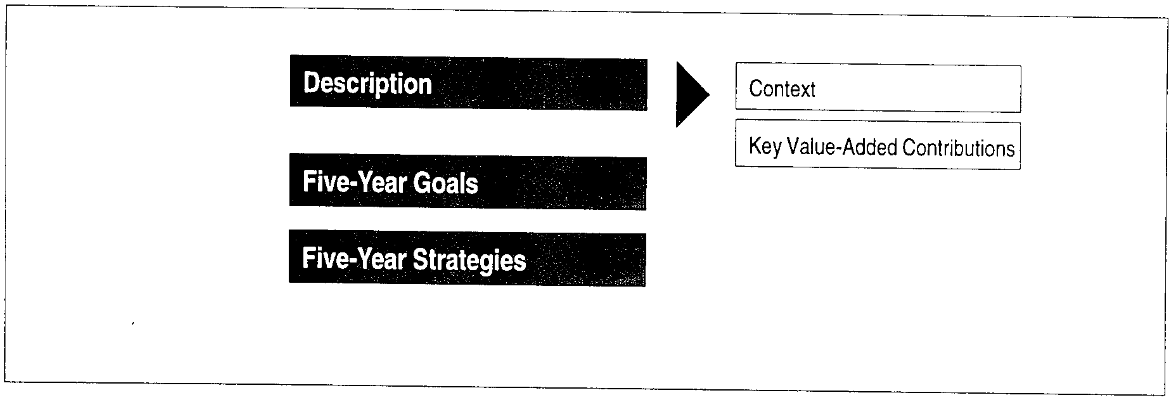
- unpredictable, inconsistent performance and quality;
- an inability to successfully implement and sustain new technologies and methods;
- strong dependence on a few highly competent individuals;
- a focus on firefighting;
- high levels of frustration and adversarial relationships across disciplines; and
- predominantly schedule-driven environments.

Too many software organizations have long relied on individual talent, which is in short supply, to produce acceptable results. Unfortunately, the only way these results can be repeated is to assign the same individuals to the next project. In such environments there is no institutionalized capability for meeting projected targets for cost, schedule, quality, and functionality. Executives in such organizations typically complain that they have little visibility into the software development process and that they are unable to make accurate projections of performance and costs. Without visibility and predictability, executives are unable to exercise management oversight or make sound business decisions regarding projects.

To improve the state of the practice of software engineering in the United States, software-producing organizations must establish an organizational capability (rather than be dependent on individuals) for developing software based on sound management practices that support a disciplined, defined, and measured software engineering process. They must be able to execute this defined software engineering process consistently across all projects in the organization, rather than have only a few successful projects, with others missing the objectives of cost, schedule, quality, and function. Furthermore, organizations must be able to learn from their experiences to improve their capability.

Establishing an organizational capability for developing software also entails defining and implementing software measurement practices. Measurement, and the ability to see and understand progress, are closely related—a developer can measure only what is visible in a process, and measurement helps to increase visibility. The CMM can serve as a guide for determining what to measure first and how to plan an increasingly comprehensive measurement program. For example, measures at level 2 focus primarily on project planning, management, and tracking, while measures at level 3 are directed toward intermediate and final products. Measures at level 4 capture characteristics of the development process itself to allow control of the individual activities of the process. At level 5, processes are mature enough and managed carefully enough to permit measurement to provide feedback for dynamically changing processes and introducing technologies across multiple projects.

In this chapter, the sections for this impact area are as follows:



3.2.1 Description

3.2.1.1 Context

The SEI utilizes its core competencies in process maturity modeling, model-based process improvement, and software engineering measurement to be a primary driving force in maturing and improving software management and organizational practice. The products and services provided by this area are all based on the CMM, which also supports advancement in the other technology areas and the maturing bases of the strategic framework (see Figure 3-1). Additionally, the transition of the CMM into practice is supported by the SEI competence in technology transition and is our key contribution to maturing the practice of software engineering. This transition is enabled by the products and services provided by the Process area in support of continuous process improvement.

The CMM has been stable for 1994-1995 and is under change control. Development of CMM version 2.0 began in 1995, with a target to deliver publicly late in 1996 or early 1997. Efforts in areas of community awareness of the CMM have continued to be a high priority. Involvement in work to influence international standards toward U.S. positions has moved from technical definition into pilot efforts. The modeling efforts have been expanded to integrate Systems Engineering and People CMMs with the CMM for Software. This has led to initial definition of an integrating architecture for maturity modeling efforts across the SEI, based on the draft international standards architecture proposed by the SEI for such models.

The CMM-Based Appraisal (CBA) effort has produced new products for assessments and evaluations that are based on the latest version of the CMM and on a common appraisal architecture: the CMM Appraisal Framework (CAF). A CAF for developing, defining, and using appraisal methods based on the CMM will lead to more consistent results across appraisal types. Over 100 individuals have been trained and authorized to lead CMM-Based Appraisals for Internal Process Improvement (CBA IPI) in the new SEI appraiser program.

In 1995, integration of the appraisal methods, software process definition, and software process measurement activities will begin. The focus of our activities in the future will be to address how we can provide a more integrated approach to CMM-based SPI efforts and build integrated process, products, and services. In addition, we will focus on addressing broader software engineering measurement needs and solutions for the future. This includes process, project, and product measures; further support of higher maturity level measures; and increased support and leverage of software data repositories.

Additional work to identify and report the results of SPI efforts is in response to some of the most prevalent questions from our community, and is reported in quantified business terms relating to cost, schedule, productivity, quality, and return on investment (ROI) in SPI (summarized in Figure 3-4).

3.2.1.2 Key Value-Added Contributions

Through our process work, we have established a community-owned, de facto standard model of organizational and management discipline and process maturity that envisions a culture of software engineering excellence. We maintain stewardship over the model on behalf of the community to ensure its continual improvement, reflecting best practices that we will identify in the future and emerging states of the art. In addition, the SEI has transitioned the model into practice in some leading-edge government and industry customer organizations.

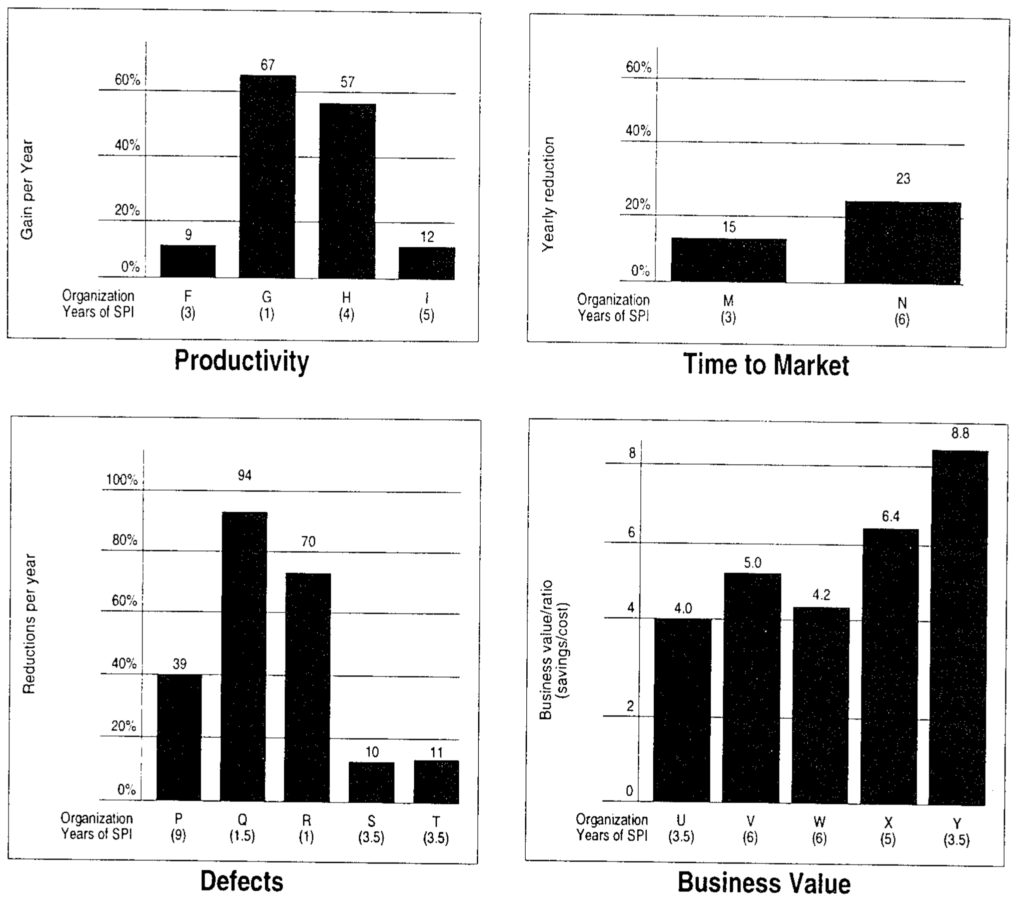


Figure 3-4: Preliminary Results of SPI

These charts illustrate some of the gains reported by organizations using CMM-based SPI. Each bar represents a data point provided by one organization. The data on the four charts do not necessarily all come from the same organizations. The first three charts show yearly productivity, cycle time, and quality gains for improvement efforts. Among these organizations, typical gains are 10% per year in productivity and quality, while some have experienced much greater improvements. The fourth chart shows the ratio of savings to costs for the process improvement effort. These organizations, all of which have been engaged in SPI for at least three years, are showing very impressive ratios in the 4:1 to 8.8:1 range.

We are increasing our competence in SPI with the objective of integrating process appraisal, definition, and measurement core competencies through CMM-based SPI. Using the model as a base, organizations are educated and trained in assessing their current state, planning for improvement, defining improved processes, and measuring those products, projects, and processes to determine business results associated with the improvements. As the products that support this training become mature, they are transitioned to expand the use of proven techniques.

The SEI is a trusted safe haven for proprietary data reflecting the results of SPI. Participants supply the results of assessments as well as the benefits of investing in SPI. We report these results to the software community so organizations can benchmark their own state and progress and make decisions on areas of SPI investments (see Figure 3-4). Additionally, as a neutral player in the community, the SEI is looked to for coordination and assistance across the community to support individual organization's development of software engineering process groups (SEPGs) (see Figure 3-5) and networking through software process improvement networks (SPINs) (see Figure 3-6).

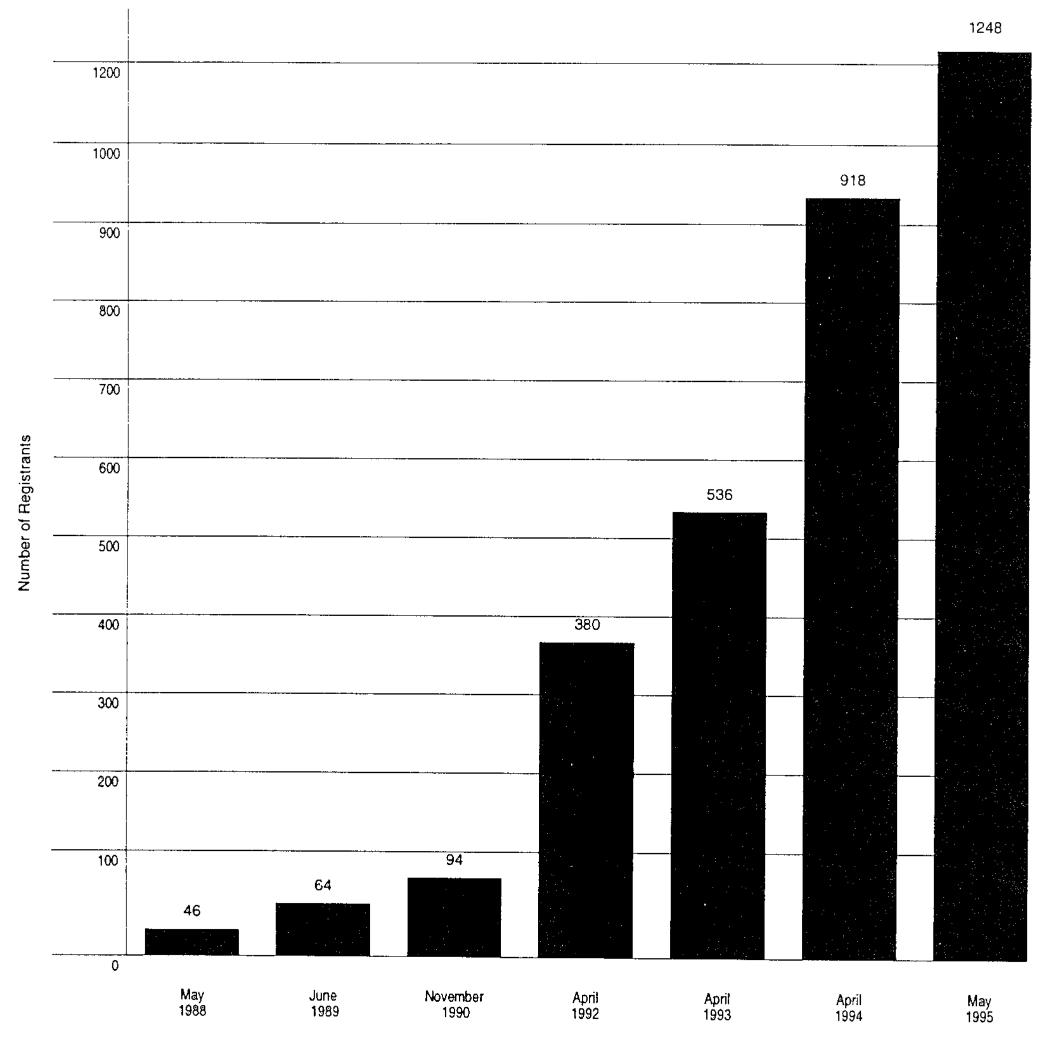


Figure 3-5: SEPG Conference Attendance

An integrated set of software engineering measures, drawn from other parts of the SEI and from the community, will further the efforts of the past in defining core measures. It will enhance the SEI's ability to provide a national service to the community in measurement activities and a core national resource for measurement expertise.

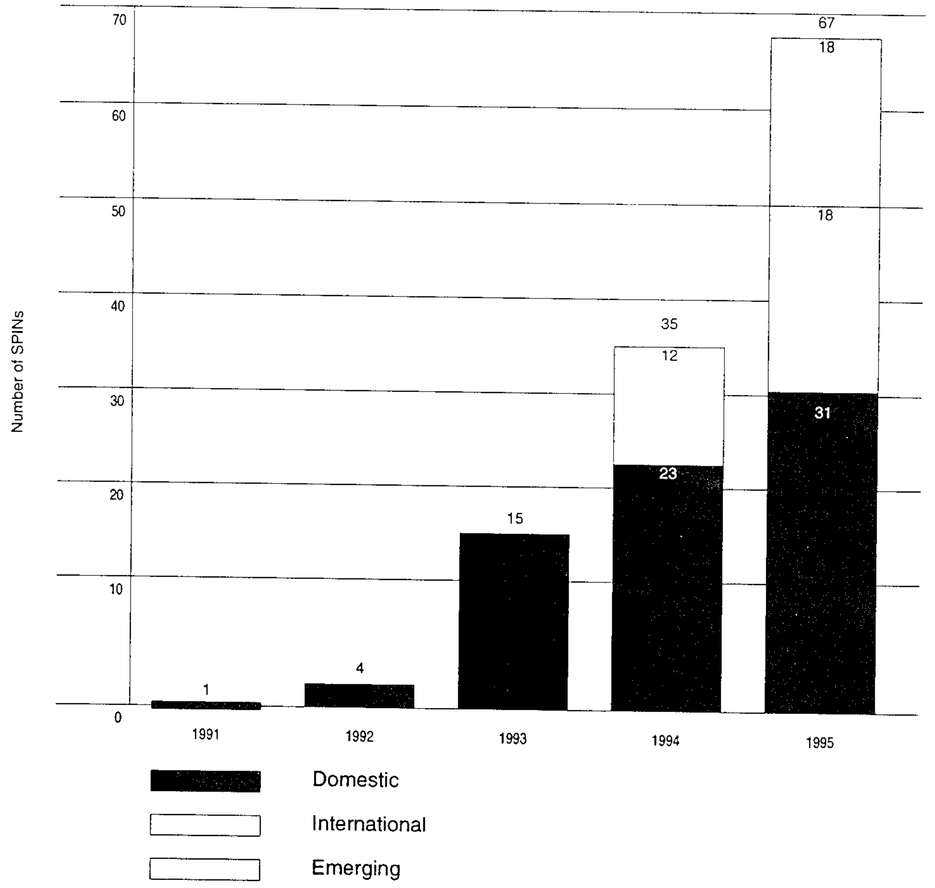


Figure 3-6: SPIN Group Growth

3.2.2 Five-Year Goals

The Process area has consolidated the four long-range goals from 1995 into three, and has reworded the statements into more "desired-state" goals. The more "action-like" content of last year's goal statements are now included in the five-year strategies. The three long-range goals are as follows:

1. *A culture of software engineering excellence is embodied in a model of engineering discipline and process maturity that is accepted and used throughout the software engineering community.*

   This primary goal is to transition the CMM for Software into the state of the practice of software engineering to improve the maturity of software development organizations as defined in the CMM. The products and services facilitate and enable such a transition.

   Evolution of the CMM, based on community input of proven organizational best practices at all levels of maturity, will support continued software process improvement. Involvement with international standards efforts will ensure the preparedness of U.S. industry in the global software market. Integration of the CMM for Software with models from related disciplines, such as systems engineering and software acquisition, will enhance the state of the practice in each discipline, including software engineering. Finally, customer support in training, tailoring, and interpreting the CMM will aid in its transition into practice.

2. *Organizations align their investments in continuous software process improvement such that the investments help them achieve their business objectives.*

   The second goal is to have organizations be able to identify their business objectives and understand the role of continuous software process improvement in helping to meet those business objectives. Additionally, our goal is to transition an integrated approach to SPI to the software community, as illustrated in Figure 3-7. An organization will be able to look to the SEI for a suite of products and services to determine where and how it should start and continue along the path of continuous process improvement. Those products and services will support the activities of an integrated approach to SPI.

   The ultimate goal here is that by the end of this century, all major commercial software-producing organizations, including those that serve the government community, will have instituted continuous SPI programs.

   A strong commercial infrastructure will be developed and in place to support SPI within the United States. This infrastructure will provide interorganization communication about lessons learned in process improvement, and wide competitive choices for satisfying the needs of the software community. Trainers and consultants will be in place for educating customers about organizational change, process improvement, and sustaining customer improvements. We will continue to pursue efforts to promote a commercial process improvement industry with quality standards for practitioners through organizations such as the International Organization for Standardization (ISO), for example, with ISO 9000 and SPICE (Software Process Improvement Capability dEtermination). We will also continue to work to ensure that the SEI and U.S. perspectives for SPI are addressed in such standards. In addition, university programs will teach SPI, including personal and team software processes.

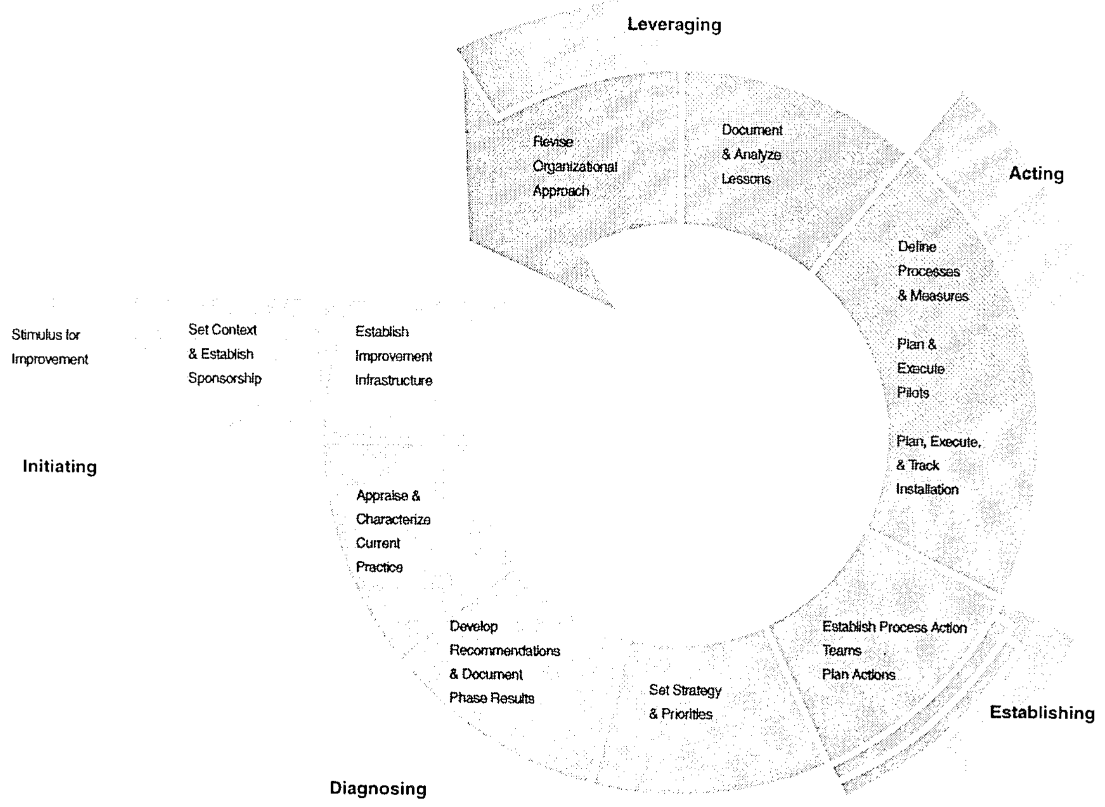

Leveraging
Revise Organizational Approach
Document & Analyze Lessons
Acting
Define Processes & Measures
Plan & Execute Pilots
Plan, Execute, & Track Installation
Stimulus for Improvement
Set Context & Establish Sponsorship
Establish Improvement Infrastructure
Initiating
Appraise & Characterize Current Practice
Develop Recommendations & Document Phase Results
Set Strategy & Priorities
Establish Process Action Teams
Plan Actions
Establishing
Diagnosing

Figure 3-7: The IDEAL^{SM}[1] Model: An Integrated Approach to SPI

[1] IDEAL is a service mark of Carnegie Mellon University

Process improvement programs, based on the SEI approach, are planned to dramatically improve the quality and reduce the costs and schedule slippage of software developed by and for government and commercial software-producing and -procuring agencies. We expect SPIN groups to continue to increase (see Figure 3-6). Total membership in the 49 (31 domestic U.S., 18 international) SPINs has grown from 3,400 in 1994 to approximately 10,000 individuals in 1995. Eighteen (9 domestic U.S., 9 international) are emerging as of this writing (see Figure 3-6). We also expect that each SEPG Conference will continue to attract in excess of 1,000 participants (see Figure 3-5).

Achievement of this goal will result in a national commitment to improvement and a service business to support it, so that by 1998, 80 percent of all defense contracting sites with more than 50 software engineers will have active process improvement programs, and 50 percent will have advanced one level from their initially measured maturity level. We expect that by 1999 this goal will address other government software producing agencies; and by 2000, commercial software development organizations. A sufficient number of contractors with defined, measured, and managed software processes will exist by 1999, so that government agencies will not need to use contractors with weaker software process capability for software intensive systems costing $10 million or more. Further, by 1999 all software systems procured from contractors with high process maturity (i.e., levels 4 or 5 of the CMM) should be delivered with fewer than one-tenth of a defect per thousand source lines of code.

3. *Organizations use quantitative information and methods effectively in managing software projects, measuring results of software products, measuring improvements in software processes, and benchmarking against industry practice.*

   This would appear to be a broadened goal from 1995; however, the definition of product and project measures, as well as process measures, remains consistent with support for transitioning the CMM into the state of the practice. These measures are interrelated and interdependent. Collection and use of meaningful data are paramount to improving productivity, quality, time to market, and ROI; and have been key elements of the Process area. Software engineering measurement is now an activity area.

Achieving these goals should make U.S. software developers among the most competitive in the world in terms of cost, quality, and time to market. This focus on process will generate requirements for process technologies that can be incorporated into software engineering environments (SEEs) which support automated processes and data collection.

3.2.3 Five-Year Strategies

Achieving these goals will require maintenance and evolution of the CMM. The CMM, developed at the SEI with much community participation, describes five stages that software organizations must achieve to realize a sustainable state of continuous process improvement. This five-stage model provides software organizations with guidance in planning a long-term process improvement program and assistance in setting priorities for near-term improvement activities.

The Process area has reworked last year's strategies to better achieve the reworked goals. Long-term strategies are described below.

- Promote the adoption, use, and advancement of the CMM for Software and related disciplines' maturity models.
- Continue to evolve the CMM for Software to incorporate innovative and effective software engineering and management practices.
- Provide core competency in maturity modeling.

- Integrate the CMM for Software with other related maturity models, frameworks, and standards.
- Define and advance a model for continuous software process improvement (the IDEAL model). Promote the adoption and use of the IDEAL model and the integrated product suite so organizations are able to transition continuous process improvement into practice.
- Develop an integrated suite of software process appraisal and improvement products to support the CMM for Software and the IDEAL model.
- Work with organizations to align their SPI programs with their business objectives. Assess and communicate the results of SPI across the software industry.
- Establish the SEI as a recognized national resource for analyzing and disseminating software engineering data, and for assessing the state of the practice of the U.S. software community.

Our strategy is to maintain the CMM as a community-owned model for which the SEI provides stewardship until a standards organization, e.g., ISO, adopts the CMM as the preferred standard. The CMM is constantly subjected to national and international review, has been stabilized at version 1.1, and has been accepted as the de facto standard for implementing SPI activities. It is now common practice in technical literature to simply say "SEI CMM level 1" without explanation.

The value of the CMM is realized by government and industry when its goals are met in practice. Thus, the Process area will work with its customers to broadly educate and inform them of the results they can obtain by maturing their software organizations' processes, in concert with other disciplines that have an impact on software.

Development of version 2.0 of the CMM began in 1995 with a workshop of 60 interested parties, coordinated and facilitated by the SEI. Results of this workshop have been widely distributed for review, and on-going focus groups are at work to continue development of several key aspects of CMM version 2.0. Future versions of the CMM must address new and changed requirements from the software engineering community. New key process areas (KPAs) need to be considered, further refinement of higher level maturity KPAs is needed, and change requests, such as having KPAs span levels, need to be investigated and incorporated into the CMM.

In addition to its role in the stewardship of the CMM for Software, the Process area has responded to various customer requests for maturity modeling support, including systems engineering, people, software acquisition, trusted software, etc. A core competency has been built in the Process area to support these requests for maturity modeling expertise. With additional maturity models and similar offerings in areas related to software organizations (e.g., systems engineering) becoming visible, the Process area has taken on the task of integrating these as appropriate. An architecture of maturity models is being developed.

In addition to the CMM, a second model (IDEAL) has been articulated (see Figure 3-7). The IDEAL model serves as an approach to a continuous improvement effort, outlining the steps needed to initiate and sustain such an effort. It has been piloted in multiple customer settings as the vehicle for transitioning improved software engineering practices into customer organizations. IDEAL itself should also be transitioned for the customer's use in continuing to improve over the long term.

To support the use of the CMM, the Process area continues to provide methods to reliably assess the maturity of an organization's development and maintenance processes, and evaluate the capabilities of their contractors. The SEI has transitioned the capability to train and conduct these diagnostic methods to third parties.

To affect the growth in the national software capability, these diagnostic methods must be coupled with the ability to plan and implement process improvement programs. This is reflected in the IDEAL model. Since these evaluation and assessment methods are rapidly growing in use and in importance in the software community, qualifications for lead assessors and assessment team members have been established, published, and enforced. Similar qualification guidance will be provided for evaluators along with the upgraded CBA software capability evaluation (CBA SCE) method.

After an assessment of process capability, an organization needs defined processes upon which to structure process improvement efforts. Level 1 organizations, in particular, are uncertain how to prioritize the activities necessary to start a process improvement effort. Several products related to process definition are essential to maintaining process improvement momentum. Identifying technology for these products, creating the products, and getting them into widespread use are underway and will be major activities during the coming five-year period.

The SEI is not supporting and promoting SPI for the sake of SPI. Rather, we want to be sure that our customers work on improvements that can be tied to their business objectives. We need to support customers' early efforts to use the IDEAL model to define their software-related business objectives, decide how they want to pursue SPI in line with those objectives, and decide how to predict and measure the results of their SPI efforts relative to those objectives. We have begun to address this difficult task more thoroughly and will continue to do so in the future.

The SEI and the Process area have a reputation as trusted, neutral brokers within the software community and are entrusted with data and information useful to the community at large. There is need for an expanded role to include broader categories of data than are currently housed and analyzed by the SEI. Going beyond process measures, there are needs for gathering, analyzing, and reporting on software engineering data, e.g., project and product measures.

The core measures defined by the Process area are a start in that direction. Additional measures, which are more broadly applied, will address broader software engineering domains and higher maturity levels. The SEI will be able to be more proactive and react more quickly to community requests with this enhanced focus on measurement.

Software measurement and database activities will also be integrated across the Process, Risk, and Disciplined Engineering areas to gain leverage from investments, technologies, and data.

3.3 Risk Management

3.3 Risk Management

The ever increasing complexity of software-intensive systems causes the amount of uncertainty in their development to grow exponentially. Engineers and managers are able to assess and manage risks on simple systems using personal knowledge and experience. Individual knowledge and experience, however, rapidly become inadequate when dealing with today's complex systems. Engineers, under schedule pressure, are forced to make technical decisions without the total picture of the risks involved, and therefore often make inappropriate decisions. No one person has the ability to understand all the risks and their interaction with each other. Technical decisions may lead to cost over runs and schedule delays that surprise management.

Surprises like these are often dealt with by seeking outside expertise to gain specific knowledge and experience. However, at some point systems becomes so complex that the interaction and communication among the managers, engineers, and experts must be facilitated and effectively managed. At this point, systematic and structured processes, methods, and tools for assessing and managing risk are imperative (see Figure 3-8). Combining a systematic approach with integrated product teams is a powerful new paradigm that addresses managing uncertainty during system acquisition and development.

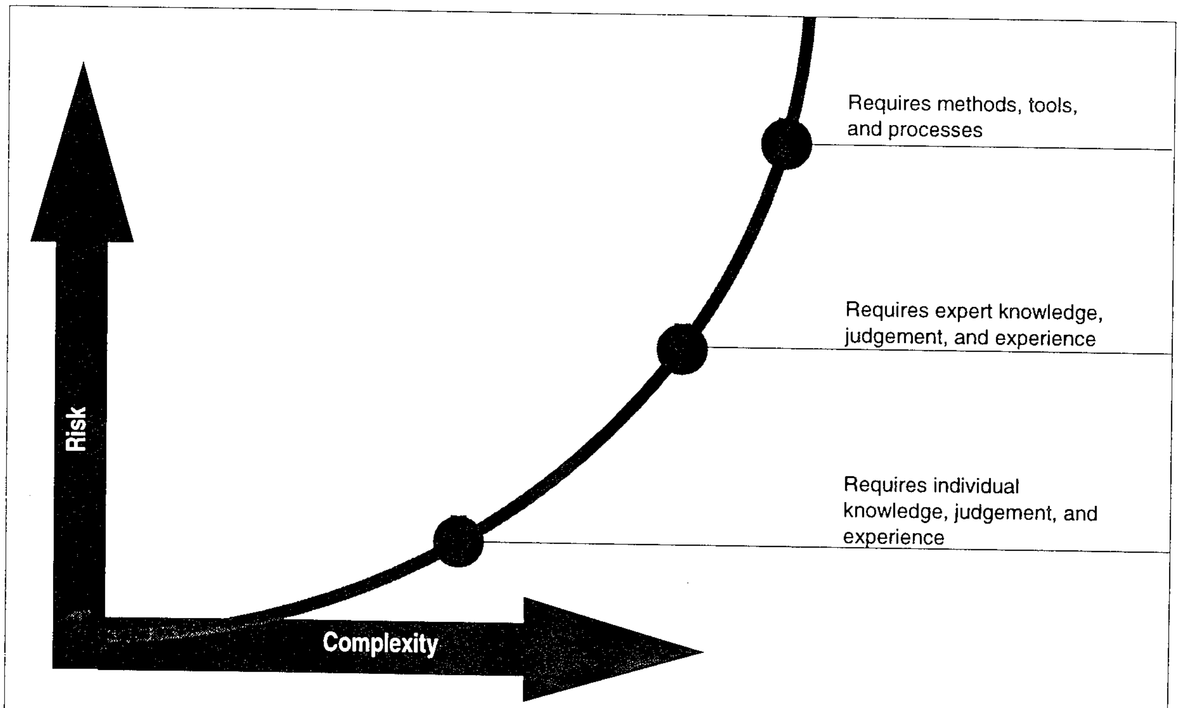

Requires methods, tools, and processes
Requires expert knowledge, judgement, and experience
Requires individual knowledge, judgement, and experience
Risk
Complexity

Figure 3-8: Risk Complexity

Integrated product teams (multi-disciplinary teams) need means to organize, communicate, and accomplish their work as effectively as possible. The lack of explicit means to organize and communicate accounts for some of the conflict seen in the integration of products and processes in system development today. Area specialists understand the need for their particular discipline in the overall system development process but not necessarily how it interacts with and affects other disciplines. There are numerous examples where multiple disciplines or factors affect decisions, including the following:

- Cost and schedule decisions affect technical performance.
- Requirement changes affect existing architectures.
- Vendor changes in computer-aided software engineering (CASE) tools affect the system design process.
- Development process changes affect product schedules.

Multi-disciplinary teams that use continuous, systematic risk management processes are most capable of making informed decisions and achieving successful programs.

A risk-aware approach means making choices about opportunities rather than reacting to surprises. Proactively addressing uncertainty is a rational approach for developing and implementing sound program acquisition and development strategies—strategies that weigh program opportunities and risks from both business and technical perspectives. For example, one can develop a program strategy for reengineering a product by weighing the advantages of the gains from new technology and processes with the uncertainties and possible failures and negative consequences. Reengineering, reuse, large-scale development, and unprecedented technology are a few examples where risk management is vital to acquisition and development programs. By focusing on risk management, programs are able to identify and analyze software technical uncertainty and present the decision makers with the right information in a timely fashion.

Since practical, well-structured risk management methods generally do not exist for software, the Risk impact area is developing processes and supporting methods that systematically identify and analyze software technical uncertainty. We seek to improve software-intensive system acquisition and development by identifying key risk management practices and the methods by which they can be integrated into software development processes and program management.

In this chapter, the sections for this impact area are as follows:

- **Description**
  - Context
  - Key Value-Added Contributions
- **Five-Year Goals**
- **Five-Year Strategies**

3.3.1 Description

3.3.1.1 Context

Software risk management contributes to maturing not only the process of software development described in the SEI strategic framework earlier in this volume, but also maturing software system technology by identifying and analyzing the technical issues associated with the development of software-intensive systems. The Risk area contributes to maturing the process by defining and modeling processes for assessing and managing software development risks. These processes are depicted in the risk management paradigm (see Figure 3-9).

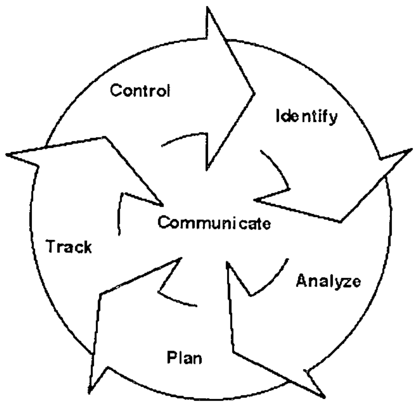


Figure 3-9: Risk Management Paradigm

During 1995, the Risk area expanded its research and development by building on its software acquisition experience and its field work in risk management. The Risk vision expanded its strategy to include the transitioning of technologies to support decision making under uncertainty in software acquisition and development. We are addressing the following issues:

- how acquisition agencies can buy smarter, improving their practice and managing risks
- how organizations can identify technical and programmatic opportunities and risks more effectively
- how organizations can manage more effectively by proactively addressing uncertainty
- how organizations can use and sustain knowledge and information within an organization

The program has examined these issues and grouped this work into the following three areas (Figure 3-10):

- software acquisition
- software risk management
- knowledge and information technology

Software Acquisition. Work began in 1994 to address software acquisition by building a Software Acquisition Maturity Model (SAMM), a model for improving the process of acquiring software-intensive systems. This activity is building on the work started within the Services and other Department of Defense (DoD)-sponsored organizations in the past three years. The SAMM work parallels other maturity modeling efforts in the SEI and is benefiting from the experience and lessons learned.

To guide implementation and institutionalization of software acquisition improvement, the SAMM must be augmented by a framework and roadmap to guide improvement activities. This framework will identify candidate practices and supporting technologies, expertise, infrastructure, and implementation guidance to satisfy the requirements of the key process areas of the SAMM. The roadmap will show a path through the possible improvement choices provided by the model, identify practices for which implementation guidebooks are needed, and include measures of success of the improvement activity.

Acquisition risk management guidelines will provide practical "how to" practices for selected key process areas. These guidelines will leverage the lessons learned in risk management and build on earlier work that provided risk management guidance to the source selection process (Figure 3-10).

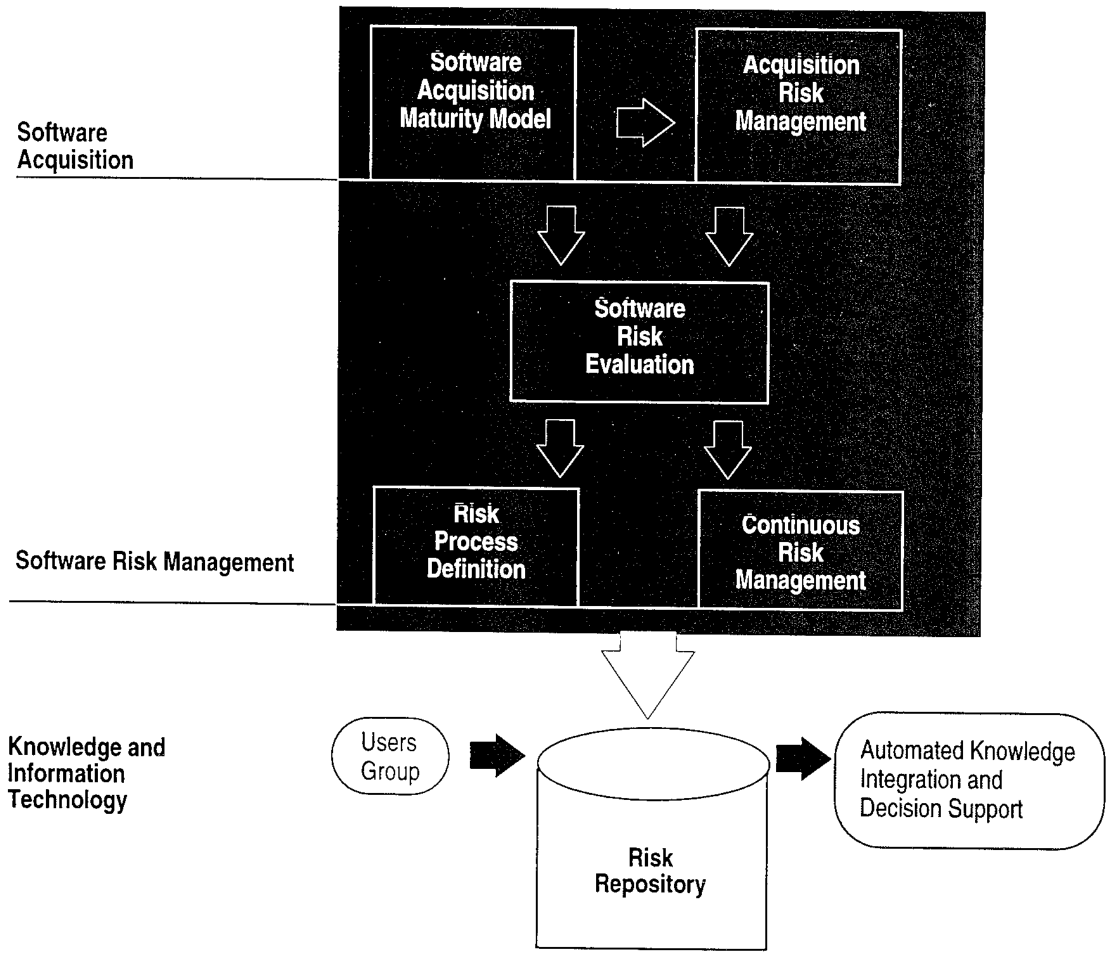

Software Acquisition
Software Acquisition Maturity Model
Acquisition Risk Management
Software Risk Evaluation
Risk Process Definition
Continuous Risk Management
Software Risk Management
Knowledge and Information Technology
Users Group
Risk Repository
Automated Knowledge Integration and Decision Support

Figure 3-10: Risk Area

Software Risk Management. Software risk management contains our most mature technology and experience. We have developed a model for continuous risk management and populated it with field-proven methods and tools, enabling organizations to adopt risk management as effective practice. We advanced our methodology, harmonized a complete suite of products, and initiated a focused effort on risk-driven metrics.

Software Risk Evaluation (SRE) identifies risks with recommendations for mitigation strategies. SRE has proven effective in acquisition and development programs as an independent risk assessment, as a diagnostic for identifying system and programmatic improvement opportunities, and as a baseline for continuous risk management. SRE has been used in both acquisition and development organizations and generally used to initiate risk process definition and continuous risk management (Figure 3-10).

During 1995 we developed a collaborative approach to installing risk management into organizations. Referred to as the Risk Clinic, it is an interactive workshop using a risk management guidebook and templates to define an organization's risk management process (risk process definition). The guidebook is the guide to tailor a risk management process based on the organization's culture and infrastructure. The addition of coaching over a transition period and the risk management guide as a reference to the Risk Clinic provides an effective means to install continuous risk management in an organization.

Along with an aggressive initiative to develop risk metrics, work is continuing with the University of Southern California-Center for Software Engineering (USC-CSE) in defining Constructive Cost Model (COCOMO) 2.0 cost drivers for integrating commercial off-the-shelf (COTS) software into software-intensive systems. This work has the potential to be to an effective means of collecting and analyzing return on investment data for managing risks.

Knowledge and Information Technology. Knowledge and information technology began with our earlier success in using contextual analysis and natural language processing to access, structure, manage, and apply data from risk assessments. We are further investigating decision making support technologies particularly automated support (for example, groupware or computer supported cooperative work). We have demonstrated a prototype and plan to develop a repository enabling community access to information about common risks, mitigation strategies, and lessons learned that match prescribed program profiles. Because the repository concept has much greater potential as an information repository for software engineering, the technology is being applied across the SEI in an effort to bring together other information sources such as SEI process improvement and measurement data to move this work toward establishing a software engineering information repository (see Section 1.3 of Volume II for further discussion of this use of this technology).

3.3.1.2 Key Value-Added Contributions

Risk management processes, methods, and tools provide the conceptual foundation and building blocks to establish and institutionalize risk management in the acquisition and development of software-intensive systems. We are adding value in the community by

- providing practical methods for conducting risk identification, analysis, planning, tracking, and control
- providing a comprehensive training course for risk management
- establishing standards in risk management by working with standards groups
- defining a maturity model to aid acquisition agencies. This activity utilizes SEI CMM expertise.
- providing a repository of software information for the community to access for software process improvement and managing risk
- providing risk drivers to cost and schedule estimation and quantitative methods to determine return on investment in managing risk

3.3.2 Five-Year Goals

To raise the maturity of the practice and management of software engineering, the Risk impact area is developing and transitioning products and services enabling software engineers and managers to more efficiently identify and manage technical uncertainties in the development of software-intensive systems. This improvement is being achieved by providing proven and tested methods and processes in the form of education, training, guidebooks, installation services, and risk data aggregated from government and civilian organizations.

There are three primary goals by 2000:

- At least three organizations are reporting higher successes in meeting cost, schedule, and performance requirements by using SEI's software acquisition products and services.
- At least five organizations are reporting higher successes because they know their risks and know how to manage them.
- At least two programs have improved decision making capabilities by adopting information and knowledge technologies.

Very little data exist today for software engineering on which to base quantitative measures of success in risk identification and management efforts. By 2000 we will have in place a system allowing quantification and tracking of method effectiveness in reducing risk, and evidence that our techniques are effective. In the meantime, we are using the following as measures of success:

- evaluations provided annually by our technical objectives and plans (TO&P) customers via "sponsor feedback forms"
- number of clients who collaborate with us in independent risk assessments and team risk management activities
- number of organizations adopting the SEI risk management process
- number of attendees at risk identification training courses

We expect that by 2000, programs will be using validated risk management methods systematically in acquisition and development throughout the life cycle. Also, a new paradigm of team risk management will bring the customer and developer together in a cooperative spirit of proactive risk management. In addition, we expect to achieve the goals shown in Figure 3-11 by 2000.

Activity Areas	Goals
Software Acquisition	• Risk management will be established in government and civilian standards with documented methods and processes. • System acquisition strategies and decisions will be based on information from proactive, systematic, and validated risk management methods. • Formal program reviews will be conducted using systematic and validated risk evaluation methods. • The Defense Systems Management College (DSMC) will have courses on software risk management in acquisition. • The DoD will have adopted team risk management as its acquisition and development practice.
Software Risk Management	• Major DoD contractors will have incorporated risk management into their software development process. • Risk management will be established as an organizational capability, executed on a continuous basis, and integrated within the context of program management. • Commercial training organizations will have continuing education offerings on software risk management. • Major government programs will have integrated customer and developer risk management activities—team risk management. • Team risk management will have integrated technical, cost, and schedule risk management into continuous, routine program management in major programs.
Knowledge and Information Technology	• The SEI will have an established national repository of risks and supporting risk reduction strategies based on information gathered from strategic partners and user networks. • Organizations are using SEI groupware adoption guidelines. • Knowledge integration technologies will be used to support organizational memory capability and decision making.

Figure 3-11: Risk Goals for 2000

3.3.3 Five-Year Strategies

The strategies in each activity area generally follow this approach:

- Define a framework or model to improve a customer capability.
- Establish guidelines to improve practice and technology adoption.
- Pilot test process, methods, and tools that provide improvement.
- Package and transition these processes, methods, and tools.
- Raise the community awareness and channel feedback of customer needs.

Our strategies are detailed in Figure 3-12.

Activity Areas	Strategies
Software Acquisition	• Build and mature a framework for software acquisition improvement. • Develop guidelines for key software acquisition processes. • Interact with the community to establish awareness, to ascertain needs, and to receive feedback. • Pilot test and transition methods and tools.
Software Risk Management	• Mature a risk management framework integral to program management. • Mature guidelines for managing risk. • Develop a cost model using risk factors as drivers/inputs. • Interact with the community to establish awareness, to ascertain needs, and to receive feedback. • Pilot test and transition methods and tools.
Knowledge and Information Technology	• Build and mature a framework to understand the knowledge integration technologies. • Develop technology adoption guidelines supporting decision making. • Develop and make accessible a risk management repository. • Interact with the community to establish awareness, to ascertain needs, and receive feedback. • Pilot test and transition methods and tools.

Figure 3-12: Five-Year Strategies for Risk Area

To be effective, we must not only provide methods for identifying and managing technical uncertainty, but we must also provide methods to facilitate the communication of risk issues. In addition to the usual cultural barriers, the common negative perception of risk makes change even more difficult. The communication process must de-personalize risks so they are viewed as opportunities for program success. All the method development and field testing activities directly address communication to enhance or enable effective communication.

In the acquisition and development areas, products will establish capabilities to evaluate software technical risks for programs, either by a customer or independent agent. The potential products cover risk evaluation and independent risk assessment methods and their application within the acquisition community.

Team risk management promotes a new paradigm of shared commitment to manage program risks as a team of customer and supplier (e.g., government and industry). Team risk management creates an environment built upon a set of systematic processes, methods, and tools that enable the customer and supplier (developer) to work together to manage risks throughout the life cycle. It is built on a foundation of risk management and cooperative team principles. This effort is being pilot tested with the Navy.

The SRE, which uses the Taxonomy-Based Questionnaire, is established as a common method for not only assessing risks, but also the impact of technology on software-intensive systems. The combined impact of the SRE method and a repository of risk information provides the community with both a diagnostic capability and a knowledge source. The SRE provides a diagnostic for discovering both technology and programmatic opportunities as well as risks. The repository provides the knowledge and information to take action and make better decisions.

To ensure a successful transition strategy, we are approaching transition systematically by targeting products to leverage limited SEI resources and to support adopting and sustaining the technologies. This includes community *awareness* activities such as conducting the annual SEI Software Risk Conference, presenting risk management tutorials at conferences, Software Process Improvement Networks (SPINs), and customer and developer working groups. These activities provide the additional benefit of providing feedback on our work. Education, training, collaboration, and publications will be our primary instruments for affecting understanding. For example, education and training include the risk identification training course and software risk management tutorials. Collaboration is addressed by the field activities to test specific methods with our government and industry partners. *Installation* is addressed by teaching leadership courses on risk management and conducting independent risk assessments. Finally, we are addressing *adoption* and *institutionalization* by conducting and transferring team risk management, risk management improvement, and independent risk assessment activities.

For a detailed summary of activity areas and work outputs, see Volume II.

3.4 Disciplined Engineering

Demand for software-intensive systems (henceforth referred to as systems) is constantly increasing, and these systems are growing in number, size, and complexity. More effective and efficient software engineering practices must be employed to cope with the challenges created by the increasing numbers of large-scale, complex systems. The challenges can be characterized as follows:

- Systems have been built in isolation, resulting in stovepipe solutions even for similar systems. Each must therefore be maintained separately. The result is continued escalation in the number of systems and the cost of their maintenance due to lack of leverage from commonality and variability.
- Systems are becoming larger; their requirements are not completely known at initiation and continue to evolve due to changes in the environment in which they are deployed. Without flexible architectures, affordable integration, and dependable upgrading, systems become outdated, have long turnaround on upgrades, and suffer in quality during upgrades.
- Systems are deployed in everyday situations and have a critical impact on our ability to function as a society. Without quantification, analysis, and prediction of quality attributes of engineering alternatives and impact of changes, confidence in the continued quality of systems in a continuously changing world remains low.

Successful practices in mature engineering disciplines are based on the use of models and architectures for system description, systematic quantification, analysis and prediction of system properties (quality attributes), and engineering trade-offs in architectural design alternatives. Organizations achieve economies of scale by recognizing product families. For unprecedented systems, identification of appropriate capabilities—and experience in their construction—is gained through rapid assembly of prototypes and quick evolution of systems into mature products that meet customers' needs. The quality of products is maintained through rigorous application of analysis and prediction, continuous improvement and refinement of quantitative models through designed experimentation, and feedback from observations of actual systems.

Several studies—for example, the Government Accounting Office (GAO) reuse report (January 1993), the National Research Council report *Software Engineering: Scaling Up* (1991), and the Defense Science Board report (1994), as well as the Deputy Director of Research and Engineering (DDR&E) (SEI Symposium 1994)—have indicated the appropriateness of using system models and architectures to advance software engineering practice. Translated in terms of software engineering, we refer to this vision of practice as model-based software engineering (MBSE). A practice of MBSE consists of system analysis and synthesis based on models, and is supported by automation of tasks, processes, and information access (Figure 3-13).

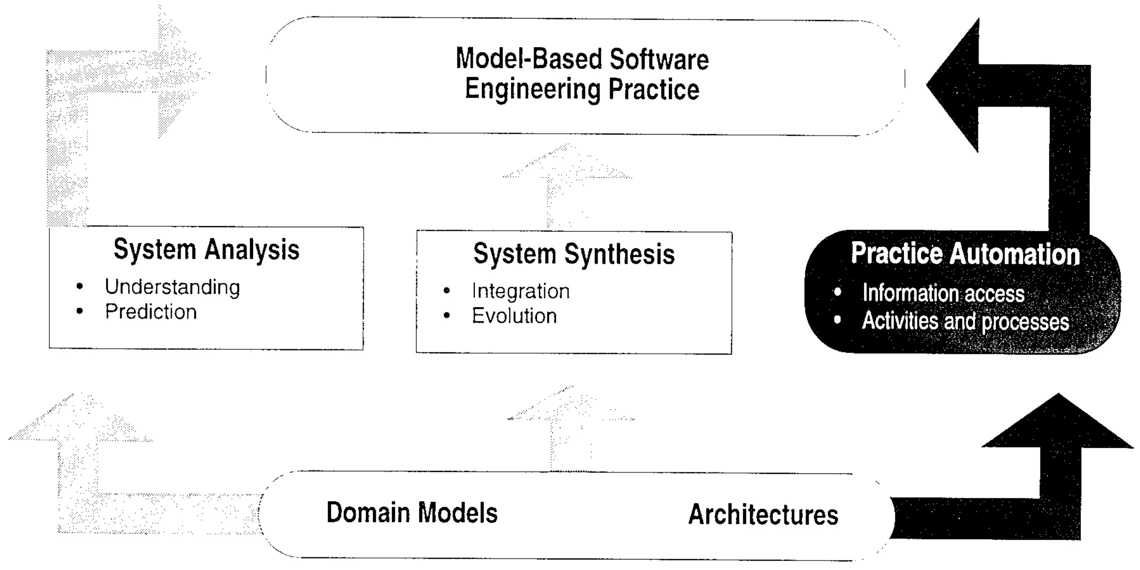


Figure 3-13: Model-Based Software Engineering (MBSE)

Based on these principles of MBSE, the problem areas outlined above are addressed in terms of three thrusts:

- **product line engineering**: engineering and reengineering of software-intensive systems from a product line perspective; utilizing domain models and architectures to leverage identified commonality and variability in a set of systems
- **evolutionary engineering architectures**: rapid, dependable evolution of systems through flexible, robust architectures and affordable system integration
- **predictive engineering**: predictable confidence in the quality of software-intensive systems through quantitative analysis and through the prediction of quality attributes

Figure 3-14 illustrates how these thrusts leverage MBSE principles, which mature from a descriptive nature to analysis, quantification, and prediction. They complement each other to improve software engineering practice.

The expected impact of this area on the state of practice in software engineering is as follows:

- reduction in the number of separately developed and maintained systems, and in their development and maintenance cost
- increase in the number of systems able to maintain the pace of change at an affordable cost both in customer need (requirements change) and in the operating environment (technology change)
- increase in the quality of systems at an affordable cost

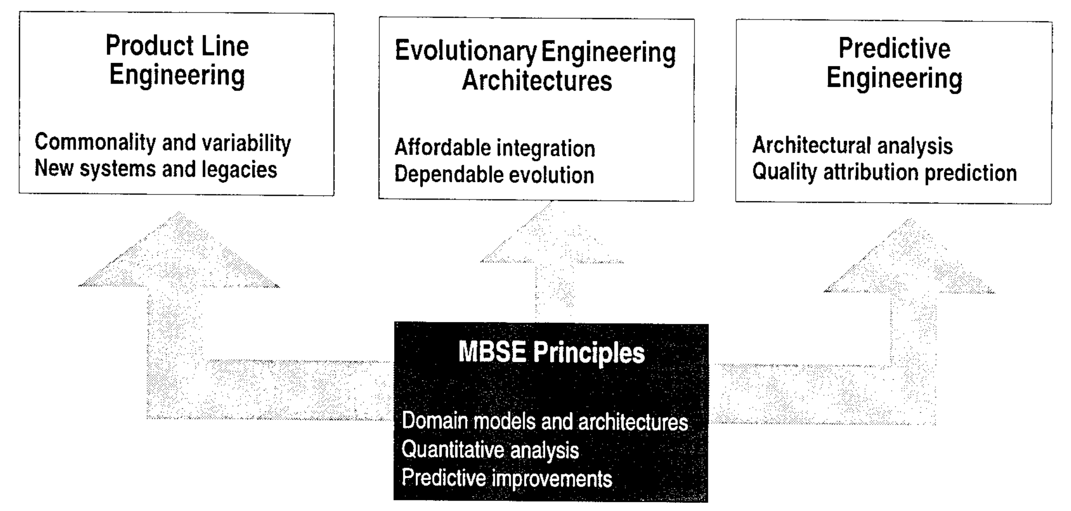


Figure 3-14: Three Thrusts of MBSE

Building on the principles of MBSE, improvements of each thrust in terms of cost, time, and quality have a multiplying effect. Our ability to adapt to change in the operating environment, when pursued in the context of product lines, reduces the number of separately maintained systems to which changes must be introduced. Similarly, improvements in quality due to engineering alternatives apply to all members of a product family. The benefits of MBSE are illustrated in Figure 3-15. In terms of efficiency and flexibility, MBSE maintains flexibility to adapt to changes in a more disciplined fashion, while increasing efficiency. In terms of redundancy and cost, MBSE reduces redundancy by recognizing commonality in systems.

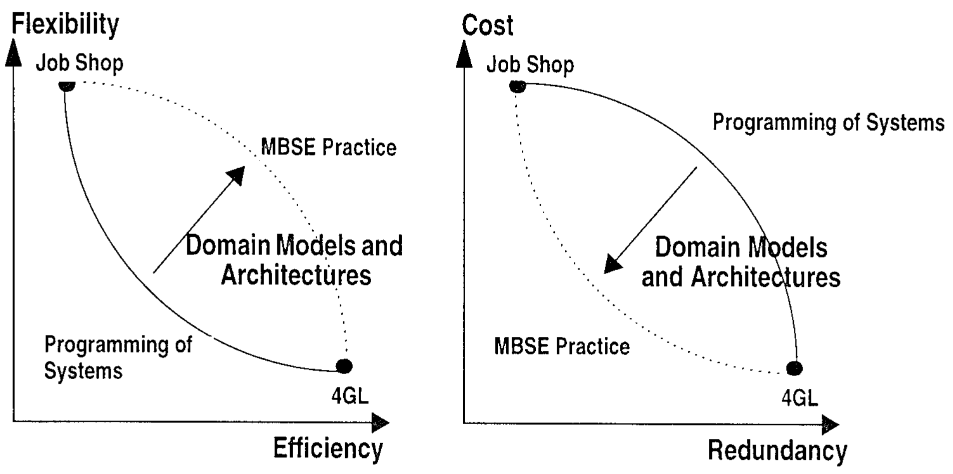


Figure 3-15: Benefits of MBSE

In this chapter, the sections for this impact area are as follows:

Description ▶ Context
Key Value-Added Contributions

Five-Year Goals

Five-Year Strategies

3.4.1 Description

3.4.1.1 Context

Disciplined Engineering focuses on maturing technology in the SEI strategic framework. A focus on technology advances in terms of a paradigm shift results in measurable improvements in software engineering practice. For Disciplined Engineering, this paradigm shift is characterized as MBSE. The focus is on improving the effectiveness and efficiency in engineering software-intensive systems by leveraging recognized commonality and variability in different systems, by increasing an organization's ability to adapt to change through continuous system evolution, and by increasing the organization's ability to improve the quality of the product through analysis and prediction of quality attributes in light of engineering alternatives.

Such a paradigm shift also has an impact on maturation of the software engineering process and of the software engineering profession. Implications for process are definition and modeling of processes to evaluate technologies to advance practice, processes to make engineering choices, and processes centered around product lines and system evolution. Implications for maturing the profession are an increasing body of engineering knowledge that practicing engineers should be familiar with and able to apply in practice.

In 1995 Disciplined Engineering focused on integrating concepts pursued by two previously separate areas (Software Engineering Techniques and Product Attribute Engineering) to contribute to the maturation of technology. One result was the principle of MBSE: the description of software-intensive systems in terms of domain models and architectures and the engineering of systems based on these models (also referred to as domain engineering and application engineering). This practice has advanced toward quantification, analysis, and prediction of product quality attributes (in the context of an Engineering Maturity Model [EMM]). A second result was the alignment of eight previously separate activity areas into four areas (architectures and models, product quality engineering, automation of practice, and maturation of practice), with a major emerging thrust in software architectures.

During 1995 it has become evident that

- The principles of MBSE were better described in terms of product line engineering, evolutionary engineering architectures, and predictive engineering.
- The work on computer-aided software engineering (CASE) environments had a strong flavor of affordable system integration of commercial off-the-shelf software (COTS) components, and had applicability to other domains.
- The work on effective access of software engineering information had synergy with work on a risk repository in the Risk impact area.

This resulted in the realignment of Disciplined Engineering in terms of three activity areas for 1996 and beyond. The three activity areas correspond to the three thrusts described earlier: Product Line Engineering, Evolutionary Engineering Architectures, and Predictive Engineering.

3.4.1.2 Key Value-Added Contributions

The SEI is in a position to address evaluation of promising technology, maturation of technology into practice, and leveraged transition that results in accelerated improvement of practice. Effective evaluation of promising technology requires a common approach for measurable assessment of the technology through designed experiments. Effective maturation of technology requires building on experiences and insights of innovators and early adopters to mature the use of particular technology into an effective practice. Leveraged transition requires cooperation with transition enablers and owners of transition infrastructure to reduce adoption barriers. The SEI is chartered to fulfill the role of evaluating, maturing, and transitioning technology, albeit with limited resources, and Disciplined Engineering is pursuing a strategy of leveraging community involvement as well as collaborating with other SEI impact areas.

The key value-added contribution of each thrust to software engineering practice is as follows:

- **Product line engineering** (engineering and reengineering of systems from a product line perspective). This thrust focuses on understanding the commonality and variability in different systems by treating the systems as a product line. The result is a reduction in the number of distinct systems to be designed and maintained and a reduction in the time it takes to produce a system (product cycle time reduction).
- **Evolutionary engineering architectures**. This thrust focuses on affordable integration of large complex systems in a flexible and dependable manner. It allows systems to rapidly change their capabilities and adapt to changes in the environment, while maintaining selected quality attributes even under hard real-time conditions. The result is reduced release cycle time at affordable cost and rapid adaptation to changes while maintaining product quality standards.
- **Predictive engineering**. This thrust focuses on our ability to systematically analyze systems, to understand the consequences of choosing among engineering alternatives, and to reliably predict quality attributes of systems. The result is engineering trade-offs with predictable results, leading to increased product quality at a justifiable cost (predictable confidence).

The key value-added contribution of the SEI to each activity area is as follows:

Product line engineering. Approaches of domain and application engineering to improving the effectiveness and efficiency of software engineering practice have been explored through a number of initiatives and programs. These include domain-specific software architectures (DSSA), Software Technology for Adaptable, Reliable Systems (STARS), Prototech, the Electronic Systems Center's Portable Reusable Integrated Software Modules (ESC PRISM), Comprehensive Approach for Reusable Defense Software (CARDS), the Software Productivity Consortium (SPC), and the SEI, as well as commercial pilots of software system product lines. The SEI is in the opportune position to build on these legacies and exploit the dual use of this technology to improve the practice in both defense and commercial settings. In collaboration with external partners, the SEI can continue the maturation and transition of product line engineering by providing decision-making frameworks and strategies for evolving current practices to an effective product line engineering practice.

Evolutionary engineering architectures. Factors influencing the need for evolvable systems include continuous advances in technology and changes in the operating environment, and the need for rapid evolution of system capabilities to meet user needs. COTS and open systems are currently seen as important contributors to affordable system integration and flexibility in system evolution. As systems increasingly control critical functions of our society, safe online evolution of systems in operational settings is becoming increasingly important. Working with external partners, the SEI has the opportunity to evaluate a number of technologies and mature them into a synergistic solution, demonstrating its viability in the context of key application systems of national importance (for example, the Federal Aviation Administration). The SEI can build on its work in open systems, evaluation of integration technologies, simplex architecture for dependable real-time systems, and evaluation of software architectures to establish a reference architecture for evolvable systems.

Predictive engineering. This activity area focuses on prediction and control of quality attributes of systems and the trade-off of these attributes across alternative designs. Researchers focus on individual attributes (e.g., reliability, security, fault-tolerance) and seldom look at combinations of attributes, while practitioners need guidance for selecting congruent technologies. For selected groups of quality attributes, the SEI, in collaboration with external partners, will identify and demonstrate the range of possible trade-offs based on experiences with real systems, and make the results available to practitioners in roadmaps and in guides to best practice. Improved practices provide predictable confidence in product quality. Mature architectures provide predictability and control over desired combinations of attributes. The SEI will survey and analyze existing and proposed architectures and provide models for selecting architectures that meet desired combinations of quality requirements.

3.4.2 Five-Year Goals

Disciplined Engineering has three long-range goals:

- **in product line engineering**: Industry leaders routinely engineer software-intensive systems from a product line perspective. The SEI customer community recognizes the benefits in terms of reduced time to market and a reduced number of separately maintained systems.
- **in evolutionary engineering architectures**: Dependable evolution of selected, large, complex software-intensive systems of national importance has been demonstrated. The SEI customer community recognizes the benefits in terms of increased responsiveness in adaptation to change at affordable cost and without loss in product quality.
- **in predictive engineering**: Software engineering practice matures toward quantitative analysis and predictability of systems through commonly agreed upon measures of quality attributes and through availability of technology to maintain predictable confidence in system quality. Early adopters will demonstrate the benefits through their piloting of predictive engineering techniques.

3.4.3 Five-Year Strategies

Disciplined Engineering's strategies include technical strategies and operational strategies.

The technical strategies are as follows:

- Use the principles of MBSE as the common foundation for the three thrusts of Disciplined Engineering.
- Evolve and apply a method for evaluation and maturation of promising technology to mature MBSE toward quantitative analysis and predictable confidence.
- Build on the work in domain engineering and reengineering at the SEI and in the community.
- Develop strategies and transition packages for effectively transitioning toward product line engineering practice.
- Build on affordable system integration through open systems, use of COTS and integration technology, and on architectures for dependable, evolving systems (Simplex).
- Develop strategies and transition packages for effective use of open systems and COTS.
- Mature technology advances in dependable system evolution for effective transition to an evolutionary engineering practice.
- Evolve an engineering decision framework, based on experiences with actual systems, and mature it towards objective, quantitative, and predictive measures of product quality attributes.

Figure 3-16 shows graphically the three thrusts along a time axis. Disciplined Engineering is pursuing all three thrusts. For each thrust, the figure shows the different degrees of technology maturation and the expected transition period to the SEI's broader customer community.

The operational strategies of Disciplined Engineering are as follows:[1]

- Collaborate with technology providers and innovators (technology risk-takers) to systematically evaluate and identify promising technology.
- Use the experience of innovators and early adopters (local, respected "missionaries" who give advice and confidence to potential adopters of a new technology) in the community to mature the practice.
- Collaborate with transition enablers and owners of the transition infrastructure to affect the early majority (those who adopt a new technology just before everyone else does) through a leveraged transition approach.

1. The terms innovator, early adopter, and early majority come from Rogers, Everett M. *Diffusion of Innovations*, 3rd ed. New York: The Free Press (A Division of Macmillan Publishing Co., Inc.), 1983.

- Adapt capabilities and mechanisms from other SEI impact areas to benefit the SEI customer community—in particular, improvement and technology transition frameworks, the software engineering knowledge repository evolving in the Risk area, interactive training product technology as an enabler for effective transition of information and knowledge, and Trustworthy Systems' approach to addressing security issues.

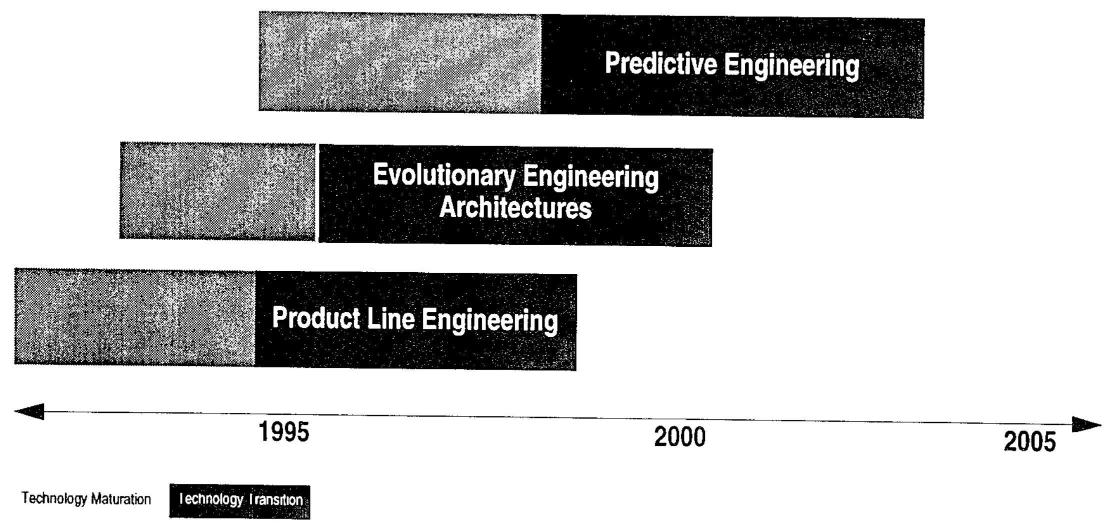


Figure 3-16: Timeline of MBSE Thrust Maturation

Figure 3-17 graphically illustrates our operational approach to working with the community in achieving technology evaluation, maturation, and leveraged transition. Disciplined Engineering cooperates with a number of the SEI's customers and strategic partners to accomplish the various tasks in the three thrusts. Customers and strategic partners include the Advanced Research Projects Agency (ARPA) Software Composition program, the ARPA Evolutionary Design of Complex Software (EDCS), the ARPA Advanced Distributed Simulation, the National Institute of Standards and Technology (NIST) Computer Science Laboratory (CSL), the NIST Manufacturing Engineering Laboratory (MEL), the Institute of Electrical and Electronic Engineers / Association for Computing Machinery (IEEE/ACM), portable operating system interface (POSIX), the International Organization for Standardization (ISO), the Federal Aviation Administration (FAA), Aeronautical Systems Command (ASC/YT), the Communications-Electronics Command (CECOM), the Electronic Systems Center (ESC), Joint Advanced Strike Technology (JAST), the Office of the Under Secretary of Defense for Acquisition (OUSDA), the Office of the Assistant Secretary of Defense (OASD); and industrial endeavors such as SEMATECH, the computer-integrated manufacturing community, and individual companies.

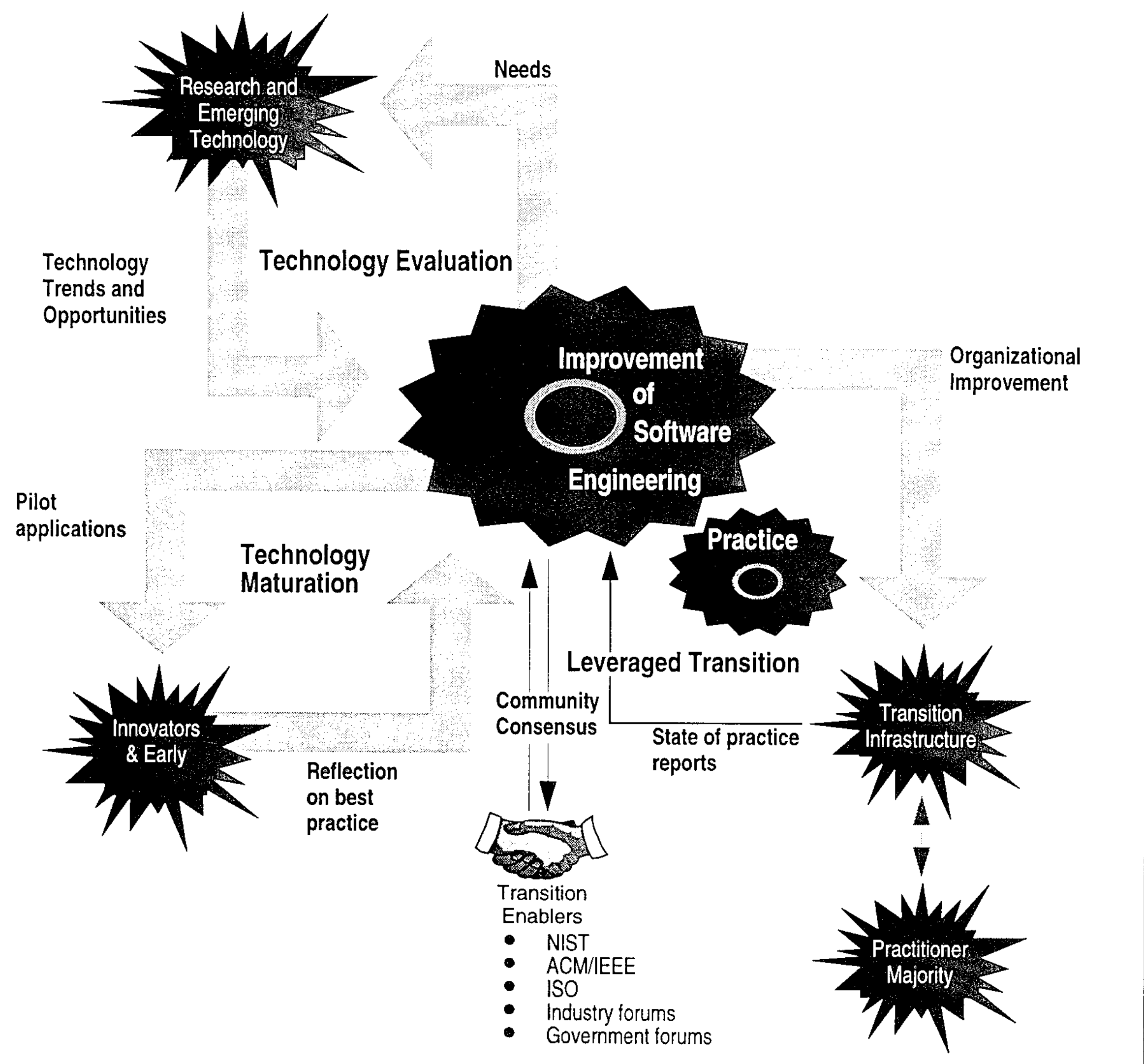


Figure 3-17: Evaluation, Maturation, and Leveraged Transition

3.5 Trustworthy Systems

In November 1988, in response to an automated attack on thousands of Internet-connected computer systems, the Advanced Research Projects Agency (ARPA) established the CERT^{SM}[1] Coordination Center at the SEI. The CERT team is chartered to work with the Internet community in detecting and resolving computer security incidents as well as taking steps to prevent future incidents. Our specific mission is to

- Provide a reliable, trusted, 24-hour, single point of contact for emergencies.
- Facilitate communication among experts working to solve security problems.
- Serve as a central point for identifying and correcting vulnerabilities in computer systems.
- Maintain close ties with research activities and also conduct research to improve the security of existing systems.
- Initiate proactive measures to increase awareness and understanding of information security and computer security issues throughout the community of network users and service providers.

When the CERT Coordination Center was created, the Internet had 80,000 host computers. Since then, the network has grown to more than 4.8 million hosts with an estimated 30 million users as of January 1995. During that same period, the computer security incident rate proportionately increased. (See Figure 3-18.)

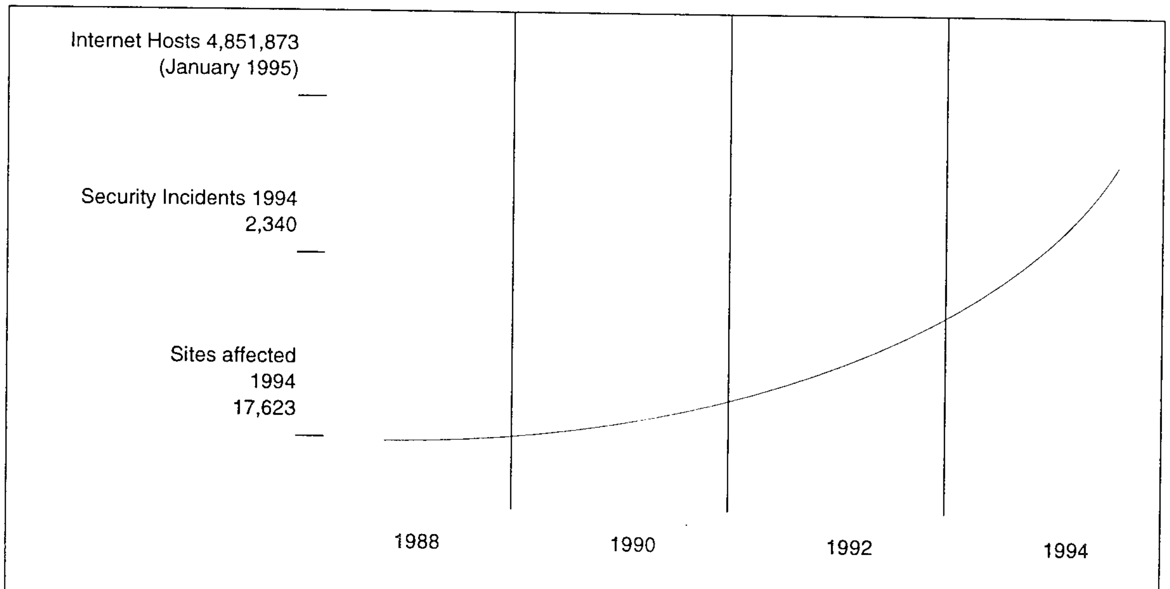


Figure 3-18: Scope of the Problem

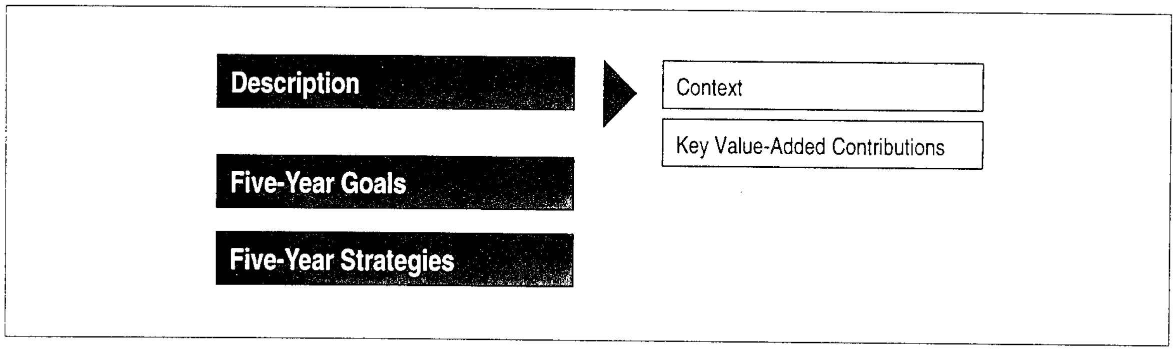
[1] CERT is a service mark of Carnegie Mellon University.

As the Internet becomes larger and more complex, as the National Information Infrastructure (NII) evolves, and as more organizations depend on off-the-shelf technology and open-access wide-area networks, the frequency and severity of unauthorized intrusions into systems connected to these networks become increasingly serious. Moreover, the Department of Defense and other government and commercial organizations increasingly rely on computer networks that extend outside their own walls. They share information with their suppliers, partners, and customers. Effective use of computer networks is becoming a necessity for many organizations; however, use of networks also leads to an increased risk that valuable or sensitive information will be lost, stolen, corrupted, or misused.

The integrity and availability of the data stored and processed on computer networks are at stake, as are the operation of the networks themselves and the organizations that use the networks. Traditional security measures are not sufficient because of the open access of the networks, the open architectures used to build the networks, and the widespread and distributed nature of the intrusions.

The Trustworthy Systems impact area is an outgrowth and planned extension of the CERT activity. Our goal is to help the software-producing and -using communities build and maintain confidence in software-intensive systems by decreasing the risks of computer security incidents. While other system security efforts provide protection through compartmentalization, classification, and extensive evaluation of products, our effort focuses on developing and transitioning information security and computer security practices that are sensitive to the cultures and meet the needs of evolving network communities. Because, increasingly, systems are built from existing software components, we also plan to explore and mature software technologies that allow system integrators to predict the security characteristics of systems built from these components. Finally, to verify system conformance with expected behavior and system operation with organizations' policies, we plan to explore and mature technologies that monitor system configurations and system performance.

In this chapter, the sections for this impact area are as follows:



3.5.1 Description

3.5.1.1 Context

From the CERT experience of the past six-and-a-half years, it is evident that the software engineering practices of open systems technology producers and vendors are not adequate to counter the threat of network and computer system intrusions. The symptoms of the failure are computer security incidents, but the root causes of these incidents include

- Requirements definition practices that fail to account for the need for simplicity in system administration practices and the need for distributed system configurations using local and wide area networks.
- Software architectures that provide user-required functionality only by sacrificing basic system control principles.
- Errors in design that lead to unexpected system behavior that can be exploited to gain unauthorized system access.
- Errors in implementation such as weaknesses in coding, review, inspection, integration, and testing practices.
- Weaknesses in configuration management practices that lead to delivery of insecure configurations to users.

In short, the rate and severity of computer security incidents serve as a barometer for the state of the practice of software engineering in a large segment of the community.

We have a solid foundation for improving the state of security of systems and networks. Our understanding of current security problems and potential solutions comes from the CERT staff's first-hand experience with compromised sites on the Internet, and subsequent analysis of the security incidents, intrusion techniques, configuration problems, and software vulnerabilities. Because of our reputation for being objective and discreet, organizations share information with us that they hesitate to share with others. As a result, we are in a position to synthesize security information from a broad range of sources—including commercial and government organizations, software developers, product vendors, and service providers. As an SEI impact area, we are able to address the concerns of government, industry, and academia. The focus of the SEI on technology transition and its role as facilitator of improvement puts us in a position to effectively transition our solutions to a broad community.

Our work in Trustworthy Systems brings SEI core competencies and the work of other SEI impact areas to bear on the problem of information security and system security in software-intensive systems. Our planned approach to the problem includes computer security incident response activities that help the network community deal with its immediate problem while allowing us to maintain an ongoing understanding of the scope and nature of the problem and of the community's needs. Our approach also includes near-term and long-term activities focused on preventing incidents. Near-term work aims at providing network and system administrators and users with tools and techniques they can use to assess and improve the security of their systems. Long-term work aims at dealing with the root causes of the problem by exploring, developing, and transitioning improved software engineering practices to technology producers and vendors who supply the wide-area-network market. See Figure 3-19 for an illustration of our approach.

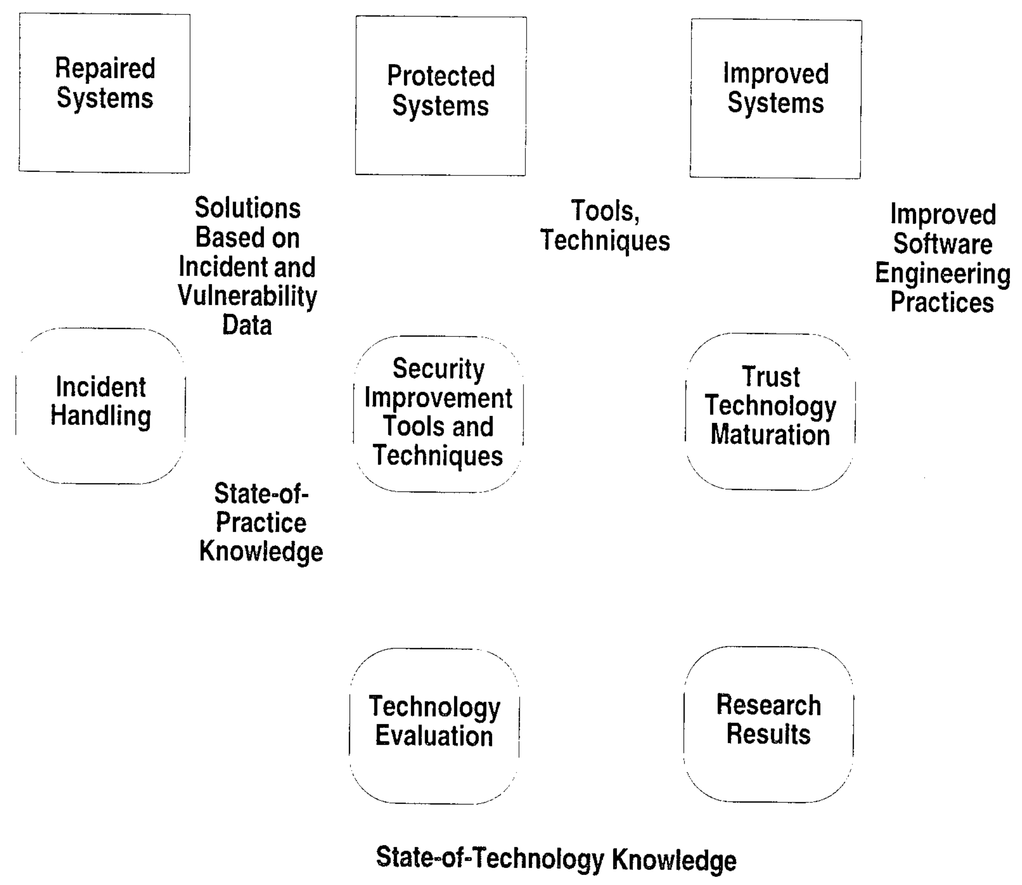

Repaired Systems
Protected Systems
Improved Systems
Solutions Based on Incident and Vulnerability Data
Tools, Techniques
Improved Software Engineering Practices
Incident Handling
Security Improvement Tools and Techniques
Trust Technology Maturation
State-of-Practice Knowledge
Technology Evaluation
Research Results
State-of-Technology Knowledge

Figure 3-19: Trustworthy Systems

3.5.1.2 Key Value-Added Contributions

To date, the bulk of our work has been security incident response and awareness activities. Through this work, we have built relationships and infrastructures that position the SEI to take the next step toward dealing with the root causes of network security problems. Key contributions include the following:

- Development and operation of an incident response capability that has facilitated the resolution of over 5,000 computer security incidents. We have captured data on those incidents to support the analysis of both security threats and product vulnerabilities that leave technology susceptible to exploitation by intruders. These analyses have enabled us to issue more than 110 advisories to the community. These advisories describe corrections to vulnerabilities or profiles of attack techniques that allow system administrators to monitor their systems for unauthorized behavior and take precautions against it.
- Development of working relationships with 42 computer software and system vendors. We work closely with the vendor community, informing them of security deficiencies in their products, and facilitating and tracking their response to the problems. We work with the vendors to improve the security they design into and deliver with their products. We also encourage them to add security topics to their standard customer training courses. It is our plan to evolve these relationships so that the SEI influence appears earlier in the product development cycle and helps the vendors address the root causes of the problems in their products.
- Leadership in the formation of the Forum of Incident Response and Security Teams (FIRST). We have helped private organizations and government agencies form their own incident response teams and worked with them to form an association that helps response teams work together. Originally called the CERT-System, this collaboration involves 44 teams that work together on incident response and prevention. The FIRST is a natural distribution channel for the security technologies and techniques we plan to mature.
- Development of two security seminars, sponsorship and organization of security conferences and workshops, and creation of press relations, all of which help raise community awareness of information and computer security issues and of the technologies and practices available to deal with those issues.

3.5.2 Five-Year Goals

It is our plan to make the Internet and other software-intensive networked systems more secure by focusing on the continued creation, coordination, and support of incident response teams; fostering the widespread adoption of tools and techniques that improve the security of existing systems; and identifying and maturing technologies that can be used to produce software and architectures that are highly resistant to attack. In particular, our five-year goals are the following:

- Each major network service provider will have an incident response team so that routine, limited-scope incidents can be handled at the regional level. In addition, an incident response and security infrastructure will be in place at 50 percent of the major Internet sites.
- Major network service providers and major Internet sites will routinely use a set of techniques and tools that enable system administrators to monitor and improve the level of security on their systems and networks. These tools and techniques will be available to the broad community through the network service providers and other components of the network infrastructure, such as software archive sites.
- Security incidents caused by errors in software architecture, design, or implementation will be reduced by 50 percent, as measured by CERT vulnerability analysis activities. Security considerations will be routinely integrated into requirements and domain models and defined in architectural specifications; and static and dynamic analysis tools will be in use.

3.5.3 Five-Year Strategies

Our strategy is to use the incident response activity to benchmark the state of the practice of information system security and to use that knowledge to help members of the network community improve their products and practices.

We combine our expertise in responding to network intrusions with knowledge of security practice and evolving technology to raise the level of security on the Internet and other software-intensive systems. We use activities that generate information and knowledge, such as incident handling and technology exploration, to drive other product development activities.

To gain leverage with our limited resources, we concentrate on working with other response teams, network service providers, technology producers, vendors, and leading network users to transition network security practices and technology to the Internet. Network service providers are particularly important given the ever-increasing size of the Internet. Service providers are well positioned to provide security improvement and incident response services to their customers. We plan to work with the service providers to raise their awareness of security issues and to facilitate their efforts to provide security services to their customers.

We also bring members from other response teams into our group for varying periods of time (several weeks to a year). We benefit from their contribution to incident handling, vulnerability analysis, and SEI product development; they benefit from the knowledge gained from their experience; and we both benefit from their role as transition agent.

To increase our leverage in the next five years, we plan to build collaborative relationships with government and industry organizations that can participate in product development and act as test sites for tools and techniques. We plan also to develop a large and diverse set of distribution channels.

Finally, we will use the work of other SEI impact areas, such as Risk and Disciplined Engineering, as a basis for a set of products and services that focus on the information system security problem. Examples include development of security risk evaluation techniques based on the work done by the Risk staff in the area of risk assessment and incorporation of security issues into the open systems handbook and course developed by the Disciplined Engineering staff. By transitioning SEI work to a new audience and by extending and repackaging some of it to address new problems, we will both broaden the impact of the SEI work and extend its scope to a new problem area.

3.6 Professional Infrastructure

A mature profession promotes high standards for professional practice and supports the rapid dissemination of new knowledge. The growth of a mature software engineering profession will contribute to substantially improved professional practice and lead to higher quality in software systems. For these reasons, maturing the profession of software engineering is part of the SEI's strategic framework. (See Section 2.3.1 of Volume I.)

Definition. There are several definitions[1] of "profession." They suggest that a profession has nine[2] components: initial professional education, accreditation, skills development, professional development, certification, licensing, a code of ethics, a code of practice, and a professional society. These are shown in Figure 3-20; the boxes represent the activity and organizational components, and the magnifying glasses represent quality assurance components.

In the figure an aspiring professional first undertakes initial professional education (the education that precedes the first day on the job; usually provided by a university); the quality of a professional degree program is assured by accreditation. To become a professional, he or she must develop skill in the application of that education (through university co-op programs, on-the-job training, apprenticeships, or other means). Certification and/or licensing assures the competence of the individual to enter professional practice. Throughout practice, there are periods of professional development, possibly resulting in recertification or relicensing. The profession ensures that its practitioners behave in a responsible manner by defining a code of ethics and a code of practice. A professional society helps assure that all the other components interact appropriately.

An important consequence of this definition is that the profession can mature separately from the development of the scientific knowledge and technology that is used by the professionals. As an example, consider the medical profession of the 1930s. The profession was reasonably mature at that time (the nine components were all in place and working well), but the knowledge and technology base did not yet include penicillin, organ transplants, or magnetic resonance imaging. But because the profession was mature, as new technologies became available, they were rapidly assimilated into the curricula of medical schools and the practices in hospitals.

1. See, for example, *International Encyclopedia of the Social Sciences*, David L. Sills, ed. New York: Macmillan Company and The Free Press, 1968.

2. Not every profession will have every one of the components. For example, some professions will have certification but no licensing (such as accounting), and others will have licensing but no certification (such as engineering).

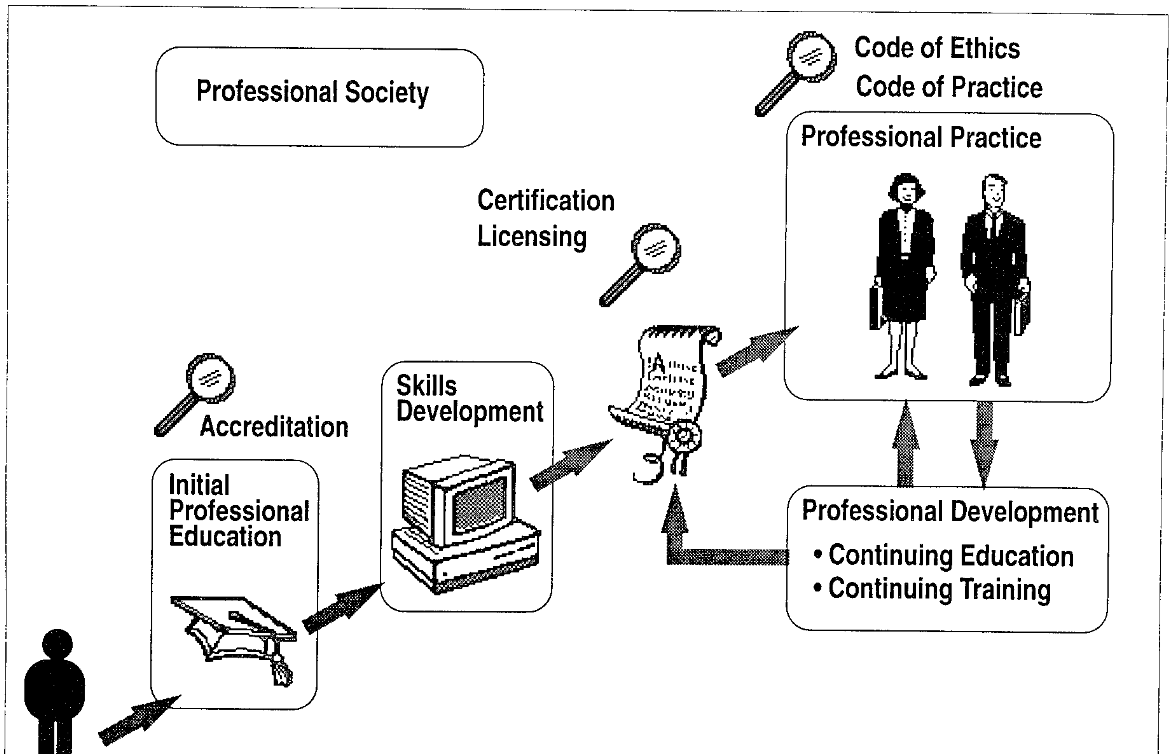


Figure 3-20: Components of a Mature Profession

Note that our definition distinguishes between the *profession* (the nine components) and the *professionals* (the people who practice the profession). We assert that an effective way to improve the practice of individuals is to accelerate the development of a mature profession. This assertion is based on the observation that a mature profession has in place the institutions, processes, and procedures necessary to ensure a continuing supply and responsible behavior of highly competent professionals. Just as a mature software process facilitates the rapid assimilation of new technology, a mature profession facilitates the rapid diffusion of knowledge and skills to professionals.

A Mature Profession. Although there is no precise definition of a *mature* profession, it is possible to characterize the maturity of a profession in terms of its components. Each of the components evolves through several stages (similar to "maturity levels"):

None	This component does not exist in any form.
Ad Hoc	Some form of this component exists, but it is not necessarily directly related to the profession being described. For example, there may be something related to computing but not specifically related to the software engineering profession.
Specific	This component exists and is directly related to the profession being described.
Maturing	This component specifically addresses the profession being described and has been improving for many years.

Figure 3-21 describes possible component stages for the software engineering profession. Nearly all the components are currently in the "ad hoc" stage, and it is an overall goal of the SEI to accelerate the evolution of some of these components to the higher stages. Specific goals and objectives for maturing particular components are detailed below and in Volume II, Chapter 5, Professional Infrastructure.

Component	Ad Hoc	Specific	Maturing
Initial Professional Education	Bachelor's degrees in computer science, engineering, mathematics, etc., are the common preparation for entry into the profession.	Bachelor's degrees exist that specifically prepare students to enter the software engineering profession; each school designs its own curriculum.	Curricula reflect the best practice; nationally accepted model curricula exist; model curricula are regularly reviewed and revised.
Accreditation of Education	Accreditation is based on computer science or engineering criteria.	Accreditation is based on software engineering criteria; ABET and CSAB have merged.	Accreditation guidelines are regularly reviewed and revised.
Skills Development	Some student project work in schools, some co-op programs, and some company training programs for new hires exist.	Guidelines have emerged for the skills needed by a software engineer for entry into the profession.	Skills development mechanisms are in place and widely used (such as apprenticeships or engineer-in-training programs).
Certification	There is ICCP certification and commercial certification related to software packages and technologies.	There is certification as a software engineer; certification standards are nationally recognized.	There is certification in specialty areas within software engineering; specialty certification standards are nationally recognized.
Licensing	State licensing as a professional engineer exists under current statutes.	Some state licensing examinations address software engineering skills specifically.	Licensing is based on appropriate examinations; NSPE and NCEE collaborate and are recognized as protecting the public in appropriate situations.
Professional Development: Continuing Education and Training	Individuals pursue professional development as they determine the need.	Professional development guidelines (curricula, expenditures per year, etc.) have emerged.	Recognized career paths exist for software engineers; education and training guidelines and curricula are nationally recognized.
Code of Ethics	Codes of ethics exist through ACM, IEEE, ASQC, ICCP; licensing statutes for engineers exist.	Code of ethics exists specifically for software engineers.	The code is widely respected and adopted; the profession has mechanisms to discipline violators.

Figure 3-21: Potential Evolution of Components of the Software Engineering Profession

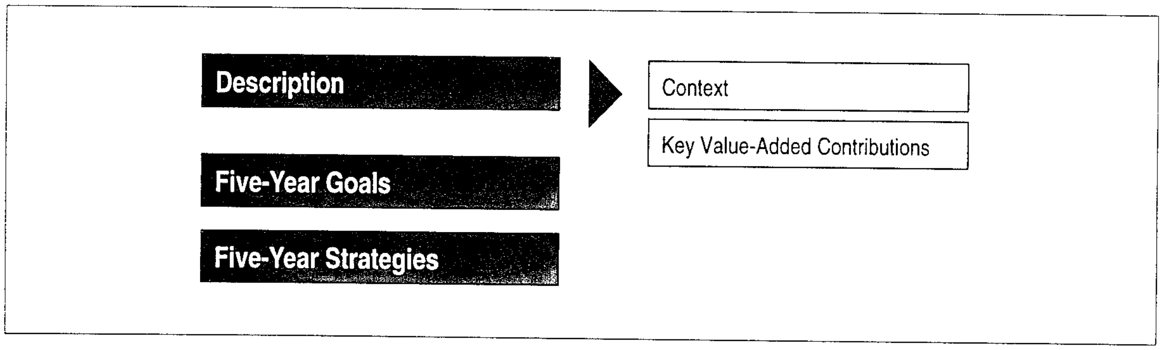
Component	Ad Hoc	Specific	Maturing
Code of Practice	Some practices are documented in textbooks or company standards.	An initial set of practices specifically addressing software engineering is published and backed by a recognized authority.	Code of practices is widely recognized, and is regularly reviewed and revised by a continuing body; content strongly influences education, training, certification, and licensing.
Professional Society	Societies include the ACM, IEEE Computer Society, and others.	A society (or subsociety of an existing society) specifically represents software engineering.	The society has an appropriate range of products and services for software engineers.

Figure 3-21: Potential Evolution of Components of the Software Engineering Profession (continued)

Acronyms used in this figure are:

ABET	Accreditation Board for Engineering and Technology
ACM	Association for Computing Machinery
ASQC	American Society for Quality Control
CSAB	Computing Sciences Accreditation Board
ICCP	Institute for the Certification of Computing Professionals
IEEE	Institute of Electrical and Electronics Engineers
NCEE	National Council of Examiners for Engineering and Surveying
NSPE	National Society of Professional Engineers

In this chapter, the sections for this impact area are as follows:



3.6.1 Description

3.6.1.1 Context

The SEI's efforts to mature the professional infrastructure have in part grown out of previous efforts in software engineering education. Late in 1994 those education efforts were evaluated and redirected, resulting in substantial changes to the 1995 work described in *SEI Program Plans: 1995-1999*. The Institute of Electrical and Electronic Engineers (IEEE) Computer Society and the Association for Computing Machinery (ACM) established a joint task force to establish the software engineering profession. This presents the SEI with a major opportunity to build upon the momentum of these groups' contributions. Because master's-level programs are now well established in the academic community, the delivery of academic courses, both in the Carnegie Mellon University Master of Software Engineering program and through the National Technological University, are being phased out in 1995. Delivery and maintenance of existing professional courses and the development of new courses will continue as needed, but will not be motivated primarily by an education-focused program. Most activities aimed at directly helping the academic community to provide undergraduate software engineering education are being phased out by the end of 1995. On the other hand, sponsorship of the annual SEI Conference on Software Engineering Education, development of guidelines for implementing the CMM key practice area related to training, and collaboration with the IEEE Computer Society/ACM industry task force are all continuing as planned.

These changes are summarized in Section 5.1 in Volume II, Chapter 5, Professional Infrastructure.

A significant new activity for 1995 is a feasibility study, the goals of which are to characterize a profession and the maturity of a profession, describe a future mature profession of software engineering, and identify further SEI work to accelerate the development of that mature profession. Preliminary results of that study are the basis for the discussion of a mature profession in Section 3.6 of Volume I.

Maturing the professional infrastructure, which means maturing each of the nine components, is a complex, long-term activity involving many people and organizations. It is important for the SEI to choose its work in this area carefully. Where possible, we will collaborate with selected organizations and, where necessary, undertake high-leverage activities to improve individual components of the profession. The feasibility study will be completed in the third quarter of 1995, at which time we will have identified those high-leverage activities and planned the SEI's involvement.

Preliminary results of the study suggest that one such activity is to create a code of practice for software engineers. A code of practice provides community-accepted methods to accomplish the tasks commonly faced by software engineers. Although much of software engineering practice is not yet ready for codification, the SEI can identify and disseminate a set of ***recommended practices***. These would include the best known practices and would be a first step toward the eventual development of a professional code of practice. The set of recommended practices will further catalyze the maturing of other components of the profession, as described in Volume II, Chapter 5, Professional Infrastructure.

We will continue to promote continuing education and training of practitioners in the field. In order to maximize our leverage, we will continue to work with national councils and committees to promote software engineering education infrastructure. We will conduct workshops with industry and government educators. These workshops identify actions to be taken by the

SEI and the education community to ensure that professional development of software engineers becomes and remains a high priority. We work directly with selected organizations to improve or document the state of the practice of software engineering education for professionals.

Additionally, we will complete our production of guidelines for software organizations to help them develop a training capability. The guidelines address such issues as determining training needs, planning and implementing curricula, establishing an infrastructure to deliver and administer courses, and evaluating commercially available courses. This work is especially helpful to organizations seeking to reach CMM level 3, which specifies a training key process area.

We will sponsor and organize the Ninth Conference on Software Engineering Education in April 1996. This unique annual conference provides a forum for the exchange of information about best practices in software engineering education among the academic, industry, and government education communities.

Detailed descriptions of these work areas and the rationale for each are presented in Volume II, Chapter 5, Professional Infrastructure.

3.6.1.2 Key Value-Added Contributions

The SEI is uniquely positioned to help leverage and coalesce the community to mature the profession of software engineering. At the highest level, our contribution is the vision that a mature profession can exist and will significantly improve the quality of software systems built in the United States. No other organization is providing or can provide the leadership to bring this vision to reality. At a more concrete level, the SEI is providing several outputs and services for the entire software community:

- The development of a set of recommended practices. Although there is a substantial amount of software engineering knowledge in books, journals, and the minds of experienced practitioners, it is not easy to find how to do a particular task. Similarly, there are high-level guidelines (such as the SEI Capability Maturity Model and the ISO 9000 series standards) that can tell an organization what to do, but they do not specify how to do it. The development of a set of recommended practices and the establishment of procedures to disseminate and maintain it helps address the long-standing need of the software engineering profession: a clear and concise description of how to do software engineering effectively. The set of recommended practices will also provide significant help to other people and organizations working to mature the software engineering profession, such as those working on curriculum design or guidelines for accreditation, certification, or licensing.
- The development of guidelines for training within software engineering organizations is an important step in the maturation of the professional development component of the profession. Currently such guidelines (where they exist at all) are developed individually and independently by software organizations. The SEI's guidelines will help the entire community move toward increased predictability of the knowledge and skills of software engineers.

- The Conference on Software Engineering Education is the only conference of its kind. The SEI has sponsored this conference for eight years, with a total attendance of 1300 educators from academia, industry, and government. For 1996 we have cosponsorship from the IEEE Computer Society; future ACM cosponsorship is also anticipated. This helps develop the professional society infrastructure for the software engineering profession. The exchange of ideas among the participants has stimulated the growth and improvement of both initial professional education and the continuing education and training components of professional development of software engineers, thus contributing to the maturation of the profession.

3.6.2 Five-Year Goals

It is our goal that in five years, each of the components of the software engineering profession exists, although not all are mature. Each is under the active stewardship of appropriate bodies, and each is maturing. How the SEI addresses each of these goals is described in the next section, Five-Year Strategies.

Specific goals for the individual components of the profession include:

Initial Professional Education. There are at least ten United States universities offering initial, undergraduate professional education specifically for software engineers (as compared to none in 1995). Model curricula for university programs in software engineering are under active development.

Accreditation. The Accreditation Board for Engineering and Technology, Inc., has established guidelines for accreditation of software engineering programs, and at least one software engineering program in a United States university has been accredited.

Skills Development. There is a consensus of the software community on the kinds of skills needed and the level of proficiency to be attained by software engineers at entry to the profession and at appropriate points in their careers.

Certification and Licensing. There is a consensus of the software community on the eventual need for certification and/or licensing of software engineers. Appropriate organizations, including the ACM, the IEEE Computer Society, the National Society of Professional Engineers, the National Council of Examiners for Engineering and Surveying (NCEE), and the Institute for the Certification of Computer Professionals, are cooperating to address the issues.

Professional Development. High-quality curricula for continuing education of software engineers have been published, and those professionals receive an average of 80 to 120 hours of continuing education and training annually. Compelling return-on-investment data is available on the value of continuing education of software engineers. There are well defined career paths for software engineering educators within large government and industry software organizations.

Code of Practice. A significant body of recommended individual and organizational practices has been documented and codified; these practices are being used as a basis for initial professional education and professional development.

Code of Ethics. A code of ethics for software engineers has been developed and is widely known among professionals.

Professional Society. The ACM and the IEEE Computer Society are providing an appropriate range of products (including publications) and services for software engineering professionals.

3.6.3 Five-Year Strategies

The SEI will mobilize organizations including professional societies, accreditation bodies, educational institutions, certification organizations, SPIN groups, national policy task forces, and DoD planning groups to contribute to the rapid maturation of the components of the software engineering profession.

Where necessary, the SEI will actively intervene to initiate, accelerate, or otherwise improve efforts to establish or mature the components of the software engineering profession.

Specific strategies for the individual components of the profession include:

Initial Professional Education. The ACM and IEEE Computer Society (IEEE-CS) plan to create a task force to develop model curricula for software engineering programs in universities. Individual staff members in the SEI will be encouraged to participate in that effort. The SEI will not assume a leadership role in improving academic software engineering education other than to make information about improved practice continually available to the academic community.

Accreditation. In the United States there are two accreditation bodies that might eventually be asked to accredit software engineering programs in universities. The Accreditation Board for Engineering and Technology (ABET) accredits engineering programs and the Computing Sciences Accreditation Board accredits computer science programs. The two organizations are currently considering a possible merger, a move the SEI supports because it would eliminate potential jurisdiction disputes when the first university seeks accreditation of a software engineering program.

Currently neither organization has accreditation guidelines specifically for software engineering. As university programs in software engineering begin to appear, the pressure to develop such guidelines will increase. The guidelines are normally produced by appropriate professional societies in collaboration with the accreditation body. The SEI will be prepared to offer opinions on guidelines.

Skills Development. We will continue our work with the IEEE-CS/ACM Industry Task Force to identify specific skills needed by software engineers.

Certification and Licensing. The ACM and IEEE Computer Society plan to create a task force to address issues of certification and/or licensing of software engineers. We will review draft documents and provide opinions on proposed guidelines.

Professional Development. We will work with national councils and committees that can influence the development of an improved infrastructure for software engineering education. We will conduct workshops with industry and government educators to identify actions to be taken by the SEI and the education community to ensure that professional development of software engineers becomes and remains a high priority. We will work directly with selected organizations to improve or document the state of the practice of software engineering education for professionals.

Code of Practice. We will develop a set of recommended practices for software engineers as a first step toward creating a professional code of practice. This includes defining ways of classifying and describing practices, developing mechanisms and templates for documenting practices, identifying and documenting current industry practices as a benchmark against which future progress can be measured, identifying an initial set of practices that are widely recognized as being "best practices," and making those practices widely available via hypertext and worldwide web technologies. We will also develop mechanisms by which we can continually collaborate with experts from the software engineering community to expand and improve the body of recommended practices.

Code of Ethics. The ACM and IEEE Computer Society have created a task force to develop a code of ethics for software engineers. No specific SEI involvement is planned.

Professional Society. We will continue to work with the existing societies (ACM and the IEEE Computer Society) rather than attempting to create a new society specifically for the software engineering profession.

3.7 Integrated Transition Strategies and Methods

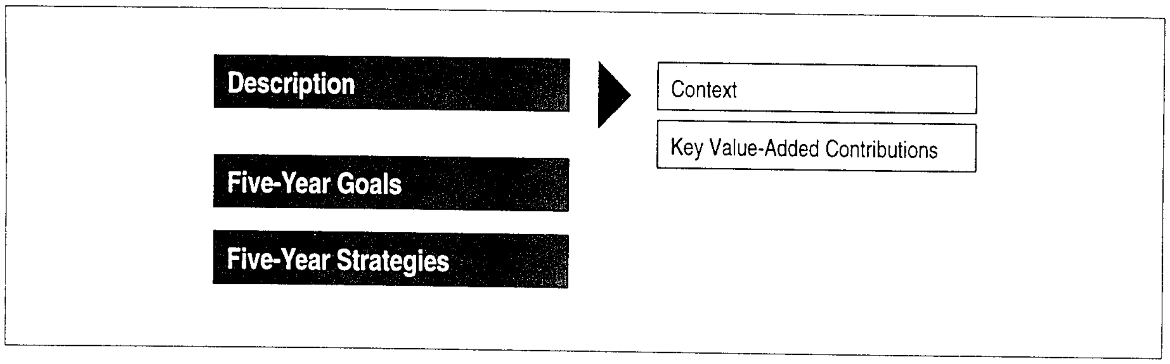
3.7 Integrated Transition Strategies and Methods

When organizations seek to improve their software management processes, their software technology, and the abilities of their personnel to use new practices effectively, they typically face difficulties in deciding what processes to put in place first, what technologies to adopt, and how to help their staff become effective in using the improved software engineering practices. Many organizations today know they need to improve their processes and technology, but they do not have effective internal methods for moving their organizations forward. They need vehicles (methods, courses, diagnostic instruments, etc.) for introducing improved engineering practices. These vehicles are also needed and used by the SEI or by others to transition improved engineering practices into particular software engineering organizations.

Lack of knowledge and skills in technology transition is not only a problem for those *adopting* new software engineering practices. *Suppliers* of technology (technology researchers, developers, and vendors) similarly lack an understanding of the technology transition process. Although technology developers are good at overcoming technical barriers that impede the effectiveness of their technology, they are typically less skilled at understanding the full extent of non-technical barriers faced by potential adopters of their technology. Technology vendors typically understand how to attract customer interest in their products but they are often less effective in getting their customers to install and use these products effectively. In short, lack of effective understanding of technology transition principles and practices impedes the transition of more effective technology from both the supplier and adopter viewpoints.

Because the development and maintenance of software systems is a group process, we are also concerned with identifying and enhancing the skills and technology needed for software engineers and managers to collaborate effectively. To address this need, we disseminate tools, methods, and practices that increase the ability of software engineers to work effectively in teams. For example, organizations may attempt to install an established software engineering process such as software inspections, but inspections may not produce the expected benefits if the organization's people are not skilled in conducting and participating in small group meetings and if other management processes for following up on the results of the review are not in place to support the use of the inspections process.

In this chapter, the sections for this impact area are as follows:



3.7.1 Description

3.7.1.1 Context

Although every SEI area has responsibility for making technology transition happen, i.e., for transitioning the improved engineering practices it is working on, the SEI undertakes some activities aimed at preserving and improving our core competence in software engineering technology transition. In particular, we adapt generalized technology transition concepts to the context of software engineering, making these concepts more usable both within the SEI and by those software organizations seeking to introduce improvements in their software engineering practices.

We are also a source of expertise on understanding and overcoming human barriers to effective use of software engineering technologies and processes. In particular, we seek to improve the skills software engineers need when working together to create software systems. Software systems development requires collaborative skills whether in eliciting requirements, creating and reviewing potential designs, reviewing code, or seeking to understand and correct system problems. For example, risk identification and mitigation involves establishing a group culture in which areas of concern (risks) can be presented and dealt with constructively. An effective method of eliciting risks so they can be dealt with requires skill in establishing appropriate team dynamics as well as knowledge of software engineering. Paying attention to people issues like these is one part of the SEI's strategy for improving the practice of software engineering.

Our work falls in two activity areas. The first area, Software Engineering Transition Practices, focuses on defining and disseminating effective ways of transitioning software engineering technology and methods into use. These transition practices are intended to be used 1) by suppliers of software engineering technologies and processes (so they produce products that are more easily adopted), 2) by adopters (so they more easily and effectively install improved software engineering practices), and 3) by intermediaries who support suppliers and adopters (so they can be more effective in transitioning improved practices). The SEI itself acts as a supplier and an intermediary to our customers, so these transition practices are used internally as well as externally.

Our second activity area, Collaborative Skills in Software Engineering, focuses on identifying and enhancing the skills software engineers and managers need to collaborate effectively. We focus on collaborative skills because the development and maintenance of software systems is a group process. Activities in this area include dissemination of tools, methods, and practices that increase the ability of groups of software engineers to work together more effectively in applying software engineering practices.

3.7.1.2 Key Value-Added Contributions

From an external point of view, ITSM adapts general principles of technology transition to the needs of software engineering organizations. We make it easier for technology providers and adopters to take the necessary steps to ensure transition success. The SEI's position in the software engineering community draws greater attention to these ideas than would be the case for other organizations.

One specific thrust of our activities is to better understand the problems software organizations face in introducing new software engineering practices, both in general and for specific practices. For example, Geoffrey Moore[1] has analyzed the difficulties of transitioning new technologies into mainstream practice. We are applying these concepts to software engineering technology transition by demonstrating the value of supporting new technologies with "Transition Packages." Transition packages are "whole products" that can expedite the introduction of methods, tools, and processes needed to install key software engineering technologies or processes. A "whole product" is one that consists of not just the core technology (such as a software quality assurance process or a computer-aided software engineering (CASE) tool) but also all the components that support moving an organization from non-use of the technology to routine, everyday use. These components include process models and guides, training built to be readily customized, consulting scenarios, document templates, and more. Most of these are labor-intensive and difficult for inexperienced change teams such as SEPGs and Process Action Teams to create. By demonstrating the value of this concept (we are demonstrating the value by creating transition packages for Level 2 Key Process Area practices), we both increase the transitionability of these software engineering processes and demonstrate to suppliers of other technologies the value of providing such support for adopters of technology.

In other cases, we provide courses and workshops that directly help organizations cope with the difficulties of introducing technological change. We train people in how to create an environment in their organization that makes change possible. We specify key indicators of transition success in an adopting organization, and we ensure that more organizations are first, aware of these indicators, and second, have processes and functions in place to make technology improvement go forward smoothly.

Finally, as a result of our contacts with adopter organizations and with organizations seeking to improve their software technology, we have access to information about the state of the practice and the benefits of software technology improvements. We are able to collect, sanitize, consolidate, and interpret this data so organizations can see how they stand with respect to their technology transition capabilities and experiences. We help motivate transition by showing how to do it.

1. Geoffrey Moore (1991). *Crossing the chasm: Marketing and selling technology products to mainstream customers.* Harper Business.

3.7.2 Five-Year Goals

By 2000, software engineering organizations at CMM level 3 and higher will know what technology transition practices and support are needed to adopt improved software engineering practices more quickly and more predictably. It these transition practices are not already installed at these organizations, they will have plans for doing so.

Our longer range goal is to ensure that effective software engineering technology transition strategies and practices are widely understood and used by technology developers, by technology vendors, and by software organizations seeking to improve their software engineering technology and practices. Thus, new or improved software engineering technologies will be appropriately selected and smoothly adopted by organizations because key technology transition practices are installed and used effectively by these organizations. Transition will happen more quickly because organizations are better prepared to make the changes needed to install improved technology and practices.

Our second long-range goal is to ensure that software engineering technologies and processes pay increased attention to the barriers inherent in the ways people are used to working together. If these barriers are not identified and mitigated, organizations and individuals will not be effective in making use of otherwise sound technologies and processes.

3.7.3 Five-Year Strategies

Much information exists on the means and strategies of effective technology transition, but this information has not generally been made accessible to those responsible for software engineering technology development and transition. Since much information is available, our strategy is to *adapt and recast* this information so it is more readily appreciated and used by technology developers and by organizations seeking to adopt improved technology and processes. We reformulate this knowledge in ways that capture the attention of our target population, and we will demonstrate the utility of our approaches by working directly with selected software engineering technology developers and organizations.

We work directly with selected organizations to provide proven methods for

- *diagnosing* an organization's software engineering needs and ability to improve
- *matching* software engineering technologies and processes with the diagnosed engineering needs of the organization
- *catalyzing* improved organizational performance via effective technology transition techniques

We use a similar strategy to address our second goal; namely, we adapt and apply existing knowledge of how groups can be energized to work together so this knowledge is effectively applied to the needs of software engineering groups and the demands (and barriers) inherent in particular software engineering technologies and processes. This work is always conducted with respect to specific software engineering technologies and processes to ensure its relevance to the practice of software engineering.

Volume I
Chapter 4 Table of Contents

4 Community Outreach

The Software Engineering Institute (SEI) has developed a range of relationships, from those that satisfy customer needs for general information about the SEI and its technical program to those that involve collaboration and technology transition. The SEI provides opportunities for customers to participate in workshops, conferences, advisory boards, and educational offerings as well as the acquisition and co-development of specific outputs and services including customer-site support. The ways in which our customers can work with the SEI are described below.

4.1 Customer Inquiry/Response

The customer inquiry and response service is provided through the SEI information line, (412) 268-5800, and FAX, (412) 268-5758, during normal working hours; and through Internet email, `customer-relations@sei.cmu.edu`. Customer Relations serves as a single SEI point of contact for the customer. By disseminating information, Customer Relations accelerates the transition of new technologies and methods. The customer relations representatives who provide this service are fully prepared to answer any question of a general nature about the SEI, to mail pertinent descriptive materials, and to follow up with members of the SEI technical staff to provide more detailed information. The SEI handles up to 1600 requests per month. We collect statistics and maintain a database reflecting the character of the inquiry/response traffic. This information is often used by others at the SEI to contact and respond to our customer community.

Another way the SEI has of reaching customers is through Visitor's Day, which the SEI hosts three times a year to familiarize software managers, practitioners, and educators with the SEI and its activities. Members of the SEI technical staff; impact area managers and sector managers; and project leaders give presentations on the technical program and SEI outputs and services. Demonstrations of SEI technical offerings are also frequently showcased.

4.2 Impact of Software Engineering Practices

In its second year in 1996, this activity is an ongoing effort to identify and quantify the impact of software engineering practices on quality and productivity in the software industry. Data and analyses focus on improvements in product quality, cost, schedule, and business value, as well as the SEI's success in transitioning technologies to the broader software community. Work processes are being defined and refined to routinely capture, analyze, and act upon data about the impact of software engineering practices. The initial focus has been on work that has high visibility within the SEI and the software engineering community. The activity eventually will address all SEI efforts that deliver outputs to external customers.

4.3 Subscriber Program

The subscriber program is an effective way for an individual to stay informed about the SEI's activities. Participants receive mailings that keep them up to date on the SEI's events, course offerings, works in progress, new outputs, and new initiatives. Anyone with a United States mailing address is eligible to subscribe. A fee of $175 (as of January 1995) has been established to help offset costs of delivery. The fee covers an entire year from the date that the subscription is activated. Department of Defense customers receive the same benefits at no cost.

SEI subscribers receive many benefits. Participants in the subscriber program

- are assigned to a Customer Relations Representative
- receive the quarterly *Bridge* magazine
- receive the annual *Technical Review*
- obtain discounts on technical reports
- obtain a discount at the annual SEI Software Engineering Symposium
- participate in the SEI subscriber electronic news network
- are notified of the SEI's conferences and events
- receive the subscriber directory
- receive a complimentary copy of the *Key Practices of the Capability Maturity Model, Version 1.1* and *The Capability Maturity Questionnaire, Version 1.1*

4.4 Resident Affiliate Program

The resident affiliate program provides the opportunity for experienced technical personnel from government, industry, and academic organizations to reside at the SEI and participate on SEI teams. Resident affiliates contribute both as software engineers and as application domain experts, providing a valuable and practical perspective. They help us understand our customers' needs by providing information about their home organizations and about the organizational and technical contexts in which these organizations practice software development.

The sponsoring organizations benefit by participating in technical activities that might not be possible in their own organizations; they obtain the results of the SEI's technical activities early in the activities' evolution and have access to the SEI's people, teams, and other resident affiliates. The resident affiliate benefits from working in a different technical context; he or she participates in the many workshops and other activities at the SEI and in the larger CMU community, and interacts with colleagues from different professional, technical, and organizational backgrounds. The SEI benefits by obtaining experience, expertise, and additional insight into the software engineering community.

Resident affiliates work on-site at the SEI for a negotiated period, usually 6 to 24 months; they may spend half- to full-time at the SEI. They devote approximately 80 percent of their time to an SEI technical team and the remaining time to technology transfer and liaison activities with their home organizations. Resident affiliates are treated as integral members of the SEI staff.

Organizations that have participated in the Resident Affiliates Program are listed in Appendix B of Volume II.

4.5 SPIN Program

A Software Process Improvement Network (SPIN) is an organization of software professionals in a given geographical area interested in software process improvement. The purpose of the SPIN group is to provide a practical forum for exchanging ideas, information, and mutual support on software process improvement. Each regional SPIN is slightly different, based upon the vision of the founders and the needs of that community.

The first SPIN was started in Washington D.C. in 1991 by several professionals who realized the need for a mechanism by which software engineering process groups (SEPGs) could band together to provide mutual support and interaction. In September 1992 the SEI agreed to serve as coordinator for the emerging SPINs. Since that time the number of SPINs have continued to grow at a rapid pace. There are currently 32 SPINs in the United States and 21 international SPINs. Active SPIN organizations are listed in Appendix C of Volume II.

The primary role of the SEI is to disseminate information from existing SPIN organizations and to assist groups of people in common geographical locations who are interested in starting new SPIN groups. The SEI maintains a directory of all currently active SPINs and points of contact in areas where interest has been expressed in forming a new SPIN. The SEI maintains an e-mail alias used to disseminate announcements of interest to the network and distributes start-up information on forming SPIN organizations. We are also able to put potential SEI speakers in touch with SPIN coordinators to enable members of the SEI technical staff to speak at many SPIN meetings.

The SPIN provides significant leverage for transition. The SEI interacts with SPIN groups, each member of which may represent one or more SEPGs. Each SEPG, in turn, represents a corporate software engineering staff often measured in the hundreds. Figure 4-1 illustrates this one-to-many effect.

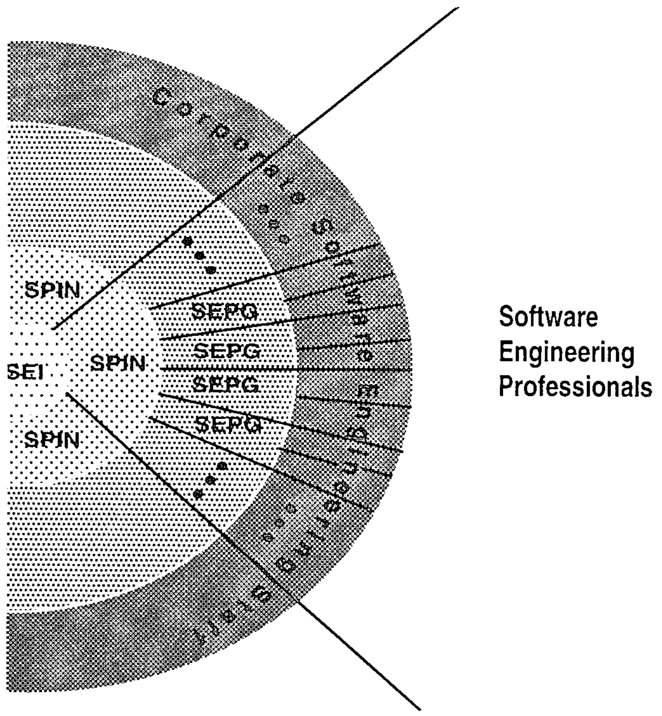

Corporate Software Engineering Staff
SPIN
SEI
SPIN
SPIN
SEPG
SEPG
SEPG
SEPG
Software Engineering Professionals

Figure 4-1: Effect of Software Process Improvement Networks (SPINs)

4.6 Public Offerings of Courses

The SEI develops and delivers courses to transition the knowledge and skills that can improve software engineering practice. Through these courses, software executives, managers, instructors and practitioners in government, industry and academia learn about proven techniques to increase profit and quality, adapt to change, build teams, mitigate risk and improve process. There are currently fourteen courses which are offered multiple times throughout the year. The SEI also makes arrangements to offer courses at customer sites.

Major SEI courses in 1995 include

- Software: Profit Through Process Improvement
- Introduction to the Capability Maturity Model
- Defining Software Processes
- Managing Technological Change
- Consulting Skills Workshop
- Software Risk Management

- Risk Identification and Analysis
- Managing Software Development with Metrics
- Engineering an Effective Software Measurement Program
- Software Quality Improvement
- CMM-Based Appraisal (CBA) Lead Assessor Training
- CMM-Based Appraisal for Internal Process Improvement (CBA IPI) Overview Seminar
- Instructor Training for Personal Software Process (PSP)
- Open Systems: The Promises and the Pitfalls

4.7 Collaboration Program

The SEI's collaboration programs are intended to create well-defined and well-managed relationships with industry customers. Through these partnerships, the SEI has access to an industry constituency that can

- provide input to the SEI technical program
- advance the maturity and accelerate the development of the SEI's technology, outputs, or services
- provide in-kind and direct funding resources

4.7.1 Technical Collaboration Program

Technical collaborations are formed for a fixed duration, involving well-defined areas of interest with one or more of the SEI's technical teams, with the end objective of a demonstrable result. Current examples include co-development of outputs of mutual interest (for example, road maps, field guides, handbooks, and training courses), technology exploration, and pilot/-field testing of new processes, methods or tools. Technical collaborations are initiated by mutual agreement and are negotiated between the SEI and the potential partner with the intent of exchanging value of mutual benefit. These collaborations may be funded by the partner or may be in-kind agreements where the partner supplies a resident affiliate, for example. Organizations that have participated in our technical collaboration program are listed in Appendix D of Volume II.

4.7.2 Strategic Collaboration Program

Strategic collaborations are long-term, corporate-level relationships between the SEI and selected industry partners. These partnerships are characterized by mutual statements of strategic intent and goals. The strategic relationship is realized by executing multiple technical collaborations, as described above.

Candidate strategic partners have demonstrated their commitment to the SEI mission and vision and the transition of SEI-developed approaches by virtue of their historical and current involvement with the SEI. In addition they have demonstrated a strong commitment to continuous improvement in the quality of their own software products and processes. The SEI seeks partners who

- are recognized leaders in several market segments (for example, system integration, manufacturing, computer systems, telecommunications, banking and finance)
- have an ability to execute technology transition roles
- can contribute to the depth and breadth of the SEI technical program

Benefits for strategic collaboration partners include

- broader, and often more immediate, access to the SEI's outputs, services, and technical staff
- an opportunity to have input into the SEI technical program
- early access to work outputs that are being co-developed, including those that may have been difficult to develop in a timely way without the benefit of collaboration

Current strategic collaboration partners include Hewlett-Packard, Hughes, Loral Federal Systems (previously IBM Federal Systems Company), and Texas Instruments.

4.7.3 Cooperative Research and Development Program

As part of the collaboration program, the SEI engages in Cooperative Research and Development Agreements (CRADAs) with industry organizations. The Federal Technology Transfer Act of 1980 and the National Competitiveness Technology Transfer Act of 1989 give organizations such as the SEI latitude to enter into these agreements. The intent is to accelerate the transition and commercialization of technologies by permitting the SEI to receive funds from industrial organizations that have a commercial interest in technology work in progress. This program is gaining momentum and will continue to contribute significantly to the SEI technical program in 1996.

4.8 Symposium / Conferences

The Software Engineering Symposium (SES) is an annual event hosted by the SEI. The purpose of the SES is to provide a forum in which people can learn about practical solutions to problems that plague the software community at large, understand the role of the SEI in assisting the development and adoption of those solutions, and meet other members of the community with similar problems and interests.

The SEI's goals in promoting and hosting the symposium are (1) to directly affect technology transition and (2) to increase awareness throughout the software engineering community of the SEI's activities, in order to broaden our impact over time. The symposium technical program includes a significant number of presentations from industry and government customers whose technology results relate to SEI outputs or tie closely with current work in progress.

This year's theme—Engineering the Future—is directed toward the changes that are likely to occur in government as the push toward rightsizing gains momentum, as well as the responses necessary within industry to stay in stride with the changing political landscape and the global competitive marketplace. The symposium brings spokespersons for the various perspectives together to offer their assessments of the nature and scope of changes within their respective areas of government and industry. This provides an important backdrop for practitioners to better anticipate the future of their organizations and their own professional development. This theme and the SEI's sponsorship of the symposium reflect our commitment to providing the enabling actions and ideas to help our customers find and adopt realistic approaches to solving their own problems.

The symposium provides a valuable opportunity to our customers and ourselves, not only to learn about mutually beneficial activities but also to interact, to learn of emerging software engineering technology applications, and to provide feedback to one another. It is one of our largest and most successful technology transition events, given that people are still the most effective means for technology transition.

Since the inception of the symposium in 1987, attendance at the event has increased to nearly 1300. Figure 4-2 shows the attendance over the past eight years. Significantly, an increasing number of presentations are given by our customers—approximately one-third in 1993, 40% in 1994, and more than half in 1995.

Many other events are held for customers who are interested in specific aspects of software engineering or the SEI's work. The SEI Software Risk Conference, held yearly, addresses the wide range of needs of SEI customers, from practitioners to managers in both the government and civilian sectors. The purpose is to provide a forum for the exchange of ideas, an opportunity to be exposed to best practice, and an awareness of current experiences in software risk management. The conference provides current information on risk management methods, theory, and practice through invited presentations, panel discussions, and workshop formats. It has proven to be an important mechanism in establishing a community of research and practice in software risk management. This effort is totally on a cost recovery basis.

The annual Conference on Software Engineering Education (CSEE), described in more detail in Chapter 5 of Volume II, invites educators, trainers, managers, and administrators from government, industry, and academia to participate in in-depth tutorials, invited keynote addresses, formal presentations of refereed papers, birds-of-a-feather sessions, and many opportunities for informal discussion.

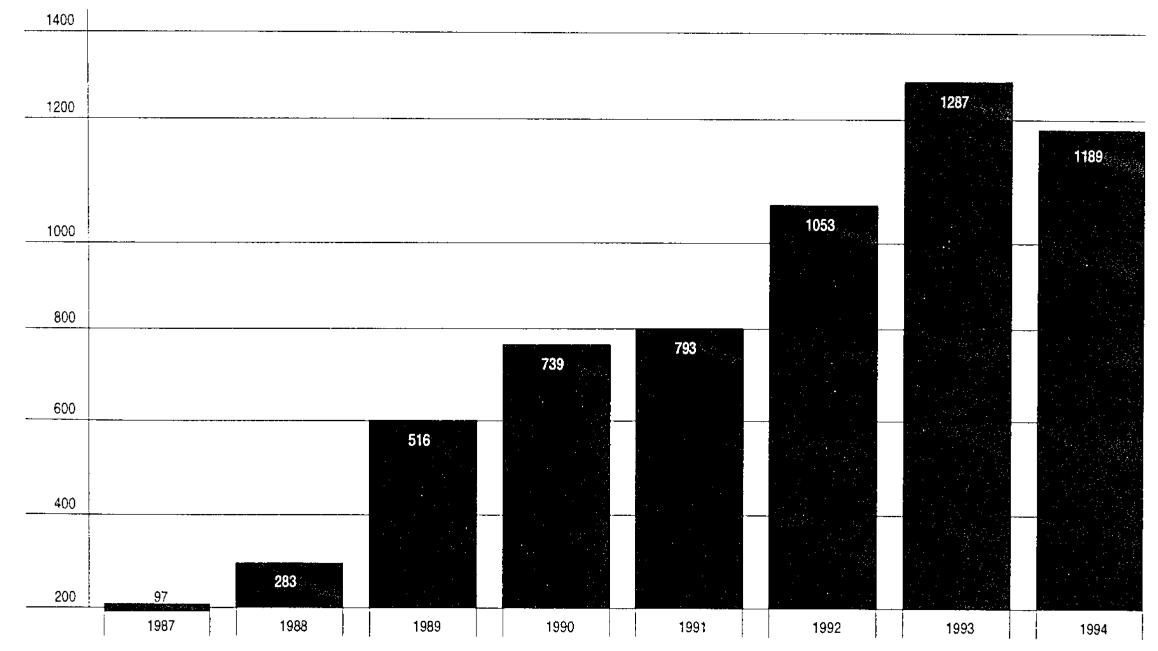


Figure 4-2: Attendees at SEI Symposia

4.8.1 Software Engineering Process Group (SEPG) National Meeting

The objectives of the SEPG National Meeting are to

- provide information on initiating and sustaining SEPG activities
- advance the state of SEPGs
- establish a network mechanism for SEPGs
- present experiences, through tutorials and papers, of successful software process improvement efforts
- enable those involved in software process improvement to discuss issues and learn from each other

Attendance at this event has grown significantly over the past seven years—from 46 in 1988 to more than 1300 in 1995. The SEI fills the enabler role in working with host SPINs to produce this annual event. The rapid rise in attendance suggests tremendous interest in software process improvement, and it indicates that the SEI is achieving a significant transition goal in helping to make this event possible.

4.9 Advisory Boards and Working Groups

In addition to oversight groups, such as the Board of Visitors (BoV) and the Joint Advisory Committee (JAC), the SEI also identifies the need for customer advisory boards or working groups. These groups provide customer guidance on current activities and future plans, and perform technical reviews of outputs. Members are selected through a screening process using project-defined criteria intended to populate the board or group with a mix of technical professionals who can help satisfy the SEI's technical objectives. The current advisory boards and working groups include the following:

- Software Process Advisory Board
- Software Process Measurement Steering Committee (MSC)
- Capability Maturity Model (CMM) Advisory Board
- Systems Engineering Capability Maturity Model (SE-CMM) Steering Group
- People Capability Maturity Model (P-CMM) Advisory Board
- Disciplined Engineering Advisory Board
- Risk Advisory Board
- Trustworthy Systems Advisory Board
- Education Advisory Board

For a description of each of these groups, see Appendix E of Volume II.

List of Acronyms

ABET	Accreditation Board for Engineering and Technology
ACM	Association for Computing Machinery
ARPA	Advanced Research Projects Agency
ASC/YT	Aeronautical Systems Command
ASQC	American Society for Quality Control
BOV	Board of Visitors
CAF	CMM Appraisal Framework
CARDS	Comprehensive Approach for Reusable Defense Software
CASE	computer-aided software engineering
CBA	CMM-Based Appraisal
CBA IPI	CMM-Based Appraisals for Internal Process Improvement
CBA SCE	CBA software capability evaluation
CECOM	Communications-Electronics Command
CMM	Capability Maturity Model
COCOMO	Constructive Cost Model
COTS	commercial off-the-shelf
CRADA	Cooperative Research and Development Agreements
CSAB	Computing Sciences Accreditation Board
CSEE	Conference on Software Engineering Education
CSL	Computer Science Laboratory
DDR&E	Deputy Director of Research and Engineering
DoD	Department of Defense
DSMC	Defense Systems Management College
DSSA	domain-specific software architectures
EDCS	Evolutionary Design of Complex Software
EMM	Engineering Maturity Model
ESC	Electronic Systems Center

ESC PRISM	Electronic Systems Center's Portable Reusable Integrated Software Modules
FAA	Federal Aviation Administration
FFRDC	federally funded research and development center
FIRST	Forum of Incident Response and Security Teams
GAO	Government Accounting Office
ICCP	Institute for the Certification of Computing Professionals
IEEE	Institute of Electrical and Electronic Engineers
IEEE/ACM	Institute of Electrical and Electronic Engineers / Association for Computing Machinery
ISO	International Organization for Standardization
JAC	Joint Advisory Committee
JAST	Joint Advanced Strike Technology
KPA	key process area
MBSE	model-based software engineering
MEL	Manufacturing Engineering Laboratory
MSC	Measurement Steering Committee
NCEE	National Council of Examiners for Engineering and Surveying
NII	National Information Infrastructure
NIST	National Institute of Standards and Technology
NSPE	National Society of Professional Engineers
OAST	Office of the Assistant Secretary of Defense
OUSDA	Office of the Under Secretary of Defense for Acquisition
P-CMM	People Capability Maturity Model
POSIX	portable operating system interface
PSP	Personal Software Process
ROI	return on investment
SAMM	Software Acquisition Maturity Model
SE-CMM	Systems Engineering Capability Maturity Model
SEE	software engineering environment

SEPG	software engineering process group
SES	Software Engineering Symposium
SPC	Software Productivity Consortium
SPI	software process improvement
SPICE	Software Process Improvement Capability dEtermination
SPIN	software process improvement networks
SRE	Software Risk Evaluation
STARS	Software Technology for Adaptable, Reliable Systems
TO&P	technical objectives and plans
USC-CSE	University of Southern California-Center for Software Engineering

Special Report
CMU/SEI-95-SR-027
January 1996

SEI Program Plans: 1996-2000
Volume II: One-Year Plans/Proposals

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, Pennsylvania 15213

Table of Contents

List of Figures

Introduction to Volume II

This document describes the work proposed for CY 1996. The strategic context for the work is provided in Volume I. The work proposals describe specific outputs in two categories. The baseline outputs are those that were approved in a previous year and for which work continues into 1996. The add-on proposals describe new outputs that the SEI is prepared to initiate in 1996. Note that anticipated funding will only enable ARPA to approve approximately one-half of these add-ons. While the SEI may seek customer funding for some of those not approved, the SEI can make no commitments at this time for any add-on outputs.

The SEI technical program encompasses work in four broad areas of interest and significance to the software engineering community at large: Software Process, Risk Management, Disciplined Engineering, and Trustworthy Systems. We have chosen to call these areas of effort software engineering technical impact areas to emphasize that they are areas of software engineering in which we intend to have significant impact. Although each of these areas represents a separate focus of enquiry, they are not independent of one another. In many cases, work in one area requires inputs from or yields insights into other areas. In such cases, expertise from the second area may be factored into the work plan.

Building on lessons from past experiences, the SEI purposefully seeks to increase integration among the separate areas. Therefore, the principles of teamwork and collaboration form the basis for the operational approach. Teams are formed to address specific problems or developments, and members are reformed into other teams as tasks are completed and new tasks are begun.

In complement to the four impact areas, the SEI technical program includes efforts that help to channel work outputs into targeted segments of the software engineering community. Two of these, Professional Infrastructure and Integrated Transition Strategies and Methods, are an explicit part of the SEI technical strategy as discussed in Volume I; and the third, Community Outreach, is a set of activities supporting our interaction with the general software engineering community. In these areas, efforts are focused on improving the transfer of knowledge to practice to lessen the transition lag and thereby speed up the maturity of software engineering practice as a whole. The intent in these areas is to apply current best understanding of transition mechanisms to transition specific software engineering technologies and to improve the effectiveness of the transition mechanisms used.

The chapters in this volume discuss the following impact areas and activity areas:

The SEI technical program represents a set of technical activities chosen to produce applicable outputs for specific segments of the software engineering community. Outputs are the key to the selection of activities, and customer needs for improved software engineering practices are the key to the selection of outputs. Thus, the following sections describe these efforts in terms of activity areas, each of which includes subsections on the problem motivating the activity, the customers and how they stand to benefit from the outputs, and the relationships among the various activities and outputs.

The chapters in this volume (with the exception of Chapter 7 on Community Outreach) follow a common structure:

For each activity area ▶

- Problem Statement
- Customers
- Rationale
- Benefits
- One-Year Objectives for 1996
- Baseline Work Outputs
- Proposed Add-On Work Outputs
- Related TO&P Activities

There are a fair number of add-on proposals described in this volume that will be performed by cross-area teams. The growth in joint technology efforts results from the mutual benefits and synergy to be gained from working on complementary technologies and products, resulting in outputs that, we believe, will deliver greater value at a lesser cost than if the work were pursued independently. The joint work outputs, the collaborating SEI areas, and the work output identifiers of the descriptions are as follows:

Work Output	Impact Area	Impact Area	Impact Area	Impact Area
Integrated Product Development (IPD) Framework (1996)	Process SP-1A	ITSM* SP-2A		
IDEAL Model Products (1996-1997)	Process SP-7A	ITSM SP-8A		
Software Engineering Information Repository (1995-1997)	Process SP-14A	Risk RM-2B	DE DE-3A	ITSM SP-15A
Risk Management for Open Systems (1996)	DE DE-9A	Risk		
Evaluation and Application of Software Process Modeling Technology (1996-1997)	DE DE-12A	Process		
Security Guidelines for Open Systems Acquisition (1996-1997)	TS TS-4A	DE		
Software Engineering Framework for Trustworthy System Development (1996-1997)	TS TS-8A	DE		

* ITSM = Integrated Transition Strategies and Methods
DE = Disciplined Engineering
TS = Trustworthy Systems

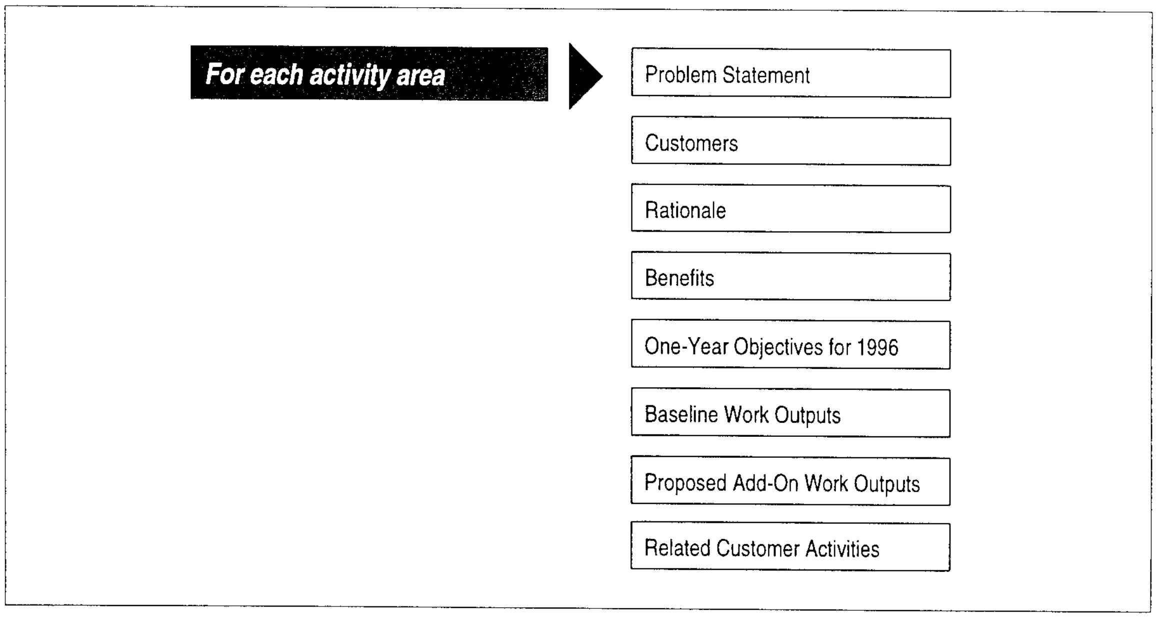
Volume II
Chapter 1 Table of Contents

1 Software Process

To improve the state of the practice of software engineering in the United States, organizations involved in software intensive systems must establish an organizational capability (rather than be dependent on individuals) for developing and evolving software. This capability must be based on sound management practices that support a disciplined, defined, and measured software engineering process. Organizations must be able to execute defined engineering processes consistently across all projects in the organization, rather than have only a few successful projects, with others missing their objectives of cost, schedule, quality, and function. Furthermore, organizations must be able to learn from their experience to improve their capability.

The Process area's focus is in three activity areas: process maturity modeling, Capability Maturity Model^{SM}/CMM^{SM}[1]-based software process improvement (SPI), and software engineering measurement. These encompass the activity areas from 1995 and also contain some additional subjects, with some refocusing for integration and increased emphasis and scope. The mapping from 1995 to 1996 is shown in Figure 1-1.

In this chapter, the sections for this impact area are as follows:



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

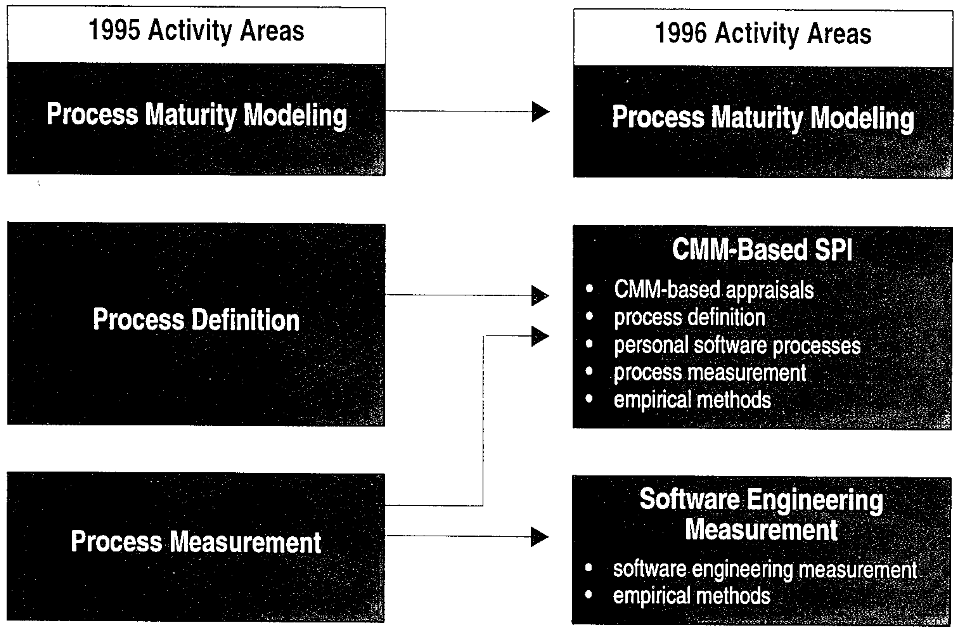


Figure 1-1: Mapping of Process Activity Areas

1.1 Process Maturity Modeling

1.1.1 Problem Statement

Across industries and government agencies today, the problems are similar. Processes are generally improvised during the course of a software development project. Even if software processes have been specified, they are often ad hoc, not followed, nor follow-able. Managers are usually focused on solving immediate crises. Schedules and budgets are routinely exceeded because they are not based on realistic estimates. There is little objective basis for judging product quality or for solving product or process problems. Therefore, product quality is difficult to predict. When hard deadlines are imposed, product functionality and quality are often compromised to meet the schedule.

The problem is that organizations are frequently not providing the infrastructure and support necessary to help projects avoid these problems. In the absence of organization-wide well-defined software processes, repeating results depends entirely on having the same individuals available for the next project. Success that rests solely on the availability of specific individuals provides no basis for long-term productivity and quality improvement throughout an organization.

Although software engineers and managers know their problems in great detail, they may disagree on which improvements are most important. Without an organized strategy for improvement, it is difficult to achieve consensus between management and the professional staff on what improvement activities to undertake first.

While the above is true for software engineering, it is equally true for systems engineering. Systems engineering needs guidance in meeting its potential as an integrating discipline; in addition, because systems engineering is an upstream supplier to software engineering, shortcomings in the systems engineering process have a major impact on software engineering in an organization.

Finally, as organizations implement process improvements, they frequently encounter difficulty in making the transition to new processes and technologies. This is particularly essential in today's world of increased competition, rapid market change, and rapid improvement in state-of-the-art technology. Organizations find it difficult to develop and retain a skilled, knowledgeable, competent, and motivated work force, which is essential to making product quality and process improvement happen and to remaining competitive.

1.1.2 Customers

Software is already an important technology in many industries and to many government agencies. Increasingly, it is becoming *the* critical technology in *almost every* industry. The customers of our products are thus potentially every product development and service organization in just about every sector of government and industry.

Software is already the critical technology for information systems organizations, which lay the information infrastructure by which the mission and business of a company or agency can operate. These, too, are customers of our products.

Many organizations out-source software development. Still others integrate software procured from multiple sources (e.g., working with multiple subcontractors and/or with commercial off-the-shelf [COTS] software). Critical in both cases is greater insight into the capability of the providers of software. Our products will help both ends of this relationship. They will help product development organizations improve their product development, manufacturing, and service processes. But they will also help the procurer select from multiple sources for software and serve as a basis for ensuring the quality of the procured or purchased software product.

Thus our customers include organizations in just about every segment, whether market driven or contract driven, whether government or industry, or whether multinational. Finally, our customers include those who provide process-based products and services to all of these types of organizations. Such organizations include education, training, and consulting businesses, as well as standards organizations.

1.1.3 Rationale

"After two decades of largely unfulfilled promises about productivity and quality gains from applying new software methodologies and technologies, industry and government organizations are realizing that their fundamental problem is the inability to manage the software process."[2]

The above is the first sentence from the first chapter of the CMM version 1.1 written more than two years ago.[3] Since that time, software has become even more critical to an organization's, and the nation's, well-being and success.

Why is this? First, the continued dramatic advances in hardware technology go relatively unmatched in software engineering. It is software that increasingly governs the introduction of new product functions and features, and from there, an organization's, and then the nation's products and services.

Thus, to retain control of their markets, product managers must be able to control software. With the introduction of the CMM for Software in 1991, organizations had an explicit roadmap to guide their software process improvement that would help address many of the problems stemming from the lack of an organization-wide well-defined process. (By "roadmap," we mean a guidebook describing evolutionary stages to higher process capability, and a description of the practices that make up each stage.) Here is evidence that the CMM is addressing an important need:

1. There has been a dramatic and growing response by the software community to the availability of the CMM: Figure 3-5 in Volume I shows the rapid growth in attendance at annual software engineering process group (SEPG) meetings since the release of the CMM for Software; in particular, note the growth just in the last year (from 918 to 1248). Other figures in Section 3.2 of Volume I also document that dramatic response to the CMM.
2. The community is increasingly demanding capability maturity models to address related disciplines, e.g., systems engineering and development of an organization's talent.

2. *Report of the Defense Science Board Task Force on Military Software*, Office of the Under Secretary of Defense for Acquisition, Washington, D.C., September 1987.

3. Mark C. Paulk, Bill Curtis, Mary Beth Chrissis, Charles V. Weber, *Capability Maturity Model for Software, Version 1.1*, Software Engineering Institute, CMU/SEI-93-TR-24, February 1993.

3. Other industries, countries, and standards organizations are copying the CMM approach. For example, Bell Canada's "Trillium" and Etnoteam's "Bootstrap" explicitly identify the CMM as the major influence and basis for much of their process maturity modeling work. The impetus to form the European Software Institute was reportedly due, in large part, to the leadership of the SEI in software process improvement. When the International Organization for Standardization and International Electrotechnical Commission (ISO/IEC) chose to develop a standard for software process assessment, SPICE (Software Process Improvement Capability dEtermination), the SEI was asked to co-manage the development of the SPICE process maturity model.

The business results of software process improvement are being gathered and reported (see Section 1.3), and they support the value of the process maturity model approach to SPI.

SEI field work over the years has revealed that software engineering problems frequently have a root cause in the systems engineering portion of a project or organization. Based on its core competence in process maturity modeling, the SEI was approached to coordinate a national effort to develop and maintain a Systems Engineering Capability Maturity Model (SE-CMM). The SEI and other leading organizations have invested significant labor in this effort. Still more recently, the SEI is developing a CMM that addresses the difficulties that organizations experience in moving up maturity levels due to the need to train, organize, align, and motivate their staff. Two TO&P partners are helping to fund this People Capability Maturity Model (P-CMM) work.

Given its experience with the CMM for Software, and more recently with the SE-CMM and P-CMM, the SEI is in a position to assist the community in developing appropriate capability maturity models that help organizations achieve their business objectives in product development and establish a common frame of reference for accelerating organizational learning. This is summarized in Figure 1-2.

1.1.4 Benefits

The CMM provides the basis for continuous SPI. The primary benefit to the software community is realized when the model is transitioned, via multiple transition enablers, into the state of the practice in organizations. Business results of SPI, based on the CMM, have been publicly reported by a few organizations and are continuing to be gathered, analyzed, and now reported by the SEI. For a summary, see Figure 3-4 in Volume I. The business results reflect increasing product development capabilities and predictability in organizations investing in SPI. The CMM-based approach strengthens an organization's ability to communicate, improve, and measure its effectiveness.

A CMM-based effort integrates well with the other Total Quality Management initiatives of organizations. It enhances those initiatives in software by the nature of its software-specific orientation. It reinforces and enhances software management process improvement.

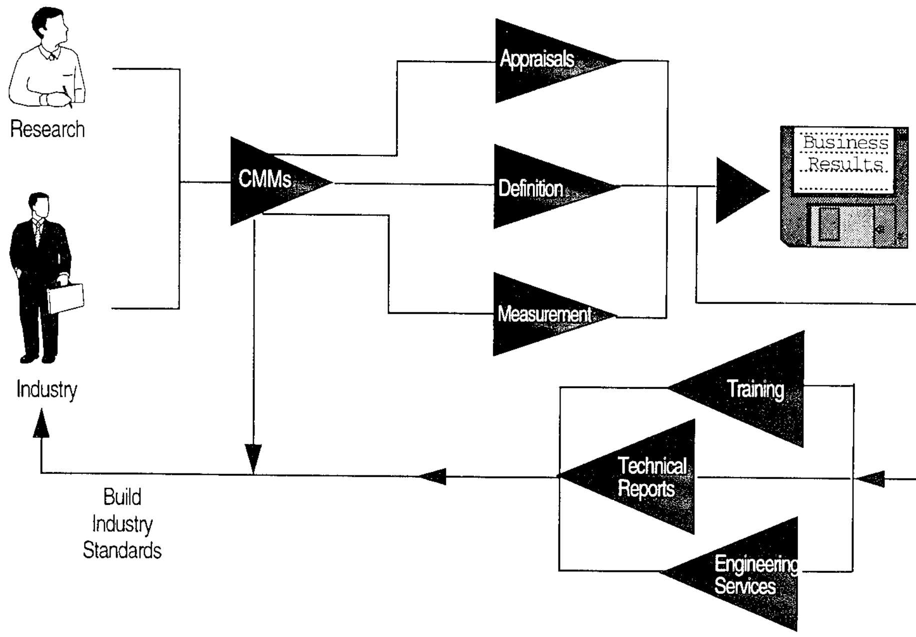


Figure 1-2: The SEI's process maturity modeling work is the basis for technology and standards that lead to improved business results

As the CMM evolves, it will address higher levels of maturity more completely. It will reflect more mature best practices and state-of-the-art practices over time. In addition, the CMM is an additive model and can evolve by integrating with other models, e.g., the SE-CMM, the P-CMM, and the Software Acquisition Maturity Model (SAMM); and be tailored through careful extensions to address the requirements of specific disciplines such as security engineering (Trusted CMM [T-CMM]).

As the basis for continuous SPI, the CMM also integrates the other outputs and services provided by the Process area. Appraisals used for internal process improvement (IPI) and for Software Capability Evaluation (SCE) for subcontractor selection are tied to the CMM. Through development based on the CMM Appraisal Framework (CAF), these appraisals will produce more consistent and repeatable findings and ratings. The follow-on activities of SPI (action planning, process definition, and process measurement) are also tied to the CMM. These uses of the model result in a commonality in language and vision for SPI, as well as a roadmap to realizing improved organizational process maturity.

In the systems engineering modeling effort, similar benefits are starting to be realized in the systems engineering community. The SE-CMM provides guidance for improving systems engineering processes and potentially provides high leverage in supporting DoD initiatives toward using commercial standards and products. As with the CMM, this model provides a shared vision of systems engineering excellence for the systems and software engineering communities.

The SE-CMM is compatible with and closely related to the CMM and P-CMM. It is also providing us with the opportunity to pilot the modeling architecture that the SEI has proposed to the ISO/IEC-associated SPICE project and providing insight into how that change in architecture might enhance version 2.0 of the CMM for Software.

Finally, the SE-CMM is starting to provide a basis and integrating framework for appraisal and process improvement efforts in the systems engineering community much as the CMM has done for the software community. It is becoming a reference model for assessing current practices; for performing supplier selection; for planning, implementing, and measuring process improvement efforts; and for determining the business results of such efforts.

For process maturity modeling, we see the trends shown in Figure 1-3.

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Address software and disciplines that impact software Provide integrated PI (process improvement) reference models Build broad community consensus	• Very few organizations are at higher maturity levels (MLs). • A lack of attention to other disciplines that impact software • Very few organizations have an integrated PI program. • A perception that CMMs are for U.S. defense contractors only	• Market leaders in competitive industries gain higher MLs. • CMMs for related eng'g disciplines are used in integrated product & process development. • World-class organizations from across the community are engaged in PI using multiple CMMs. • Increasingly, PI also addresses people & systems eng'g.	• Improvement efforts tie process, people, and technology, in multiple disciplines, in synergistic fashion. • Integrated CMMs are owned by global eng'g communities. • Widespread adoption of CMMs and higher MLs, especially in highly competitive industries, e.g., consumer software	• # of customers who base SPI on CMM • % of Fortune 100, 500 who do SPI • ROI across industries • Improvement in community MLs • # or % of other relevant disciplines covered by CMMs • % using 2 CMMs • Synergism in CMMs reflected in higher quality, productivity, time-to-market
Harmonize with standards	• Improvement efforts are confounded by multiple software process-related reference models.	• PI based on CMMs is generally adequate to address most process standards.	• Standards in disciplines that impact software are strongly based on CMMs.	• % of each process standard which is CMMs-based • % of each CMM addressed in a std

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Enable efficient PI across disciplines	• Organizations are uncertain on how to efficiently integrate PI activities across disciplines. • It takes years to go up MLs.	• PI is more frequently tied to business objectives and outcomes. • Leading organizations within competitive industries are engaged in rapid PI using CMMs.	• Strong ROI from multiple CMMs-based PI • In highly competitive industries, within one year, an organization can climb one or more MLs across multiple CMMs.	• Reduction in cycle time needed to radically improve business outcomes • % of SEEs & EEs (engineering environments for non-software disciplines) based on CMMs & supportive of PI

Figure 1-3: Trends in Process Maturity Modeling

1.1.5 One-Year Objectives for 1996

Address software and disciplines that have an impact on software. The objective here is to address disciplines that are relevant to improving the state of software engineering practice. The current version of the CMM, version 1.1, lacks guidance at maturity levels 4 and 5 and does not harmonize well with ISO 9001, both of which the community needs. In 1996, we plan to fold in the work of various process focus groups, to produce a draft (for broad community review) of a revised CMM that addresses these two needs. We also plan to release the next version of the SE-CMM to fold in the lessons learned after more than one year of broad review and use. Finally, we plan to maintain the P-CMM and provide support to the development of the SAMM.

Provide integrated process improvement reference models. This is really a multiyear objective whose subobjectives specific to 1996 are (1) to develop a draft framework (i.e., a structure for relating information that is not formalized into a model) for all process modeling work, a harmonizing framework in which the SEI CMMs all fit, and (2) to produce a draft process maturity modeling guidebook that guides the development of CMMs for other disciplines that fit with those developed/codeveloped by the SEI.

Build broad community consensus. For each CMM it is important to develop strong community consensus on a vision of exemplary practice. Toward this end, the SEI will continue to expand its correspondence and review groups to include other industries and to provide more direct access to CMMs under development. We will seek to improve our capability to address the many queries and requests received each day on how to implement a specific practice or process area of one of the CMMs.

Harmonize with standards. The key here is to identify process standards that the community may need to comply with, and to influence the content of these standards to ensure that (1) the standard does not contain a vast amount of new unproven material and (2) the standard includes content from the CMMs to the extent appropriate. Depending on the outcome, it may be necessary to write change requests against a CMM to better harmonize with that standard. The SEI's ongoing leadership and coordination within the U.S. on SPICE standardization will continue in 1996, leading to completed field experiences and proposed changes to the emerging standard and, as appropriate, the CMM.

Enable efficient improvement across disciplines. A number of CMMs have been released. How can an organization efficiently address several disciplines in one improvement program? Our process maturity modeling plans include development and piloting of improvement guidelines to help organizations more rapidly reach higher maturity levels across several disciplines simultaneously. For 1996, we plan to develop guidelines that address software and systems engineering improvement and incorporate people issues into software process improvement programs. We also plan to develop a framework to address the application of multiple CMMs in Integrated Product Development (IPD).

1.1.6 Baseline Work Outputs

This section describes the baseline work outputs for the activity area related to process maturity modeling. These outputs were approved for 1996 through the basic funds allocation process. Appendix A lists all baseline and proposed add-on work outputs.

SP-1B CMM Version 2 (1996-1998)

When completed, CMM versions 2.0 and 2.1 will be published, probably as SEI technical reports. These will be a revision to the current CMM version 1.1. This task is a continuation of two related tasks from 1995, "CMM Maintenance" and "CMM Version 2.0."

Purpose

The purpose is to revise CMM version 1.1 to address change requests, harmonize with new and evolving standards (e.g., ISO 9001, SPICE, MIL-STD-498), and expand the descriptions of higher maturity organizations.

Customers

The broad software community is the customer. SEI's field experience has demonstrated that the CMM is broadly applicable across different industries, organization sizes, and organization types (i.e., whether product driven or service driven, and for market-driven as well as contract-driven organizations). The emphasis of the CMM, however, will remain on control of large, complex software systems.

Collaborators

The broad software community is providing resources toward the development of this product by becoming involved as resident affiliates, submitting change requests, participating in workshops and focus groups, and participating in reviews and pilots.

Approach

- Establish external focus groups to contribute to v2 efforts (1995).
- Accumulate change requests, etc., until 1 July 1995.
- Disseminate prototypes of proposed new key process areas, architectures, etc., for review (1995-1996).
- Develop requirements specification and review (1995).
- Develop drafts of CMM v2 and review (1996).
- Obtain closure on CMM v2 in collaboration with reviewers and CMM Advisory Board (1996-97).
- Release CMM v2.0 in early 1997.
- Synchronize release of CMM v2.1 for widespread use with related products (e.g., CMM-Based Appraisal for Internal Process Improvement [CBA IPI] and CMM-Based Appraisal for Software Capability Evaluation [CBA SCE]) and planned releases of ISO 9001 and SPICE (probably in early 1998).

SP-2B Maturity Model Integration Framework (1996-1997)

This work encompasses creating and disseminating a framework (i.e., a structure for relating information that is not formalized into a model) for describing the current and intended relationships of existing and potential maturity models to each other. It also involves development of guidelines for maturity model developers and interface specifications to help in revising current CMMs to fit more synergistically into the integrated framework. (By "interface specification," we mean an identification of where there is content overlap between the maturity models and a description of how each model is to handle that overlap. For example, "training" is addressed in each CMM. An interface specification would identify "training" as an area of content overlap and would describe where and how the different aspects of training are to be addressed.)

This task is based on a feasibility study conducted by the SEI in 1994 investigating the need for integrating the SEI's CMMs, and on subsequent work in 1995 toward creating a draft integrated framework.

Purpose

The purpose is to ensure that (1) organizations using multiple CMMs can use them synergistically and (2) maturity model developers, both internal and external, have guidance in producing new CMMs that will fit with others already built.

Customers

Customers include organizations that are trying to use multiple CMMs, maturity modelers both internal and external to the SEI, providers of CMMs-based products and services, and SPI and other process improvement vendors.

Collaborators

Collaborators would potentially include external model developers.

Approach

- Provide draft CMMs integrated framework for review (late 1995).
- Develop draft "CMMs Developer's Guidelines" for external parties wanting to build maturity models that interface with SEI CMMs (1996).
- Finalize the framework (1996).
- Develop interface specifications between SEI CMMs as input to the CMM version 2 development and as input to the SE-CMM revision (facilitates revision and integration of P-CMM and the other CMMs) (1996-97).
- Finalize the "CMMs Developer's Guidelines" (1997).

SP-3B SPICE Product Suite (1996-1997)

The SEI participates in and tracks various standards efforts that develop reference models for software engineering. At present, the most significant effort underway is SPICE, an international project whose program of work, a standard for software process assessment, was approved by ISO/IEC JTC1/SC7 (the major international software engineering standards group). The major document of concern is the SPICE Baseline Practices Guide, as it could document required practices that are too prescriptive, are not economically sensible, or otherwise distract an organization from pursuing a sound SPI program. The Baseline Practices Guide includes practices on customer interface, project management, software development and maintenance, support processes, organizational asset development, and process management.

This task is a continuation of the 1995 task, "ISO-SPICE Technical Report." The change in name reflects the relative autonomy of the SPICE Project from the ISO/IEC organization and the change in focus from development of technical reports to piloting and revising a suite of products.

Purpose

The SEI has three objectives in helping develop the SPICE product suite: (1) ensure that the Baseline Practices Guide and other components of the SPICE product suite do not contain practices that put organizations embarked in CMM-based SPI at a disadvantage, (2) try to ensure that these organizations have an advantage in maturing their software processes, and (3) try out promising ideas that might otherwise not be tried and incorporate the lessons learned into our own products.

Customers

Customers include the U.S. software community and broader worldwide community when it serves U.S. interests.

Collaborators

Major collaborators in the U.S. involved in SPICE include Allied Signal, AT&T, and Bellcore.

Approach

- Broadly disseminate information and opportunities to participate on SPICE (1993-97).
- Represent U.S. on SPICE Management Board which controls project schedule (1993-97).
- Pilot test "continuous model" architecture in SPICE to better understand the merit for its possible use in CMM version 2 (1993-97).
- Apply lessons learned from use of this architecture in our process maturity modeling work (1994-97).
- Coordinate SPICE trials in the U.S. (1995-97).
- Coordinate U.S. positions on the SPICE product suite and on its possibly becoming a standard (1993-97).

1.1.7 Proposed Add-On Work Outputs

This section describes the proposed add-on work outputs for the activity area related to process maturity modeling. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

SP-1A Integrated Product Development (IPD) Framework (1996)

IPD is a systematic approach to product development that achieves a timely collaboration of necessary disciplines throughout the product life cycle to better satisfy customer needs. The characteristics of IPD are

- Integration of the processes of product development.
- Integration of input and output of work products and processes.
- Integrated planning.
- Integration of requirements.
- Efficient data flow and communication.
- Integration of skills/disciplines.
- Timely collaboration to make decisions and perform tasks.
- Coverage of all disciplines, skills, and activities involved in product development.

This task is based on (1) practical experience obtained through implementation of Integrated Product and Process Development (IPPD) and Concurrent Engineering concepts applied at several organizations participating in the SE-CMM and IPD collaboration and (2) a decision by the steering group of that collaboration that there were now sufficient experiences with these concepts and capability maturity modeling to develop a practical integrated product development framework.

Purpose

The purpose is to establish a common framework for integrating the disciplines applied in IPD that can be used to appraise IPD projects and guide them in establishing and improving the processes used in building high-quality products rapidly and successfully. A secondary purpose, served by item SP-2A below, is to understand, in particular, the aspects of teaming philosophies and practices that are increasingly being used in organizations.

Customers

Customers include organizations that are trying to implement IPD and that may be using multiple CMMs to improve their capability and competitiveness. Eventually, customers of organizations such as those described above may also wish to assess their "IPD capability." In addition, maturity modelers, both internal and external to the SEI, will not only have guidance (outputs of SP-2B) on how to make their models fit with the CMM for Software, the SE-CMM, and the P-CMM, they will also be able to reuse the common practices and, instead of repeating these, establish an appropriate set of good practices specific to their discipline.

Collaborators

Initial efforts in the development of an IPD Framework began in 1995. Collaborators for 1996 work are still being determined; however, steering group members in the broader SE-CMM-related program of work for 1995 included Hughes Aircraft, Lockheed Corp., Loral Federal Systems Co., National Institute of Standards and Technology (NIST), Office of the Secretary of Defense, Software Productivity Consortium, and Texas Instruments. Collaborators will provide, at a minimum, in-kind resources to help in developing and reviewing the IPD Framework.

Approach

- Develop a draft IPD Framework based on the draft "CMMs integrated framework" (SP-2B), the results of a study of highly effective software development teams (SP-2A), and an analysis of what is common across CMMs and what is specific to each discipline (1995-96).
- Populate the framework with successful IPD practices in the field and IPD-related practices in the CMM, SE-CMM, and P-CMM, and review these practices (1995-96).
- Identify and include the specific sets of practices unique to software engineering and to systems engineering, and to other appropriate disciplines (e.g., marketing and manufacturing) (1996).

- Coordinate the results of this effort with future revisions to the CMMs integrated framework (final version due in 1996), the SE-CMM (version 2.0 due in 1996), and CMM (version 2.0 due in 1997).

SP-2A Assessment of Highly Effective Software Development Teams and Environments (Feasibility Study) (1996)

The report produced by this feasibility study will discuss to what extent the SEI should initiate increased efforts focused on factors contributing to the effectiveness of software development teams. The output will describe what makes software teams effective and the role of software teams in an increasingly team-based corporate environment. If this is an area in which the SEI can make a significant and useful contribution, we will recommend additional work to be pursued and explain why this work will be valuable. The output of this study will help support the Integrated Product Development (IPD) Framework effort (SP-1A).

Purpose

The increasing use of teams for a variety of purposes in U.S. industry means there is more information available that could be applied to the process of creating software development teams. In addition, software development teams are increasingly a part of integrated product development teams and the SEI needs to understand their role in such situations. The proposed study will provide information on the current use of software and integrated product development teams and the potential that advanced teaming concepts have on the productivity, quality, and cycle time (time to market) of software development.

Customers

The SEI has received numerous requests from clients for team building and team development for product development teams.

Collaborators

Clients having special experience in developing software teams and integrated product teams will be asked to collaborate as part of this study.

Approach

- Review studies and literature on the use, structure, and outcomes of teams that develop software (and similar design-intensive products).
- Examine the state of the practice regarding the formation and implementation of product teams. State-of-the-practice data will be gathered from typical product organizations and compared to the practices and results obtained in advanced organizations using teams.
- Report the findings and recommendations for possible follow-on work.

This work is proposed as a research project. The literature on teams will be examined to assess the state of the art in software teaming and integrated product development technology. This examination of the state of the art will be contrasted with a study of the state of the practice at client locations.

SP-3A SE-CMM Version 2.0 (1996-1997)

This work encompasses the production of version 2.0 of the SE-CMM and its maintenance. Version 2.0 will need to reflect the lessons learned from using version 1.0 in the 1995-96 timeframe for the purpose of improving an organization's systems engineering process capability. This task is a continuation of two 1995 tasks, "Systems Engineering Capability Maturity Model (SE-CMM) Development and Integration" and "Systems Engineering Capability Maturity Model (SE-CMM) Maintenance."

Purpose

The purpose is to revise the SE-CMM based on the lessons learned through its use in assessment and process improvement programs within the systems engineering community and to extend it to address post-delivery phases of the systems engineering life cycle.

Customers

Customers include the U.S. systems engineering community and broader worldwide systems engineering community, when it serves U.S. interests.

Collaborators

Collaborators for 1996 work are still being determined; however, steering group members in the broader SE-CMM-related program of work for 1995 included Hughes Aircraft, Lockheed Corp., Loral Federal Systems Co., NIST, Office of the Secretary of Defense, Software Productivity Consortium, and Texas Instruments.

Approach

The continuing role of the SEI in 1996 is to

- Lead and expand community involvement.
- Work with the SE-CMM Steering Group in the disposition of received change requests.
- Coordinate the production of version 2.0 of the SE-CMM, using the CMMs integrated framework (SP-2B) to adjust the SE-CMM architecture and using the IPD Framework to adjust the practices (late 1996) (SP-1A).
- Continue to gather review comments and track lessons learned from implementing the SE-CMM (through the remainder of 1996).

1.1.8 Related Customer Activities

In 1994-95, the U.S. Army and the Office of the Secretary of Defense funded development of the People CMM (P-CMM), including supporting products. The P-CMM is an adaptation of the CMM focused on developing an organization's talent. The motivation is to radically improve the ability of software organizations to attract, develop, motivate, organize, and retain the talent needed to steadily improve software development capability. Although the P-CMM is being initially developed with a focus on software and information systems development, the principles and many practices of the P-CMM apply equally to systems engineering. In 1996, we plan to continue piloting the P-CMM and its supporting products, including one or more modified appraisal methods. We intend to investigate carefully how people issues should best be incorporated into an organization's SPI program to maximize efficiency of resources and the effectiveness of the improvement. Sources for funding this in 1996 might include the current funding sources.

In 1994-95, the National Security Agency (NSA) funded the tailoring of the CMM to include trusted system requirements, its incorporation into CBA IPI, and one or more pilot tests of the approach. It is possible that this work will be extended in late 1995 and/or 1996 to include a tailoring of the SE-CMM with an additional process area to address concerns related to systems engineering security. This work helps prove the feasibility of CMM extensions and adaptations to meet the needs of specific disciplines or domains such as security engineering.

In 1994-95, the Process area provided process maturity modeling expertise to the SAMM development effort. The purpose of the SAMM is to improve the practices of software acquisition organizations. The SAMM will closely complement the CMM, be based on best acquisition practice, and include the perspective of U.S. DoD contractors. It is described in Chapter 2 of Volume II. This work is expected to continue into at least part of 1996.

In 1994 through early 1995, the Process area delivered training on the CMM to the software community. To address the high demand for course delivery, and to shift our resources from training to product development, an Introduction to the CMM instructor training course was developed in 1994 in collaboration with Motorola, Citibank, Defense Information Systems Agency (DISA), Process Enhancement Partnership, and Software Technology Transition. In 1995, the SEI used this course as part of a licensing strategy to transition the capability to teach the Introduction to the CMM course to outside organizations to better meet the high demand for CMM training. We expect that most of the revenue generated from these licenses will be used in assuring the quality of the CMM training. Also in 1995, we are codeveloping an Advanced CMM Workshop with Motorola University.

1.2 CMM-Based Software Process Improvement

This section discusses the activity area related to CMM-based software process improvement and IDEAL^{SM}[4], including CMM-based appraisals, process definition, personal software processes, process measurement, and empirical methods.

1.2.1 Problem Statement

Many software organizations do not know the current state of their software engineering practices and management processes. Of those that do know their current state, many do not know how to go about improving to reach a desired state. Furthermore, once they have improved the current state, they need to revisit their software engineering practices and management processes to see if they need further improvement, perhaps in other areas.

Many software organizations lack the capability to define manageable processes and perform them with fidelity. They often lack process change control, which results in abandoning the process in time of crisis, rather than managing and adjusting the process. Frequently, software organizations do not use quantitative methods effectively in managing software projects and in measuring improvements in software products and processes. Data gathered to reflect the results of development and improvement efforts are not consistently defined, collected, interpreted, or reported.

In addition to organizational processes, and in support of them, related individual processes must be defined, measured, and followed with fidelity. Quality techniques that are compatible with the CMM need to be taught and applied at the individual software engineer's and engineering team's level to enhance transition of organizational processes and disciplined software engineering techniques.

The software engineering community needs data about the state of the practice of software development processes to determine

- Where organizations are relative to the state of the practice.
- Which innovations in the practice contribute to improved performance and capability.
- The value of and return on software process improvement/investments.

1.2.2 Customers

Customers include the software development community, developers of software, acquirers of software, and suppliers of appraisals in government and industry. SPI champions, sponsors, and agents for change are also customers. However, since it is not process improvement for process improvement's sake, the business results of appraising, defining, measuring, and

4. IDEAL is a service mark of Carnegie Mellon University.

evaluating the results of processes and changes to processes are meaningful to managers and other stakeholders of an organization as well. In addition, decision makers who must allocate scarce resources to SPI efforts need information on the value and progress of SPI.

Specifically, clients include

- Military departments or services:
  - Air Force (Air Force Materiel Command, Electronic Systems Center, Standard Systems Center, Warner Robbins, Software Technology Support Center, Det 25).
  - Army (Communications-Electronic Command [CECOM], Missile Command, Tank-Automotive Command [TACOM]).
  - Navy (Naval Oceanographic Office [NAVOCEANO]).
- Defense agencies (DISA, Defense Financial Accounting Service, Defense Communications Agency, Defense Mapping Agency).
- Civil agencies (National Security Agency, National Oceanic and Atmospheric Administration, Federal Aviation Administration, Treasury, Patent and Trademark Office, Veterans Administration).
- Industry (Loral Federal Systems Company, Xerox, Citibank,Texas Instruments).

1.2.3 Rationale

In other engineering and manufacturing disciplines, the development processes are easily seen, and the measurement, evaluation, and improvement of those processes, along with supporting technology, is a given. In the software engineering discipline, which is people intensive, visibility of process is low and change is very slow. Management placing high priority on software process improvement and the development of software engineering skills and discipline is sorely needed. As shown by the most recent SEI data on software process assessments, the majority of software engineering organizations are at the Initial level of software process maturity (see Volume I, Figure 3-2).

Some organizations, particularly DoD contractors and some government organizations, have applied CMM-based SPI to improve their software process maturity to levels 2 and 3. The process improvement processes that these organizations applied independently need to be documented and communicated to the rest of the software community so more organizations can attain similar benefits. These improvement processes include appraising the current state of the engineering practice and management processes, developing improvement recommendations, prioritizing improvement efforts, defining best practices and process changes, defining measurements that are meaningful, following and measuring the processes, and determining the benefits of SPI (see Figure 1-4).

Additionally, support for higher capability maturity, i.e., moving organizations beyond level 3 if there is business value in doing so, requires research, communication, documenting, and

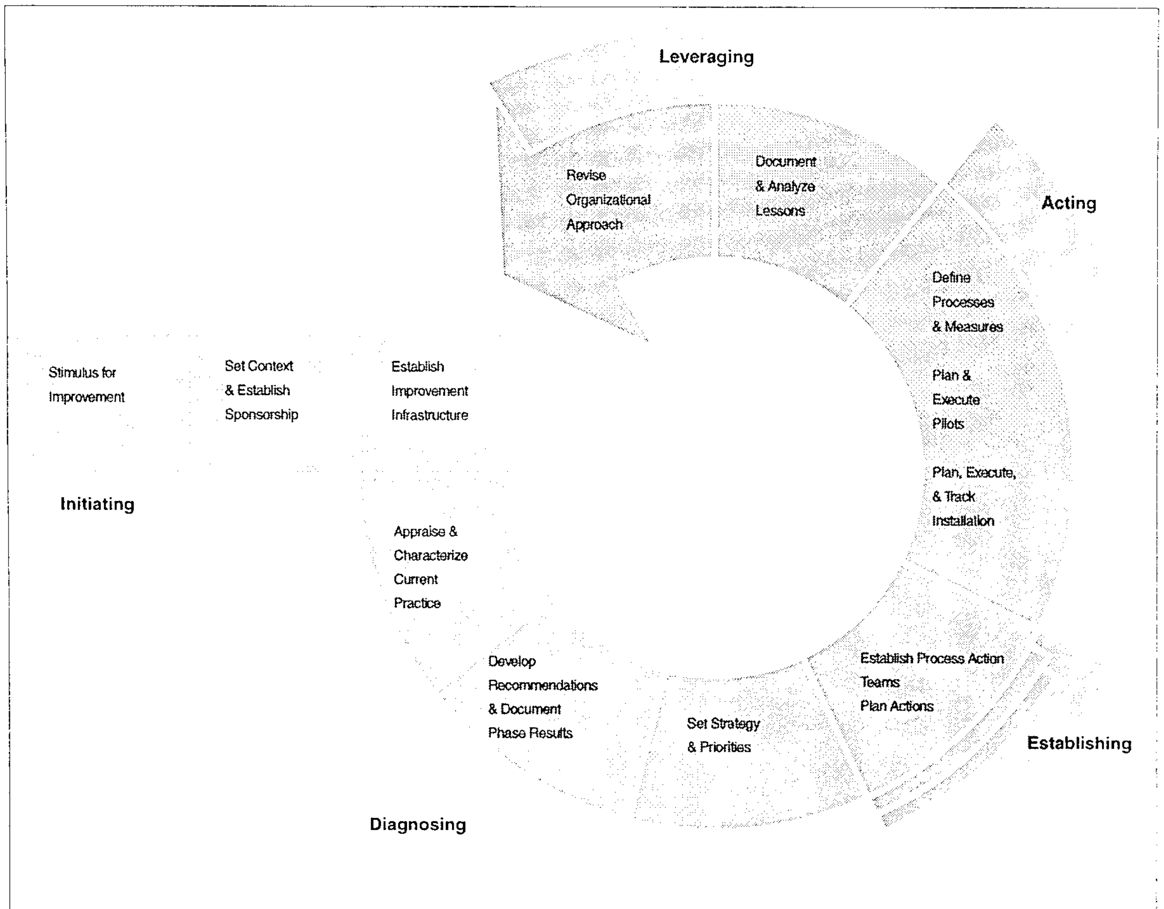

Leveraging
Revise Organizational Approach
Document & Analyze Lessons
Acting
Define Processes & Measures
Plan & Execute Pilots
Plan, Execute, & Track Installation
Stimulus for Improvement
Set Context & Establish Sponsorship
Establish Improvement Infrastructure
Initiating
Appraise & Characterize Current Practice
Develop Recommendations & Document Phase Results
Set Strategy & Priorities
Establish Process Action Teams Plan Actions
Establishing
Diagnosing

Figure 1-4: The IDEAL Model: An Integrated Approach to SPI

transition of process technology into those organizations. There are very few organizations reporting that they are CMM levels 4 or 5. Understanding best practices in such areas as level 4 and 5 appraisals, process definitions, process metrics, and process value is needed if organizations are to continue to mature and improve.

1.2.4 Benefits

In response to the software improvement community's expressed concern for more and better data about the results of SPI, the SEI conducted an initial study of 13 organizations' experiences with CMM-based SPI. The goals of this effort were to

- Collect and publish data that management can use to help guide decisions about investment in SPI.
- Provide managers and practitioners with SPI experiences linked to the adoption of CMM practices.

Thirteen organizations participated in the initial study. They represent DoD contractors, commercial organizations, and military organizations. The organizations represent a wide range of process maturity levels. The range of application areas is also diverse, including telecommunications, embedded real-time systems, information systems, and operating systems. The participating organizations were Motorola; Northrop; Bull HN; Schlumberger; GTE Government Systems; Siemens Stromberg-Carlson; Hewlett Packard; Texas Instruments; Hughes Aircraft Co.; U.S. Air Force, Tinker Air Force Base (AFB) Logistics Center; Loral Federal Systems (formerly IBM-FSC); U.S. Navy Fleet Combat Direction Systems Support Activity; and Lockheed Sanders.

The results were organized and presented by several categories: cost of the process improvement effort, productivity, schedule, quality, and overall business value of the SPI effort. These quantities were measured with different data definitions in the various organizations, so we restricted our analysis to changes within each organization over time. In Figure 1-5, each category of data is presented with the range of data values that were reported.

Category	Range	Median	Number of Data Points
Years Engaged in SPI	1 - 9	3.5	24
Yearly Cost of SPI per Software Engineer	$490 - $2004	$1375	5
Productivity Gain per Year	9% - 67%	35%	4
Early Defect Detection Gain per Year	6% - 25%	22%	3
Yearly Reduction in Time to Market	15% - 23%	19%	2
Yearly Reduction in Post-Release Defect Reports	10% - 94%	39%	5
Business Value (savings/cost of SPI)	4.0 - 8.8:1	5.0:1	5

Figure 1-5: Categories of SPI Data

An in-depth discussion of these data is provided in Chapter 3 of the technical report titled *Benefits of CMM-Based Software Process Improvement: Initial Results* (CMU/SEI-94-TR-13).

For CMM-based software process improvement, we see the trends shown in Figure 1-6.

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Integrated software process improvement	• A few projects/organizations have their own approach.	• A visible model is in limited use and experience using the model is documented.	• Widespread use of the model is the practice.	• Maturity profile is improving. • SPI cycle time is decreased.
CMM-based appraisals (CBAs)	• A few organizations have piloted CBAs. • Training and appraisers are available.	• Many organizations will be using CBAs to baseline the current state of their software engineering practices and management processes.	• Most organizations will use CBAs to support continuous SPI. • Certification of organizations' software process capabilities.	• Many more organizations reporting the state of their software process maturity.
Organizational processes and measures	• Few organizations have standard processes and process measures defined.	• Defined method for defining software processes and measures is available and taught. • Methods are in use.	• Widespread use of methods for defining software processes and measures • Automation support and vendor support are available.	• Maturity profile is improving toward level 3. • SEI offerings are widely available from third party suppliers.
Individual processes and measures	• A few have completed Personal Software Process (PSP) training.	• PSP training is widely available. • Industrial use data are becoming known. • Application in team environment is defined and taught.	• Many software engineers use PSP techniques and report results. • Results of team software process use are becoming known.	• Maturity profile is improving toward level 4. • Widespread offerings of PSP are available.
Benefits of software process improvement	• Initial study reported. • Initial business objectives of SPI identified.	• Additional studies validating the CMM and continuous process improvement are published. • More organizations identify their business objectives.	• Value of the CMM and improvement are widely accepted. • Business objectives tie to SPI.	• Investment in continuous software process improvement doubles. • Business value reported.

Figure 1-6: Trends in CMM-Based Software Process Improvement

1.2.5 One-Year Objectives for 1996

Provide support in use of CMM-based software process improvement:

- Define an integrating framework for SPI (the IDEAL model).
- Position the Process area's products and services in a visible way along the IDEAL model.
- Maintain and extend the CMM Appraisal Framework (CAF), the CBA IPI assessment method, and CBA SCE method.
- Extend and exploit the CAF, CBA IPI, and CBA SCE to other related CMMs.
- Support transition partners, TO&P, and Cooperative Research and Development Agreement (CRADA) customers.
- Provide community involvement support for conferences, software process improvement networks (SPINs), SEPGs, classes, tutorials, etc.

Develop advanced support for software process definition:

- Complete documentation and pilot test of a software process definition method.
- Complete documenting specifications for customers' software process guides.
- Focus on higher maturity issues, e.g., process architectures, tailoring project-specific processes, process technology support.
- Continue transition of personal software process.
- Initiate the development of team software process and software process improvement for small teams/organizations.

Study and report on software process improvements and their effectiveness:

- Conduct CMM validity studies.
- Investigate and produce a process value method for higher maturity organizations.
- Demonstrate and validate a system for storing and disseminating information on software practices that have been implemented and are compliant with the CMM.
- Produce preliminary report and summary of the business value of CMM-based SPI in collaborative efforts with several customers.

1.2.6 Baseline Work Outputs

This section describes the baseline work outputs for the activity area related to CMM-based software process improvement. These outputs were approved for 1996 through the basic funds allocation process. Appendix A lists all baseline and proposed add-on work outputs.

SP-4B Community Involvement (1996-2000)

Several events during the year request support from the Process area. They include invitations to present at SPIN meetings, conferences other than strategic SEI events, contributions to journals, participation in related collaborative efforts with others, one-time tutorials, overviews and classes, and the hosting and running of the Process Program Advisory Board meeting twice a year.

The SEI supports the development of SPI programs throughout government and industry sectors. To introduce these programs broadly in both sectors, we must continue to support the formation and development of SEPGs and to educate and train members of fledgling SEPGs in skills and tactics that have proven to be successful in executing improvement programs.

Purpose

The purposes of the Process area's involvement in the software community are to

- Provide support for SPINs as requested, e.g., briefings, panelists, keynotes.
- Provide support for conferences/symposia in addition to the SEPG Conference and the SEI Symposium, e.g., Software Technology Conference, International Software Process Workshop, International Conference on Software Engineering, Defense Systems Management College.
- Support external collaborations, e.g., University of Southern California (USC), Feedback, Evolution, and Software Technology (FEAST), John Wiley's Journal on Software Process.
- Provide funding for support of the Process Program Advisory Board.
- Respond to three support requests, e.g., meetings with sponsors and other influential parties, comparative reports, and briefings.

Customers

Customers include sponsors and other influential parties; collaborators and partners of the SEI and the Process area; and organizers and attendees of SPINs, SEPGs, symposia, and conferences.

Collaborators

In-kind support exists for community involvement activities, e.g., John Wiley (publisher), and strategic partners (resident affiliates). Some symposia/conference speaking is supported by an honorarium or partial travel reimbursement.

Approach

- Identify key events to be supported.
- Receive and review "pop-up" requests from sponsors to support events, meetings, briefings, and reports.
- Receive and review external requests for speakers, panelists, etc.

SP-5B CMM Appraisal Framework (CAF) (1996-1998)

The CAF technical report describes the common requirements used in developing appraisal methods based on the CMM. The CAF provides a framework for rating the process maturity of an organization against the CMM. It includes a generic appraisal architecture for CMM-based appraisal methods and defines the requirements for developing CAF-compliant appraisal methods. The CAF supports the diagnosing phase of the IDEAL process improvement life cycle.

This effort maintains the baseline CAF, taking into account issues and concerns raised by internal and external users, and extends the CAF to support changed, evolved, or new maturity models. Thus, whenever any of these events warrant, a new version of the CAF will be published as a technical report. This task is a continuation of the 1995 task, "Appraisal Architecture and CRF Report."

Purpose

The purpose of this task is to foster CAF consistency between appraisal methods that are developed for different purposes. For example, the results of a CBA IPI and CBA SCE are used for different purposes and audiences, but should reflect similar outcomes, particularly in relation to the CMM. The CAF also fosters consistency between different appraisal teams.

Customers

Customers include sponsors of appraisals (both for process improvement and source/vendor selection), appraisal team leaders, and appraisal method developers.

Collaborators

Collaborators include the CMM Advisory Board, CMM-based SPI Advisory Group, and authorized lead assessors.

Approach

- Gather feedback from appraisal sponsors, lead assessors, advisory boards, and others on the results of CBA IPI and CBA SCE.
- Determine the impact on the CAF and appraisals of CMM version 2.0.
- Upgrade the CAF to reflect necessary impacts of above, as well as other CMMs, SPICE.
- Maintain and communicate the CAF as a core element of the SEI appraisal methods.

SP-6B CMM-Based Appraisal for Internal Process Improvement (CBA IPI) (1996-1998)

Appraisals are an integral part of the diagnosing phase of the IDEAL approach to software process improvement. CBA IPI is an appraisal method used by organizations as a tool to gain insight into their software development capability and to assess their software processes based on the CMM v1.1. This task is a continuation of the 1995 task, "Appraisal Methods."

CBA IPI method, associated training, and materials consist of

- CBA IPI Method Description (technical report).
- CBA Lead assessor training materials.
- CBA IPI assessment kits.

Additional planned work products are

- Extension of applications of CBA IPI (technical report).
- CBA IPI compatibility with SPICE (technical report).
- Appraisals for emerging SEI maturity models (plan).
- Linkage between diagnostics and software process metrics (technical report).
- Workshops, bulletin boards, etc.

Purpose
The CBA IPI method and associated training were completed in 1995. Until the method has stabilized and been transitioned into the community, it requires support and enhancement. The SEI needs to maintain a competence in the CBA IPI and participate with the community in its enhancement, acceptance, and transition. This is a vital part of transitioning the CMM into the state of the practice of software engineering.

The CBA IPI will also influence the emerging SPICE assessment and evaluation methods. It is integral with the new version of software capability evaluations (CBA SCE), and it is part of the basis for consistent appraisals across various CMM development activities, e.g., systems engineering.

Customers
Customers include sponsors of internal process improvements (e.g., industry and government organizations involved in SPI), lead assessors and assessment team members, other communities developing CMMs for their improvement programs (e.g., systems engineering, trusted systems, people, software acquisition), and developers and users of SPICE standards.

Collaborators
Collaborators include authorized lead assessors, CMM Advisory Board, CMM-Based SPI Advisory Group, SE-CMM Steering Group, People CMM Steering Group, and SPICE appraisal method developers.

Approach

- Continue to train and authorize lead assessors in CBA IPI.
- Participate in assessments, responding to issues of CMM interpretation, assessment fidelity, etc.
- Communicate with the assessment community to improve results, consistency, etc.
- Improve assessment methods based on experience.
- Evolve the method for CMM version 2.0, other CMMs, SPICE, etc.

SP-7B CMM Validity Studies (1996-1998)

The work proposed is the design and execution of a coordinated series of empirical studies that will generate evidence to assess the accuracy and value of the fundamental ideas and claims about the CMM. Outputs will include presentations at the SEPG Conference and SEI Symposium, as well as technical reports.

Purpose

The CMM is the intellectual core of most of the work in the Process area. The content and the approach of the CMM, however, is regularly debated in the software engineering literature and professional meetings, and our customers frequently ask for evidence that the CMM does, in fact, embody a good approach to software process improvement.

Customers

There is tremendous interest across the entire SPI community in the issues addressed by these studies. There are also internal SEI stakeholders who are interested in the results, whether they confirm, qualify, or disconfirm claims about the CMM, as one input to be used in the process of continuous improvement.

Collaborators

We have numerous collaborators among government organizations, defense contractors, and commercial companies. The list includes such organizations as Citicorp; Motorola; U.S. Air Force, Tinker AFB Air Logistics Center; Hughes Aircraft Co.; and Loral Federal Systems.

Approach

In order to focus and prioritize our efforts, we have used extensive input from customers, as well as suggestions from advisory boards and internal SEI stakeholders, to identify a set of critical assertions about the CMM and about the effect that CMM-based software process improvement is presumed to have on software-dependent organizations. These assertions are organized in five categories:

- Predictability and performance by maturity level.
- Key process areas.

- Moving up the maturity scale: time, cost, and progress reporting.
- CMM usability and implementation issues.
- Factors influencing success of SPI efforts.

To date we have published an initial study (CMU/SEI-94-TR-13, *Benefits of CMM Based Software Process Improvement: Initial Results*) of the improvements in quality, cycle time, productivity, cost, and business value in 13 organizations. We have also conducted an appraisal follow-up survey and a study of appraisal findings, both of which were presented at the 1995 SEPG Conference and which will be documented in 1995 technical reports.

We have only addressed a few of the assertions in any depth thus far, and we have not fully addressed any of them. The work will continue to address the ideas and claims about the CMM that our customers consider to be the most important.

SP-8B CMM-Based Appraisal for Software Capability Evaluations (CBA SCE) (1996-1998)

Capability evaluations are a diagnostic tool that form part of the suite of products associated with a maturity model. The evaluation tool provides an acquirer with the answer to the question, "How is their process capability?"

This effort maintains the baseline method via an SEI-chaired configuration control board; develops extensions of the method to support changed, evolved, or new maturity models; pilots applications of the method in new technology domains or sectors of the economy; and supplements the method in support of the underlying maturity models. Thus, whenever any of these events occurs and regardless of the source of funds that caused the event, the complete product suite including method description, instructional materials, technical reports, and implementation guidance must be updated and promulgated. This task is a continuation of the 1995 task, "Appraisal Methods."

Purpose

The SEI is committed to continue to maintain and continuously improve the capability evaluation method in support of the CRADA partners to whom we have licensed the evaluation method. Royalty payments are anticipated to be sufficient to cover only direct support of CRADA partner efforts, including quality assurance, handling referrals to the partners, interpreting method guidance, etc.

This effort is interdependent with CMM version 2.0, the CAF, the CBA IPI, and other Process area efforts. The SEI needs to maintain a competence in the area of SCE.

Customers

Customers include government source selection authorities, industry organizations letting software subcontracts, including outsourcing information technology support, SCE lead evaluators and team members, and CRADA partners (licensees) providing SCE training.

Approach

- Continue to upgrade the method to support a lead evaluator community and evaluation team members.
- Maintain lead evaluator guidelines, catalog, authorizations, etc.
- Participate with the community to gain feedback on the method.
- Communicate with the sponsor and evaluator community to improve results, consistency, etc.
- Evolve the method for CMM version 2.0, and other CMMs if appropriate.

SP-9B Method for Defining Software Processes (1996)

This small effort is to complete the work begun in 1995 to publish version 1.0 of the Method for Defining Software Processes technical report. This output is a recommended "how to" method for defining software processes (including designing and evolving them). The method will support and operationalize the acting phase of the IDEAL model. It will also support continuous improvement of defined processes. This task is a continuation of the 1995 task, "Software Process Definition Guidelines."

Purpose

This method addresses a vital practitioner question arising during the acting phase of the IDEAL model: "How does one develop and evolve process models and definitions?" Accomplishing this is key to instituting continuous process improvement, satisfying the Organizational Process Definition key practice area (KPA) of the CMM, and achieving related organizational goals.

Customers

Customers include process definition and improvement practitioners, e.g., SEPGs and process action teams (PATs).

Collaborators

Collaborators include reviewers from government and industry involved in SPI.

Approach

- Complete development and documentation of the prototype method as part of the Software Process Definition Guidelines series.
- Produce a draft technical report documenting the prototype method and submit for editorial review.

Pilot testing of this version (1.0) of the method, and development of a subsequent version, is proposed as SP-5A, below.

SP-10B Product Specification for Process Guides (1996)

The output of this effort is the completion of a set of recommendations for content, structure, and reviews of process guides. Process guides are documentation providing guidance to the intended performers of a process. They are embodiments of the results of process definition work. This task represents completion of the 1995 task, "Software Process Definition Guidelines." It builds upon the work on information content for enactable process representations.

Purpose

These recommendations address a vital practitioner question arising during the acting phase of the IDEAL model: "What does a process definition look like, and what should it contain?" Developing process definitions that can be put into practice is critical to instituting continuous process improvement, satisfying the Organizational Process Definition KPA of the CMM, and achieving related organizational goals.

Customers

Customers include process definition and improvement practitioners (e.g., SEPGs and PATs), and software engineering practitioners indirectly as users of the resulting process guides.

Collaborators

Collaborators include reviewers from government and industry involved in SPI.

Approach

- Develop a practitioner handbook documenting these recommended product specifications as part of the Software Process Definition Guidelines series (1995).
- Submit a draft handbook documenting these results for peer review (1995). The handbook will contain an example process guide, recommended outline and structure for process guides, recommended content, and a review procedure with extensive checklists.

Work funded by 1996 basic funds in this baseline activity will complete the review, revision, and publication of the handbook.

1.2.7 Proposed Add-On Work Outputs

This section describes the proposed add-on work outputs for the activity area related to CMM-based software process improvement. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

SP-4A Advanced Process Definition Reports (1996-1997)

This output addresses advanced topics in process definition and modeling for organizations advancing to CMM level 3 and beyond, and will include two technical reports, case study examples, an annual focused workshop and report, and several presentations and publications on the subject. The two reports will focus on guidance for developing a software process architecture for an organization, and composing and tailoring project-specific software processes from a populated architecture and life cycles. The motivating vision for this architecture and tailoring work is to support the definition of reusable process components, which can be composed and tailored to meet the needs of individual projects and products. In this context, a software process architecture provides a framework for those reusable process components. The case studies will provide samples of processes that have been implemented in practice, are consistent with a relatively high level of CMM maturity, and illustrate experiences of improving a process over time. The focused workshops will address topics such as process representation experiences, evolution of defined processes, process architectures, and process tailoring. The results of these workshops will be documented in a summary report, intended for widespread dissemination.

Purpose

Very few organizations have a clear understanding of what is required to develop process architectures and project-level tailoring guidelines, and to measure, analyze, and evolve processes at CMM levels 3, 4, and 5. Other than the CMM, the SEI has little additional information or examples to assist organizations in understanding and implementing these rather complex recommendations. The purpose of this output is to lead and catalyze the software community with guidance on process architectures and tailoring approaches. This guidance is important as more software engineering practitioners advance to CMM level 3 and beyond.

Customers

The primary customers of this output are software process change agents within organizations who are working to achieve CMM level 3 or higher. In addition, this work will link well with the CMM v2.0 revision, since the results will lead to a better understanding of at least four of the KPAs within the CMM (Organization Process Definition and Integrated Software Management at the Defined level, Quantitative Process Management at the Managed level, and Process Change Management at the Optimizing level).

Collaborators

All of the identified outputs will require collaboration with leading-edge software process improvement practitioners and applied researchers with expertise in the specific topics outlined above. These people will play a key role in the workshops and case study examples, and in validating draft reports. Potential collaborators include personnel from the USC Center for Software Engineering, McGill University, Boeing Computer Services, and TRW.

Approach

- Conduct site visits at selected high-maturity or process innovative organizations (1996-1997).
- Identify, collect, sanitize, and publish case study examples (1996-1997).
- Hold focused workshops, and publish workshop reports (1996-1997).
- Develop and publish architecture report and tailoring report (1996-1997).
- Support pilot testing and gather feedback (1997).

SP-5A Method for Defining Software Processes, Version 2.0 (1996-1997)

This work is follow-on to output SP-9B. A technical report, tentatively entitled "Software Process Definition Guide: Method for Defining, Designing, and Evolving Software Processes," will contain a recommended "how to" method for defining, designing, and evolving software processes. Whereas task SP-9B is producing version 1.0 of the method and report, this proposed task will entail pilot testing of that prototype, and subsequent elaboration, refinement, and revision to produce a new version.

Purpose

The purpose of this output is to enable organizations to more effectively and expeditiously define and continuously improve their software processes, thus assisting these organizations in achieving higher levels of process maturity.

The technical report will provide detailed descriptions of the steps comprising the acting phase of the IDEAL model for software process improvement. The method defined will

- Support CMM-based improvement strategies.
- Promote and employ process management and quality management principles.
- Lead to development of processes that can be continuously improved.
- Be systematic and well-defined.
- Be manageable.
- Be consistent with the "product specifications" and examples being produced for enactable software process representations (e.g., process guides described in SP-10B).

Customers

The primary customers of this output are any process change agents in organizations that are implementing an IDEAL method for continuous process improvement. The output will be targeted at early adopters, and early and late majority practitioners. Other customers include those within the SEI working on enhancing the IDEAL model.

Collaborators

Collaborators will include pilot test sites, e.g., resident affiliates' organizations, customer support clients, and reviewers involved in SPI.

Approach

- Pilot test the prototype method and gather feedback (1996-1997).
- Revise the method based upon pilot tests (1996-1997).
- Elaborate and refine the method (1996-1997).
- Publish a new version (1996-1997).
- Develop practitioner-oriented presentation materials, e.g., a handbook and/or training (1996-1997).

SP-6A Measuring the Impacts of the Personal Software Process (PSP) (1996-1997)

The work proposed here is divided into three successive phases, starting 1Q96 and ending in 1997. During the first phase, the work will produce a technical report containing a rigorous evaluation of the PSP training. In Phase 2 we will develop and pilot a handbook to assist PSP instructors with the gathering of data from their students. As more PSP instructors are trained and conducting classes, efficient and standardized mechanisms for recording and reporting data will be needed. In Phase 3 (1997) we will study the impact of teams of PSP-trained software engineers on project performance and software process improvement.

Purpose

The goal of the personal software process is to make software engineers aware of the processes they use to do their work and how they perform those processes. Software engineers set personal goals, define the methods to be used, measure their work, analyze the results, and adjust their methods to meet their goals. It is a strategy for professional self development and enhanced productivity.

For organizations considering investing in their personnel via PSP training, the benefits and limitations of this new methodology are of vital interest. Investment by organizations in the PSP and continued transition of the PSP by the SEI should be based upon rigorous evaluation of its impacts at both the individual and organizational levels. This effort proposes to conduct those evaluations.

Customers

Customers of this work are those organizations deciding whether or not to adopt the PSP as part of their training for software engineers. This currently includes software development and maintenance organizations as well as universities training software engineers. The potential market for the PSP will likely expand as evidence of its effectiveness increases.

Collaborators

This work requires collaboration from PSP instructors, PSP students, and software organizations with software engineers who have been trained in the PSP. This work requires each of these groups to play a role in generating data and sending it to the SEI for analysis. Currently, PSP instructors send data to the SEI. However, some problems have been noted with the existing process. Follow-up procedures for obtaining data from students and organizations after training have yet to be developed.

Approach

The cornerstone of this work will be the analysis of actual data from PSP students and later from their software projects and organizations. While some analyses have been done suggesting short-term benefits, these results are based upon a limited amount of data and incomplete analyses. To collect the necessary data to answer the questions about the impacts of PSP, we will establish consistent data collection routines and develop tools to help students and instructors record the data. Regarding the benefits to organizations of having PSP trained software engineers, we intend to work with software organizations to compare the on-the-job performance of PSP engineers with that of other similar engineers. Additionally, we will track the PSP engineers' performance and use of PSP methods over time.

SP-7A IDEAL Model Products (1996-1997)

The primary product of this effort will be an integration of products which implement the IDEAL model, the SEI's recommended approach to implementing SPI. A summary technical report will contain an up-to-date description of the model's basic process sequence, integrating process management principles, appropriate implementation strategies, organization structures, field-usable forms and templates, and job task skills and knowledge proven to be effective in implementing SPI programs. In addition, early analysis work will yield an effectiveness report describing how well the SEI is satisfying the knowledge and skill needs of those responsible for implementing SPI programs, and what the SEI can do to enhance its clients' capabilities.

Purpose

The purpose of these products is to provide the software community with a viable and proven model for implementing SPI, assist organizations in understanding and preparing for what is required to make improvement happen, and understand how existing SEI products and services support their improvement objectives. The SEI needs these products to better transition SPI methodologies and techniques into industry, and help guarantee seamless products and services for organizations that desire a comprehensive SPI package.

Customers

The primary customers are champions, sponsors, and change agents in organizations presently implementing or considering initiating a SPI program, particularly those who desire to use SEI process improvement products, training courses, and materials. In addition, the SEI will benefit from this work through alignment of the various segments of the SEI involved in customer SPI activities, leading to improvement of our overall effectiveness in developing, maintaining, and transitioning SPI products to our customers.

Collaborators

This work requires the collaborative assistance of champions, sponsors, and change agents in organizations that have been successful in implementing SPI programs (defined as those who have achieved higher levels of CMM maturity). These collaborators will play a key role in identifying key knowledge and skills that led to their successful implementation of SPI. In addition, the SEI will need volunteer organizations to use field-usable forms, templates, and materials as part of IDEAL model-based improvement field tests.

Approach

- Collect, document, and publish existing knowledge and implementations of the IDEAL model (1996).
- Produce an effectiveness report of the SEI's existing SPI products and services (annually).
- Develop and pilot test IDEAL model-based forms, templates, and materials (1997).
- Consolidate lessons learned and publish a technical report on the IDEAL model (1997).

SP-8A Applying Lessons Learned for Developing Software Process Improvement Teams (1996)

Often teams are formed by software organizations trying to improve their software process. These teams spend large amounts of time organizing and operationalizing themselves before they become productive. This time can often be drastically reduced, and the team made more effective faster, if the process of forming teams is planned and managed. In a resource-constrained environment, this can often determine success or failure for the improvement effort. This work will develop a set of interventions to guide the experienced facilitator in helping client organizations understand how to design, staff, and support teams to effect software process improvement.

Purpose

In working with many organizations, we have found that the ad hoc methods used by these organizations to form and deploy action teams have resulted in much time and energy being consumed by the team to define itself and to reach some consensus on how it will approach the assigned problem. We have informally developed a set of interventions that make it easier for teams to get started. Using these methods, we have found that teams become productive much more quickly and efficiently.

Customers
This work is applicable to organizations undertaking software process improvement. It is focused on management groups that authorize and sponsor action teams as part of their strategy to improve.

Collaborators
We will seek as collaborators those organizations that are beginning to form process improvement action teams. We will ask these organizations to provide funding to support pilot presentations of the workshop materials.

Approach
This effort will develop a workshop, a set of materials, and follow-up interventions designed to help software organizations charter, form, and deploy software process improvement action teams in an effective and efficient manner.The SEI has extensive experience with client organizations striving for improved software capability. In working with the community and to address specific client needs, the SEI has prototyped conference tutorials and materials.

Using this experience and existing materials as leverage, an educational design will be developed and reviewed with clients and the Education and Training Review Board (ETRB). Formalized materials will be developed, piloted, and evaluated. Final materials will be reviewed by the ETRB.

SP-9A Addressing Software Process Transition Barriers (1996-1998)
The focus of this effort is to address the major factors that inhibit the broad adoption of proven process methods into software practice. The transition inhibitors to be addressed are

- Inadequate management awareness of the benefits of using mature software processes.
- Insufficient conviction by managers and engineers that the use of process methods can substantially improve organizational and personal performance.
- The long lead time required to motivate and train engineers and managers to follow process methods.

The work, along with other SEI activities, is aimed at addressing these factors both in the long and short term.

Purpose
The purpose of this work is to provide support to government and industry organizations in identifying transition inhibitors to process improvement and addressing those inhibitors within the organization.

Customers

Customers include decision makers within government and industry organizations who are interested in improving their business results, in addition to managers and engineers who are interested in improving their organizations and their personal effectiveness, efficiency, and quality management.

Collaborators

This work requires support and collaboration from organizations throughout the software industry: the Institute of Electrical and Electronic Engineers (IEEE) Computer Society (which partners with the SEI in increasing the awareness of process benefits); maturing organizations (such as Motorola's software laboratory in Bangalore, India) which provide examples of key process improvement activities that have overcome inhibitors to change; maturing organizations (such as Union Switch and Signal) which provide the environment for introducing new techniques and analyzing their effects; universities and colleges (such as Embry-Riddle Aeronautical University) which provide transition support to undergraduate and graduate students, including data and feedback on the effects of new techniques for process improvement; and publishing companies (such as Addison-Wesley) which provide additional avenues of transition support.

Approach

- Increase awareness of process benefits through such means as recognizing outstanding software process improvement by helping to select the recipient of the IEEE Software Process Achievement Award, publishing and presenting on process improvement results, and maintaining an awareness of key process improvement activities.
- Convince managers and engineers to emphasize process improvement through introduction of the PSP, work with organizations to install and measure the costs and benefits of PSP, and convince management to support PSP introduction in their organizations.
- Accelerate process improvement by introducing the concepts and demonstrating the benefits of measured and managed processes to engineers during their formal education periods. This includes integrating PSP into undergraduate and graduate education programs and establishing university software development centers where students get realistic experience that is related to their course topics. For example, provide contract software development and support to small businesses in the immediate vicinity of the university.

SP-10A Process Value Method (PVM) for Higher Maturity Organizations (1996-1997)

The PVM will guide decision makers through a series of steps to define and structure the decisions that they need to make, gather relevant information, estimate values for missing information, analyze the available information, and interpret the results of these analyses with respect to explicit decision criteria.The PVM will provide an estimate of a proposed improvement's performance, e.g., in terms of business value, software engineering efficiency, or software product quality. The method will take into account the existing organizational environment and accommodate varying magnitudes of process change, degrees of implementation across the organization, and quality of information inputs. The method will allow decision makers to compare various alternative changes against each other and with the status quo in terms of relevant decision criteria

This is a proposed full-scale follow-on to the PVM feasibility study undertaken in 1995. Early indications from the feasibility study are that simulation modeling (e.g., through process modeling or system dynamics modeling) will be a promising approach. This will allow analysis of process changes of varying magnitudes, from small changes up to the implementation of a full CMM KPA. Higher maturity organizations are most likely to benefit from the PVM, because they are more likely to have the necessary process discipline and baseline data needed to calibrate the model. However, the types of process changes to which the PVM is applicable are not limited to higher maturity KPAs, nor even to the CMM.

Purpose
Making informed decisions about investments in software capability is difficult at best. Even when experience reports and lessons learned are available to assist in the process, it is difficult to translate these into a decision-making process that uses them to best advantage. The purpose of PVM is to capture knowledge and experience in a practical, usable form, which will assist higher maturity organizations in making rational, effective decisions under these uncertain conditions.

Customers
The PVM is designed for the increasing number of software organizations that have achieved higher maturity levels (3 and above). The method will support decision makers who are leading software process improvement initiatives, like project managers or SEPGs.

Collaborators
This method will require collaboration by higher maturity organizations, such as those with whom we currently have data-sharing relationships based on prior collaborations.

Approach
- Develop a method (1996).
- Pilot test the method (1996).
- Package and disseminate the method (1997).

The 1995 feasibility study will develop a detailed plan for pursuing this work in 1996-97.

SP-11A Applied Process Research Collaborations (1996-1997)

This work will foster and influence external research relevant to software process issues. The focus is on applied research, i.e., addressing practical issues that are emerging today in leading-edge organizations, and will be facing the broad community within just a few years. This effort will entail

- SEI participation and collaboration in selected promising external research efforts.
- SEI sponsorship of a symposium on applied process research, in conjunction with the 1996 SEPG Conference.

The strategy here is to gain leverage from SEI resources by helping to focus others on emerging, real-world problems that need to be solved effectively, in order to help our customers and the broad software engineering community. The SEI is in a unique position to perform this role, due to the respect we enjoy in both the practitioner and research communities, and our understanding of both real-world needs and research advances. As results and solutions emerge, they will become targets for future SEI technology transition efforts.

Purpose

The effort is intended to

- Bring the SEI's familiarity with real-world process problems and the needs of the practitioner community to bear within the research community.
- Foster development and growth of an applied process research community.
- Intellectually contribute to promising external/collaborative research efforts.
- Increase the depth of SEI staff's familiarity and insight into very promising external applied process research efforts, leading to easier transition of these in the future.
- Increase the SEI's and other researchers' understanding of leading-edge practitioners' views of emerging process issues.
- Encourage research and development work that is directly relevant and applicable to forthcoming major community needs.
- Encourage relationships and alliances between researchers and leading-edge practitioners, who otherwise might not connect with each other.

Customers

The direct customers of this effort are (1) applied process researchers (e.g., at the University of Southern California, Imperial College, AT&T Bell Labs, McGill University, SEI, and STARS [Software Technology for Adaptable, Reliable Systems] contractors/participants such as Loral and Boeing) and (2) leading-edge government and industry organizations (e.g., Boeing Computer Services, TRW).

In addition, because this work is intended to increase the quantity and applicability of applied research conducted in the Process area, it will provide a greater pipeline of technologies for the SEI to transition to the software engineering community. This makes both the SEI and the community secondary customers of this activity.

Collaborators

Collaborators in this effort are the Advanced Research Projects Agency (ARPA) and key applied process researchers. These are likely to include the USC Center for Software Engineering, FEAST principals, the Committee on Software Process of the IEEE Computer Society Technical Council on Software Engineering, etc.

Approach

The approach to this work comprises two major thrusts. The first is to organize and conduct a symposium, approximately one day in duration, attracting both researchers (academic and industrial) and leading-edge practitioners. This Symposium on Applied Process Research will provide a common ground and forum for discussing emerging process issues and challenges. The symposium is proposed to be held in conjunction with the 1996 SEPG Conference, which offers the advantages of encouraging practitioner participation and providing the logistical infrastructure of an existing major conference.

The second thrust is to engage in technical collaborations on promising external research endeavors. SEI personnel are frequently sought out as participants and collaborators because of our expertise in the software process arena. Individual collaborations funded under this activity will be limited to relatively low effort, whereas larger effort collaborations should be "spun-off" as separate funded undertakings. These collaborations will primarily entail participation in technical meetings. Identifiable outputs would include meeting co-organization, meeting participation, presentations/briefings, and position papers.

SP-12A Information on CMM-Compliant Practices (1996-1998)

The outputs are information on software practices that are mapped to CMM maturity levels and key process areas. The practices will be implemented examples that are compliant with the CMM.

Purpose

As the number of organizations adopting the CMM as the foundation for their software process improvement efforts has increased, so has the number of requests for the SEI to provide examples or pointers to CMM-compliant practices and processes. The intent of this task is to facilitate the sharing of example processes (through a publicly available and accessible query system) that meet specific CMM attributes. The output of this task is a process example library (PEL). A PEL is similar to a process asset library (PAL) except artifacts contained in the PEL will be processes that have been implemented. This is in contrast with a PAL, which typically contains generic artifacts that need tailoring and function definition.

Customers

The primary customers of this information are all communities involved in CMM-based SPI. This information will especially benefit organizations beginning a SPI program and those that have been assessed at a specific maturity level and are striving for higher levels. Also, assessors and evaluators who must rate processes against the CMM will find the information useful for training purposes as well as during an appraisal or evaluation for doing online comparisons with processes that have been found to be compliant to the process being appraised or evaluated at hand.

Collaborators

This effort will require the assistance of higher maturity organizations to provide the process and practice descriptions and information. It will also require enabling technology (risk repository) developed in the Risk area (SP-15A, RM-2B).

Approach

- Identify and document customer requirements (1996).
- Using the enabling technology, design and prototype the system for storing and disseminating the information (1996).
- Define the criteria and process for including and storing information in the system (1996).
- Implement the system and make available for broad public use (1997).
- Continually populate the system and maintain the information (e.g., modify the system relative to updates to the CMM) (1997-1998).

SP-13A Software Process Improvement for Small Organizations (1996-1997)

The primary output is a methodology enacted through a series of several organization interventions (workshops), focusing on transitioning the knowledge and skills required to apply SPI in small organizational settings. This SPI methodology is based on the IDEAL model and is structured around four interventions to take place at the organization's site, each in a three - five day period. Tentatively, these workshops are (1) Sponsorship Building and Contracting Workshop, (2) Diagnosing Workshop, (3) Analysis and Action Planning Workshop, and (4) Improvement Workshop.

Purpose

A growing number of very small organizations (staff less than 50 members) have indicated that they need assistance in improving their organizations' software processes; however, these organizations cannot afford to plan and implement software process improvement as presently prescribed by the SEI through the IDEAL model and existing products and services. One-time tailoring of these products is not the answer since they were designed to support primarily large organizations. The purpose of this output is to provide small organizations a cost effective methodology for building sponsorship, identifying a few process areas for improvement, developing an improvement strategy, and taking action to improve the selected processes.

Customers

This output is directly targeted at organizations desiring to initiate a SPI program but whose entire staff size is 50 members or fewer.

Collaborators

Due to the limited understanding of the needs and issues of small organizations in implementing an effective SPI program, this methodology's development model will be based on a prototype and test approach. Such a strategy requires access to several small organizations willing to collaborate over the next several years on the development of this methodology. These collaborators will provide funding support, site staff and resources for improvement efforts, data on SPI investments, and access to present and future process and performance data.

Approach

- Collect, analyze, and document small organization requirements for a SPI program (1996).
- Construct a suitable SPI program based on the IDEAL model of improvement (1996).
- Identify appropriate intervention workshops and establish an appropriate time frame (1996).
- Build the workshops and conduct pilot tests at selected organizations (1996-1997).
- Collect and analyze pilot-site SPI investments, in addition to process and performance changes (1996-1997).

1.2.8 Related Customer Activities

Clients will integrate the IDEAL model into new and existing software process improvement efforts. They will employ qualified lead appraisers to conduct diagnoses and associate business objectives with the priorities they develop for software process improvement. They will use defined methods for defining software processes and measures, predict them, improve them, and spread their use. Clients will document the lessons learned from these actions and report their benefits.

Efforts in transitioning continuous software process improvement to clients will involve training, tailoring of products to specific client needs, support during initial process definition and measurement definition stages, and transition of the improvement capabilities to the client. It also involves aiding the clients' start-up in identifying the business value that they are receiving from continuous SPI.

Clients anticipated in 1996 include the Federal Aviation Administration (FAA), the U. S. Air Force, XEROX, Loral, Citicorp, Mobil, U.S. Treasury, and the software engineering community at large.

1.3 Software Engineering Measurement

1.3.1 Problem Statement

There are two thrusts that are addressed in the Software Engineering Measurement activity area:

1. Development of software measurement technology, processes, and practices; and development of the methods and support for integrating this technology into management and technical software processes.
2. Creation of an information resource that enables organizations to learn from the experiences of others and that provides reference points for comparison.

Measurement forms the basis of engineering and management decision making in most disciplines. Yet few software organizations use measurement and those that do tend to use measurement data in a limited way, primarily for cost estimation. Most do not even know their own error rates. Increased and improved measurement is essential to improving the practice of software engineering and management. Better measurement of product quality will help ensure the delivery of systems that more effectively satisfy user requirements. Better measurement of productivity and costs will facilitate more efficient planning of system development and maintenance. Currently, too few organizations use quantitative methods and information to manage software projects effectively and to measure improvements in their software products or processes. Many software engineering organizations also do not have credible data on how software engineering improvements and innovations contribute to improved software engineering and business. Such data are essential to make effective decisions and to develop effective improvement strategies that contribute to business goals.

The software community needs both a clearer picture of software development capabilities and a quantitative basis from which to measure overall improvement. Customers often call with questions regarding national performance standards on software engineering. They often request data to include in proposals, data upon which to base estimates, information on how their quality and productivity compare with national averages and distributions, and information on how to evaluate the performance of software suppliers.

There are no standard methods for measuring and reporting data for software products and processes. Comparisons across domains or across the nation are impossible. A U.S. company cannot know if its software quality is better or worse than the national average because no such national information is available. The quantitative standards available in other disciplines are missing, and there is seldom a clear understanding of how a measure on one software project can be compared or converted to a similar measure on another. With quantitative information, national goals can be set to help keep the software community competitive and focused on continuous improvement of products and processes.

Also, a recent report from the SEI Blue Ribbon Panel (dated February 1994) recommended "that the SEI establish a central repository of software measurement information to help depict the current state of the practice." The report further elaborated on this idea by stating that the SEI could manage a critical mass of data that offers a comprehensive software picture, and that the SEI should be proactive in creating a central information repository and in sharing the information with the software community.

1.3.2 Customers

Customers for our products and services include those using or wanting to use data to manage and improve their software organizations and projects. Representative groups include

- Industry and government software engineering process groups (SEPGs).
- Industry project managers, DoD program managers.
- Business and product managers.
- Process action teams in industrial, military, and federal agency organizations.
- Software process appraisers.
- Practitioners engaged in developing, maintaining, or acquiring software systems and also implementing commercial-off-the-shelf integration activities.

Software process improvement champions, sponsors, and agents for change are also customers as well as those who provide SPI-based products and services. Example organizations may include education, training, consulting, and standards organizations.

1.3.3 Rationale

The need for data definition, collection, analysis, and use of the results is recognized as crucial in software development. However, many organizations want and need assistance in getting started; understanding measurement techniques and interpreting quantitative information; defining measurement processes (Figure 1-7), communicating about and believing in measured results; and using measures to estimate, plan, and control projects results.

As well as learning from their own experiences, organizations seek to learn from and compare themselves to other organizations. Therefore, products are needed to support individual organizations in their efforts to improve their management and technical capabilities, and information is needed so that organizations can seek ideas for benchmarking and improvement.

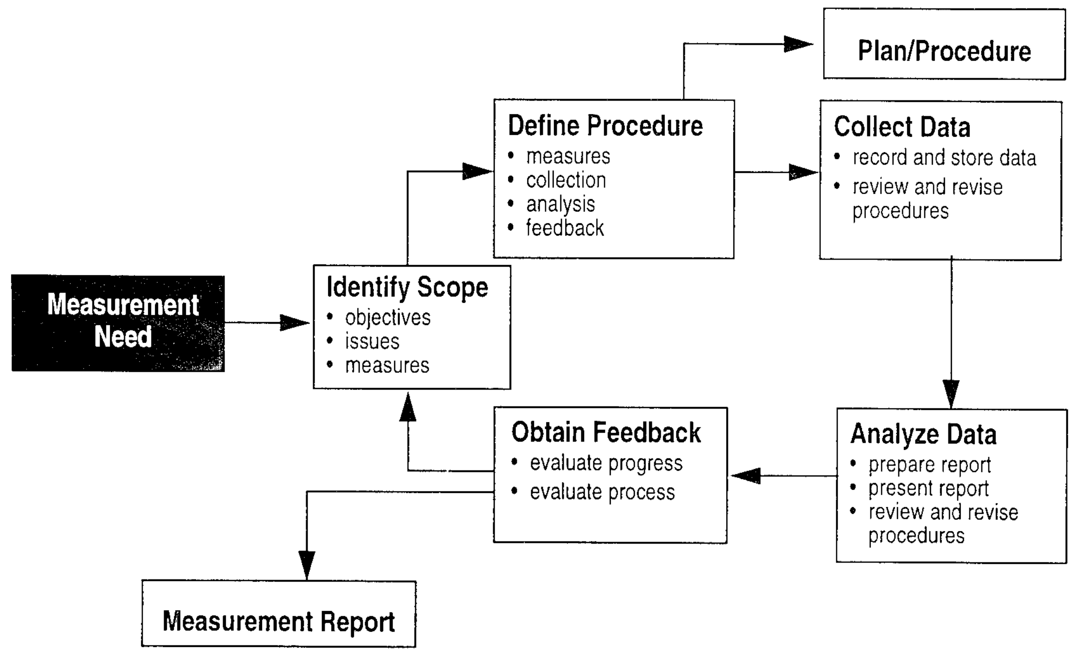


Figure 1-7: Defining the Software Measurement Process

The SEI is positioned to support organizations in using quantitative information and methods effectively in managing software projects, measuring results of software products, measuring improvements in software processes, and benchmarking industry practices. The SEI can serve as a national resource for software engineering information (Figure 1-8) and software measurement practices, processes, and methods (as shown in Figure 1-9). Data from industry and government organizations, other collaborators, and published reports can be maintained. A primary benefit to our customers will be the ability to find, through a single source, the most comprehensive compilation of software engineering data on project and process performance, process improvement, risk mitigation, and improvement tactics.

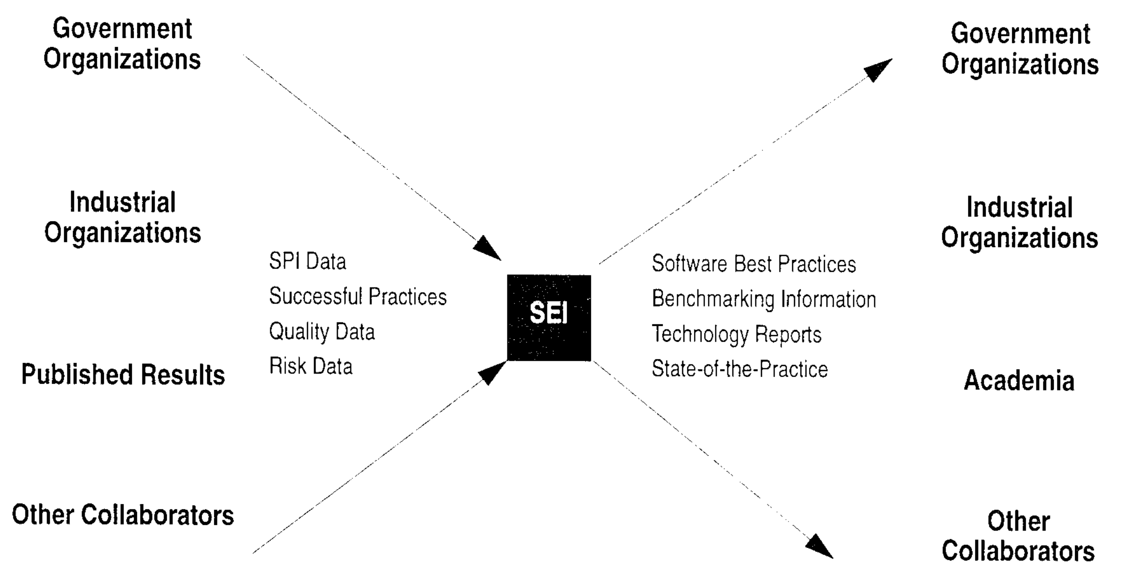


Figure 1-8: The SEI Serves as a National Resource for Disseminating Information to the Community

The SEI is well positioned to work with the broad software engineering community to synthesize, package, and transition best practices and research findings to help organizations install and sustain effective software technologies. We have an established track record for gathering confidential data from software organizations. As shown in Figure 1-8, we would seek to gather data on software process improvement, quality, and productivity, as well as information on successful software practices. We would then analyze and disseminate information back to the community. This information would include, for example, software best practices, benchmarking information, technology reports, and state-of-the-practice reports. Building on the SEI's success as an impartial stakeholder for government and industry, it makes sense for the SEI to maintain and disseminate data and information on the state of software engineering practice.

1.3.4 Benefits

The application of measurement processes, practices, and methods is needed to support advancement in process capability. The SEI is developing products and services to implement the key practices of the CMM for Software. There is a recognition of the need to begin a measurement program at levels 1 and 2 (i.e., not to wait until level 4), and to build a comprehensive measurement program as an organization matures.

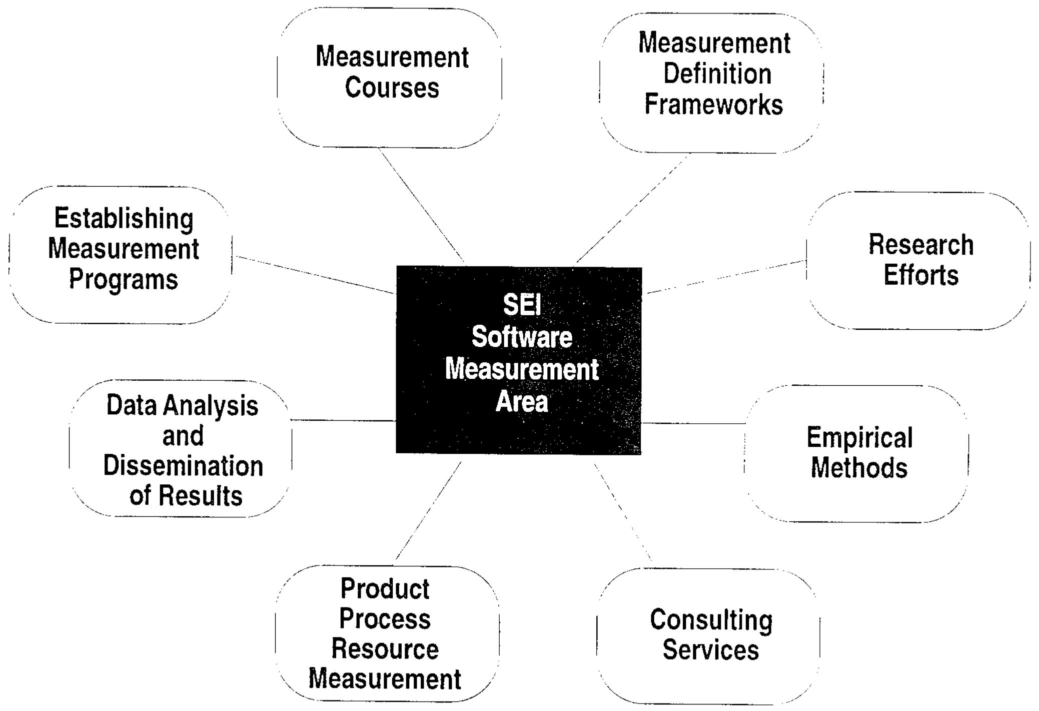

Measurement Courses
Measurement Definition Frameworks
Establishing Measurement Programs
Research Efforts
SEI Software Measurement Area
Data Analysis and Dissemination of Results
Empirical Methods
Product Process Resource Measurement
Consulting Services

Figure 1-9: The SEI Serves as a National Resource for Software Measurement Practices

Measurement is an indicator of progress in process improvement programs, and it can be an indicator of bottlenecks and inhibitors to progress as well. Measurement and feedback on the software process and product also provide significant potential for insight into areas needing improvement. Mature software processes and products are managed, defined, controlled, measured, and focused on orderly improvement.

Monitoring and reporting process improvement in the software community and making available empirical methods to analyze and support organizational software process improvement are also important contributions. We will ensure the development of well-founded techniques for measuring organization performance and change.

Our efforts are also focused on improving the ability of organizations to meet their business objectives. Assessing an organization's current process establishes a baseline measure of its process maturity and findings to address during an improvement effort. By improving its process maturity, a software organization will improve its business performance. To assist software organizations' efforts to improve, we will work with both those developing innovations to improve the state of the practice as well as software organizations adopting new innovations promising to yield improved performance. These innovations should be aligned with the business objectives for both the software innovation developers and their customers, the software engineering community (see Figure 1-10).

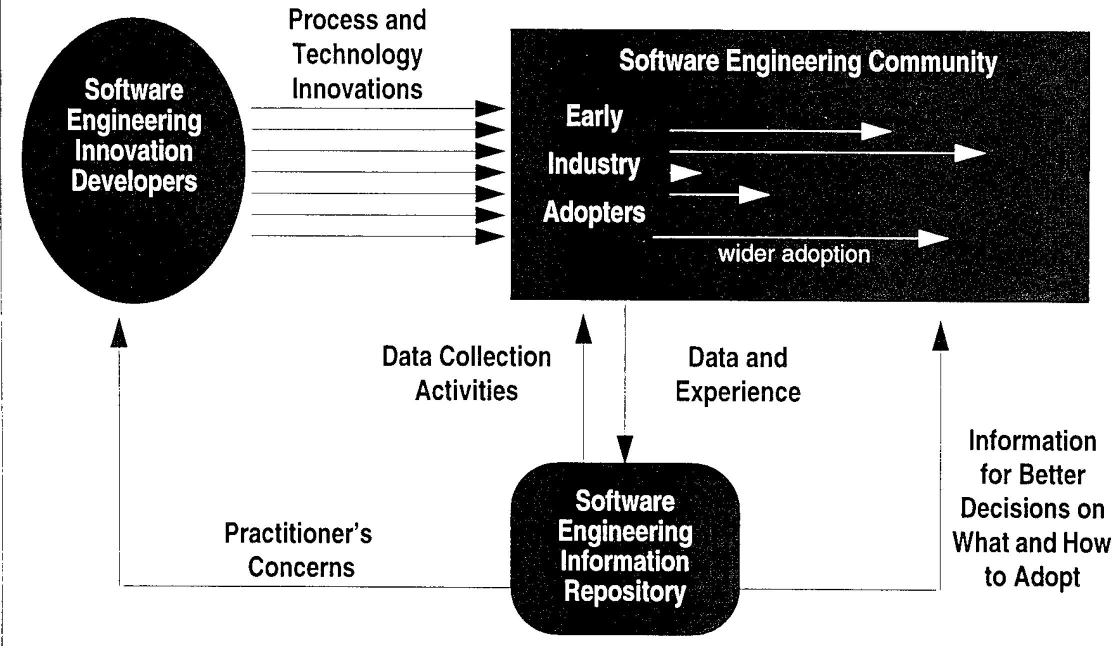

Software Engineering Innovation Developers
Process and Technology Innovations
Software Engineering Community
Early Industry Adopters
wider adoption
Data Collection Activities
Data and Experience
Information for Better Decisions on What and How to Adopt
Practitioner's Concerns
Software Engineering Information Repository

Figure 1-10: Role of the Software Engineering Information Repository in the Software Community

Better data on the effects of software engineering innovations will aid decision making for software organizations investing in software engineering innovations. These data can also be used to inform software engineering developers about the practical concerns of software organizations. Monitoring improvement in the community will help sustain the commitment to software engineering improvement and continue to build and support the infrastructure necessary to sustain and broaden improvement efforts.

For software engineering measurement, we see the trends shown below.

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Measurement definition frameworks	• Measurement definition frameworks developed for software size, effort, schedule, quality.	• Measurement definition frameworks completed for L2 and L3 KPAs.	• Measurement definition frameworks for L4 and L5 KPAs	• Number of measures for which frameworks are developed • Number of organizations able to sustain effective software measurement programs
Best practices	• Technical reports • Software measurement course • Consulting efforts	• Software Measurement Handbook developed. • Statistical practices course	• Organizations possess knowledge and skills for building comprehensive measurement programs.	• Number of organizations reporting increased process maturity • Number of organizations able to sustain software measurement programs • Number of people trained in software measurement practices
Measurement and the CMM	• Measures and the CMM TR	• Measurement practices integrated in CMM V2	• Integrated into standards efforts	• Number of customers reporting increased process maturity
Software engineering information repository	• Anecdotal SPI information collected • Process assessment data	• State of the practice of SPI technical reports • SEI is a national resource for software data. • Performance data • Successful software practices	• Expand database to house additional software data on specific tools and technologies	• Number of organizations submitting data • Data and information on the state of the practice of software engineering
Capability to support software engineering improvement within software organizations	• Based on past experience • Prone to fads and quick fixes	• Some organizations conducting studies based on models of their software process	• Common practice among high maturity organizations • Some use by low maturity organizations	• Used by organizations • Number of software engineers trained in method • Type of data submitted to the SEI

Figure 1-11: Trends in Software Engineering Measurement

1.3.5 One-Year Objectives for 1996

1. Lay the basis for establishing the SEI as a recognized national resource for the collection, analysis, and dissemination of software engineering data and for assessing the state of the practice of the U.S. software community.

   This will entail providing a safe haven for data from software organizations on their processes and performance and reporting on trends in the software community. This will also include supporting organizations in using quantitative information and methods effectively in managing software projects, measuring software product attributes, measuring improvements in software processes, and benchmarking industry practices.

2. Take a leadership role in national measurement activities.

   This is a continuing activity in serving in an advisory role to the National Software Data and Information Repository and coordinating our products and services with others in the community.

3. Define, with other SEI impact areas, an integrated set of software engineering measures and practices.

   A core competency in software engineering measurement will be established at the SEI. The focus will be on integrating software measurement-related activities and database activities across SEI impact areas.

4. Provide organizations with the tools to support identifying, defining, collecting, analyzing, and improving measures of continuous improvement in software development.

   We will work with government and industry organizations to help them formulate and adopt measurement processes, select appropriate measurements to clearly communicate essential information needed for effective management decision-making, and define and implement a software process measurement system. We will also help our customers by showing them how measurement supports SPI and addressing business value measurement.

5. Work with organizations installing software engineering measurement programs that will help them answer questions on achieving business goals.

   This will entail coordinating with the effort sponsored by the Joint Logistics Commanders (JLC) to complete a software measurement handbook that addresses measurement for process management, evolution, and improvement. We will synthesize, package, and transition best practices into the community.

1.3.6 Baseline Work Outputs

This section describes the baseline work outputs for the Software Engineering Measurement activity area. These outputs were approved for 1996 through the basic funds allocation process. Appendix A lists all baseline and proposed add-on work outputs.

SP-11B Software Measurement Handbook (1995-1996)

The primary output provided by this activity is a software measurement handbook. Production of this handbook will be coordinated with the JLC-sponsored measurement guide "Practical Software Measurement." While the JLC guide is focused primarily on the DoD acquisition program manager, the focus of the SEI handbook will address a much broader perspective including government and industry software engineering process groups and project managers.

Three areas of software measurement should be discussed in three complementary guidebooks: project/program management, process improvement, and product engineering. The JLC-sponsored effort is addressing measurement for program/project management purposes. The SEI work will focus on measurement for process management /process improvement. Future joint efforts will include a third guidebook addressing product engineering.

Purpose

This software measurement handbook will provide information that will enable software development or support organizations to use software measurements to manage and improve software processes that affect their business objectives.

Customers

The customers for this output include industry and government software project managers, developers, and SEPGs.

Collaborators

The collaborators for this effort include the writing team of the JLC-sponsored measurement guide, "Practical Software Measurement."

Approach

The general approach includes synthesizing, packaging, and transitioning best practices. The approach is as follows:

- Identify key process management issues.
- Define a generic measurement process.
- Structure the effort to provide assistance to people addressing or managing their organization's process improvement program.
- Produce draft handbook (late 1995).
- Produce final handbook (1996).

SP-12B Process Appraisal Information System (PAIS) (1995-1997)

Three types of outputs are provided by the process appraisal information system: semi-annual updates of the community maturity profile, reports on software process issues identified by organizations conducting process appraisals, and an information resource supporting other SEI research and development efforts. This task is a continuation of the 1995 task, "SEI Process Database."

Purpose

The SEI process appraisal information system provides a general data collection, storage, retrieval, and reporting capability based upon the results of software process appraisals (SPAs), internal process improvements (IPIs), and interim profiles (IPs) performed in the software community. The PAIS provides the SEI with actual data on process issues and improvements in software organizations including process strengths and findings, action plans for improvement, process maturity ratings, and actual project and organizational performance. The data gathered and functionality provided support multiple SEI work efforts as described in the outputs above.

Customers

Existing customers of the PAIS include software process improvement champions, software organization managers, the software community, as well as SEI researchers and product developers. SEI Customer Relations reports that the updates of the community maturity profile developed by the PAIS are frequently requested by our external customers. Data and results from the PAIS are frequently included in SEI presentations and have been used in the development of new SEI products.

Collaborators

The PAIS depends upon those leading process appraisals to submit data and results to the SEI. The list of contacts and relationships between the PAIS and those conducting software process appraisals, both as internal consultants or third parties, has been growing. This reflects the trust and confidence that the community has in the PAIS and its operation. The PAIS now has contacts in more than 100 companies spanning more than 450 software organizations.

Approach

The general approach for the PAIS is to work with our collaborators to identify and collect data from software organizations that meet the needs of both internal and external SEI customers. This approach translates into two basic steps: (1) expanding the scope of the data gathered and (2) returning useful information related to software process improvement to customers. From an internal operations perspective, the PAIS subscribes to several principles:

- Protect the confidentiality of the data.
- Increasingly automate the process.
- Work closely with both customers and collaborators.
- Quickly and accurately acknowledge in writing all contributions of data.
- Report only data for which we have documentation.

1.3.7 Proposed Add-On Work Outputs

This section describes the proposed add-on work outputs for the activity area related to software engineering measurement. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

SP-14A Software Engineering Information Repository (1995-1997)

This proposal is a joint effort of the Risk and Process areas and involves developing a general software engineering data and information repository through integration of the risk repository and the PAIS. Actual data from industry and government organizations as well as data from published reports could be maintained. Funding this proposal would also include the definition and collection of data.

The technologies that are the bases of the risk repository and the PAIS are flexible and adaptable, allowing for expansion of the repository to include new types of data and information. Relating the different types of data that can be handled with these technologies would create a unique and powerful resource for the SEI and the software engineering community.

1996 outputs include an information service on software engineering issues, operational plans, a guidebook on data collection and analysis, and a technical report on trends in software engineering. These outputs support producing an information service on software engineering issues using the SEI repository. This would be a retrieval service providing customers with online information. This service may initially be a human-assisted batch service, but eventually it would be an online, interactive service.

Purpose

The purpose of this effort is to establish a repository of software engineering data that would provide the community with a picture of the U.S. software industry as a whole including national baselines, trends, and performance data. These data would address issues such as how many U.S. companies are involved in software development today, how much software is in the U.S. inventory, are software systems getting larger, will there be enough programmers in five years, and what is the average quality of U.S. software data.

Community needs for information include figures upon which to base estimates, data to assess how their quality and productivity compare with national averages and distributions, and data to evaluate the performance of software suppliers. A primary benefit to our customers would be the ability to find, through a single source, the most comprehensive compilation of software engineering data on project and process performance, risks, process improvement, and mitigation and improvement tactics.

Customers

Customer communities that would benefit from this information include those in government, academia, and industry seeking information on national performance standards on software engineering.

Collaborators

Industry and government organizations as well as other collaborators will be asked to contribute software information.

Approach

We have begun this effort with a feasibility study in 1995. The initial focus of this effort is for the Process area and the Risk area to integrate their repositories and then to address how the repository can be expanded to include broad software engineering data, plan the products and services of a broader software engineering repository, and address the operational and maintenance costs of establishing such an information repository. Our detailed approach is as follows:

- Develop a plan and a design document to address how the repository would be supported, how customers would retrieve from and enter data into the repository, and how the repository would be maintained, to name a few.
- Define initial data definition and collection procedures including a standard project data sheet to be used in the collection of project characteristics.
- Generate operational documentation and procedures that document the use of the repository, including retrieving data, analyzing data, and entering data.

- Develop a users' manual that will describe in detail how to use the repository for retrieval and analysis of data.
- Analyze other reports on the software engineering field to provide insight into areas that the SEI should be investigating, and publish comprehensive articles on the state of the practice. This report will detail how to generate the data for a state-of-the-practice report.

SP-15A Understanding Organizational Conditions Leading to Successful Software Process Improvement (1996)

As part of our work on technology transition practices (see Chapter 6 of Volume II), considerable information has been gathered about an organization's history, culture, likely forms of resistance to change, organizational structure, sponsorship, etc. This information has proven very useful to organizations when developing improvement implementation plans. This work will create a body of knowledge to be added to the PAIS that will lead to enhanced and improved methods, training materials, and facilitation capabilities making software process improvement more certain, effective, and efficient.

Purpose

The software engineering community is increasingly interested in improving its processes, technology, and people skills. However, only anecdotal evidence of the value of this work is generally available. It will be easier to motivate the improvement process and evaluate the effectiveness of improvement strategies if the current data possessed by the SEI are analyzed and if follow-up interviews are made with clients to ascertain the longer range effects of improvement programs initiated with the help of the SEI. Proper organizational climate and alignment appear critical to effective organizational change. Applying this knowledge to software process improvement can dramatically increase our client's ability to succeed. Little is currently known about the organizational factors contributing to successful improvements in software organizations. This work will increase our knowledge about what has proven to be successful in the software community.

Customers

Potential customers for the results of this effort are those in the software engineering community interested in learning about improvement experiences of software organizations as well as those organizations engaged in software process improvement.

Collaborators

The SEI will revisit collaborating organizations that originally provided data to us in order to gather progress data about their improvement efforts.

Approach

The proposed work consists of consolidating existing data at the SEI that have been gathered but not analyzed, supplementing the data with follow-up interviews of some of the organizations contributing to the existing data, and integrating this data into the PAIS. This work will be a collaborative effort between the Process area and Integrated Transition Strategies and Methods. Considerable data on client organizations have been collected as a result of customer improvement activities. These data provide insight into how to evaluate an organization's readiness to change, effective change strategies that have been actually used, and baseline data for post-hoc analysis of improvement effectiveness. The PAIS is currently organizing and collecting appraisal data. The addition of this organizational analysis data along with follow-up information to the PAIS would allow more detailed analysis and insight into successful software process improvement.

By sanitizing, aggregating, refining, organizing, and then integrating these data into the PAIS, we will produce information such as best organizational practices, case histories, improvement lessons learned, winning strategies, organizational readiness profiles, trends, etc. We will also be able to specify more clearly what data should be collected in future efforts.

1.3.8 Related Customer Activities

Client support with services and government agencies will complement the development of the basic-funded products. Clients anticipated in 1996 include Ascocarp, Defense Mapping Agency, and the DISA.

The TO&P activities may involve the following engineering services, training services, and the development of tailored guides/handbooks:

Engineering services

- Assist with the formulation and adoption of measurement processes (assistance would include, as necessary, guidance and training in developing, piloting, deploying, and using measurement for decision making).
- Assist in selecting appropriate measurements to clearly communicate essential information needed for effective management decision making.
- Assist with the definition and implementation of a software process measurement system.
- Collaborate to build a software measurement implementation plan.
- Assist customer in installing a software measurement program.
- Support internal infrastructure (e.g., SEPG, measurement working group) to institutionalize software measurement processes that support SPI efforts.
- Assist customers in using and analyzing data.
- Support customers in baselining their software measurement processes and measures.
- Support customers in identifying, defining, collecting, analyzing, reporting, and evolving the software measures, processes, and systems.

Training services

- Train customers on software measurement principles, practices, and processes through offering "Engineering an Effective Software Measurement Program" course.
- Tailor course to address specific customer needs.
- Develop new courses that address specific customer needs.

Handbooks

- Develop tailored customer software measurement handbooks.
- Develop customer-tailored data collection guides.

Volume II
Chapter 2 Table of Contents

2 Risk Management

The SEI has established a strong foundation in risk management in the form of a model and methods for establishing risk management in acquisition and development programs. Looking to the future and building on previous work, we are expanding to three activity areas: Software Acquisition, Software Risk Management, and Knowledge and Information Technology. (See Figure 2-1.) These three activity areas support each other and are strongly tied to a central theme of making better decisions by managing uncertainty.

The Software Acquisition activity area is tied to the SEI's previous experience in software process improvement, the Capability Maturity Model (CMM), and software risk management. The Software Risk Management activity area is the continued evolution and transition of a continuous and robust risk management paradigm to the acquisition and development communities. The third activity area, Knowledge and Information Technology, is exploring more advanced methods and technologies with the potential for order-of-magnitude improvements in the way organizations deal with uncertainty and gain understanding. Together these three activity areas address both the acquisition and development communities' need to make informed decisions in uncertain conditions.

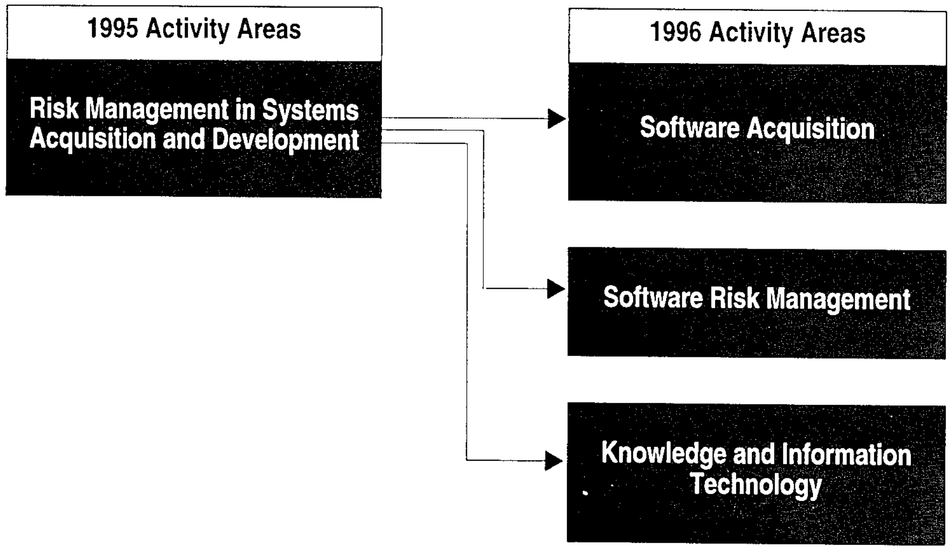
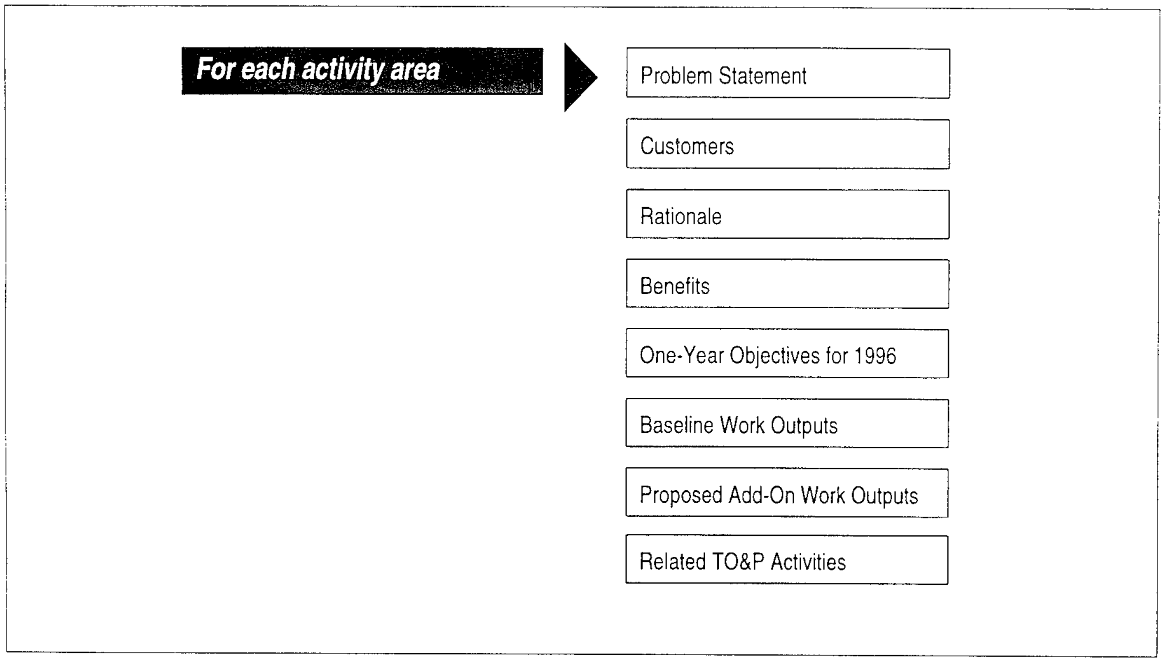

1995 Activity Areas
Risk Management in Systems Acquisition and Development
1996 Activity Areas
Software Acquisition
Software Risk Management
Knowledge and Information Technology

Figure 2-1: Mapping of Risk Activity Areas

In this chapter, the sections for this impact area are as follows:



2.1 Software Acquisition

2.1.1 Problem Statement

Acquisition and development of large software driven systems continues to suffer large cost and schedule overruns, with attendant operational problems. While industry is improving its capability and performance through use of the CMM for software, many acquisition organizations continue to operate in an unstable environment. Staffing is based on the availability of individuals, resulting in a random composition of acquisition skills. Very few program team members have the requisite software acquisition or application domain skills and little documentation exists to define procedures or capture corporate memory. Consequently the software acquisition typically proceeds in an ad hoc manner.

The SEI and other agencies have developed and transitioned technologies to improve narrow areas of the Department of Defense (DoD) software acquisition practices. Although each technology provides the potential for improvement in specific areas, the technologies have not been integrated by an overall framework that serves as a model and roadmap to make lasting improvements. The SEI can lead in implementing and institutionalizing the Software Acquisition Maturity Model (SAMM) based on our leadership in its development. Since the SEI is providing expertise in managing the efforts of diverse teams and in gaining widespread review and acceptance of the SAMM, it can also provide leadership in the development of software acquisition guidebooks.

2.1.2 Customers

The potential customer base is nationwide and spans defense, non-defense, government, and commercial industry. Support has been received from

- DoD acquisition organizations (Army, Navy, Air Force)
- Office of the Secretary of Defense (OSD) command, control, communications, and intelligence (C3I)
- Defense agencies
- Federal agencies
- Industry

2.1.3 Rationale

In 1995, the SEI led a joint government-industry team to develop and pilot test the SAMM. The SAMM, similar in concept to the CMM for software, provides a means for assessing and improving the maturity of organizations that acquire software intensive systems, both for the government and for industry. The model defines key process areas (KPAs) in software acquisition, such as Requirements Definition and Management, Software Risk Management and Organizational Process Definition and Improvement. While planning the implementation of the SAMM, it has become apparent that the DoD needs a software acquisition improvement framework that can be tailored by each DoD component and that provides a roadmap for assessment and improvement.

There are a number of current technologies at the SEI that have been and can be used for improvement of software acquisition management. A software acquisition improvement framework is needed to incorporate these and other efforts into a coherent program to give a focused, integrated, and formalized means for improving software acquisition processes and practices. This framework would act as a means for organizing and structuring an improvement model, technologies, and implementation guidelines to be developed and transitioned to the acquisition community. The technologies and practices may be existing or provided by the community later (e.g., DoD Software Acquisition Best Practices) or developed by the SEI. These include

- a model for assessing and improving software acquisition organization maturity (SAMM)
- criteria for selecting, adopting and integrating appropriate processes, methods, and tools for software acquisition process improvement
- guidelines for implementation of the selected processes, methods, and tools

2.1.4 Benefits

This work continues and expands the development and implementation of the SAMM and Acquisition Risk Management guidelines. The SAMM will provide customers a structured model for improving their processes and methods. Assessment techniques will enable benchmarking and clearly defining strengths, weaknesses, and potential improvement areas. The Software Acquisition Improvement Framework will identify candidate practices and supporting technologies, expertise, and infrastructure required to implement, institutionalize and measure the success of a chosen implementation.

The broader community can be leveraged during development and testing of the SAMM to provide practical guidelines to the entire software acquisition community. Current state-of-the-practice in acquisition organizations is between experimental and guided. High level government guidance is sometimes supplemented by local practices and experience databases; however, most organizations rely heavily on individual expertise and experience. The software acquisition guidebooks will move the practice to controlled technology maturity level. One example is the Acquisition Risk Management Source Selection (ARMSS) Guidebook. This work will improve the practice in software acquisition (Figure 2-2).

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Software Acquisition Maturity Model	• Acquisition organizations ad hoc	• 4 programs assessed • 2 programs start improvement activities	• DoD standard is level 3 • 50% of acquisition organizations at level 2	• Programs report improved acquisition success
Software Acquisition Improvement Framework	• None exists	• Initial framework for improvement	• Well populated framework	• Programs following roadmap report measured results
Acquisition Risk Management Source Selection	• No insight, at source selection level, into program and offeror risks	• Concept adopted by Defense Systems Management College (DSMC) • Acquisition community using ARMSS guidebook	• Accepted as routine practice in all major acquisitions	• ARMSS used to monitor contract accomplishment

Figure 2-2: Trends in Software Acquisition

2.1.5 One-Year Objectives for 1996

The objectives for the Software Acquisition activity area are the following:

- Release version 1.0 of the SAMM.
- Develop a draft of the Software Acquisition Improvement framework.
- Select and develop Acquisition Risk Management guidebooks.
- Publish the Acquisition Risk Management Source Selection Guidebook.
- Pilot test software acquisition improvement products above.

2.1.6 Baseline Work Outputs

This section describes the baseline work outputs for the activity area related to software acquisition. These outputs were approved for 1996 through the basic funds allocation process. Appendix A lists all baseline and proposed add-on work outputs.

RM-1B Software Acquisition Maturity Model (1996)

The SAMM is a model for benchmarking and improving the software acquisition process. The model follows the same architecture as the SEI CMM. This model, however, is focused on the customer and the acquisition process. Each maturity level indicates an acquisition process capability and contains several KPAs. Each KPA contains goals and common features and organizational practices intended to institutionalize common practice. It provides a means for measuring software acquisition lifecycle improvement showing organizational effectiveness and the value of improvement.

The SAMM provides a principle-based, public model for assessing and improving a software acquisition organization's maturity. The SAMM will provide software acquisition organizations with many of the concepts needed to improve and keep pace with maturing software development organizations. The SAMM is a model for a software acquisition improvement framework enabling organizations to create a roadmap and plan for incremental improvement in their capability to acquire software from software development organizations.

Purpose

The SAMM provides acquisition managers the foundation and criteria to measure the efficacy of their acquisition processes. The SAMM is a lifecycle model that acquisition managers will use to place a stake in the ground to mark their current capability and use as a benchmark for the improvements they want to achieve. With the SAMM, acquisition managers will have a basis for understanding their acquisition process capability, how to measure it, and how it can be improved.

Customers

Customers include the following:

- Air Force

  -Electronic Systems Center (ESC)
  -Aeronautical Systems Center (ASC)
  -Space and Missile Center (SMC)
  -program executive officers (PEOs)

- Army

  -Communications-Electronics Command (CECOM)
  -Missile Command (MICOM)
  -PEOs

- Navy

  -Naval Air Systems (NAVAIR)
  -Space Warfare Systems (SPAWAR)
  -PEOs

- OSD, C3I
- Defense Information Services Agency (DISA)
- Federal Agencies

  -National Oceanographic and Atmospheric Sciences Agency (NOAA)
  -Federal Aviation Administration (FAA)
  -National Institute of Standards and Technology (NIST)

- FFRDC

  -Aerospace
  -MITRE

- Industry

  -Software Productivity Consortium (SPC)
  -Abacus Technologies
  -Lockheed-Martin Marietta
  -TRW
  -Computer Sciences Corporation (CSC)
  -GEC-Marconi
  -Adsystech, Inc.

Collaborators

Collaborators include the following:

- DoD (Army, Navy, Air Force)
- DISA
- NOAA
- Industry (Kaman Sciences, Loral Federal Systems, Logicon)

Approach

- Publish SAMM version 1.0 including the Model Description and Key Practices.
- Document Implementation Guidelines which address tailoring the SAMM for project size, acquisition stage, acquisition type, etc.
- Develop and document training material to implement the model.
- Initiate implementation planning.
- Plan for transition and maintenance of the model.

2.1.7 Proposed Add-On Work Outputs

This section describes the proposed add-on work outputs for the activity area related to software acquisition. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

RM-1A Software Acquisition Improvement Framework (1996-1997)

The Software Acquisition Improvement Framework uses the SAMM as the foundation to explicitly map an acquisition organization's improvement strategy. The Software Acquisition Improvement Framework is a structure to generate a tailored roadmap to assist the organization in successfully adopting and integrating practices and technologies into their acquisition process. A software acquisition improvement roadmap will help the organizations to incrementally implement the technologies and practices in a phased improvement. The roadmap will describe both a short and long-range integrated plan for initiating and managing an improvement program. Guidelines for tailoring are an integral part of the framework. Tailoring the framework to generate a roadmap is necessary for situational acquisition specific characteristics and to support current and future organizational needs. Clear understanding of what will be done and when it will be done is invaluable for a low risk improvement program.

Purpose

The framework will define three dimensions (SAMM, technologies, and implementation) as depicted in Figure 2-3. The framework will characterize attributes that support the cost, schedule, and technical decision-making needs associated with the software acquisition process. The first dimension is represented by the SAMM and supported by an assessment tool to baseline an acquisition organization's KPAs (current state). The KPAs are further defined by those practices necessary for improvement. A software acquisition technology catalog of best practices will provide a source of acquisition best practices and supporting technologies for continuous improvement. The second dimension characterizes software engineering technologies, required skills, and expertise that underpin good practice. The third dimension identifies the transition mechanism for moving the technology into practice. This dimension specifies the training, supporting infrastructure, and practice guidelines. Training is for individuals to improve their ability to use the practices and technologies and is supported by the P-CMM (People - Capability Maturity Model). Practices will be described in guidebooks such as our earlier work, the ARMSS.

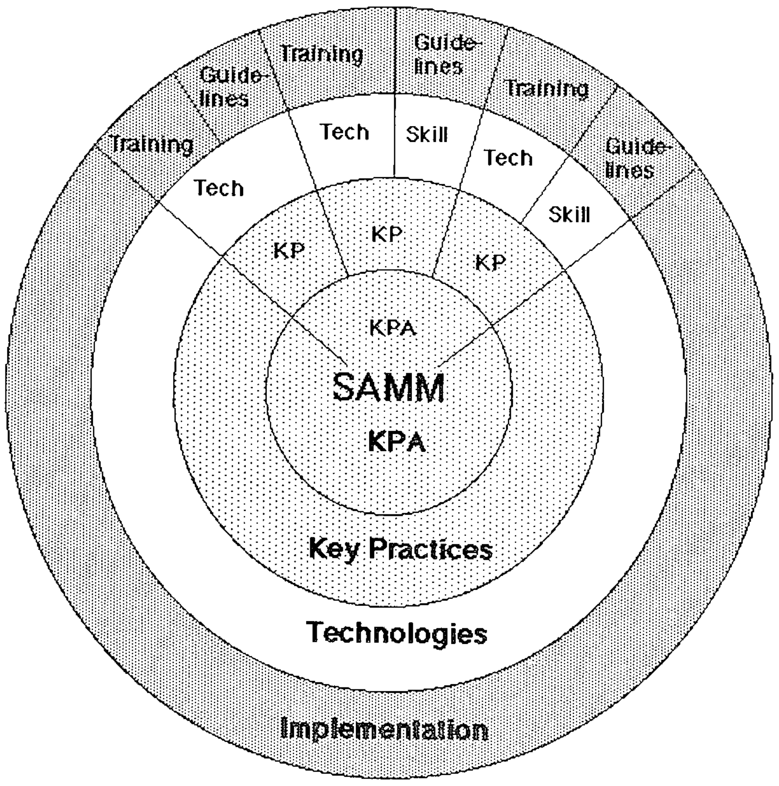


Figure 2-3: Software Acquisition Improvement Framework

Customers

The customers for this output are government and industry acquisition organizations.

Collaborators

Potential collaborators include acquisition organizations and programs.

Approach

- Conduct requirements elicitation and acquisition domain analysis
- Develop software acquisition technology catalog to populate the framework.
- Develop and document Software Acquisition Improvement Framework.
- Develop and draft tailoring guidelines.
- Pilot test framework/roadmap.

RM-2A Software Acquisition Maturity Model Guidebooks (1996-1997)

Guidebooks for selected practices contain instructions for meeting the goals of the commensurate KPAs. They will contain bibliographies of efforts that are successful in meeting the KPA requirements, instructions for performing the activities, and guidance in the selection of support documentation and tools. An example is the Acquisition Risk Management Source Selection Guidebook that provides specific guidance for the source selection evaluation board and the prospective bidders to identify and mitigate risks in the proposed work. The guidebooks could be both descriptive and prescriptive depending on the characteristics of the practice.

Purpose

The purpose of the SAMM Guidebooks is to give program managers, PEOs, and software acquisition managers methods and guidance in moving up the SAMM maturity ladder. This work will help institutionalize the framework and improvement efforts begun by the SAMM. The Guidebooks will be similar to the SEI Ada Adoption Handbook and provide guidance for key processes within the framework of the SAMM.

Customers

The customers for this output are government and industry acquisition organizations.

Collaborators

Army (CECOM) will collaborate, with other collaborators to be determined.

Approach

- Select key practices.
- Develop draft guidebooks.
- Pilot test draft guidebooks.

2.1.8 Related TO&P Activities

Continued technical objectives and plans (TO&P) funding will be sought to support pilot testing of the SAMM, and new TO&P funding will be sought for the Software Acquisition Improvement Framework and SAMM Guidebooks. TO&P and cooperative research and development agreement (CRADA) customers provide the primary opportunity for the technical staff to work in real program environments. In some cases, methods are developed in collaboration with client organizations and in all cases this collaborative activity provides the testing ground for developing software acquisition improvement methods. These efforts are vital to gather and test best practices in acquisition. This information is made available for the entire software acquisition and development community. These collaborative activities with clients are a vital part of the transition strategy for rapidly improving the state of the practice of software acquisition.

2.2 Software Risk Management

2.2.1 Problem Statement

Software acquisition, development, and maintenance programs have suffered large cost overruns, schedule delays, and poor technical performance. The increasing complexity of systems has exacerbated the situation. The failure to appropriately deal with uncertainty in the acquisition and development of complex, software-intensive systems has contributed to the situation. This failure to adequately address uncertainty has, in large part, been due to the lack of a systematic approach to identify, communicate, and resolve software development risks. Often when dealing with risk the focus has been on the symptoms of cost overruns and schedule delays rather than on product acquisition and development root causes. The causes span all aspects of software development and maintenance programs, as shown in Figure 2-4. There is a need to define, verify, and establish systematic risk management processes, methods, and tools that address all potential sources of risk for all areas of system acquisition and development.

Based on the data from 30 risk assessments, we found that programs need a structure to assess software risks, to gather data about risks, and to share information with other projects regarding issues, mitigation strategies, or lessons learned regarding software.[1] Unfortunately, numerous textbooks on risk management offer little on how to implement a risk assessment or management process. The SEI is chartered to gather this type of information and share it with the community. The SEI is evolving several product lines to enable the community to make better decisions about risk management.

1. Some organizations such as consulting firms do possess a risk assessment and management process but treat those processes as proprietary and keep their data confidential.

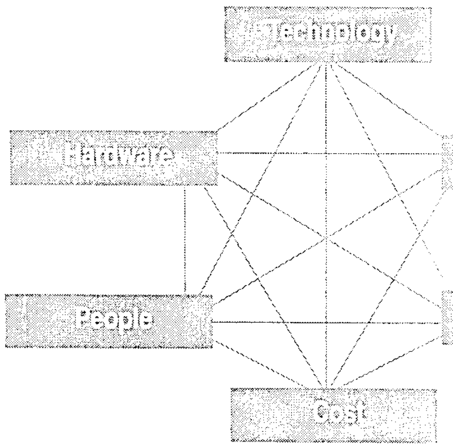


Figure 2-4: Risks Within a System Context

2.2.2 Customers

The potential customer base spans defense, other government, and commercial industry. Specifically, clients include the following:

- Navy Program Executive Office for Air Anti-Submarine Warfare, Assault and Special Missions
- Marines (Marine Corps Tactical Systems Support Agency [MCTSSA])
- Army (CECOM)
- U.S. Treasury Department
- Loral
- Northrop Corporation
- Harris Government Information Systems Corporation
- Chrysler Technology Airborne Systems Inc.
- Allied Signal

2.2.3 Rationale

Every organization involved in developing software-dependent systems has the problem of being able to identify risks early enough to take corrective action and avoid surprises that could jeopardize the success of their programs. These organizations also need to be able to assess and plan technology improvements to build tomorrow's complex systems. As systems become more complex, a continuing improvement initiative is needed to stay abreast of technology to increase efficiency and to take advantage of the latest techniques.

In a survey conducted at the 3rd Annual Risk Conference in 1994, only 10% of the respondents reported that they do risk management on a regular basis. Since 1991 we have conducted risk assessments and participated in developing risk mitigation strategies. The data from this field work show the need for a cost-effective and practical means to identify, analyze, track, and mitigate risks. However, feedback from SEI workshops and this field work has revealed that risk management processes, methods, and tools are simply not widely known or used. Teams with multiple and varied disciplines require continuous processes and methods to identify, manage, and communicate the risks in the acquisition and development process.

The Software Risk Evaluation (SRE) method is increasingly being adopted by industry and government as an integral part of their acquisition and development processes for risk identification. Industry, in particular, is requesting that the Taxonomy-Based Questionnaire be expanded to cover the entire system lifecycle from concept through post-deployment software support (PDSS). Specific areas of interest are performance, maintainability, security, and post-deployment support. The SRE continues to be an instrument for independent review and diagnosis to assess performance or initiate an improvement process. The SRE also provides a baseline to install continuous risk management into an organization or program. The Risk Clinic takes this baseline risk information or independently defines a continuous risk management process for a program. This is accomplished in a workshop environment to adapt best practices to a program's existing infrastructure. A transition phase of coaching allows the inculturalization to sustain the effort. Finally, a paradigm of team-based, collaborative customer-supplier risk management, called Team Risk Management, capitalizes on the powerful concept of team-based approaches such as integrated product teams.

The qualitative risk identification methods developed by the SEI have been effective in the field. Quantitative methods and risk metrics in risk management are the next logical step to enhance these methods, providing more precise values for characteristics of risks such as their likelihood, impact, and time frame. Such increased precision makes the information more visible and "real," particularly to engineers and management.

A significant portion of our effort in the Software Risk Management activity area is TO&P or CRADA funded (see Section 2.2.8, Related TO&P Activities).

2.2.4 Benefits

Risk management provides the processes and methods to enable organizations to systematically identify and manage software technical risks within their programs. Using the SEI risk management paradigm

- improves predictability of results in acquisition and development
- improves communications among team members
- enables the program to identify and analyze risks
- provides a structured process for developing risk mitigation strategies
- ensures that program software personnel participate in program risk management

Simple, widely accepted methods significantly enhance the ability of an organization to analyze its risks and potential mitigation strategies. Qualitative methods serve quite well in many situations, particularly when there are few data on which to base judgment. On the other hand, quantitative methods used appropriately have the advantage of rigor and acceptance in engineering and software development. They also provide greater value in decision making by providing a clearer understanding and discriminating the multiplicity of factors associated with software development risks. Trends for this activity area are shown in Figure 2-5.

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Software Risk Evaluation	• No widely accepted mechanisms to deal with program software risks	• Systematic application of SRE method	• Integrated into overall framework of program management through DSMC	• Number of applications • Continual use by clients • Institutionalized • Shows up in contracts
Continuous Risk Management Methods	• Rely on individual knowledge and experience	• Systematic methods and tools in use • Team-based methods in use	• Integrated into overall framework of program management	• Number of vendors • Used by clients • Institutionalized • Shows up in Request for Proposals (RFPs) and proposals
Risk Cost Model	• Cost and schedule not directly related to technical risk factors	• Cost and schedule directly related to technical risk factors	• Calculated return on investment	• Number of vendors • Number of users
Risk Metrics	• Ad hoc	• Risk-driven approach for selection of metrics • Quantitative measures adopted	• Predictive quantification methods identify risks • Accepted as routine practice	• Shows up in RFPs and proposals • Number of users

Figure 2-5: Trends in Software Risk Management

2.2.5 One-Year Objectives for 1996

The objectives for the Software Risk Management activity area are

- Harmonize SRE, training, and continuous risk management products and services.
- Publish the Risk Clinic as the model for installing continuous risk management into organizations and programs.
- Develop risk metrics and quantitative methods for risk management.

- Develop risk-driven cost model with the University of Southern California-Center for Software Engineering (USC-CSE).
- Extend the application of risk management in specific domains (e.g., Open Systems).

2.2.6 Baseline Work Outputs

There are no baseline work outputs for the activity area related to software risk management.

2.2.7 Proposed Add-On Work Outputs

This section describes the proposed add-on work outputs for the activity area related to software risk management. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

RM-3A Software Risk Metrics (1996-1998)

Software Risk Metrics (SRM) is the quantitative means to plan, track, and control the risks that are inherent to acquisition and development of software-dependent systems. The SRM effort will define, develop, and transition quantitative methods and associated tools for installation within a program's ongoing risk management activities.

The SEI Risk area has developed and evolved many methods and tools, most of which have addressed the risk identification and analysis activities. By transitioning these methods and tools the SEI has succeeded in improving the state of the practice of risk management for software-dependent systems. But this effort has resulted primarily in advancing the capability to identify and analyze risks. Risk metrics and measurement are required to improve risk mitigation activities to

- objectively communicate risk performance and progress
- assess and minimize the impact of program risks
- facilitate the mitigation of program risks

Purpose
Managers will use SRM to both communicate program performance to senior management and external organizations such as oversight agencies and to execute their internal program management functions. Practitioners will use SRM for planning, tracking, and controlling risks.

Customers
Customers for this output include acquisition and development program offices.

Collaborators

The Navy will collaborate on this output.

Approach

- Define concepts and application (1. Data characteristics of risks, 2. Visual information-processing, 3. Objects of measurement and their inter-relationships).
- Define basic risk measures, collection, and use guidelines.
- Pilot test risk metrics.
- Develop implementation guidebook.

RM-4A Risk Cost Model (1996-1997)

The risk cost model is an extension of the work in progress in the Risk impact area in collaboration with USC-CSE. This work is determining the Constructive Cost Model (COCOMO-2) cost drivers for estimating the "risk," in terms of cost and schedule, of integrating commercial off-the-shelf (COTS) software into both a deliverable system and the environment wherein the system is being built. Because of shrinking DoD budgets, more and more systems will be built using COTS software. Many projects have suffered adverse effects because of an inability to predict the cost and schedule needed to integrate COTS software. The outcome of this work will give program managers a tool to make better estimates of integrating COTS software and a means to calculate return on investment for risk mitigation.

The risk cost model activity is also tied into the repository work and risk metrics work. Data collected on Software Risk Evaluations and contained within the repository will be used to help develop the cost drivers for COCOMO-2. This information will be valuable in establishing a methodology for risk measurement and reporting, especially in the tracking and controlling functions in the SEI risk paradigm. The COCOMO-2 will be integrated into the SRE process to help estimate the cost of risk mitigation strategies and the cost to the program if a risk happens. This will help establish a return on investment (ROI) for risk management.

Purpose

The Risk Cost Model will provide acquisition and development program managers a model to quantitatively assess the cost and schedule impact of risks identified by software risk evaluation. Used in conjunction with planning mitigation strategies, the model gives management a way to assess return on investment to further evaluate the strategy. Periodic model data snapshots can provide cost and schedule trend data which give managers additional insight to program status and forecasted outcome.

Customers

Customers for this output are the software acquisition and development community.

Collaborators

Collaborators include USC-CSE and members of the software engineering community associated with the USC-CSE.

Approach

- Map the SEI risk taxonomy and COCOMO-2 cost drivers.
- Develop draft COCOMO-2 risk cost driver guidebook.
- Pilot test the model and guidebook.
- Publish the model and guidebook in 1997.

2.2.8 Related TO&P Activities

TO&P funding will be sought to support continuing pilot testing and enhancements to risk management processes, methods, and tools. TO&P and CRADA customers provide the primary opportunity for the technical staff to work in real program environments. In some cases, methods are developed in collaboration with client organizations and in all cases this collaborative activity provides the testing ground for developing risk management methods. In addition to testing through pilot studies and collaboration with clients, these efforts provide empirical data to improve and update the Taxonomy-Based Questionnaire and to populate the SEI risk repository.

For example, TO&P funding will be used to evaluate and extend the Risk Clinic, risk planning, and risk training activities and to conduct pilot tests of the risk cost model and the risk metrics. Organizations that are interested in active collaboration with the SEI will be sought as partners in pilot testing. The benefits afforded these organizations include early access to proven methods for planning, tracking, and controlling of risks and an integrated method for estimating and tracking software risk costs.

Through these efforts, not only are SEI methods being proven in field conditions, but also best practices are being merged and integrated into a total risk management framework. This information is made available for the entire software acquisition and development community. These collaborative activities with clients are a vital part of the transition strategy for rapidly improving the state of the practice of risk management.

2.3 Knowledge and Information Technology

2.3.1 Problem Statement

The complexity of many software-intensive systems coupled with rapidly evolving technology results in formidable challenges for the software engineering community. These challenges are evident in the need for software engineering practitioners and managers to (1) possess knowledge of increasingly broad domains, complex systems and processes; (2) facilitate the sharing of knowledge across diverse specializations; (3) integrate voluminous and complex information into the practice of software engineering; and (4) translate this information into design and management decisions. The effective use of this knowledge and information is vital for successful risk identification and management in software-intensive systems development.

The fact that an overwhelming majority of software-intensive programs overrun costs and schedules and/or fall short of performance requirements because of a failure to identify risks is symptomatic of the ineffective use and management of information. A major contributor to this problem is the lack of accessible data about the development, use, and effectiveness of software products and practices. Both government and industry lack in-depth data on the sources of risks, mitigation strategies, and lessons learned concerning the development of software-intensive systems. This situation is further complicated by a lack of technologies to manage knowledge and information in the development processes within organizations. Given these conditions, there is an opportunity to improve software risk management practices by enhancing both access to and application of knowledge and risk information.

2.3.2 Customers

Customers include the software acquisition and development community.

2.3.3 Rationale

Software engineers are forced to resort to non-empirical arguments in deriving or evaluating software development risks because data is not available. For example, 67% of identified risks from a broad spectrum of programs arise in only 14 out of a possible 70 risk areas (see Figure 2-6). This provides the empirical basis for concentrating the development of methods and tools that would attack issues in these risk areas before expending the effort to develop methods and tools for the remaining 56 risk areas.

In addition to the lack of ready access to broad-based risk management information, there is a need to make information real to users and relevant to the specific issues (risks) being addressed. The process of making information relevant can be viewed as transforming information into knowledge and integrating that knowledge into an organization's processes. Knowledge integration technology facilitates making information accessible, useful, and relevant within an organization by collectively considering the dynamic interaction between people, knowledge, and processes. The effective use of knowledge and information can reduce risks and uncertainties characterizing complex software development programs and facilitate the management of those risks.

Risk mitigation requires understanding a situation and relating current conditions to previous risks, problems, and solutions. The knowledge required includes domain and process knowledge as well as specific knowledge about the program itself. For example, consider a risk with-

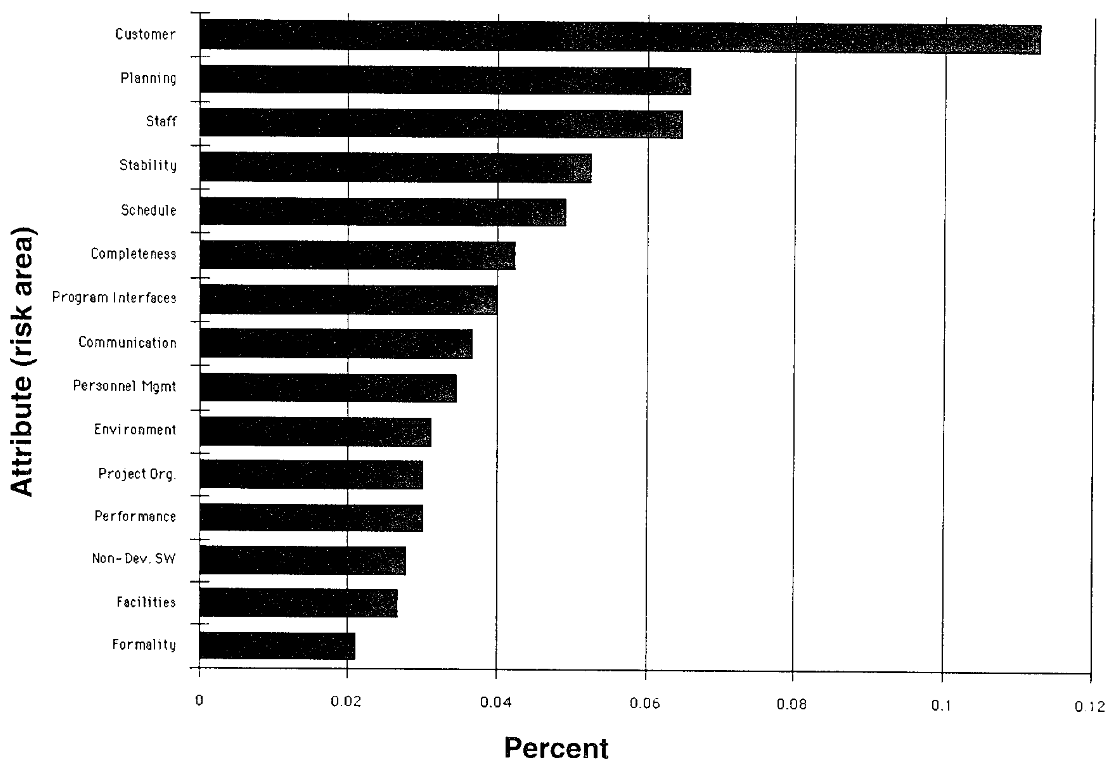


Figure 2-6: Percent of Risks by SEI Taxonomy Attribute

in a development program where there is a lack of expertise in object-oriented design on a project that is required to use the C++ language. The knowledge required to mitigate this risk for an organization includes the need to more fully understand the nature of object-orientation, its relationship to other approaches, the impact of object-orientation on the development process, the current staff's level of knowledge, and how similar risks have been successfully mitigated. This issue not only requires knowledge in object-orientation, management, and software process, but also the integration of this knowledge into the organization, i.e., it must become part of the interactions and shared understanding of management and technical staff. Achieving this integration involves more than simply acquiring information about these issues. The information must be structured and "fit" into the specific context of the organization—molded into organizational knowledge. Knowledge integration technology provides methods and tools that facilitate acquiring, structuring, and using (sharing, applying, managing) information within the context of the organization itself.

As software development efforts continue to become increasingly complex undertakings and the information challenges associated with software development become more formidable, an organization must increase its ability to interact efficiently as a collaborative team while providing management with visibility into programs (e.g., to communicate risks and opportunities). To accomplish this improved capability, awareness, and understanding, it is vitally important to recognize, apply, and effectively manage the collective knowledge within the organization.

To improve the accessibility of critical information across organizations and to integrate that information as knowledge into organizations, it is the premise of this activity area that

- A broad collection of data across organizations—a repository—is required to capture vital software risk and development information to enable improved risk and program management.
- Collective knowledge (organizational memory) exists within a software engineering organization that can be described, analyzed, managed, and used to improve risk management and software development practices.
- Methods and tools are required to support the access and integration of knowledge and information.

The Software Engineering Risk Repository (SERR) enhances the accessibility of critical software risk information for organizations throughout the software engineering community. Knowledge integration technology enables the effective use of organizational memory by facilitating the acquiring, sharing, structuring, applying, and managing of knowledge within an organization. Collectively, SERR and knowledge integration technologies enhance an organization's capabilities to deal with complexity and risk in software development by making knowledge accessible, real, and relevant. The relationship between SERR and knowledge integration technology is shown in Figure 2-7.

The problems and issues identified in this section support the reasons for our major thrusts and point to several community needs: access to risks faced on similar programs and corresponding mitigation strategies, methods to enhance the analysis of risk data, and models and tools to facilitate the process of risk management and the integration of this knowledge into organizations.

2.3.4 Benefits

SERR will provide the software community with easy access to risk management data. Program managers will be able to obtain information on common risks and mitigation strategies, thereby making informed decisions on the technical direction of software development programs. When all risks become known and managed, a program will reduce the number of issues becoming crises and thus control and manage routine and critical problems with much improved probability of success.

Appropriately automated tools for knowledge integration within an organization are the basis for codifying theory, principles, and best practices in carrying out all steps required for software acquisition development risk management. Hence, broad access to and integration of

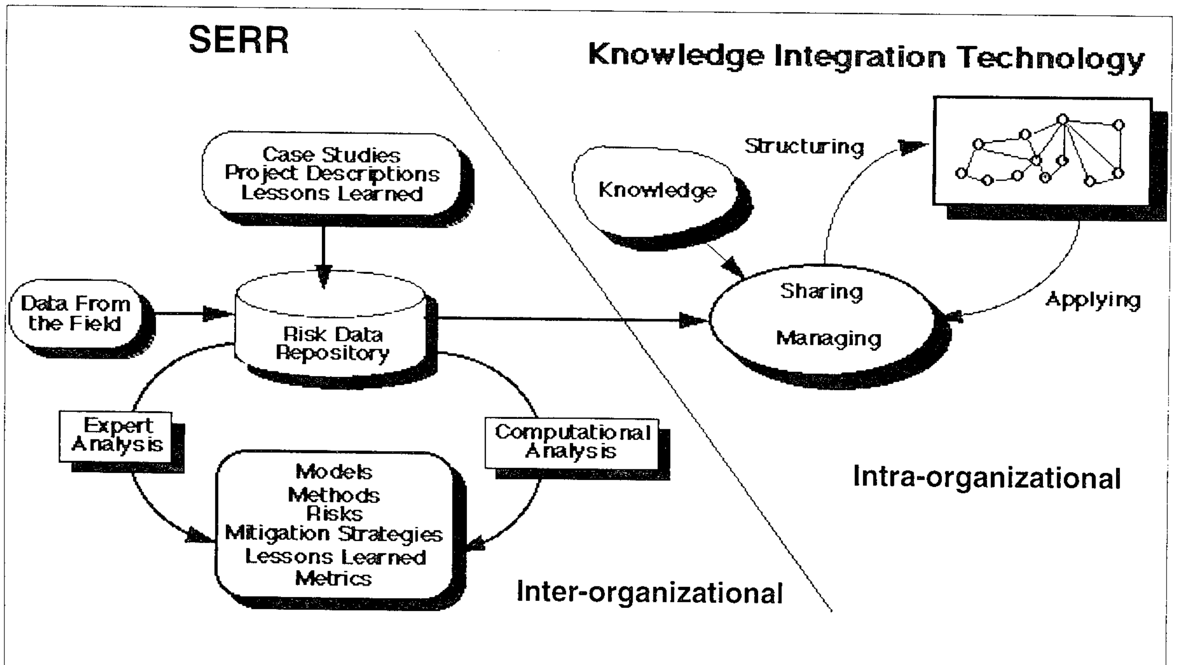


Figure 2-7: Software Engineering Risk Repository (SERR)

risk knowledge and information coupled with appropriate support tools will significantly add to the arsenal for dealing with software acquisition and development risks.

The implementation of SERR and knowledge integration technologies for application in software acquisition and development programs will

- improve predictability of results in acquisition and development
- provide a program with data on common risks, mitigation strategies, and lessons learned
- improve risk management practices
- enhance decision making capabilities regarding risks and program issues generally
- improve communications and collaboration among team members

Trends for the key risk management items of knowledge integration and information repository are shown in Figure 2-8.

Easy access to risk management data provided by the repository and the use of knowledge integration technology within an organization will provide program managers with the ability to make informed decisions on the risks they are facing, thereby increasing the probability of success.

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Knowledge Integration in Risk Management	• Bits and pieces with limited coverage and effectiveness	• Automated tool use • Defined methods	• Knowledge-based collaborative tools	• Number of vendor tools • Assessments (evidence of effective tools)
Software Engineering Risk Repository	• Limited high-level abstractions of risk and strategies	• Prototype database with empirical data • Limited access and use	• Widespread access and use • Knowledge-based tools to identify risks and strategies • Widespread community input of data	• Numbers of users • Number of successful predictions

Figure 2-8: Trends in Knowledge and Information Technology

2.3.5 One-Year Objectives for 1996

The objectives for the Knowledge and Information Technology activity area are:

- Release version 1.0 of SERR for access by the software community.
- Initiate beta testing of version 2.0 of the repository in the 4th quarter of 1996.
- Establish a method of cost recovery for use of the repository.
- Define a framework and preliminary guidelines from the use of knowledge summarization, analysis, and visualization technologies.
- Define the nature of groupware adoption services and finalize the groupware adoption guidelines. (See Section 2.3.8, Related TO&P Activities).

2.3.6 Baseline Work Outputs

This section describes the baseline work outputs for the activity area related to knowledge and information technology. These outputs were approved for 1996 through the basic funds allocation process. Appendix A lists all baseline and proposed add-on work outputs.

RM-2B Software Engineering Risk Repository (1996-1998)

SERR (Figure 2-9) is an on-line interactive document retrieval system containing and making available information on the development of software-dependent systems and the transfer of software technology. Using Mosaic as the front-end, natural language queries are made to retrieve documents from the information base. The ultimate goal of SERR is to provide a forum where the transfer, reception, and evaluation of advanced software engineering process technologies can be communicated, interpreted, and negotiated.

As part of SERR development, joint activities with the Process area are taking place to integrate the Risk Repository with PAIS (Process Assessment Information System) to form the Software Engineering Information Repository (SEIR). See Section 1.3 of Volume II for further discussion of the joint activities.

Purpose

SERR will not only serve the SEI as a means of collecting, analyzing, and reporting on software risk data, but also will provide the technology for the SEIR to aid the software community at large in their software improvement initiatives. Both acquisition and development organizations will be able to see common risks based on a program's characteristics and what strategies have been effective in mitigating those risks. This technology has the potential to support repository initiatives for other areas such as Disciplined Engineering, and architectures work.

Customers

Customers for this output are the software acquisition and development community.

Collaborators

Potential collaborators include clients with whom we previously collaborated on risk assessments.

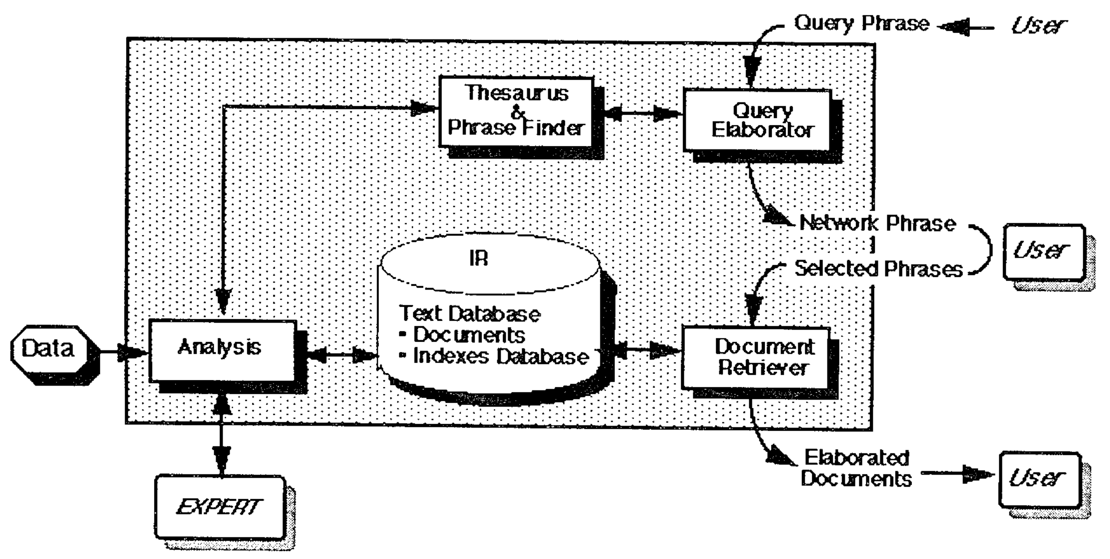


Figure 2-9: Software Engineering Risk Repository (SERR)

Approach

Each version contains increased functionality:

- Release SERR version 1.0 March 1996; features data access and statistical reporting.
- Release SERR version 2.0 March 1997; features interactive elaboration of queries and multiple vocabularies.
- System test version 2.0.
- Release SERR version 3.0 March 1998; features project profiling.

RM-3B Knowledge Summarization, Analysis, and Visualization (1996)

This work, which is a follow-on to earlier feasibility work, Knowledge Integration Technology (Program Manager's Assistant), will investigate knowledge summarization, analysis, and visualization (K-SAV) methods and tools. These technologies are intended to address both the understanding and the information overload problems by facilitating the interpretation, representation, and content analysis of the large body of unstructured textual information associated with software acquisition and development programs. The focus will be on analyzing the effects of these technologies on risks within a program by applying these tools to the analysis of risk information itself. This activity area will result in products that help clients make better and more informed choices regarding adoption of technologies that support the use of knowledge and information within organizations. This is a cross-cutting technology with risk management as its initial test bed.

Purpose

Knowledge integration technology will improve decision making by program managers. The technology includes the ability to analyze and structure risk data and present that data from multiple perspectives. One form of this technology currently being explored provides a graphical representation of the key aspects of risk resulting from a risk assessment. The graphics summarize not only the critical aspects of risk but also the relationship between these aspects. These relationships may be useful in identifying root causes and interdependencies that can be employed in formulating risk mitigation strategies.

Customers

It is anticipated that there will be a diverse set of potential customers for this technology. Initially the Risk area will provide a test environment for evaluation. Ultimately, the outputs will provide program managers and their staff with guidance on the adoption and use of the technology. In addition, it is expected that the results of this work will be useful to clients interested in developing K-SAV tools.

Collaborators

Collaborators include acquisition and development program managers and their staff.

Approach

The approach will include the following steps:

- Define a framework for representing and assessing K-SAV methods and tools for use in software development and maintenance programs.
- Document the evaluations of approaches by detailing the results of the use of these methods and tools in projects internal to the SEI (e.g. the risk program) and applied to SEI internal data.
- Develop a preliminary structure, procedures, and criteria for making decisions regarding the adoption of K-SAV methods and tools.

RM-4B Software Engineering Information Capture (1996-1997)

This effort builds upon 1995 work of documenting the experience, lessons learned, and value added of building interactive information products (Amore, CMM CD-ROM, World Wide Web, etc.) and associated information product architectures.

This work focuses on collaborating with Motorola in studies of the impact of software engineering information capture techniques and technologies on software cycle time, productivity, and quality. This work also includes completing the lessons learned in developing the current and past set of software engineering information products, refining the existing workshop, and developing new information products (e.g. CD-ROM, World Wide Web, and network accessible digital video [Informedia]) to enable others in the software engineering community to more easily create electronic information products. This revised workshop can then be offered as a public product.

Purpose

This effort will result in outputs that enable more effective capture and delivery of software engineering information by employing information technology products in software development efforts conducted by geographically distributed teams using a variety of representations (video, audio, text, etc.). Whereas the repository and K-SAV work is text-based, this provides for a multi-media approach. The results of this task also enable software engineering practitioners to use and distribute software engineering information relevant to individual projects. Further, it will allow projects to more effectively capture, use, and distribute to SEI customers information relative to ongoing work.

Customers

Potential customers of this work include individual software engineering practitioners and software development teams either co-located or at geographically distributed sites.

Collaborators

Collaborators include the following:

- Motorola's Corporate Software Engineering Research and Development Center
- Carnegie Mellon University Computer Science Department
- Massachusetts Institute of Technology (MIT)
- Advanced Research Projects Agency-Software and Intelligent Systems Technology Office (ARPA-SISTO)
- National Science Foundation (NSF) Division of Information, Robotics and Intelligent Systems
- Microsoft Corporation
- Intel Corporation
- Digital Equipment Corporation (DEC)

Approach

- Continue to investigate the application of Informedia with Motorola in establishing an organizational memory of software project reviews, requirements gathering, experiences, and best practices.
- Document the results of the investigation and lessons learned.

2.3.7 Proposed Add-On Work Outputs

Joint proposals with the Process area are described in Section 1.3 of Volume II.

2.3.8 Related TO&P Activities

Software Engineering Risk Repository

TO&P funding will be sought to pilot test the SERR. The testing will be to address user interface issues as well as the application of the technology in risk management.

Groupware Adoption Guidelines and Services

The primary functions of groupware technologies are to support the non-intrusive capture, efficient sharing, and use of knowledge within an organization. Building on the knowledge integration technology feasibility study, this effort will extend the groupware investigations to include a pilot implementation study. The intent of the study will be to assess the impact of groupware and to evaluate, modify, and finalize adoption practices for groupware technology. The completion of the pilot study and documentation of a service product to support clients in groupware adoption in software risk management and software engineering management generally, are planned as the principal outcomes of this work in 1996.

The potential clients include DoD, civilian government, and commercial suppliers involved in large software development programs. The outputs of this effort will provide program managers and their staff with guidance on the adoption and use of groupware.

Volume II
Chapter 3 Table of Contents

3 Disciplined Engineering

Demand for software-intensive systems (henceforth referred to as systems) is constantly increasing, and these systems are growing in number, size, and complexity. More effective and efficient software engineering practices must be employed to cope with the challenges created by the increasing numbers of large-scale, complex systems. The challenges can be characterized as follows:

- Systems have been built in isolation, resulting in stovepipe solutions even for similar systems that must be maintained.
- Systems are becoming larger; their requirements are not completely known and continue to evolve due to changes in the environment in which they are deployed.
- Systems are deployed in everyday situations and have a critical impact on our ability to function as a society.

Successful practices in mature engineering disciplines are based on the use of system models and architectures, systematic quantification, analysis and prediction of system properties (quality attributes), and decisions on trading off various attributes. Organizations achieve economies of scale by recognizing product families. For unprecedented systems, identification of appropriate capabilities—and experience in their construction—is gained through rapid assembly of prototypes and quick evolution of systems into mature products that meet customers' needs. The quality of products is maintained through rigorous application of analysis and prediction, continuous improvement and refinement through designed experimentation, and feedback from observations of actual systems.

The benefits of a mature engineering discipline are achieved because modeling can be used to guide the development of systems. The same rigor is needed in software engineering; hence, we call our vision of software engineering practice model-based software engineering (MBSE). The vision of MBSE is pursued through three activity areas:

- **product line engineering**: engineering and reengineering of software-intensive systems from a product line perspective; utilizing domain models and architectures to leverage identified commonality and variability in a set of systems
- **evolutionary engineering architectures**: rapid, dependable evolution of systems through flexible, robust architectures and affordable system integration
- **predictive engineering**: predictable confidence in the quality of software-intensive systems through quantitative analysis and through the prediction of quality attributes

Figure 3-1 illustrates, in terms of activity areas, the changes in Disciplined Engineering relative to last year's characterization.

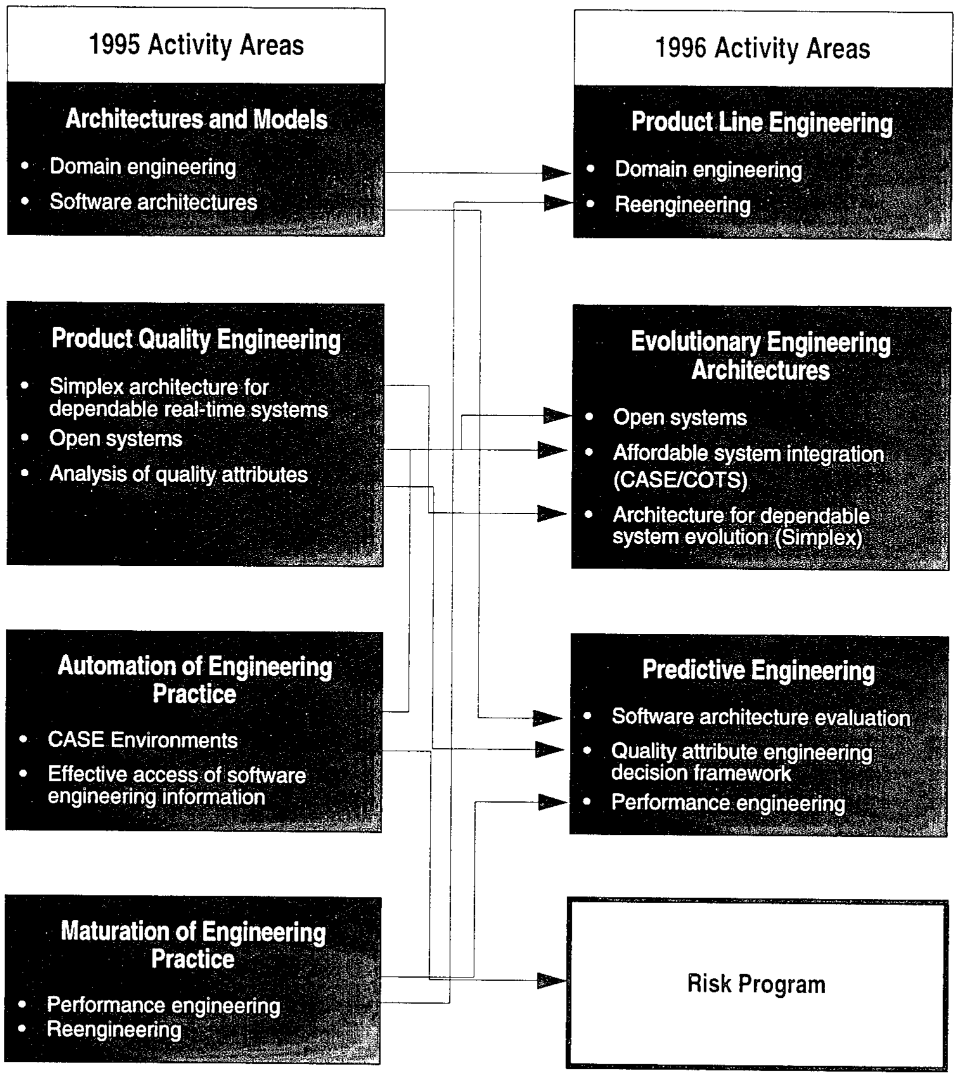
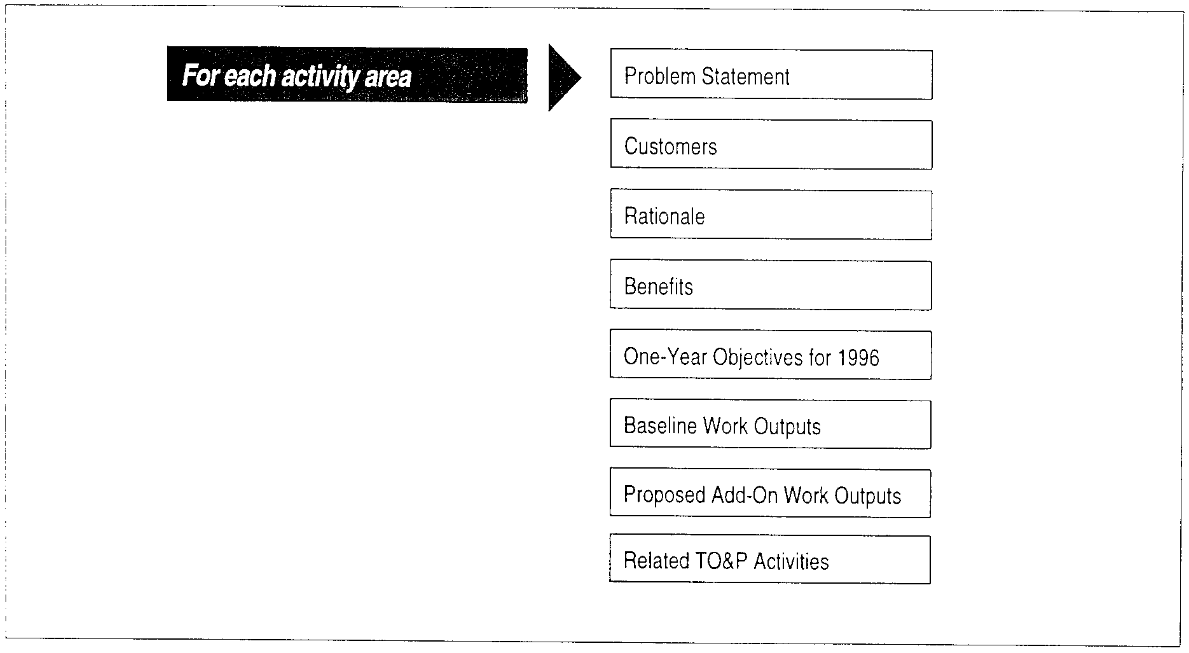

1995 Activity Areas
1996 Activity Areas
Architectures and Models
• Domain engineering
• Software architectures
Product Line Engineering
• Domain engineering
• Reengineering
Product Quality Engineering
• Simplex architecture for dependable real-time systems
• Open systems
• Analysis of quality attributes
Evolutionary Engineering Architectures
• Open systems
• Affordable system integration (CASE/COTS)
• Architecture for dependable system evolution (Simplex)
Automation of Engineering Practice
• CASE Environments
• Effective access of software engineering information
Predictive Engineering
• Software architecture evaluation
• Quality attribute engineering decision framework
• Performance engineering
Maturation of Engineering Practice
• Performance engineering
• Reengineering
Risk Program

Figure 3-1: Mapping of Disciplined Engineering (DE) Activity Areas

In this chapter, the sections for this impact area are as follows:



3.1 Product Line Engineering

3.1.1 Problem Statement

A product line represents the systems, both existing or planned, that address similar customer needs. These systems are generally variations of a theme, such as command and control centers, communications systems, and simulation systems. Product lines and systems within those product lines can be identified and a development process established to support future system development, as well as support evolution of legacy systems within the product line.

Current models of software development and evolution assume a stovepipe approach in which each application is developed and maintained independently of other applications. By focusing on commonalities among systems, a product line approach offers the potential of significant benefit by leveraging existing models, rather than constantly re-inventing the wheel.

However, despite the fact that a number of organizations are moving toward a product line approach, there are no established guidelines or methods.The Product Line Engineering area will address this need by developing and testing approaches for supporting the evolution to product line development. Where available this area uses and transitions existing technology. Some of this technology has been developed and tested by the SEI; technology from other sources will also be used and matured through practical application.

This activity will address the need for product line engineering guidelines to identify product line assets to facilitate software reuse, rapid prototyping, and system understanding for re-engineering. To identify product lines and supporting assets such as a generic architecture, domain model, or components, the effort must include

- an inventory method to enumerate existing systems and potential product lines
- guidelines for product line identification
- rules for boundaries and interconnections among systems and product lines
- plans for current and future evolution of the product lines
- lists of members of those product lines

The success of this activity will help organizations achieve

- core competency and processes for developing applications in the product line
- development and legacy products supportable through base requirements, standards, and common development tools and process environment
- a "platform" or domain-specific software architecture and product line assets. These assets provide common building blocks for the product line.
- clarification of the relationships between products
- a means of explicitly documenting product understanding, i.e., representing the development "culture" as a technology base

3.1.2 Customers

The activity area customers are technology developers, software developers and maintainers, and systems acquisition groups. Customers with an interest in the technology of product line engineering include

- programs sponsored by the Department of Defense (DoD): e.g., Software Technology for Adaptable, Reliable Systems (STARS), domain-specific software architectures (DSSA), Prototech, Comprehensive Approach for Reusable Defense Software (CARDS), the Defense Mapping Agency (DMA), and the National Security Agency (NSA). Some of these programs are nearing completion and the activity will seek to transition results and lessons learned. Others are in the early stages, and understanding product line engineering will influence the program goals and their approaches.
- development organizations, both government and industry, building competency in architectures and models for emerging systems and legacy systems undergoing reengineering
- the DoD acquisition community, including policy organizations such as the DoD Software Reuse Initiative, wishing to evolve to product line approaches

Product line engineering will also serve customers in the following domain areas:

- training and simulation
- armament
- command, control, communications, and intelligence
- air traffic control
- telecommunications
- tool building

Product lines should contribute to the effective management and development of products within an organization. However, the management structure for these product lines may cross existing management boundaries (program executive officer, etc.). Achieving a true product line approach may require breaking down organizational barriers.

3.1.3 Rationale

The escalating costs of developing and maintaining software are well documented. One reason for such rapidly increasing costs is that applications, in most cases, are developed and maintained according to the standard stovepipe approach. Applications tend to be focused on specific organizational and customer groups. As organizations change, the applications need to undergo additional changes to reflect the changes in the organization.

The product line approach offers specific advantages over a program-oriented development strategy. The cost and time of developing and evolving systems are significantly reduced. Organizations build core competencies. Products are engineered by recognizing changes in base requirements. While creating and managing the product lines incurs some cost, the advantages to a customer organization in reduced cycle time and cost and in increased quality justify the effort to identify, build for, and organize for product line development. In addition to addressing development problems, the product line approach has potential for offering substantial benefit to reengineering of existing systems. When legacy systems are viewed as examples of product lines, rather than as individual, unique systems, the application of a specific reengineering activity is broadened considerably.

Product lines must be managed similar to a software development effort. There is an analysis phase, followed by architecture, design, and implementation. The benefits will accrue only when the definition of product lines is accompanied by strong management support for further analysis, design and implementation of domain or product assets.

Existing domain analysis methods cover only a small portion of the effort required to create, sustain and manage product lines. STARS[1] characterizes the process as including creation, management and utilization of assets as well as a formalized improvement process to learn from experience. CARDS[2] promotes a domain-specific reuse and software engineering approach linking product line resources from a reuse asset base, fed by domain (or product line) engineering, to application engineering. The product line resources are utilized in developing new applications, and reflection on their application must lead to improvement, change, or retirement. In this way, the resources to support product lines will be sustained in a life cycle, apart from that of the applications supported by the product line. Only a strategic approach that focuses on long-term investment and sustainment will achieve many of the product line benefits.[3]

The SEI has been a pioneer in the use of a product line approach. Several pilot efforts (which are described in the section below), including the Portable Reusable Integrated Software Modules (PRISM) program and the air vehicle simulator effort at Wright Patterson Air Force Base, have shown promise for significantly reducing the costs and time needed for system development and maintenance. Because of these successes, the expansion and consolidation of the concepts offer promise for the broader software engineering community.

The Air Force, for example, has initiated two well-documented product lines: command centers and air vehicle simulators. The generic command center developed by the PRISM program offers the basis for creating multiple command centers from a collection of assets. These assets include commercial off-the-shelf software (COTS) tools as well as wrappers to provide a common interface to similar tools. From the perspective of Air Force management, PRISM constitutes an investment in a product line approach for customers who need command centers. PRISM products are currently being applied in several prototype efforts.

As another example, the air vehicle simulator effort at the Aeronautical Systems Command (ASC/YT) (Wright Patterson Air Force Base) has developed a standard architecture for related simulator systems. Simulators have generally employed standard interfaces to hardware components that provide cockpit displays and platform motion. The software architecture provides a single structure to support analysis for partitioning simulator requirements and coordinating simulator interactions. Further efforts are aimed at establishing procurement qualifications for companies that produce systems in this product line.

1. Creps, Richard E; Simos, Mark A.; & Prieto-Diaz, Ruben. *The STARS Conceptual Framework for Reuse Processes*, Technical Report, 13 November 1992. Department of Defense.

2. Martin, Lorraine. "Comprehensive Approach to Reusable Defense Software." Briefing Materials. 1994.

3. Platt, Lewis E. Hewlett-Packard Presentation. 24 August 1993.

3.1.4 Benefits

A recent study has concluded that the product line approach "allows an organization to derive maximum benefit from software process improvement."[4] There is also a strong correlation between product lines and process maturity. This product line approach focuses on efforts to create assets for future software development. These assets constitute an investment for future work. By creating assets that support development of multiple systems in a product line, an organization

- reduces cycle time and cost
- eliminates redundancy
- reduces risk
- produces applications from common assets
- improves quality of its delivered products
- manages its legacy assets more efficiently
- evolves a common marketing strategy
- establishes core competency around strategic business interests

To achieve these benefits, product lines should be organized around ongoing business activities. These activities will address specific mission areas of interest: command and control, aircraft simulators, etc. A collection of product lines can then be organized into a product family managing multiple lines and the product line platform or assets that go into building members of a product line. Assets that cross product lines take advantage of "economies of scope," a benefit that comes from developing one asset use in multiple contexts. Figure 3-2 summarizes the concepts that support product line identification.

Concept	Definition	Example
Assets	A component that may be used to build a product including architectures, tools, COTS, GOTS, etc.	Mapping system, PRISM C^2 architecture
Product line	A business area that produces and markets a collection of similar products.	Air defense, mission planning, aircraft flight crew simulators
Product family	Common assets and architecture for a collection of similar products.	C2, surveillance systems, simulators
Products	A system delivered to a customer. A result, instance, or element of a product line or family.	JSTARS, B-2 flight crew simulator, B-2 maintenance simulator

Figure 3-2: Product Line Concept Definitions

4. Besselman, Joe & Rifkin, Stan. "The effect of software process improvement on the economics of procurement." SEI Symposium, Pittsburgh, Pa., August 1994.

Figure 3-3 illustrates the desired impact of this activity area.

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Domain engineering	• Best practice report in place • Standard training in place	• Role of product lines provided in guidebook • Successful pilots of domain engineering	• Guidebook and training capability transitioned	• Training offered by two or more vendors • Used by community • Institutionalized
Evolution of legacy systems and system understanding	• Categorization of key issues • No common terminology	• Reengineering taxonomy for system evolution • Reengineering decision matrix	• Adoption handbook • Business case	• Used by community • Institutionalized • Reengineering approach shows up in proposals and requests for proposals
Business strategies for MBSE	• The SEI is defining a method for making strategic decisions • Community interest	• Examples from pilots • Standard training in place	• Best practice report (e.g., decision aids) • Community acceptance	• Organizations implementing MBSE • MBSE seen as corporate reengineering tool

Figure 3-3: Trends in Product Line Engineering

3.1.5 One-Year Objectives for 1996

- Evaluate product line approaches in terms of suitability for transition and promote product line concepts and methods as a key technology for software engineering. Obtain community acceptance of this approach.
- For the community, define and provide examples of product line approaches and their relationship to legacy system evolution. This will give practitioners an understanding of the relationship between product line engineering and reengineering.
- Promote within the community a common understanding, interchange, and identification of areas for additional research. Through this role, we will identify the barriers to a paradigm shift to product line development and means to address the current lack of technology to support product line engineering.

3.1.6 Baseline Work Outputs

This section describes the baseline work outputs for the activity area related to product line engineering. These outputs were approved for 1996 through the basic funds allocation process. Appendix A lists all baseline and proposed add-on work outputs.

DE-1B Business Strategies for Model-Based Software Engineering (MBSE) (1996)

This activity results in guidelines and checklists of considerations for managers. The guidelines will enable project and other managers to make business decisions on whether an organization should engage in architecture-focused technology, including domain-specific reuse, product line engineering, and reengineering. The guidelines will also include training in a method to make these decisions.

Purpose

This activity focuses on business case analysis for software engineering based on software architectures and models. The results from this activity, together with the technology transition assessment and planning method, will provide essential components for bridging the gap between early adopters (trial/pilot use) and the late majority (adoption/institutionalization).

Customers

Potential customers for this activity include DoD and commercial organizations needing a decision support framework for product line and reengineering assessment. This task is already working with Army and Air Force customers and, through affiliates, with industry.

Collaborators

This activity involves collaboration with experts at Carnegie Mellon University (Graduate School of Industrial Administration, the Heinz School of Public Policy and Management) as well as the SEI's customers in industry and government who have expressed interest (including those involved in STARS and CARDS). Industry has expressed a strong need for and interest in collaboration to support successful introduction of domain application engineering based on a domain model and architecture. The SEI is in a unique position, representing neutral ground for companies to foster interdisciplinary collaboration (including business, industrial administration, and policy making on the Carnegie Mellon University campus as well as at the University of Pittsburgh).

Approach

The method for evolving business strategies for MBSE includes techniques for

- identifying the domain of core competence and related core assets in an organization
- baselining organizational, investment, and development strategies for systematic software engineering based on architectures and models (reuse, product line engineering, reengineering)

- assessing suitability of systematic reuse and rapid application development based on domain models and architectures for a product group
- developing a business case for the above, and techniques for evolving an organizational strategy for transition engineering to an architecture focus in software engineering

During 1995 this task produced

- Reuse Opportunity Analysis Method (ROAM) description
- case study summary data base
- method training
- pilot identification

During 1996, this task will continue with

- pilot studies
- ROAM Guidebook
- revised training

Assuming continued support from customers and collaborators, potential follow-on work is

- 1997 - case study survey, MBSE database
- 1998 - MBSE decision business model

DE-2B Domain Engineering Guidebook (1996)

This task (described in last year's plan, *SEI Program Plans: 1995-1999*, as the MBSE Guide) will develop guidance to support organizations using domain engineering to transition to a product line engineering approach.

Purpose

As companies in the electronics industry realize the importance of software to the success of their products and of time-to-market, they are seeking new ways to identify commonalities, increase productivity and evolve to product line development for software. This task will develop guidance to support organizations using domain engineering to transition to a product line engineering approach.

Customers

Organizations that can develop products and services to support the evolution to a product line approach to software development must

- recognize that software is greater than code
- leverage from previous efforts yielding “economies of scope” (i.e., the ability to use the same product in multiple contexts)
- have a planning and control function in place to track actuals against plan

We are currently working with customers from the Air Force, Army, and industry to transition preliminary versions of this work.

Collaborators

This task will build on actual experience in working with SEI customers such as the ASC, Communications-Electronics Command (CECOM), the Electronics Systems Center (ESC), and commercial organizations. It will also build on previous work in domain engineering with commercial customers. This work established the MBSE framework relating the technology of modeling, domain engineering, and application engineering. The SEI has already developed a training program for introducing domain engineering and a guidebook for the domain engineering process.

Approach

During 1996, this task will produce

- a domain engineering trainers’ guide to support organizations wishing to transition from pilot to institutionalization of domain engineering
- a revised domain engineering guidebook relating MBSE and domain engineering to product line approaches. This will build on our on-going TO&P efforts with ASC and ESC.

Proposed follow-on work is

- MBSE comparative study (1997)
- MBSE support environments (1999)

DE-3B Report on the State of Program-Understanding Technology (1996)

The output of this task is a framework and method for evaluating program-understanding technology and making appropriate choices in its use on real projects.

Purpose

This activity will evaluate the state of program-understanding technology, both research technology and advanced commercially available products.

Customers

Customers include NUWC and other government and commercial customers trying to improve the management and evolution of legacy systems.

Collaborators

This work will include collaboration with a resident affiliate from the Naval Underwater Warfare Center (NUWC).

Approach

This task contributes to the advancement of promising trends in system understanding technology such as program-understanding technology, abstraction to higher levels of analysis, alternative scenarios based on user perspective, and understanding of tasks most frequently used by system maintainers. A potential new task for system understanding will build on the results of this task.

During 1996, the task will produce

- a framework for program-understanding technologies
- trials of the framework on samples of current technology

Proposed follow-on activity is to develop a roadmap for system understanding technology.

DE-4B Domain Analysis for System Understanding (1996)

This task is based on a feasibility study, which was done as a part of the guide to best reengineering practice, a task in 1995. In 1996 this work will produce a methodology report documenting the application of domain analysis techniques to system understanding for reengineering. The report will examine a number of prominent domain analysis methods (Organization Domain Modeling, Feature-Oriented Domain Analysis, Defense Information System Agency Domain Analysis Method) and contrast their applicability.

Purpose

The goal of this task is to add formalization to the domain analysis and domain engineering methods currently in practice at the SEI and elsewhere. Of primary importance are

1. defining the types of features which make one system different from other related systems within a product family
2. understanding the interplay within system formation: development environment, user environment, concept of operations, system environment, product attributes

By understanding the characteristics of domains that cross product families, the causes of differentiation within those domains, and the rules of composition of domains into products, this task will provide guidance for supporting system understanding.

Customers

Customers include the NUWC and other government and commercial customers trying to improve the management and evolution of legacy systems.

Collaborators

This work will include collaboration with a resident affiliate from the NUWC.

Approach

System understanding is a key element of reengineering. It reflects the process of creating and maintaining an understanding of a system through analysis, elicitation, and capture. System understanding relies on a variety of sources of information. These may include:

- artifacts related to the system under reengineering
- knowledge of builders, users, or maintainers regarding the system
- evolutionary history of the system

This information, which is of use in reengineering, is also a product of *domain engineering*. This task will examine domain engineering as a process for identifying capabilities and design of a class of systems (i.e., a domain), both those which are common to all members of the class as well as those which differentiate the members. The task will identify how a domain model may be used to understand where product binding decisions have been made or how to design for product change. A *domain design*, another product of domain engineering, represents a generic design for applications that are part of the class. This task will also examine ways in which the domain design can further promote system understanding and support the evolution of legacy systems.

The resulting method will provide a structure for analysis, elicitation, and capture of knowledge and experience related to the system undergoing reengineering. It will also support decision making regarding the evolutionary migration path from the existing to the desired system.

3.1.7 Proposed Add-On Work Outputs

This section describes the proposed add-on work outputs for the activity area related to product line engineering. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

DE-1A Domain Analysis and System Building (1996-1997)

This task will produce a guidebook for using the products of domain analysis in both architecture selection and application engineering through generators. The guidebook will discuss methods for capturing domain information, such as object-oriented analysis, and the use of common vs. application-specific information in making architectural decisions. This work will contribute to DE-2B Domain Engineering Guidebook (1996).

Purpose

This work builds on our understanding of domain analysis to establish the relationship between domain analysis/engineering and system building. The task will produce methods for mapping domain models to

- generic design, components and generators
- software requirements for applications

Customers

The current customer base, which includes DoD development labs, DoD contractors, and commercial development organizations are looking for support in this area and will benefit from the work. For example, we are currently working with two DoD customers whose goal is to develop a generic design derived from domain models. A commercial customer is building a domain model to be used by their marketing organization. This work will benefit all of these customers by providing methods for relating the products of domain analysis to subsequent development activities.

Collaborators

A member of this activity area is collaborating with staff from MITRE, Loral, and ESC to understand the development of assets in support of application engineering. This task will also involve collaboration with internal architecture efforts and will seek new TO&P or affiliate support.

Approach

The SEI has promoted the link between key technology areas (e.g., requirements and architecture) and domain engineering. More recently, our work with customers has established specific ties between domain models and requirements engineering during the application engineering process. The project has also piloted the connection between domain models, architecture selection, and generic design for classes of systems. An add-on task can formalize those connections. Two activities comprise this task:

Activity 1: Domain Analysis and Architecture Selection

This activity will examine the connection of domain analysis to architecture for a family of systems. The Feature-Oriented Domain Analysis (FODA) method looks primarily at "user-visible" aspects of a domain. The term user-visible includes not only the human user but also system users where applications within the domain or product family interact with other systems. The more complete aspects of a domain will include

- operational features
- system configuration and component connections

- development environment
- user environment (including system mission)
- quality (non-functional) attributes

These are aspects that affect the selection of architecture for the domain. Some of these aspects are already captured under the "context features" of FODA. The add-on task will formally describe all aspects contributing to product line understanding and building.

Activity 2: Domain analysis and application generators

The SEI has piloted a domain model front-end to an application generator. This pilot was based on the "001" tool code generation technology. Our business strategy effort is comparing the flexibility of component techniques with the ease-of-use of application generators. This activity will examine generator technology to determine when it pays to build an application generator and when component technology makes more sense. The activity will also look at the front-end activities of identifying the requirements for the generator and look at where domain analysis can add discipline to generator development and improve the interface.

DE-2A Disciplined Evolution of Legacy Systems (1996-1997)

This task is developing systematic techniques for large-scale reengineering. In 1996 this task will produce

- a preliminary taxonomy of incremental reengineering transformations that can be used to describe and characterize a broad spectrum of reengineering efforts and that form the core elements of the maintenance and evolution scenarios developed in the first phase
- a preliminary framework for the decision-making processes associated with the incremental transformations that are part of the reengineering taxonomy
- a technical report on maintenance activities, including: 1) a summary of the tools, techniques and sources of information that are particularly effective in addressing specific maintenance activities, and 2) a decision framework to help software engineers evaluate reengineering decisions from an architectural perspective
- a report on architectural evolutionary patterns and the state-of-the-practice in dealing with those patterns

Purpose

Specific types of maintenance problems and scenarios (e.g., porting to new operating systems, replacement of a system component such as a database, performance enhancement, etc.) will be identified, along with maintenance techniques that are effective in addressing them. Particularly critical are those techniques that assist in the understanding of software systems (i.e., program understanding), both at the intra-component and inter-component (architectural) levels. This work will be used to develop a decision framework to support and guide organizations in the disciplined evolution of their legacy systems.

Customers

Customers are the DoD, other government organizations, and commercial organizations with large-scale existing legacy systems, such as the DMA and the Federal Aviation Administration (FAA).

Collaborators

Collaborators are the reengineering community, the Software Technology Support Center (STSC), and government contractors.

Approach

The reengineering field has traditionally focused on a relatively narrow set of issues to support disciplined evolution of legacy systems. During 1995 we prototyped a categorization for understanding reengineering issues such as the operational system, system integration/test facilities, and software development/maintenance facilities. Each step in this categorization requires some form of technical, economic, or programmatic decision-making, such as establishing the reengineering goals and objectives, eliciting and validating requirements, reverse engineering of the legacy system, performing impact assessment, and forward engineering of the system.

This activity builds on the categorization previously developed and provides the software engineer with a practical means for describing the tasks to be performed on a reengineering project, and a systematic approach for making sound reengineering decisions. This task will define a set of incremental transformations that are suitable for describing a wide range of evolution efforts. This would allow a reengineering practitioner to define a customized process to accomplish unique reengineering project goals and objectives by selectively combining and appropriately sequencing individual transformations.

The task develops a set of decision-making criteria based on the taxonomy of incremental reengineering transformations. These criteria will guide the reengineering practitioner in the decision-making processes associated with each incremental reengineering transformation, thereby supporting disciplined evolution. It will also identify and categorize specific software system evolution scenarios (e.g., porting to new operating systems, replacing components, enhancing performance, etc.). This categorization will enhance our knowledge of the manner in which systems evolve. Tools and techniques that are particularly effective in program understanding, as well as widely available resources and centers of expertise, will be identified.

As proposed follow-on work in 1997 and 1998, the preliminary taxonomies will be refined and codified, with a long term result leading to a roadmap for the Disciplined Evolution of Legacy Systems.

DE-3A Use of SEI Repository for Developing Case Studies in Reengineering (Feasibility Study) (1996)

The deliverables associated with this task include

- a set of case studies of reengineering, including an analysis of common themes and issues
- an analysis of the effectiveness of using the SEI repository, together with lessons learned for both SEI and non-SEI participants
- recommendations for future research in reengineering

Purpose

This activity is directly connected to the proposed SEI repository since it develops a set of case studies on reengineering to place in the repository. This will enable us to gather and analyze information about the experiences of other organizations in the discipline of reengineering. The task uses the repository as a demonstration of how an individual area can use this resource, and how other areas within the SEI can use such a general resource for the specific needs of a given activity.

Customers

The repository will benefit customer organizations with legacy system and reengineering needs by providing case studies, exemplary data, and metrics.

Collaborators

Collaborators will be organizations that provide case studies, such as the National Aeronautics and Space Administration (NASA), the NUWC, and the FAA.

Approach

This task will involve a joint effort with the SEI areas of Risk and Disciplined Engineering. Risk will provide training on the tools and techniques of the SEI repository. Disciplined Engineering will perform the technical work and analyses.

The analysis will include a set of structured on-site case studies on reengineering projects. These projects will be selected to provide maximum insight into the other proposed tasks in program understanding and software evolution. The case studies will address a number of issues, such as 1) architectural understanding, 2) programming language and structure of code, 3) dependencies on idiosyncratic language or platform features, and 4) the cumulative effects of continuous maintenance rework.

The individual cases will be entered into the repository, and analyses will be performed to understand abstractions from the data. Tools and techniques that have been useful in a particular set of circumstances will be analyzed in terms of how they can be further applied, or in terms of their specific limitations. We will also pay particular attention to problems and failures as lessons to avoid in the future.

DE-4A Impact of Object-Oriented Technology (Feasibility Study) (1996)

This work will produce in 1996 a single report or a series of white papers which will identify for the community the essential issues related to the impact of object technology on software engineering practices as well as to how other technologies (for example, software architecture, domain analysis, reuse, open systems) affect object technology. It will also provide a preliminary survey of available resources documenting study or experience with these issues.

Purpose

The purpose of this task is to establish an objective source that provides guidance to customers relative to the total impact on software development from a shift to object technology. Another purpose of this feasibility study is to provide a focus for potential follow-on work.

The rapid and widespread adoption of object-oriented languages and techniques has surpassed even the most optimistic estimates. The move to object technology by a software development organization involves much more than a language shift. The impact on the organization's software engineering will be experienced not only in language but in analysis, design, architectures, reuse, testing, tools, metrics, and management. In short all existing software engineering practices may need to be reexamined and new ones embraced.

To date, most organizations have not understood or grasped the full impact on their practices and organization due to the immaturity of the technology, the lack of experienced personnel, the immaturity of the process infrastructure, and the need for up-front technology investigation investment. While the promise of object technology is great, unless the ramifications on other software engineering practices are well understood, for many organizations (including the government as Ada 95 is embraced) the risk is also great.

Customers

Though the object movement is pervasive, thorough understanding of the technology and its ramifications is not. The customers are as broad as the object-oriented development community desiring improvements in their ability to integrate new object technology.

Collaborators

The SEI has been very active in the object technology community and has a representative on the Object-Oriented Programming Systems, Languages and Applications (OOPSLA) Executive Committee. Potential collaborators include other members of the committee. These connections will be leveraged and utilized in surveying the community and validating the eventual evaluation.

Approach

We will interview and survey software developers and object technology experts to build on existing understanding of the current state of object technology, especially as regards the relationship with other software engineering practices and technologies. We may hold a workshop to facilitate this activity.

We will identify the relationship and impact of object technology on other technologies that are examined by Disciplined Engineering as contributing to the advancement of the software engineering practice. We will distribute this identification and evaluation in written output to the community.

This approach relies upon interaction with the software development community and exploitation of existing ties with the object community.

3.1.8 Related TO&P Activities

Several ongoing TO&P activities are related to this area:

- The Tank and Armament Command (TACOM) is applying our Reuse Opportunities Analysis Method to identify product lines and reuse strategies. The Communications-Electronics Command (CECOM) is also considering this approach.
- The ESC is using a product line approach to restructure its acquisitions approach.
- The NSA is applying our research on maintenance scenarios to the ongoing maintenance of legacy systems.
- We are working with commercial companies who are pursuing the product line approach.

3.2 Evolutionary Engineering Architectures

3.2.1 Problem Statement

The high costs and low productivity of many software development and maintenance projects are well documented. At the same time, budgets for software development within both government and industry are coming under increasing scrutiny. A number of organizations, including the U.S. government, realize that one way to reduce the burden of development and maintenance is by constructing systems from collections of existing components. Furthermore, many pre-existing components can be purchased as commercial off-the-shelf (COTS) products from third-party suppliers. This leads to a vision of software development as primarily the task of selection and integration of existing components that conform to industry-wide standards. Hence, such systems are "open" in the sense that they permit easy addition of new components that support those standards.

However, many barriers currently block the realization of that vision. Typical issues being faced include

- misunderstanding and confusion about the technologies and techniques that support "open systems" and COTS-based approaches to system integration and evolution
- lack of understanding of the role of interface standards in the acquisition of COTS-based systems
- acquisition guidelines and regulations that preclude or hinder the use of COTS components when developing a new system
- lack of visibility and access to COTS component internals to establish key component qualities such as security, reliability, and performance
- system instability during maintenance and evolution of the system due to the frequency, variability, and inconsistency of periodic new releases of COTS components
- loss of schedule control during development and maintenance when relying on another developer's products and on interface standards controlled by external organizations and agencies
- lack of effective software architectures and tools that are designed to support the selection, integration, and evolution of component-based systems

Once developed, maintainers of COTS-component-based systems have to deal with asynchronous vendor-driven component upgrades to keep up with technology trends. The system evolution stakes are particularly high for mission-critical systems where downtime is expensive and failures may lead to disaster. For example, recently the Electrical Power Research Institute (EPRI) reported that although new microprocessor-based controllers are inexpensive and have advanced capabilities, the risk of upgrading power plants is so high that most plants are forced to keep obsolete equipment. In fact, the average equipment age is approaching 28 years old. Owing to the increasingly frequent failures of aging equipment, the cost of generating power has gone up in recent years even though fuel costs have come down.

Similar situations can be found in many other industry segments and in defense systems. The lack of effective technologies to support the evolution of mission-critical systems will lead to the erosion of the competitiveness of U.S. industry and the erosion of the technological superiority of U.S. defense, especially when many fewer new defense systems will be developed in the post cold war era.

The software engineering community requires practical advice, techniques, and heuristics in making the move toward COTS-based, open systems approaches to system development. In particular, help is required concerning how cost-effectively to acquire systems assembled from COTS components, how standard interfaces (i.e., open systems) can help achieve the goals of portability and interchangeability of components, how to evolve systems composed of COTS components safely and efficiently, and what techniques and mechanisms can be used to integrate the components to achieve the system qualities they require. Also required is a reference architecture for open, component-based software systems that will

- support the integration of third-party products to combine complementary solutions and foster competition leading to accelerated innovations
- ensure the dependability and safety of such horizontally integrated computing systems
- support the replacement of COTS components and the insertion of new technologies into deployed systems quickly, reliably, and safely

3.2.2 Customers

There is a wide range of customers in this area. Examples include

- policy makers within government and industry who require help in gaining a deeper understanding of the programmatic, technical, and economic issues associated with building and evolving systems constructed of COTS components
- project managers responsible for acquisition and introduction of systems within an organization, needing effective evaluation and selection techniques, and better adoption processes
- COTS-component vendors and integrators who would like to enhance their components and integration strategies to meet the needs of their customers
- project members who build and evolve large, complex application systems and would like better tools and techniques to use when constructing them from COTS components

3.2.3 Rationale

Before systems integration of COTS components can have more predictable, widespread effect on the software engineering community, many barriers must be addressed, and solutions developed.

A workshop held at the SEI in January 1995 brought together a number of project managers responsible for acquisition of systems with technical experts in the area of COTS integration and open systems to discuss the problems being faced in this area by the software engineering community. Feedback and discussion during that workshop identified a number of topic areas that require much greater attention in the future:

- reports of lessons learned by system integrators to help others see what worked and what didn't
- in-depth analysis of a range of technical issues connected with describing appropriate system architectures, interface standards, and so on
- investigation of how to measure system qualities (e.g., usability, maintainability, and portability) for systems composed of COTS components
- examination of the impact of changes in roles and responsibilities associated with the move to this new way to develop and maintain software
- describing approaches to manage the evolution of systems containing COTS components

Similarly, during the August 1994 Workshop on Dependable and Renewable Industrial Systems, sponsored by the National Institute of Standards and Technology (NIST), people identified that a fundamental problem affecting U.S. industry competitiveness is the inability of existing industrial computing architectures to introduce new computer system technologies into existing plants easily, quickly, safely and reliably. This problem has fundamental implications for U.S. industry's productivity, agility, and quality since these properties are a function of the technologies used by the business. Indeed, the current architectures have a built-in obsolescence effect. For example, Honeywell pointed out the difficulties in adding hardware or software functions into existing plants. They noted that it took more than 10 years to make effective improvements in critical manufacturing processes in a large plant. This means that in spite of the fact that the U.S. is the acknowledged leader in technology innovations, the impact of innovation is not materialized quickly.

These findings are consistent with a number of other studies that have taken place over the past few years. They all point to the need for leadership in the community and to the importance of gathering current best practices, and of supporting standards organizations and policy makers in their efforts to direct the community to more coordinated and consistent ways of building and evolving COTS-based open solutions.

3.2.4 Benefits

Considering software development as a process based on assembly and evolution of collections of existing components is important because it raises the possibility of a number of gains and improvements. These include

- reducing software development costs as a consequence of constructing large parts of a system from existing components that have been previously developed and tested

- reducing the expense of maintaining and evolving software by delegating maintenance of COTS components to their suppliers
- improving reuse across programs through reducing individual, purpose-built, “stove-pipe” software development in favor of larger purchases of existing products for consistent use across the whole organization
- keeping deployed systems modern through easy, reliable, and safe component replacement and technology insertion
- promoting a competitive component marketplace in many technical domains with the result that system integrators will be able to choose from a range of third-party-supplied components

Figure 3-4 illustrates the desired impact in this activity area.

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
COTS integration	• The SEI has extensive experience regarding COTS integration in the CASE domain • CASE testbed in place	• Extended experience in two further domains • Best practice guidebook developed	• Able to reliably predict impact of different COTS integration strategies	• Use of best practice guides • Used by community • Institutionalized
Open systems	• Pilot course developed • Categorization of key issues • No common understanding	• Open systems risks identified • Open systems handbook	• POSIX standards completed • Open systems business case established	• Attendance at courses • Institutionalized • Improved use of concepts in proposals and requests for proposals
Safe evolution of complex systems	• Simplex demonstration available • Wide community interest	• Pilots applied in industry • Extended application investigated	• Measurable benefits experienced • Community acceptance	• Increasing number of users • Cost effective, predictable COTS use in real-time systems

Figure 3-4: Trends in Evolutionary Engineering Architectures

3.2.5 One-Year Objectives for 1996

- Continue existing initiatives in improving people's understanding of open system issues. The notion of an "open system" has received wide currency. However, interpretations of this concept vary, with a great deal of misunderstanding and confusion. In 1996 we will build on our existing investment in an open systems course for program office personnel to deepen the understanding of open systems concepts within the software engineering community and to provide practical guidance for acquisitions of secure systems.
- Explore COTS integration mechanisms and techniques. Existing work at the SEI over the past few years has looked at data-oriented techniques for the integration of computer-aided software engineering (CASE) tools. In 1996 this work will be expanded to look at event-based mechanisms (e.g., message passing systems) for integrating CASE tools. Also, the techniques examined in the CASE domain will be applied in other application domains where COTS components are integrated.
- Improve approaches to safe evolution of complex real-time systems. An existing demonstration system developed at the SEI illustrates a technique for the safe evolution of complex real-time systems that include COTS components. In 1996 this demonstration will be enhanced to consider more directly the evolution of distributed mission-critical and discrete sequential control systems.

3.2.6 Baseline Work Outputs

This section describes the baseline work outputs for the activity area related to evolutionary engineering architectures. These outputs were approved for 1996 through the basic funds allocation process. Appendix A lists all baseline and proposed add-on work outputs.

DE-5B Open Systems Standards (1996)

The output of this work is an approved POSIX.21 standard.

Purpose

As POSIX.21 WG chair, the SEI will lead this open system standard draft through the balloting process. The SEI has provided technical leadership in the POSIX standardization effort by being the POSIX.21 WG chair since the group's inception.

Customers

Customers are all those in need of an application program interface for real-time distributed processing.

Collaborators

Collaborators are other government, industry, and academic members of the .21 WG.

Approach

This effort is a continuing effort from 1994 through 1995 and is expected to terminate in 1996, with the standards document going into ballot in that year. During 1996 we will continue to provide the WG chair and as such provide technical contributions to the standards document. Because this is a consensus-based standardization process it is difficult to enumerate contributions other than "leadership" for the approach, but one example of major SEI contribution in 1995 has been a draft formal specification of the standard, which will improve validation and balloting success. This effort has been funded through a combination of basic and TO&P funds.

DE-6B Open Systems Evidence Module (1996)

The output of this work is an evidence module for inclusion in the existing open systems course, *Open Systems: The Promises and the Pitfalls.*

Purpose

Additions will be made to the course in the form of evidence of open system deployment, strengthening the validity of the assertions and advice in the course.

Customers

Students of the course—personnel in the DoD and other acquisition agents—will be the primary customers for this new module. In addition, the information will eventually be incorporated into briefings and other course companion products, so consumers of those also become customers of this work.

Collaborators

Collaborators in this work include those programs that will be contacted for data that pertain to this line of investigation. There is also potential sponsorship from DoD programs interested in the enhancement of the basic course. In addition, use will be made of those from other projects within Disciplined Engineering who have expertise in the formulation of business cases and return-on-investment arguments.

Approach

Building on the work of 1995, we are in the process of collecting and understanding the successes and lessons learned in applying open systems approaches to real DoD programs. In 1996 this information will be analyzed and used in enhancing the course, *Open Systems: The Promises and the Pitfalls*, in two possible ways. One is to develop a new course module that will provide quantitative information that can be used in the development of business cases for open systems. The other is to use the acquired information to verify or correct assertions and advice that are given in the course materials, based on the strategies and approaches used by real programs.

DE-7B Roadmap for Environment (Integration) Technology (1996)

We will provide a handbook for integrators of CASE environments. The handbook will contain guidelines that describe methods by which CASE integrators can effectively and efficiently integrate CASE tools acquired from different vendors.

Purpose

This task will provide guidance to software engineers who are required to make choices about the selection and integration of CASE technology. Currently, there are many possible strategies and technologies that can be selected to accomplish CASE integration. This task will result in practical guidance that will enable software engineers to navigate their way through the many possibilities.

Customers

Customers are all organizations in the DoD and elsewhere who are struggling to effectively use collections of CASE tools.

Collaborators

This task is being carried out with the help of a wide range of members of the CASE community (vendors, integrators, users). In addition to informal collaboration and communication with this community, the SEI will attend and host workshops with CASE vendors, integrators, and users.

Approach

Building on the work of 1995 (and previously), we are in the process of understanding and assessing current CASE integration technologies and approaches in the form of a roadmap that describes the current CASE environment landscape. During 1996 a set of guidelines for CASE integrators will be developed based on the lessons learned from case studies and analyses that take place during 1995.

DE-8B Software Engineering Environments Technology Evaluation, Integration, and Measurement (STEIM) (1996-1997)

The outputs of this activity are quantitative results regarding the effective use of integration technologies based on hands-on testbed investigations. This task continues as outlined in the original proposal in the 1995 plan, *SEI Program Plans: 1995-1999*. Its generalization to COTS is occurring naturally as evidenced in our collaboration with the NIST Manufacturing Engineering Laboratory (NIST MEL) on the common object request broker architecture (CORBA) object request broker (ORB) experimentation.

Purpose

Quantitative data regarding the effective use of integration technologies will enable software engineers to make intelligent decisions concerning the engineering trade-offs that must be made. Community involvement will be sought in carrying out this task, providing the benefits of a wide body of experience to this important collaborative initiative.

Customers

Customers are the software engineering community, but particularly managers selecting CASE technologies who require return-on-investment data, and technology integrators requiring predictable integration mechanisms and strategies.

Collaborators

Collaborators are the NIST Computer Science Laboratory (NIST CSL), NIST MEL, the Advanced Research Project Agency (ARPA) research community, and (potentially) SEMATECH.

Approach

In 1995 a significant initiative was begun aimed at identifying and measuring the effectiveness of various CASE integration strategies and technologies. By the beginning of 1996 we hope to have a prototype measurement approach that has been validated on a number of case studies. During 1996 we intend to extend this work in two directions. First, we will pilot this approach with a number of organizations to show real, practical benefit from the methods defined. The approach itself will be refined and packaged for ease of transition into a number of software engineering organizations. Second, we will expand our investigation to system integration of COTS integration strategies in general, based on our experiences in the CASE domain. This work is intended to be an on-going initiative that continues through 1997.

DE-9B Dependable Real-Time Systems Handbook (1996-1997)

The output for this activity is a handbook for practitioners on the development of dependable and evolvable real-time systems.

Purpose

This handbook will provide a reference architecture, its associated fault coverage model, and a collection of quantitative methods to help ensure predictability, reliability, and safety of mission-critical systems. Special emphasis is given to the techniques, to support the safe replacement of COTS components and the safe online testing and insertion of new technologies in deployed systems.

Customers

Customers are government agencies that fund the research and development of mission-critical systems, and the community that develops them.

Collaborators

Collaborators are graduate students and faculty members at Carnegie Mellon University (CMU). Other collaborators are MITRE, NIST, and (potentially) SEMATECH.

Approach

The Simplex architecture is designed to support the evolution of mission-critical systems. It makes the upgrade of computers, networks, and application software easier and safer. It accomplishes these objectives by encapsulating three advanced technologies—generalized rate monotonic theory, membership protocols, and the theory of analytic redundancy—while giving users a simple “plug-and-play” application environment.

The Simplex architecture demonstration has proven to be an effective tool to inform developers what is possible to safely insert new technologies into deployed mission-critical systems. There are two major aspects in the maturation of Simplex architecture.

Improve the existing architecture description.

1. Provide a detailed informal description of Simplex architecture.
2. Express in architecture description language (ADL) for those aspects that can be described by Garlan and possibly by Luckham.
3. Document those aspects that cannot be effectively described by the existing ADLs; for example, semantic dependency that is transparent to interfaces, such as signal-to-noise ratio and phase lags in data.
4. Work on possible extensions to existing ADLs to address specific concerns in real-time fault-tolerant applications.

Integrate new technologies. Simplex architecture adds value by integrating three promising technologies (real-time scheduling, hardware fault tolerance, and analytic redundancy) that were developed separately by different communities. Different assumptions about these technologies were made by different researchers. Thus, it is important to

1. Document a unified and consistent set of assumptions that were made, and the rationales
2. Prove the logical consequence of certain changed assumptions. What are the properties that are preserved? what are the properties that are no longer true? what are the properties that are new?
3. Develop comprehensive tutorial materials based on the experiences gained from applying Simplex architecture. These include TO&P-funded experiences with applying Simplex architecture to manufacturing applications (supported by NIST); re-implementation of a Simplex architecture demonstration in Ada 95 in a mixed C/C++ and POSIX.1a environment (supported by Joint Advanced Strike Technology [JAST]); and other potential applications, such as submarine control (with Electric Boat) and work with Wright Patterson Air Force Base on future F16 weapons control architecture.

DE-10B Airborne Radar Study (MITRE) (1996)

The result of this study is the examination of the Simplex architecture in a realistic setting.

Purpose

This activity will apply the Simplex architecture in a real setting, obtain feedback on how it works in practice, and gain confidence in its application in industry settings.

Customers

Customers of this task are developers of tracking systems for the DoD.

Collaborators

This work is performed in collaboration with MITRE.

Approach

We have been applying the Simplex architecture in a proof-of-concept demonstration in cooperation with another federally funded research and development center (MITRE) in a simulated environment that models the problems encountered in the Airborne Warning and Control System (AWACS) program upgrade. This work is led by MITRE, and the milestones are determined by MITRE. The first MITRE prototype was already successfully demonstrated at the 1994 SEI Symposium and to its Air Force sponsors. A joint MITRE-SEI demonstration to the Advanced Research Projects Agency (ARPA) sensor system community in 1995 is planned.

This is a collaborative project between the SEI and MITRE. The benefit to the SEI is a pilot insertion of Simplex technology in an important DoD application domain.

DE-11B Report on State of Practice in Process-Centered Environments (1996)

The output from this task will be a series of reports describing the findings of our study on the current use of process-centered environments and discussing the results of a workshop involving the participants of the study. A first report (1995) will describe the in-depth interviews with users of process automation technology. A second (1996) report will describe the results of the two surveys conducted. A third report (1996) will provide guidelines to researchers and vendors of process automation products on the needs of technology end-users. The workshop for study participants will be organized for 1996.

Purpose

This task will identify inhibitors to the successful introduction of software process automation, and disseminate the findings to the software process community.

Customers

Customers are those organizations who are considering adopting software process automation technology.

Collaborators

This activity is being supported by Nolan Norton and Company, a division of KPMG Peat Marwick. The study also has an information exchange agreement with Cap Gemini Sogeti, who is conducting a comparable study in Europe.

Approach

Within the SEI, the CASE environments group is leading this effort, with support from the Transition Models and Empirical Methods groups. This information-gathering and analysis study is structured into four parts:

- Conduct a series of in-depth interviews, in order to gain an understanding of the challenges encountered by end-users of the technology (1995).

- Develop a survey questionnaire on the basis of the interviews and disseminate it to the wider technical community. This survey will be distributed twice, the second distribution taking place 9 to 12 months after the first distribution. This will allow us to assess the evolution, progress, or failure of process automation projects (1995-96).
- Develop guidelines for researchers and vendors, based on the insights gained from our end-user interactions.
- Organize a workshop of study participants. This will be held to provide the participants (end-users, researchers and vendors) with a broad overview of the findings and to foster interaction between end-user organizations and those who are developing the technology.

3.2.7 Proposed Add-On Work Outputs

This section describes the proposed add-on work outputs for the activity area related to evolutionary engineering architectures. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

DE-5A Lightweight CASE Integration for Architecture-Based Evolutionary Design (1996-1997)

Outputs of this task include a report that describes the characteristics of patterned architectures and the CAE cycle ("conjecture, analysis, evaluate" cycle), and that provides opportunities for a next generation of CASE design tools. In addition, a prototype integration of tools will be produced to support the CAE design cycle that conforms to designers' needs.

Purpose

This task will investigate lightweight (i.e., low cost and impact) approaches to CASE integration in support of architecture-based evolutionary design approaches.

There is ample evidence that the current generation of CASE tools for engineering design are not adequate for the demands of software engineering in today's complex software environment. At best, these tools are used passively—to document designs, rather than as active aids in engineering problem-solving (a central aspect of engineering design). It is well recognized that "power designers" operate from a perspective quite different from that provided by current generation design CASE tools: They operate from a CAE cycle rather than from the linear design refinement model supported by most tools. This CAE model also mirrors that found in mature engineering disciplines.

By improving our understanding of the way engineers design software systems, we will be able to provide improved automated support for the engineer's task. As a result, we will be able to demonstrate that current technology can be applied successfully to assist engineers in the CAE life cycle.

Customers

The beneficiaries of this work will be

- tool vendors, who will gain insight into the kinds of tools needed by software and systems designers of complex applications
- corporate technologists, who will gain insight into how to adopt software architecture principles in practice without waiting for the "big technology" solutions
- corporate CASE advocates, who will gain insight into how to identify real opportunities for demonstrating the value of CASE on large projects
- researchers in software architecture and software environments, who will be provided with a low-risk, incremental counterpoint to the more ambitious and risky, large-scale technology approaches to software architecture

Collaborators

Collaborators are members of the ARPA research community who have been active in this area.

Approach

This work will be carried out by members of the CASE and architectures groups at the SEI. There are two parallel threads of this work:

- a design-theoretical thread that is concerned with understanding the use of architectural patterns in a CAE design cycle; and,
- a tool integration thread that is concerned with providing technology assistance to designers to help reduce the CAE cycle time.

The first thread involves surveying designers of industrial-strength patterns (Northrop's Teacherless Blackboard, the SEI's Structural Models, and other object-oriented design methods) to understand the information and automation needs of designers as they formulate conjectures, perform analysis of conjectures and evaluate partially-fabricated solutions—and as they transition between these steps (in particular, how designers use the results of an evaluation to produce the next, more refined conjecture).

The second thread mobilizes the knowledge obtained above by developing "opportunistic and lightweight" tooling for a particular architectural pattern and design problem:

- By opportunistic we mean that the nature of complex system design implies hard limits on the scope and extent of tooling—no single design tool will be sufficient. Instead, there are likely to be a range of opportunities for design automation, and the first thread provides a framework for identifying and prioritizing these opportunities.
- By lightweight we mean that the tool support must be quickly assembled from widely-available, mature tool technologies. By exploiting the structural and generative properties of patterned architectures, innovative use of tools not otherwise associated with engineering design can be effectively used (e.g., spreadsheets, program generators, 4GLs, templates).

Unlike research approaches that are developing an all-encompassing technology base for architectural-based design (DSSA), the approach taken in this task is to work with designers to identify and prioritize design tools that can help them reduce CAE cycle time. While the more ambitious research efforts may produce a breakthrough in architecture-based software development, their very scope and magnitude (and relative risk) obscure opportunities to transition more stable aspects of architectural design into practice along with effective and usable, albeit limited, technology support.

DE-6A Affordable Use of COTS Components (1996-1997)

The primary outputs of these tasks will be of benefit to a wide community interested in applying practical techniques to building applications from COTS components. They will consist of

- a document describing the best current practice in COTS integration as supported by a high-level survey and several detailed case studies
- a report describing the relationship between open systems, COTS, and non-developmental items (NDI), containing strategic advice for program managers required to make use of COTS and NDI in implementing an open system approach
- a report containing guidelines on how to select, test, evaluate, and integrate COTS components for use in mission-critical applications

Purpose

As evidenced at the recent SEI / Microelectronics and Computer Technology Corporation (MCC) Symposium ("Systems Integration of COTS Components"), there is a great deal of interest in the topic of building applications from collections of COTS components. However, perhaps the main lesson from that symposium is that experiences, approaches, and techniques currently in use for integrating COTS components vary widely. The consequence of this is that there is little uniformity of understanding, a lack of knowledge and technology transition from one organization to another, and no consistent experience base on which to draw to predict the costs and benefits of applying a COTS approach to systems integration.

This task will investigate integration and evolution of COTS components. The concentration will be on affordable ways to predictably and safely design, build, and evolve systems from existing components.

Customers

This work will be of value to a wide range of DoD customers, including the industry organizations providing solutions to the DoD, and DoD program mangers selecting, acquiring, and managing these systems.

Collaborators

This work will necessarily involve frequent interactions with members of the DoD community to supplement, refine, and validate the work that is carried out.

Approach

This work will be carried out by members of the CASE, open systems, and architectures groups within the SEI. Currently, we envisage three main tasks that will be carried out as part of this work.

1. We will analyze a number of organizations that are currently developing application systems through the integration of COTS components. Building on previous studies in the CASE domain, it is likely that this study will undertake a wide ranging survey in the real-time systems domain to obtain a general impression of the current state of integration in this area. This will be accompanied by a detailed investigation of a small number of organizations to discover some of the detailed practices that they employ.
2. The relationship between open systems, COTS, and NDI will be explored. This work will directly address concerns voiced by a number of SEI customers who find themselves in the position of being advised or mandated to follow open systems based approaches without the necessary understanding to determine what this means. This task will explore the relationship between open systems, COTS, and NDI, and provide techniques that aid decision makers in selecting COTS and NDI components based on their support of an open systems approach.
3. We will explore in detail the use of COTS in the real-time and/or fault-tolerant domain. The Simplex architecture demonstration is constructed from COTS components. In developing this demonstration, it was a frustrating experience to find various undocumented defects in COTS components that are advertised for mission-critical applications. In further discussions with industry developers, we find that there is a serious lack of comprehensive information regarding the selection, evaluation, and integration of COTS components for mission-critical applications. DoD systems are increasingly relying on COTS components, and yet most DoD contractors have little experience to deal with selecting, evaluating, and integrating COTS components. To mitigate the cost, schedule, and technical risks of COTS-based systems, we will work with system developers to develop comprehensive guidelines and validate the guidelines via selected experimentation.

DE-7A The Evolution of Distributed Mission-Critical Systems (1996-1997)

The output of this work is a report and a feasibility demonstration of a distributed application of Simplex architecture.

Purpose

The purpose of this task is to demonstrate safe and reliable evolution of real-time networks based on open standard components by the end of 1996, to demonstrate evolvable network technology during 1997, and to transition distributed concurrent system technology in 1998. In addition, the matured network and distributed system technology will be documented in a handbook.

Customers

Customers are developers of large-scale defense or industrial systems; for example, those that generate and distribute electrical power. EPRI has provided us with its internal study of its communication needs and has prepared a 1996 plan that includes the SEI's participation.

Collaborators

Collaborators are graduate students and faculty members at Carnegie Mellon University (CMU), the Navy's Next Generation Computing Resources (NGCR) High Performance Network Working Group, and (potentially) EPRI.

Approach

Many mission-critical systems are distributed in nature. Examples include defense systems, flexible manufacturing systems, and telecommunication systems. The future distributed mission-critical system will also include a larger number of mobile components. While computers and networks have revolutionized the capability of distributed systems for the production of goods and delivery of services, the current distributed computing infrastructure often introduces formidable barriers to continuous process improvement, equipment upgrades, and agility in responding to changing markets and new threats in defense systems.

Consider the following anecdotal scenario from industry, related to implementing quality control. In this case, the quality control system identified the variability in one of the critical production steps as the source of a continuing product quality problem. To further determine the causes of the uncontrolled variability, it was necessary to collect data on the operation of the critical cell. The natural place to obtain this information was from the on-line controller, since it accessed the real-time sensor data. However, it was not clear how the data acquisition and communication over the network would affect the timing of the program, which was controlling a high-speed spindle. Any further attempts to improve the performance had to be abandoned.

Currently, the Simplex architecture is able to support the evolution of uni-processor and multiprocessor applications for continuous process control and signal processing systems. It is important to mature the Simplex architecture to support safe, predictable, and on-line networking evolution in the near term and to support the evolution of concurrent distributed system as whole in the future.

DE-8A The Evolution of Discrete Sequential Control Systems (1996)

The output of this work is a report and a feasibility demonstration of a Simplex architecture for discrete sequential control applications.

Purpose

Industrial systems can be divided into continuous process control (e.g., motion control and the making of chemicals) and discrete sequential control (e.g., the making of vehicles and the packaging of semi-conductor chips). The existing Simplex architecture is limited to continuous process control. The purpose of this effort is to expand the domain of Simplex architecture and make it more complete.

Customers
Customers include developers of discrete control systems such as manufacturing systems and communication switches.

Collaborators
Collaborators include graduate students and faculty members at CMU; NIST; and (potentially) SEMATECH. Experiments will be conducted at CMU's Plasma Deposition Laboratory.

Approach
We will work with CMU's Electrical and Computer Engineering Department to incorporate the sequential control technology into Simplex architecture. The Simplex architecture will then be ported to the Department to investigate experimentally its effectiveness in mitigating the risk in updating both continuous and sequential controls. Furthermore, we will work with Wright Patterson Lab engineers and industrial partners to define a model of sequence control problem in the context of safe online upgrade of sequencing and coordination of distributed motions. A proof-of-concept demonstration will be given at the 1996 SEI Symposium.

DE-9A Risk Management for Open Systems (1996)

This work produces two outputs. One is an extension of the risk assessment framework to take open systems risks and issues into account. The second is a new module for the open systems course, *Open Systems: The Promises and the Pitfalls,* addressing risk management for open systems.

Purpose
The results of this joint effort with the SEI's Risk area are both to enhance the open systems course by integrating the risk technology into the course and to develop draft guidelines for software acquisition. These guidelines could then be pilot tested with the Software Acquisition Maturity Model key process pilot tests of acquisition guideline material. The open systems course, currently well received, would be enhanced to discuss managing the pitfalls and changes to adopt open systems. The integration of both risk and open systems work increases the value of both and provides an opportunity for springboarding off existing work.

Customers
Customers include future customers of both the open systems course and related products and the risk assessments and related risk products.

Collaborators
Collaborators will be DoD program offices and contractors willing to pilot this work with the SEI.

Approach

This work is a joint effort between Disciplined Engineering and Risk at the SEI. The approach is two-fold:

Open systems issues will be integrated into risk assessments, and risk management integrated into the open systems course. Risk and Disciplined Engineering will work together to integrate open systems issues and concerns into the existing taxonomy and instruments for risk assessment. Risk management issues will be integrated into the open systems course in the form of a specific course module on risk management for open systems.

Open systems guidelines will be integrated into software acquisition guidelines. The mutual understanding and cooperative work will culminate in an extension of current software acquisition guidelines to include guidelines for open systems.

DE-10A Open Systems Assessment Instrument (1996-1997)

The output of this work is an assessment instrument—consisting of a framework and a questionnaire—to assess a system's openness and to assess an organization's readiness to move to open systems.

Purpose

Those responsible for systems today frequently need to know how open their system is, and to have guidance on how to make it more open. The information provided by such an assessment capability would be beneficial to those acquiring or maintaining a system as well as to the vendors who are trying to satisfy the demand for open systems.

Customers

Customers are those government programs that are interested in a means of assessing their current state of openness in order to plan the migration of their system to greater openness.

Collaborators

Collaborators may include government programs that are willing to act as experimental subjects as we work to learn more about what it means to assess a system's—and an organization's—openness.

Approach

The effort will be based on the extensive experience within the SEI in the development of such assessment instruments. Other SEI impact areas that will be consulted include Risk and Process. From such prior work, we expect to gain a greater understanding of the kind of framework that would be suitable, the kinds of questions that are most effective, interview processes, etc.

The basic approach will be to work with our collaborators to determine the feasibility of an assessment instrument or set of assessment instruments to use for this purpose and then to design that/those instruments. An effective part of this approach may be collaboration with a government program that is interested in assessing its openness and is willing to be our subject as we work to evolve our knowledge of how to assess openness. It is often the case that such instruments evolve as they are put to use and more is learned about eliciting the appropriate information. This effort will also include some testing of the evolving instrument and inclusion of that feedback into enhancements.

DE-11A Open Systems Interactive (1996)

The SEI has evolved a 3-day course, *Open Systems: The Promises and Pitfalls*, that addresses open systems issues. This course is accompanied by a handbook, which contains relevant reference material. Additions to the course (through the outputs DE-6B, DE-9A, DE-10A, TS-4A) will be included in the Open Systems Interactive output as appropriate.

This task will make this body of open systems knowledge available in interactive form on CD-ROM media.

Purpose

The purpose of the interactive CD-ROM is to package and distribute the existing course in a way that traditional (paper) media cannot: the CD-ROM can mimic elements of the existing course, particularly the rich interaction students have with the instructor and with the other students. As described below, interactive media does not simply reinvent lectures in a new medium, but provides an effective learning format for widespread distribution.

Customers

Customers include DoD contractors, the DoD, and acquisition authorities. The material will subsequently be tailored to a broader audience, including personnel from other government agencies and the industrial and commercial sectors.

Collaborators

Possibilities exist for collaboration with interested multimedia experts at CMU and possibly in industry as well.

The Office of the Under Secretary of Defense's Open System Joint Task Force (OSJTF) has indicated interest in collaborating with the SEI and providing TO&P sponsorship. The Defense Acquisition University may have an interest in a multimedia presentation of the course, as well.

Approach

Front-end analysis. The approach to developing the CD-ROM will start with a clear definition of the audience for which it is intended, the ideas it is to convey, and an instructional approach that is suitable to best meet the instructional goals. Since there are multiple ways for CD-ROM to be used as an instructional medium, the approach will take a careful look at the exact utility of the course material in relation to the career functions to which it is directed. Among other considerations, this approach will consider the sheer volume and scope of content that is practical for a CD-ROM implementation, site equipment requirements such as the number and types of delivery platforms available to distribute the course, and the treatment of the presentation to the student. Currently, CD-ROM provides a wide variety in this regard, from generic video presentation to fully interactive simulation and training.

Development of initial architecture. If the initial analysis indicates the need, the next step will be to create an architecture for a learning product. This architecture will specify the components of the instructional system and the interactions between these. Using this architecture, a prototype will be developed using a single module from the existing course that reflects the characteristics that are important to the defined approach and architecture. As mentioned above, the issues of the instructional approach will be related to the needs of the target audience and career functions to which the course is directed. Therefore, depending on the nature of the educational mission associated with the identified career positions, the presentations could be mostly informational in nature, or could require a great deal of hands-on interaction with the system, or both.

Final development. Finally, based on the revisions to the approach and architecture resulting from a systematic study of student comments and observations of their use of the prototype, a full system implementation will be undertaken.

DE-12A Evaluation and Application of Software Process Modeling Technology (1996-1997)

The outputs from this task will assist organizations in more effectively selecting and applying software process modeling technology (SPMT). Some examples of SPMT include use of word processing, graphical and outlining tools such as Microsoft Works, MacDraw and Inspiration, IDEF0 tools for process modeling, use of STATEMATE® for process modeling, and simulation.

The outputs will consist of reports, presentations, briefings, and other guidance. Consistent with the activities described below in "Approach," the primary outputs will be

- a state-of-the-practice report on SPMT, particularly focusing on the application of SPMT to software process modeling and definition, in support of software process improvement (SPI). Coverage is expected to include technological capabilities and features being applied in practice, and corresponding usage experiences.
- guidance on the evaluation and application of SPMT. Topics are expected to include (1) guidance to help customers make SPMT product selection choices appropriate for their objectives, (2) guidance in the effective application of SPMT to software process modeling and definition, and (3) identification of needed capabilities not currently available and an explanation of how they would be applied.

Purpose

The overall purpose of these outputs is to improve the effective use of SPMT as applied to the process definition, process implementation, and process evolution activities entailed in SPI. More specifically, these outputs are intended to provide concrete guidance for practitioners in the selection and application of SPMT, subject to their particular objectives and circumstances. These outputs are also intended to provide feedback to SPMT researchers and developers, regarding the needs and experiences of the community using these technologies in support of process definition, implementation, and evolution.

Customers

Customers are practitioners involved in process definition and related activities in support of SPI; for example, members of software engineering process groups (SEPGs), process action teams, and other process improvement teams. Researchers, developers, and vendors of SPMT are also customers, as are sponsors of research and development of SPMT (e.g., ARPA).

Collaborators

Collaborators are practitioners who have acquired and used SPMT for process definition and related activities (i.e., innovators and early adopters), as well as researchers, developers, and vendors of SPMT. Planned participation includes eliciting their experiences, lessons learned, and views regarding objectives and capabilities for SPMT, as well as reviews and feedback regarding the emerging outputs.

Approach

This task is a collaborative effort between the SEI's Software Process and the Disciplined Engineering impact areas. This task complements a task focusing on technology in support of process automation [see DE-11B Report on State of Practice in Process-Centered Environments (1996)] and is closely tied to baseline work in Process on the process definition method.

This effort is explicitly focused on SPMT as applied to the process definition, process implementation, and process evolution activities entailed in SPI. The effort entails the following major activities:

- Collect experiences, lessons learned, and views regarding objectives and capabilities for SPMT, from knowledgeable practitioners, researchers, developers, etc. Assimilate and organize into a framework of objectives (e.g., understanding of process, management of process/project, consistent performance of process) and technological capabilities (e.g., describing, adapting/developing, validating, changing, automating process) (1996).
- Develop a state-of-the-practice report describing objectives being pursued, features being applied in practice, and corresponding usage experiences and lessons (1996).
- Develop guidance to help practitioners make SPMT product selection choices appropriate for their objectives; such as an evaluation method and criteria for SPMT, and results of trial application to selected existing products (1996-1997).

- Develop guidance in the effective application of SPMT. Address issues of technology use to support process definition and modeling practices called for in the Capability Maturity Model (CMM) (i.e., specific practices outlined in the key process areas: Organization Process Definition, Integrated Software Management, Quantitative Process Management, and Process Change Management). Address issues of technology use to support the emerging SEI method for defining software processes (SP-5A) (1996-1997).
- Integrate this work with other SEI investigations into process automation (i.e., process centered environments study DE-11B). Investigate guidance for automating a process that has previously been defined/modeled using SPMT tools not oriented toward automated execution (1996-1997).
- Continue to track emerging technology and its application (with respect to objectives and capabilities), and provide feedback on needs and applications based upon SEI insights and knowledge of both state-of-the-art and state-of-the-practice (1996-1997).

3.2.8 Related TO&P Activities

A number of existing TO&P activities directly support the initiatives described in this area.

In the open systems area, support is provided for course development by the Office of the Secretary of Defense (OSD) Open Systems Joint Task Force, and for development of the POSIX standard by the Navy's NGCR program.

In the systems integration context, support for investigating COTS integration mechanisms techniques is provided to the DoD I-CASE program in support of the evolution of the I-CASE environment, to the Office of the Secretary of Defense (OASD) for Information Management in defining DoD policy on CASE environments, to NIST in its support of integration techniques in both the CASE and manufacturing engineering domains, and to the NSA as it maintains numerous software systems.

In the safe evolution of complex real-time systems, current support is provided by the ONR which provides funding for developing a Simplex architecture demonstration and development of the theoretical foundation of Simplex, CMU's Advanced Real Time (ART) technology project which provides graduate students, faculty time, and funding for performing cooperative research of some theoretical issues raised by the Simplex architecture, NIST which provides funding for integrating the Simplex architecture with its enhanced machine controller to support the safe integration and evolution of machine controllers, and JAST which provides funding to port the Simplex architecture to Ada 95 and to investigate the potential use of the Simplex architecture in advanced avionics applications.

The activities in this activity area align with the goals of ARPA's new software program, Evolutionary Design of Complex Software (EDCS). Disciplined Engineering will be collaborating with ARPA and EDCS researchers.

3.3 Predictive Engineering

The predictive engineering area is based on the following predicates:

- Predictability and control are the hallmarks of a mature engineering discipline.
- Predictability and control result from understanding the relationship between structure and behavior.
- This understanding comes from a combination of and interaction between empiricism, analytic practice, and experience.

This area plans to address predictive engineering by developing systematic ways to relate quality attributes (such as performance and dependability) expressed by a system to the system's architecture.

3.3.1 Problem Statement

Most organizations that develop, procure, or maintain software do not have the ability to predict with any confidence or accuracy how long the process will take, how expensive it will be, how the software will perform when it is fielded, how difficult it will be to maintain over its life-cycle, or whether or not the functionality will be satisfactory to the customer for whom it was intended. There are no formal, repeatable, reliable models of prediction that allow a developer or customer to tell, before a system is delivered,

- whether it will meet all of its requirements
- whether its performance will be sufficient
- what it will cost to perform adaptive, corrective, or perfective maintenance on the system
- how long the development effort will take and what it will cost
- how a change to improve one quality of the system (e.g., performance or security) will affect other qualities (e.g., maintainability or portability)

What has elevated structural engineering (and other engineering disciplines) above the morass of ad hoc techniques, unpredictable outcomes, and test-based verification is the advent of a sufficient body of knowledge—codified, accessible, reliable, standardized, and based on quantitative formalisms—that allow system building to proceed with the assuredness of high-confidence predictability.

Software engineering is not yet at this stage of maturity, but it is possible to encourage trends in that direction. The Predictive Engineering area takes this as its problem to address—evolving the community towards predictable, repeatable software system development.

3.3.2 Customers

Customers for these outputs are software engineering researchers (developing technology for quality attributes), developers, and technical managers (making trade-offs between quality attributes).

Specific customers with technology interests include

- DoD-sponsored programs: e.g., STARS, DSSA, Prototech, Software Composition, and CARDS
- development organizations wanting to build corporate competency and assets in architectures and models
- the DoD acquisition community evolving to architecture- and reuse-based approaches

Work in predictive engineering will also serve customers in the following domain areas:

- training simulator community
- real-time embedded system community
- command and control community
- tool builders

3.3.3 Rationale

Software quality requires mature technology to predict, control and make trade-offs among different quality attributes and requires an understanding of the interrelationship between quality attributes and software architecture. If either is lacking, even a mature organization will have difficulty producing products with predictable performance, dependability, or other quality attributes. Designers often focus on achieving some narrow goal (e.g., performance) while neglecting its impact on other attributes (e.g., dependability). The result is that systems often fail to meet requirements, that is, they lack quality.

The problem arises not just on customized software or software developed under proprietary standards. Open components and subsystems are designed to meet the same interface standard, but different components or subsystems could emphasize different (and perhaps conflicting) quality attributes. Integrating open components or subsystems of different provenance often has an adverse effect on overall system quality.

Poor quality eventually affects cost and schedule because serious problems are often not discovered until the system integration phase or beyond, when remediation would require extensive rework.

This area focuses on the prediction and control of those attributes that determine a software product's quality and on the use of architecture technology to build systems with specific quality attributes such as performance, dependability, and interoperability. The ultimate goal is the ability to quantitatively evaluate and trade off multiple quality attributes, as suggested in Figure 3-5, to arrive at a better overall system *before* the system's design is too detailed (or has been cemented in an implementation) to easily change.

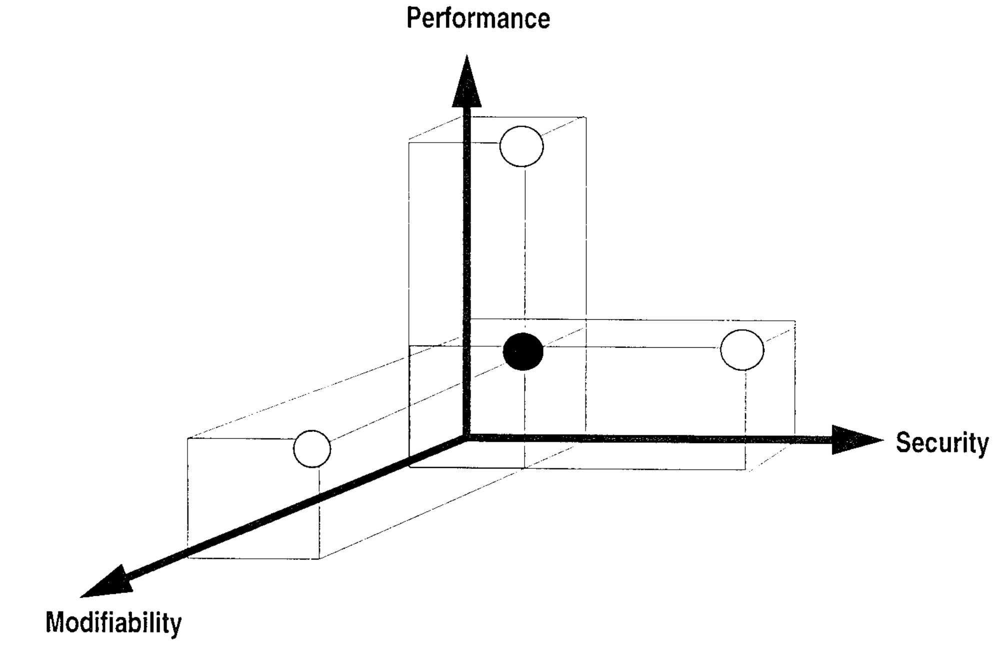


Figure 3-5: Trade-offs Among Product Quality Attributes

Other engineering disciplines recognize architectures as a key component for managing complexity. Similarly, other engineering disciplines use models as the appropriate level of abstraction for reuse. Yet techniques for effective evaluation and analysis of architectures and models, and synthesis of systems based on architectures and models, are not widely understood or known.

Successful program and product planning can occur only when related architecture and reuse activities are brought together to share technology and advance the engineering practice. New architecture technologies continue to emerge, and we must be able to build on related efforts. This must be a consensus-building process between different parties, be they system developer or customer, product-line manager, or parties in a software component industry.

In summary, the effects that particular architectural design decisions have on system qualities often cannot be determined in a quantitative or repeatable way. Developing systematic ways to relate the quality attributes of a system to the system's architecture provides a sound basis for making objective decisions about design trade-offs and enables engineers to make reasonably accurate predictions about a system's attributes that are free from bias and hidden assumptions. This ability is on the critical path of making software engineering a mature engineering discipline. The Predictive Engineering area will endeavor to stimulate the maturation and dissemination of methods that allow for quantitative analysis of software architectures.

3.3.4 Benefits

This activity area will provide the following benefits to customers:

- shared views among software professionals of appropriate architectures and models for similar systems and system functions
- evaluations of the maturity and effectiveness of architectural representations and their use for describing and analyzing actual systems
- increased ability of systems engineers to make informed trade-offs regarding quality attributes of systems
- linking of domain and application engineering to provide a systematic approach to gain the benefits promised by reuse
- examples of how abstraction techniques (e.g., domain analysis, architectural modeling) can be used in support of engineering of software-intensive systems
- increased ability to predict and control product attributes through architecture-level analysis and synthesis
- frameworks for analyzing existing development practice in order to address the needs of integrated product development teams concerned with evolving product families

Figure 3-6 projects the state of the practice for the near-term, in light of the anticipated benefits.

3.3.5 One-Year Objectives for 1996

One-year objectives for the activity area include

- production of a guidebook for addressing the architectural mismatch problem in systems integration
- production of a well-tested framework for characterizing architecture representation technologies, in the form of a guidebook for choosing an architecture representation strategy
- production of a guidebook detailing the best practices for evaluating an architecture with respect to desired quality attributes, and combinations of quality attributes
- production of a white paper which serves as a guide for developing engineering frameworks (which can then be applied to the development of a security engineering framework)

- production of an engineering framework for guiding engineering decisions concerning quality attributes
- testing and validation of an engineering framework for guiding performance engineering decisions

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Prediction and control of quality attributes	• Problems with quality attributes commonly discovered late in development • Lack of quantitative methods for predicting behavior of systems with combinations of non-functional requirements	• Performance engineering rationalized • Improvement demonstrated for some strategic partners • Better control of performance achieved through architecture-level synthesis and analysis	• Control of combinations of quality attributes demonstrated • Better control of combinations of quality attributes achieved through architecture-level synthesis and analysis	• Ability to predict behavior of non-functional attributes (such as performance and dependability) • Engineers using mature attribute engineering techniques • Systems behave as predicted from the start
Architecture representation and evaluation technology, and taxonomic guidebook	• No consensus on best practice • Developers using ad hoc techniques • Lack of quantitative evaluation methods	• Availability of architecture technology is known • Advantages have been demonstrated • Several high-visibility projects are actively exploiting it	• Architecture technology is routinely used in large-scale development efforts • Common architecture metrics emerge	• Use of architecture technology routinely appears in proposals and requests for proposals • Engineers routinely trained in architecture technology and methods

Figure 3-6: Trends in Predictive Engineering

3.3.6 Baseline Work Outputs

This section describes the baseline work outputs for the activity area related to predictive engineering. These outputs were approved for 1996 through the basic funds allocation process. Appendix A lists all baseline and proposed add-on work outputs.

DE-12B Quality Attribute Engineering Framework (1996-1997)

This engineering framework is intended to be a guide that facilitates making technical engineering decisions. (In the 1995 plan, *SEI Program Plans: 1995-1999*, it was titled Report on Quality Attribute Trade-Offs.) The guide will focus on identifying the factors (such as the properties of a software architecture) that affect quality attributes with an emphasis on dependability.

Purpose

Acknowledging that a truly useful engineering decision framework will need to deal with multiple quality attributes, the purpose of this work is to extend the performance engineering framework (see DE-13B Performance Engineering Framework) to include dependability. In effect it will provide a language of discourse for the performance and dependability attributes.

Customers

Potential customers of this output include software engineering practitioners and acquisition agents. For example, a lead engineer responsible for the development and/or evolution of a system could be aided by a framework that offered guidance for making decisions and trade-offs concerning the quality attributes of the system. Acquisition agents could benefit by having a framework that could be used to ask important high-level questions about quality attributes of the design as development progresses.

Collaborators

A project member is a member of the IFIP working group 10.4 on dependability. Potential collaborators include other members of this working group.

Approach

This work builds on work in product quality attributes in 1995. This work also will be aligned with DE-14B Assessment of Architecture Evaluation Practice and DE-13A Guidebook for Addressing Architectural Mismatch.The primary goal for 1995 is to develop a strategy for generalizing the performance engineering framework to include multiple quality attributes. During 1995 we will have done some preliminary work in codifying dependability decisions and determining how to make systematic trade-offs between performance and dependability. During 1996 we will develop the first draft of a dependability engineering framework. Also during 1996, we will continue to apply what we learn about quality attribute decision analysis to the analysis of software architectures.

DE-13B Performance Engineering Framework (1996)

It is well recognized that software systems performance is a problem. (By performance we are referring to software timeliness usually thought of in terms of latency and throughput.) Often systems perform more than 100 times slower than desired when first tested, and the time required to improve performance is a frequent cause of cost and schedule slips despite a considerable amount of knowledge that is available to deal with performance problems (queueing theory, scheduling theory, simulation tools, etc.).

This engineering framework is intended to be a guide that facilitates making technical engineering decisions. (This work was titled in the 1995 plan, *SEI Program Plans: 1995-1999*, as Performance Engineering.) The guide will focus on identifying the factors (such as the properties of a software architecture) that affect quality attributes with an emphasis on performance.

Purpose
The purpose of this work is to develop a performance engineering framework that will foster broad awareness of performance engineering knowledge and practices. The framework will offer a structure for codifying, assessing and improving performance engineering.

Customers
The potential customers for this output are the same as for DE-12B Quality Attribute Engineering Framework, which includes software engineering practitioners and acquisition agents.

Collaborators
Potential collaborators include some of the participants in the performance engineering workshop held in June 1995, who were drawn from industry, government, and academia and who shared practical experience in engineering the performance of large software-intensive systems.

Approach
Our approach relies on interaction with the performance engineering community. During 1994 and 1995, we gathered information from the performance engineering practitioner community via interviews and a workshop to gain a better understanding of the types of and causes for performance problems. We also explored several frameworks and presentations of state-of-the-practice data.

The most promising approach was based on engineering decision making. The premise of the approach is that engineering can be viewed as a decision-making process and that engineering decisions serve as loci for considering architectural, design, implementation, and evolution alternatives.[5] Therefore, a framework that highlights key decisions and facilitates decision making has the potential to raise awareness of the impact that decisions have on system performance and other attributes. A first draft of the performance engineering framework will be produced in 1995. During 1996 the framework will be tested in organizations and extended to dependability as a second attribute. (See DE-12B Quality Attribute Engineering Framework.) If successful, the framework will be applied to security as an additional attribute. (See TS-8A

5. Gruber, T.R. & Russell, D.M. "Generative Design Rationale: Beyond the Record and Replay Paradigm." (KSL 92-59). Palo Alto, California: Knowledge Systems Laboratory, Computer Science Department, Stanford University, 1993.

Software Engineering Framework for Trustworthy System Development (1996-1997), in Chapter 4 of Volume II.)

We will also examine how other engineering disciplines codify and disseminate technical engineering information.

DE-14B Assessment of Architecture Evaluation Practice (1996-1997)

Architecture evaluation is concerned with trying to understand whether a particular set of architectural decisions will permit or prohibit the construction of a system that will meet its behavioral, performance, and quality requirements. An assessment will produce a guidebook of successful architecture evaluation experience. (In the 1995 plan, *SEI Program Plans: 1995-1999*, this work was titled Guide to Software Architecture Assessment Practice.)

Purpose

Practitioners currently have no standard guidelines, models, methods, criteria, or engineering discipline to help them decide when an architecture is sound or to choose an architecture as the best or most appropriate among a set of competing designs. This often leads to the intolerable situation in which the inappropriateness of the earliest design decisions is not discovered until after implementation is complete. The purpose of this task is to understand the state of the practice in evaluating architectures for quality attributes, performance attributes, and functional attributes that can be predicted from architecture-level information.

Customers

Customers for this work include software acquisition agents (who can use the framework for evaluating competing architecture-based proposals, and for specifying what architecture-level information must be present in a proposal in order for it to be evaluated), and software development organizations (which can use the framework to chart specific quality-based life-cycle evolution paths for their systems, and who can use the framework to choose among competing designs to solve a system problem).

Collaborators

Our collaborators will be industrial and DoD sources who have experience in evaluating a system at the architectural level.

Approach

The approach will be two-fold:

First, software developers and procurers will be interviewed in a set of structured case studies to understand how architectural trade-off decisions are made in practice.

Second, the SEI's Software Architecture Analysis Method (SAAM)[6] will continue to be evolved into a more formal, more repeatable, more domain-independent method that can be used routinely by domain experts to make measured evaluative architecture decisions. SAAM will be applied to actual industrial-scale architectures such as an FAA air traffic control system, to give it a grounding in real-world applicability.

DE-15B Evaluation of Architecture Representation Technology (1996)

Architecture representation is a key process in the utilization of architecture as the unifying concept of system development. Representation may be accomplished by using an ADL or some other language that is capable of capturing architecture information in a less comprehensive fashion. Often, developers represent architecture informally, using circle-and-line diagrams with poorly defined semantics. The advantage of the ADL approach is the built-in capability for completeness and consistency checking, as well as analysis (such as performance modelling) that can be performed about the system based solely on architecture-level information. This evaluation will take the form of a survey of available ADLs, comparing and contrasting them with each other and a set of non-ADL languages that can be used to represent an architecture.

Purpose

The purpose of this activity is to produce a unifying characterization framework for comparing ADLs that can be used to compare two languages or representation technologies on a common basis. A developer who wishes to use an ADL will be able to quickly pinpoint the candidates under consideration into the framework, and make judgements about which ADL may be most appropriate, given the particular context of that developer's project.

Customers

Customers for this work will include system builders, who can use the common framework to evaluate languages for their potential suitability to their projects; language developers, who can use the framework to suggest areas that might be of special concern to system builders; and technology sponsors, who can use the framework to understand the areas in which competing languages overlap or fail to address particular technical areas.

Collaborators

Collaborators include the CARDS project; language developers such as those at Stanford, CMU, the Naval Postgraduate School, Teknowledge, the University of Texas; researchers who have studied and compared ADLs, such as Honeywell and the University of Southern California; and application communities in which ADLs may play an important role, such as Advanced Distributed Simulation and PMA-264.

6. Kazman, Bass, Abowd, Webb: "SAAM: A Method for Analyzing the Properties of Software Architectures," Sixteenth International Conference on Software Engineering, Italy, 1994.

Approach

The approach is to use a feature-oriented domain analysis organizational approach applied to the domain of architecture description languages, augmented by a set of development scenarios carried out in various application domains. Features are categorized broadly as applying to the language itself, to systems describable with the language, and to the process of producing a system using the language. The framework for characterizing different languages, then, is the union of the features found in the domain of ADLs. These features can be used to describe a particular language by describing to what extent that language exhibits the feature.

The framework will be applied to a broad selection of ADLs, in order to produce an ADL survey and also to validate the framework in terms of its ability to discriminate among ADLs by including the appropriate set of features.

3.3.7 Proposed Add-On Work Outputs

This section describes the proposed add-on work outputs for the activity area related to predictive engineering. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

DE-13A Guidebook for Addressing Architectural Mismatch (1996-1997)

The term "architectural mismatch" was coined by David Garlan to describe what happens when supposedly-compatible, off-the-shelf software products are integrated and fail to work together correctly. Architectural mismatch is one aspect of the more general COTS integration problem. A guidebook will treat the issue of kinds of information that constitute *de facto* (as opposed to documented) interfaces to software components. Emphasis will be on semantic information, as opposed to traditional syntactic treatment of interfaces, and how that information can be used to avoid architectural mismatch.

Purpose

Architectural mismatch represents a major obstacle to the routine use of independently-developed software components, reuse, and architecture-based development. The purpose of this activity is to produce a guidebook of software component interface information, to allow for systematic production of integrable, composable, independently-developed software components. A guidebook will suggest ways in which vendors and consumers alike of software components can avoid the architectural mismatch problem.

Customers

Customers are the current customer base, plus DoD contractors, DoD and industrial acquisition authorities, software vendors, system integrators and developers.

Collaborators

Collaborators include COTS vendors, DoD acquisition offices, and software architecture theoreticians.

Approach

The planned output is a guidebook of design assumptions or agreements that components embody about each other. Beyond what is normally thought of as "interface" information, these agreements or assumptions may include areas of performance (e.g., expected return time), side effects, resource utilization, structure (e.g., which component has control of the system's overall control loop), environment, re-entrancy, and many others. They are almost never documented and result in system development failures. Given such a guidebook:

- Component vendors can consult the catalog to see if a component is making unwritten assumptions about or requiring unadvertised agreements of other components. The designs may be modified, or the documentation may be improved, making the component easier to integrate into new systems.
- System-builders can consult the catalog to help ask effective questions when considering a particular component for use in a new system.
- Integrators can consult the catalog when faced with non-cooperating components, to get direction towards possible solution strategies.
- Acquisition authorities or procurement agents can consult the catalog when writing requests for proposals, to aid in the expression of requirements for a component to be procured.

This activity will benefit from previous and ongoing SEI work in quality attribute engineering, open systems standards, architecture representation, and performance engineering.

3.3.8 Related TO&P Activities

TO&P activities relevant to this work include those in which the SEI is involved in helping customers evolve to a more mature engineering practice. We can gain insight from these tasks in terms of how customers act today in order to predict system qualities based on architecture-level criteria. Examples include those customers involved in transitioning the Simplex work, those customers using Structural Modelling to build real systems, the FAA, the Coast Guard, USN PMA-264, and the USAF Alr Staff.

The activities in this activity area align with the goals of ARPA's new software program, EDCS. Disciplined Engineering will be collaborating with ARPA and EDCS researchers.

Volume II
Chapter 4 Table of Contents

4 Trustworthy Systems

As briefly described in Volume I, Trustworthy Systems will focus on three activity areas:

1. Incident Handling
2. Security Improvement Tools and Techniques
3. Trust Technology Maturation

In the first activity area, Incident Handling, we address immediate problems. Through the second activity area, we foster the widespread adoption of tools and techniques that improve the security of existing systems. Through the third activity area, we identify and mature technologies that can be used to produce software-intensive systems that are highly resistant to attack. Figures 4-2, 4-4, and 4-6 illustrate the activities in each area and how each provides input into the next.

Some of this work was described in different sections of last year's plan, as shown in Figure 4-1. Last year's Incident Prevention activity area has been divided into two areas: Security Improvement Tools and Techniques, and Trust Technology Maturation.

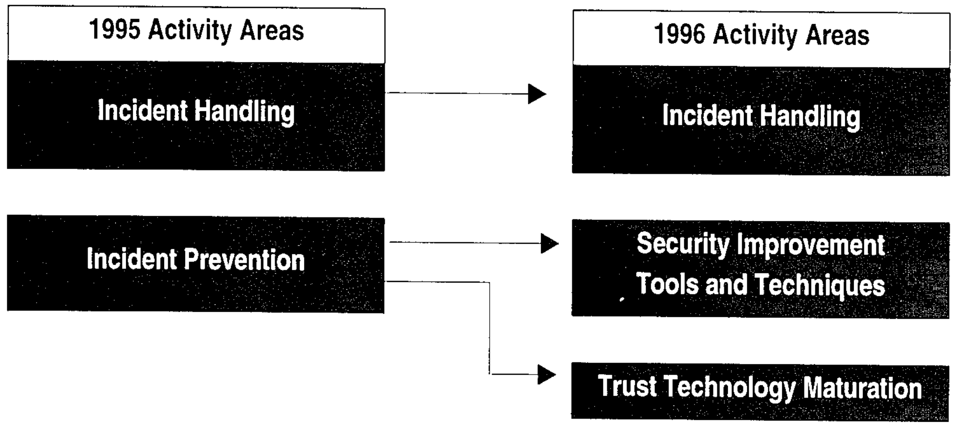
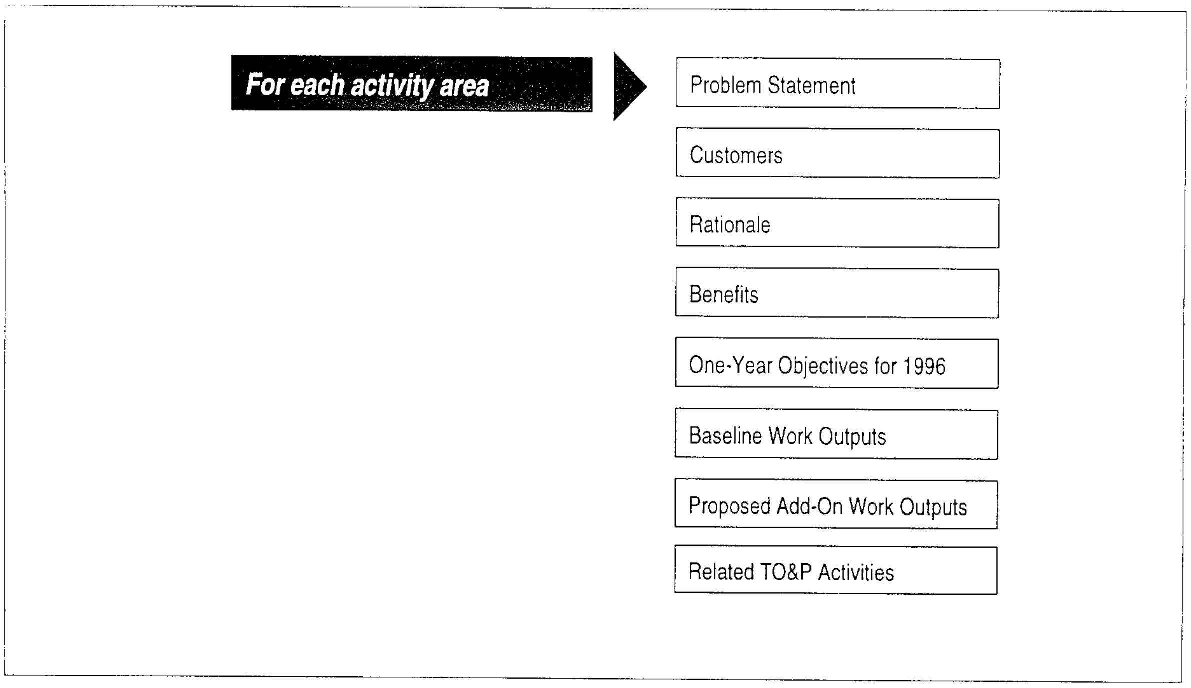


Figure 4-1: Mapping of Trustworthy Systems Activity Areas

In this chapter, the sections for this impact area are as follows:



4.1 Incident Handling

4.1.1 Problem Statement

As the Internet and National Information Infrastructure (NII) become larger and more complex, the frequency and severity of unauthorized intrusions into systems connected to these networks become increasingly serious problems. The data stored and processed on these networks is at risk, and the need to protect information and resources is critical. Without a centralized response and coordination facility, the resolution of the security problem is difficult at best.

Wide area networks provide an environment in which intruders (often referred to as hackers and crackers) form a well-connected community and use network services such as email, netnews, and bulletin boards to quickly distribute information on how to exploit vulnerabilities in systems. From hobbyists to serious attackers, the intruder community dedicates time to developing malicious programs and sharing information. They have even developed their own publications, and they regularly hold conferences that concentrate specifically on tools and techniques for defeating security measures in networked computer systems.

In contrast, the legitimate, often overworked system administrators and users on the network frequently find it difficult to take the time and energy from their normal activities to stay current with this information, much less design patches, workarounds (mediation techniques), tools, policies, and procedures to protect the computer systems for which they are responsible. Moreover, the legitimate network community is not as well coordinated as the intruder community; administrators and users work independently, trying to protect their own systems from harm as best they can. Figure 4-2 illustrates how our incident handling activities address the immediate problems facing the network community. When compromised sites contact us, we draw on our incident records, vulnerability analysis results, and seven years of experience to help the site.With each incident, we add to our store of knowledge. During our incident handling activities, we may interact with vendors, other response teams, and technical experts. The outcomes of our effort include guidance on practice and software patches developed by vendors. In addition to working with the compromised site, we package our knowledge into advisories, bulletins, and checklists (described later in this section) and disseminate them broadly to the network community, thus helping to raise the level of security overall.

4.1.2 Customers

Customers of the Incident Handling activity area include network service providers, security and system administrators, operations managers, and incident response teams. Individual users also look to us for information and advice.

4.1.3 Rationale

Incident handling activities help system administrators and managers at sites affected by security incidents to deal with issues they have never faced before. In many cases, our effort limits the damage by enabling site personnel to stop an intruder before they would have otherwise detected the intrusion. Because we gather data from many domains and synthesize it in our incident and vulnerability analysis work, we are often able to understand the scope of the problems and to identify patterns and trends that are not evident to those whose information is limited to one domain or one aspect of a problem.

The Internet is doubling in size each year, and the incident rate is doubling with it. Thus, incident response coordination is essential if the 80,000 new Internet users who appear each month are to get help. Many of these new users are not aware of the threats and are ill prepared to deal with security incidents when they occur.

In addition, our incident handling work allows us to maintain a first-hand understanding of the state of the practice of information and computer security. The work helps us understand the root causes of the problems as well as the effectiveness of tools and techniques aimed at dealing with the problems.

4.1.4 Benefits

When a security breach occurs, the Incident Handling staff helps affected sites to identify and correct problems in their systems and to develop system safeguards and security policies. We

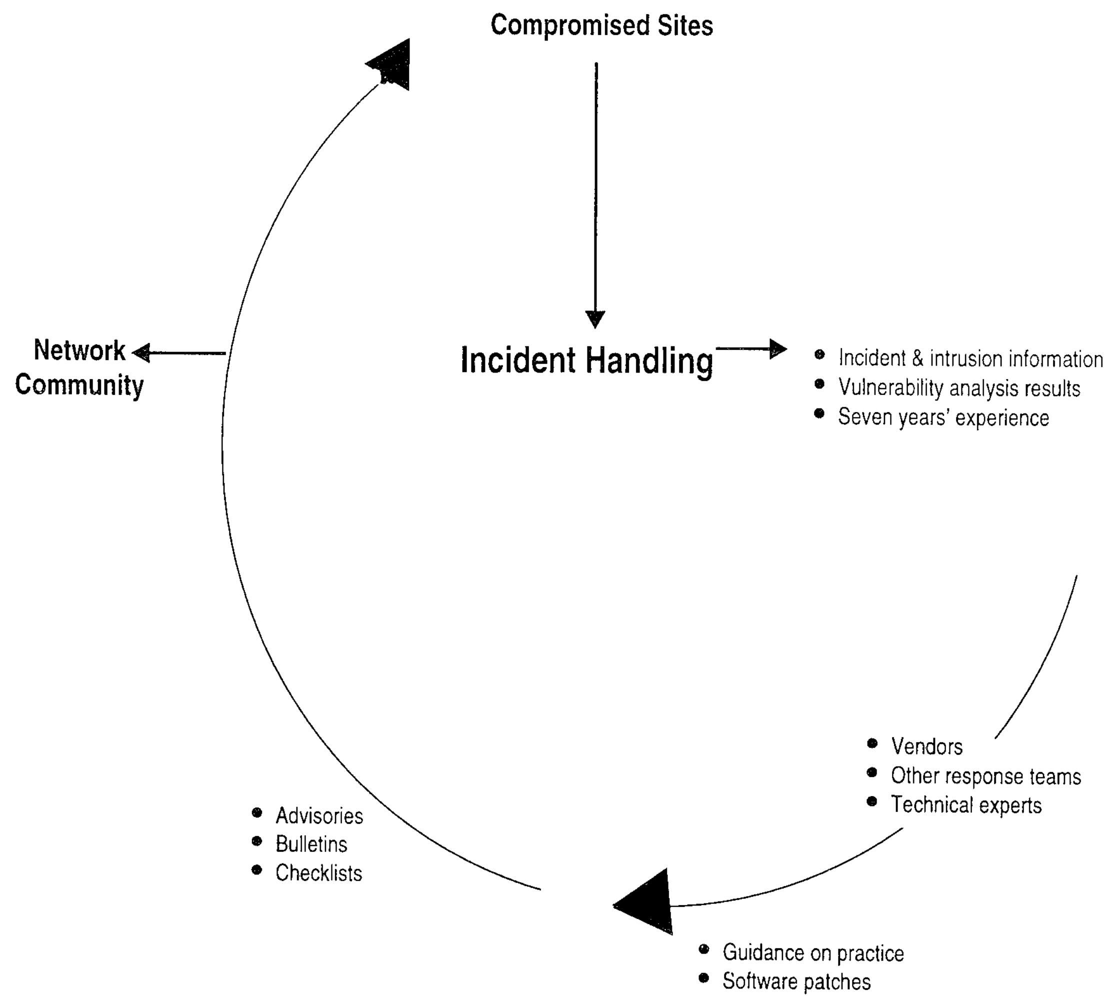


Figure 4-2: Solving Immediate Problems

coordinate with other sites affected by the same problem, work with computer vendors to identify and correct deficiencies in their products, and, when an affected site explicitly requests, work with law enforcement and investigative agencies. When new problems are discovered, or widespread attacks develop, we issue advisories alerting the Internet community to the problem and recommending steps to prevent or recover from an attack.

The lessons learned through the Incident Handling activity area are captured and made available to users of the Internet through a public archive of security information and products. The archive includes security tools, a security checklist, all our advisories, and "tech tips," along with answers to frequently asked questions.

In the future, a compendium of intruder tools and exploitation scripts will increase the efficiency and effectiveness of incident handling activities, thus benefitting the network community as a whole.

The trend chart for this activity area is presented in Figure 4-3.

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Incident response teams	• CERT and 43 other teams active	• 50% of major service providers have response teams that handle routine, limited-scope incidents	• All service providers and many major sites have response capability	• Time to resolve incidents decreases
Incident response guidelines and policies	• Incident response teams follow documented policies and practices	• Service providers agree on incident handling policies and practices • Guidelines are widely available	• 50% of major service providers require evidence of policies before connection	• Percentage of sites with policies; percentage of users able to state policies
Vulnerability corrections	• Vendors' response uneven, with much variance in time to correction	• Major vendors' response is uniformly fast • Increased electronic distribution of corrections	• Major vendors have fast channels to their customers	• Time from vulnerability report to correction in customer hands

Figure 4-3: Trends in Incident Handling

4.1.5 One-Year Objectives for 1996

During 1996 it is our plan to

- Continue our current level of incident response while refining procedures to cope with the continuing increase in security incidents and to support rapid problem identification, classification, and resolution.
- Expand our contacts with the network service provider and service user communities, and use those contacts to establish additional incident response teams in those communities.
- Establish incident handling policies, practices, and environments that the service providers share. Extend our vendor contacts by building working relationships with network technology producers and vendors to more effectively advise them of security deficiencies in their products, help them to resolve the problems, and facilitate the distribution of corrections.

4.1.6 Baseline Work Outputs

There is currently no basic funding for the Incident Handling activity area; therefore, there are no baseline work outputs.

4.1.7 Proposed Add-On Work Outputs

There are no proposed add-on work outputs in the Incident Handling activity area.

4.1.8 Related TO&P Activities

The Incident Handling activity area is funded entirely by a TO&P with the Advanced Research Projects Agency (ARPA) Computer Systems Technology Office (CSTO). Our sponsor has charged us to concentrate primarily on incident handling and vulnerability analysis. We have listed potential work outputs below, though our ability to develop them depends upon the resources that are available after we have met our primary responsibilities.

Potential Work Outputs

Incident Response Guidelines

Incident response guidelines for vendors, technology developers, and system integrators will contain the information needed for these groups to work with incident handling teams to eliminate security vulnerabilities in a timely manner. This includes training material so that the vendor, technology developer, or system integrator will be prepared to deal with the rapid resolution of a security vulnerability in emergency response situations.

CERT Advisories

A CERT advisory is a communication to the network community alerting them to vital security information. An advisory may alert readers to a specific set of ongoing activities that are occurring within wide area networks, or it may describe a vulnerability that is being exploited and recommend corrective actions for sites to take. In developing advisories, our staff works with vendors of the exploited technology and coordinates response to the problem with other security experts. The advisory is widely distributed through mailing lists and newsgroups, and copies of all advisories are available by anonymous FTP (File Transfer Protocol).

CERT Vendor-Initiated Bulletins

A CERT vendor-initiated bulletin is a means to distribute vendor-specific security information to a wide audience efficiently. These bulletins contain text written by vendors about a security problem in their products that requires immediate attention. The CERT staff works with the vendor to ensure that the potential bulletin meets criteria similar to that for our own advisories, such as identification of the scope and impact of the problem and sufficient information for system administrators to take action. Bulletins are distributed to the same audience as CERT advisories.

Vendor Workshop

The Vendor Workshop provides a forum in which technology producers (vendors) come together to address important security issues. A primary goal of the workshop is to apprise vendors of emerging threats, as identified through our handling of security events, and to help stimulate the move to improved security in commercial products.

Handbook

In cooperation with several other response teams, we will begin this year to catalog tools and exploitation scripts used by intruders and describe symptoms of their use. The resulting handbook will ultimately include information about how to respond to intrusions involving those tools and how to protect systems from them. It will enable incident handling personnel to be more effective and efficient in responding to intrusions and in helping sites protect their systems.

4.2 Security Improvement Tools and Techniques

4.2.1 Problem Statement

Over the past six-and-a-half years, the CERT Coordination Center has worked to identify and address security problems in the Internet community, and has handled thousands of security incidents. These incidents have usually occurred because organizations don't fully understand the complex technical and administrative issues that affect the security of their information infrastructure. Over the years, information infrastructures have evolved and expanded from homogenous mainframe environments to environments that are both heterogenous and distributed. Not only is the technology distributed, but the administration of the technology is also distributed, leading to a fragmentation of system administration policies and practices. Typically, staffing is not kept at sufficient levels to allow system administrators to maintain knowledge relating to all the new technologies being incorporated into the environment. It is not surprising that what often results is an infrastructure burdened with a collection of configuration errors and product vulnerabilities that make the organization highly vulnerable from a security standpoint.

Until the engineering practices of technology producers and product vendors are changed to include security engineering techniques and stronger security assurance practices, there will be an ongoing need to surround information system products with policies, practices, and tools that reduce operational systems' vulnerability to attack—tools that help a site understand the configuration and state of their networks and systems, that can be used to audit the network behavior of sites, and that assist administrators in configuring systems and networks.

Figure 4-4 illustrates how our security improvement activities address this need. We work primarily with security-conscious sites, which are concerned about protecting their systems from intrusions. We draw on incident handling data, past experience, and the results of site evaluations and technology evaluations to develop tools and techniques for protecting networked systems. After testing at selected test sites, we incorporate the tools and techniques into our courses, checklists, and guidelines for the network community.

4.2.2 Customers

Customers of the Security Improvement Tools and Techniques activity area include system acquisition organizations, security and system administrators, system managers, network service providers, and software and technology producers. Ultimate beneficiaries are all users of networked systems.

4.2.3 Rationale

For the community to have confidence in existing networked systems and for the state of security to improve in those systems, proactive activity is required in two areas: 1) training to raise awareness of security issues and build skills in using tools and techniques that improve security and 2) development and transition of tools and processes that can be used to improve security. In response to this problem, we will produce a set of tools and guidelines that will help the Department of Defense (DoD) and critical national infrastructure organizations address their immediate, acute needs. Because the security of infrastructures can be maintained only through an ongoing activity that addresses evolving technologies and changing threats, future activity in this area will address these changes.

4.2.4 Benefits

Our planned training activities increase the Internet community's awareness of information security and computer security issues. By drawing on our incident handling experience, we are able to produce seminars and courses that are both pragmatic and relevant to customers. Customers receive state-of-the-practice knowledge accompanied by guidelines for applying that knowledge to their situation. More importantly, increased awareness will lead customers

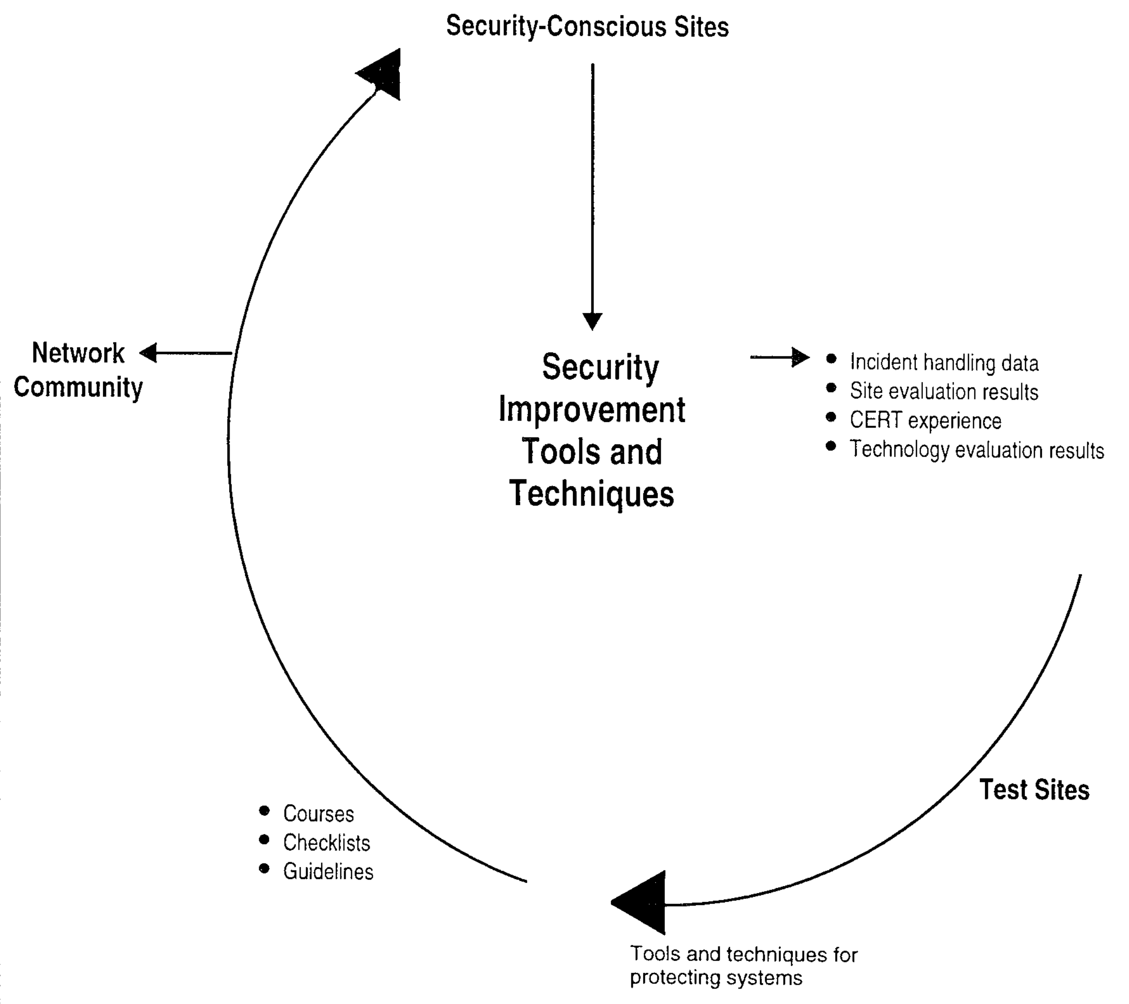

Security-Conscious Sites
Security Improvement Tools and Techniques
• Incident handling data
• Site evaluation results
• CERT experience
• Technology evaluation results
Test Sites
Tools and techniques for protecting systems
• Courses
• Checklists
• Guidelines
Network Community

Figure 4-4: Protecting Systems

to expect products with improved security characteristics. This change in customer attitude is necessary to give technology producers and vendors the incentive they need to invest in improving the security attributes of their products.

As awareness increases, there will also be an increased demand for tools and processes that improve the security of existing systems. Because the systems used for our incident handling work are an attractive target for would-be intruders, we have an excellent source of requirements and a testbed for new technology that tests and enhances security in existing systems.

Thus, our customers get tools and techniques that have already been tested in practice in our complex and high-pressure incident response activities. We will develop security tools and techniques incrementally, as we gain opportunities to test them in practice with organizations that collaborate with us. This will give us first-hand information about the needs of our customers and the success of our approaches. The test sites will benefit from the knowledge we bring from incident handling and our experience with other organizations. Finally, by distributing our tools and techniques widely, we help to raise the level of network security without undue investment of time and effort on the part of individual sites.

By anticipating problems and exploring solutions, we can promote good security practice and the use of effective technology as the Internet continues to expand. The ultimate benefit is an increase in the network community's confidence in the Internet, and thus the NII.

The trend chart for this activity area is presented in Figure 4-5.

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Information and systems security policies and practices	• Limited number exist	• Model policies and descriptions of best practice are widely available	• Major sites have implemented policy and policy awareness programs	• Percentage of users able to state policies
Tools to enhance and monitor system security	• Small number of public domain enhancement tools exist • Prototype monitoring tools in test	• Enhancement and monitoring tools available • Service providers and major Internet sites routinely use the tools	• Tools are available to the broad community through network service providers and components of the network infrastructure	• Percentage of sites with tools in use
Product security guidelines	• Little attention to security • No documented guidelines	• Product security guidelines are available and are in use by 50% of the major vendors	• Vendors routinely follow guidelines to improve the security of their products	• Security metrics

Figure 4-5: Trends in Security Improvement Tools and Techniques

4.2.5 One-Year Objectives for 1996

During 1996, we plan to

- Develop information packages that raise the community's level of understanding of technologies and administrative practices that improve security.
- Collect and distribute public domain tools that allow system administrators to test for the presence of known vulnerabilities and security weaknesses in their system configurations.
- Distribute network monitoring tools that allow system administrators to verify that their systems are being operated in accordance with their security policy and to detect anomalous behavior.
- Work with network service providers to broaden information and tool distribution mechanisms for disseminating information broadly.
- Conduct the second annual management conference on information and system security.
- Develop guidelines on product security characteristics and features that are required for secure operation on wide area networks such as the Internet.

4.2.6 Baseline Work Outputs

The Security Improvement Tools and Techniques activity area currently has no basic funding and, therefore, no baseline work outputs.

4.2.7 Proposed Add-On Work Outputs

In this section we describe the proposed add-on work outputs for the activity area related to security improvement tools and techniques. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

TS-1A Security Risk Taxonomy (1996-1997)

The security risk taxonomy is a practical classification scheme that contains a hierarchy of security attributes (for example, security policy, security architecture, and network security management). Those doing security risk evaluations can use the taxonomy as the basis for evaluation guidelines for determining the security of products and environments, whether they want to evaluate a single application or collections of networked systems. A prototype version of the taxonomy has already been developed, and several field tests have been done. In 1996, we will analyze the data collected from these and additional field tests, and will continue to develop the taxonomy with a focus on producing a set of metrics. We will use the taxonomy to define specific ranges of the allowable values and map these values to a scale or set of scales that are linearly ordered. In 1997, we will publish the taxonomy and a process for using it.

Purpose

One purpose of our taxonomy work is to provide a framework to guide our development of improvement tools and techniques so that we can ensure that each element of the security taxonomy is addressed by defined measurement techniques and associated improvement strategies. Additionally, the taxonomy will provide a framework to guide technology developers in specifying and developing software that can be used as the foundation for secure operational systems.

Customers

The customers for this work are information security practitioners and technology developers.

Collaborators

We plan to collaborate with the National Science Foundation and Federal Bureau of Investigation.

Approach

1. Conduct field tests using an existing prototype taxonomy.
2. Analyze field test data and incorporate it into the prototype.
3. Checkpoint 2Q96: Evaluate the stability of the taxonomy based on the number of changes required after field testing. Determine if more field tests are needed or if the taxonomy is stable enough to proceed.
4. Create an initial set of metrics for self-evaluation.
5. Publish a technical report to transition knowledge to the DoD, civilian, and commercial sectors.

TS-2A Security Risk Evaluation Methodology (1996-1997)

Using the taxonomy developed in TS-1A, we will develop a methodology for evaluating security risks in information infrastructures. In 1996, we will concentrate on review and analysis techniques for policy and procedures. In 1997, we will integrate this work with the output of TS-3A into a comprehensive security risk evaluation methodology that takes into consideration both qualitative and quantitative data. This methodology will enable evaluators to integrate the outcome of policy and procedure reviews with analyses of technical configurations. Finally, we will document the methodology and publish guidelines for using it.

Purpose

Building on a prototype model based on the work of the SEI Risk impact area, we will develop a comprehensive security risk evaluation methodology that will allow an organization to define a desired security state, measure its current state, and make decisions regarding what to do to gain measured improvement.

Customers

Our primary customers are information security practitioners and managers.

Collaborators
We plan to collaborate with the National Science Foundation and Federal Bureau of Investigation.

Approach
1. Conduct field tests of the security risk evaluation methodology.
2. Analyze and incorporate field test experience.
3. Checkpoint 2Q97: Evaluate the stability of the methodology by considering the results of field testing. Determine if more field tests are needed or if the methodology is stable enough to proceed.
4. Codify expertise into transitionable policies and procedures.
5. Develop a set of guidelines that includes a description of the evaluation methodology.

TS-3A Security Analysis Toolkit (1996-1998)
The toolkit is a set of measurement tools that, together, assess the security of an installed technology base and its corresponding configuration. The toolkit will contain both publicly available tools and a prototype network auditing tool developed at the SEI. The currently available network monitoring packages infer network activity through the use of host-based audit trails. Our prototype will audit events on the network itself. It will provide a comprehensive network transaction audit trail for a complete wide-area-network segment using basic auditing types appropriate to network activity. This will provide the necessary abstractions to easily describe and detect network anomalies and attacks that would otherwise be difficult to detect. As an example, a network sweep that scans a number of hosts on a network would require coordination of multiple host-based audit logs to abstract the activity, while a network-based audit trail could easily detect the scan through the network primitives.

Purpose
The purpose of this work is to develop a measurement toolkit for assessing an installed technology base. Using the tools, network administrators will be able to monitor the behavior of their networks in an ongoing fashion. They will get real-time notification of network security events and, over time, data on long-term network use trends.

Customers
Our primary customers are network administrators.

Collaborators
We plan to collaborate with the National Science Foundation and Federal Bureau of Investigation.

Approach

1. Review and analyze publicly available tools.
2. Mature a prototype network auditing tool (Argus) by refining the basic audit types and creating a client program for audit analysis that may be used to support the security analysis of a network.
3. Select tools to be included in the toolkit.
4. Checkpoint 3Q97: Check the tool set against the taxonomy (TS-1A) to determine if coverage of attributes is adequate before integrating the tools.
5. Integrate tools into a tool suite.
6. Develop a guide for using the tool suite.

TS-4A Security Guidelines for Open Systems Acquisition (1996-1997)

There has been increased pressure for the DoD to increase the use and integration of commercial-off-the-shelf (COTS) products. However, most of these products have not been built using comprehensive security assurance, and organizations have little assurance of the security of either the COTS elements themselves or the composite security of the constructed systems and networks.

The security guidelines for open systems acquisition are a set of guidelines developed from the analysis of seven years' vulnerability data identifying product characteristics that have caused problems in the past. The guidelines will describe product attributes that are important to the sustained security of the infrastructure into which they are integrated. These guidelines complement TS-1A and TS-2A. While TS-2A is focused on operational security, this work is product-oriented. Acquisition organizations can use the guidelines to help them select technologies that will minimize the insertion of vulnerabilities into their infrastructures.

Purpose

The purpose of this work is to guide DoD and other organizations in the acquisition of open systems technology, with an emphasis on security characteristics, and to enable them to improve the overall security of their information system infrastructures. A secondary purpose is to influence the development of trustworthy technology.

Customers

The customers for this work are technology acquisition organizations and developers of open systems.

Collaborators

We have no external collaborators for this activity.

Approach

This work is a collaborative effort between Trustworthy Systems and Disciplined Engineering.

1. Analyze seven years' worth of vulnerability data to abstract security characteristics and identify common deficiencies.
2. Checkpoint 1Q97: Evaluate the results of the analysis to determine if a checklist can be created.
3. Based on the analysis, develop a set of product attributes that contribute to the overall security of the technology.
4. Develop guidelines to be used in the acquisition of open systems, including checklists of minimally acceptable security features, auditing mechanisms, security-related software that should be pre-installed on shipped systems, system features that should be enabled by default, and system features that should be disabled by default.
5. Collaborate with the Disciplined Engineering staff to integrate security information into previously developed SEI products on open systems, such as the course and handbook developed in 1995.

4.2.8 Related TO&P Activities

The Security Improvement Tools and Techniques activity area is funded entirely by a TO&P with ARPA CSTO, the same TO&P as the Incident Handling activity area described in the previous section. We have listed potential work outputs below, though our ability to develop them depends upon the resources that are available after we have met our primary responsibilities of incident handling and vulnerability analysis.

Potential Work Outputs

Security Checklists

Checklists for vendors, technology developers, and system integrators will be derived from the CERT vulnerability and incident handling data. These checklists will provide guidelines to help prevent the recurrence of common security flaws introduced by poor software engineering and integration practices, by poor choices for default system configurations, and by undue administrative complexity.

Product Security Guidelines

The product security guidelines will describe characteristics and features needed in products to ensure their secure operation on wide area networks. These guidelines will be developed from our analysis of seven years' vulnerability data to identify underlying causes of security problems we see today. Technology producers will be able to use the guidelines to help them develop products that will be more secure in today's global networks.

Network Management Tool

We plan to beta test a prototype network auditing tool that will help network managers detect security problems. After analyzing the field test results, we will begin the transition of this tool to the wider Internet community.

Incident Impact Survey Report

We plan to conduct a survey of sites that have reported incidents to the CERT Coordination Center in order to learn what impact they have experienced as a result of security incidents. We will publish the results in a report and disseminate it broadly. This concrete knowledge of the potential effects of network breaches will provide organizations with a basis for decisions on assigning resources to network security and will increase their understanding of the potential consequences of insufficient security.

4.3 Trust Technology Maturation

4.3.1 Problem Statement

The CERT Coordination Center daily sees the real-world damage (in the form of system intrusions) caused by the lack of foundation upon which to build sound software engineering practices for today's systems. For example, security enhancements and corrections for security vulnerabilities are typically made in the form of "patches" and are not reflected in the architectural description of the system. This disconnect between the architecture and the implementation of a system all too often has an unexpected and undesirable impact on security, as well as on the other attributes of software quality.

The symptoms of the problems are computer security incidents, but the root causes are failures in the requirements definition, domain modeling, software architecture, software design, software implementation, and software maintenance practices in use today. In the Trust Technology Maturation activity area, we focus on dealing with the root causes of information system security problems by exploring, developing, and transitioning improved software engineering practices to technology producers and vendors. See Figure 4-6 for an overview of our activities. As the figure shows, we work primarily with technology producers and vendors, drawing on our software engineering expertise, security expertise, and research to identify software engineering practices that enhance security. In doing so, we may work with software development and software acquisition organizations. We package our knowledge in the form of technical reports, guidelines, and models, which we disseminate broadly to the software engineering community.

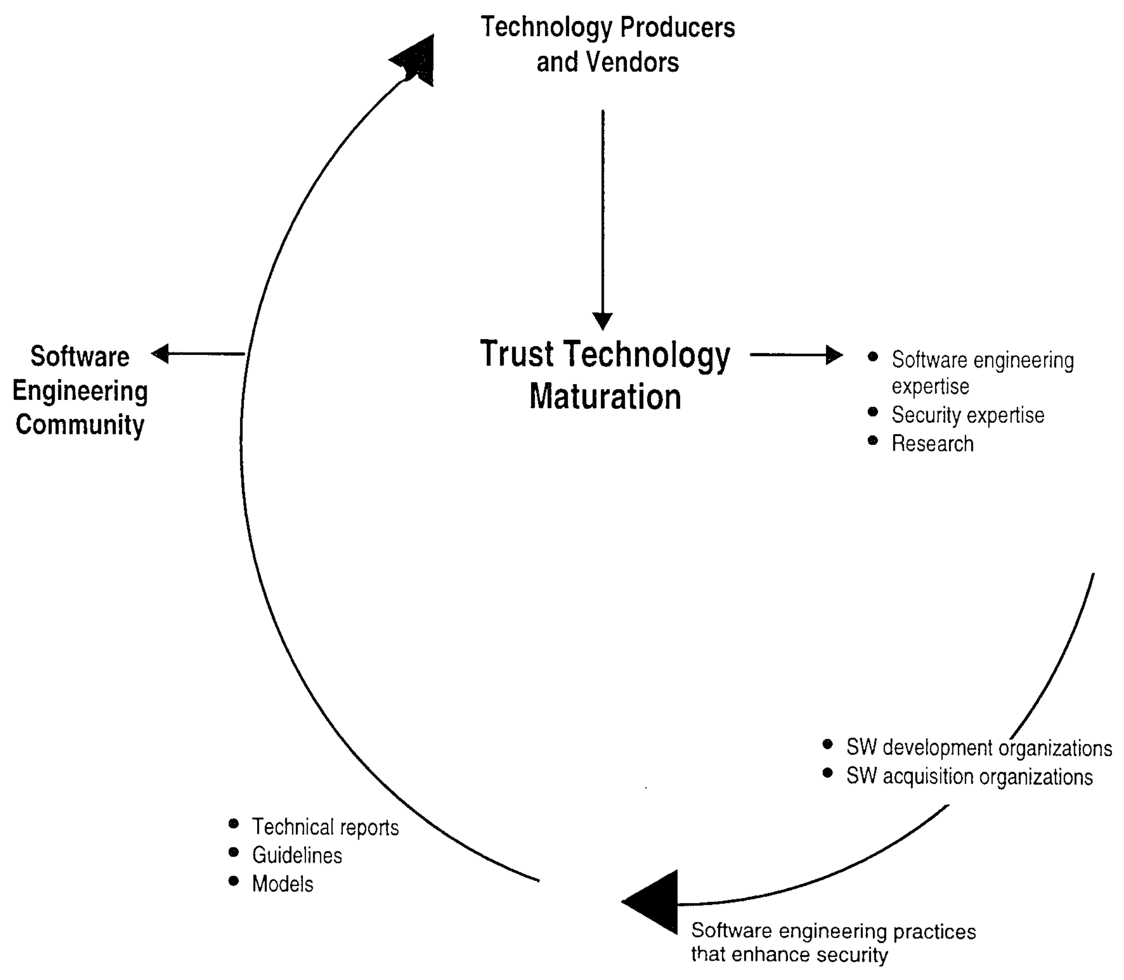

Technology Producers and Vendors
Trust Technology Maturation
Software Engineering Community
• Software engineering expertise
• Security expertise
• Research
• SW development organizations
• SW acquisition organizations
• Technical reports
• Guidelines
• Models
Software engineering practices that enhance security

Figure 4-6: Improving Systems

Once problems are introduced into the network, it is necessary to respond quickly and efficiently. We face this challenge daily in our Incident Handling activity area; thus, we have a real-world testbed for the state of the art in the technology and methods for responding to incidents and vulnerabilities. The Trust Technology Maturation activity area also looks to the future of incident and vulnerability response, developing the technologies and practices to transition to the rapidly changing environment of unbounded systems. In unbounded systems, boundary conditions cannot be fully determined; that is, it is not possible to identify, define, and characterize the extent of administrative control over a domain, all access points to a system in the domain, or all signals that may appear at those access points. These characteristics must be considered if a system is to operate securely.

4.3.2 Customers

Our customers are software engineers and software engineering managers, software engineering improvement groups, and software acquisition organizations that affect the software development practices of their contractors. Other customers include fledgling response teams, security managers, network service providers, and researchers who rely on the results of the long-term analysis of security engineering, vulnerabilities, and incidents.

4.3.3 Rationale

For the community to have confidence in networked systems and for the state of security to improve, there must be a long-term view of how security will be integrated into new products and techniques. Our tasks and products will provide this two-to-five year focus through the maturation of technology such as security engineering, vulnerability analysis, and domain analysis.

To produce new security architectures and tools that will secure the critical military and government use of unbounded systems, it is necessary to extend the SEI work in domain analysis models, software and network architectures, incident response technology and practice, and software engineering tools development. Our work will expand the SEI expertise in these areas. To accomplish this, we will use a model of the future Internet and extrapolate intruder activities and vulnerability information to this model.

4.3.4 Benefits

The DoD and other communities will be able to use the information we collect to identify critical security elements and to characterize the threats and risks of using the Internet and the future NII for government traffic. Our work ultimately will allow the DoD and other beneficiaries to operate highly secure, bounded systems (with a known degree of risk) within larger, less secure, unbounded networks over which they have no administrative control.

Within five years, the tools and architectures based on the results of this work will be in widespread use in the Internet or in whatever unbounded system technology replaces the Internet for DoD and government use.

The trend chart for this activity area is presented in Figure 4-7

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Security in software architecture, design, and implementation	• Little attention to security in any phase of software engineering practice	• Initial set of software engineering practices is available for avoiding vulnerabilities	• Security considerations are routinely integrated into requirements, domain models, and architectural specifications	• Percentage of security incidents caused by errors in software architecture, design, or implementation
Tools for analyzing security in software	• Do not exist	• Prototype static and dynamic analysis tools emerge	• Early adopters use static and dynamic analysis tools	• Percentage of software development efforts using tools
Security level exhibited in commercially available products	• Little attention to security • Security often implemented as post-design patches	• Security model and metrics are available to guide design activities	• Early use of key practices that support measurable improvement in security	• Security metrics

Figure 4-7: Trends in Trust Technology Maturation

4.3.5 One-Year Objectives for 1996

It is our plan in 1996 to

- Develop a database support system for vulnerability and incident analysis.
- Create an initial set of engineering practices that software developers can use to avoid introducing vulnerabilities into their products. Examples of these practices are the use of code analysis tools and code review techniques that focus on security.
- Investigate alternative security architectures.

4.3.6 Baseline Work Outputs

The Trust Technology Maturation activity area currently has no basic funding and, therefore, no baseline work outputs.

4.3.7 Proposed Add-On Work Outputs

In this section we describe the proposed add-on work outputs for the activity area related to trust technology maturation. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

Background

Currently, little of the basic technology in security engineering and system integration applies to unbounded systems but, instead, assumes that the capability exists to identify, define, and characterize the extent of administrative control over a system, all access points to that system, and all signals that may appear at those access points. In unbounded systems such as the current Internet and future NII, these *boundary conditions* cannot be fully determined. On the Internet today, the cornerstone of security is based upon the notion of a firewall, which is an attempt to create a logically bounded system within a physically unbounded one. Our research will demonstrate that "bounded-system thinking" within unbounded domains leads to security designs and architectures that are fundamentally flawed. The firewall concept in particular is severely limited and can be circumvented easily by exploiting the fundamental differences between bounded and unbounded systems. We approach ensuring security on the Internet and the future NII through a paradigm shift toward security engineering using tools and architectural techniques built on domain models that are derived through the analysis of unbounded systems. Further information about this approach is included in the proposed work described below.

TS-5A Description of the State of Security Architectures of Unbounded Domains (1996)

We plan to publish a survey report describing the current state of the art in security architectures (theory and practice). It will include related research in domain analysis and nonfunctional attributes. The report will identify significant gaps and recommended research areas/directions that are not currently under investigation. This effort serves as a feasibility study for work in TS-7A.

Purpose

The purpose of this work is to describe the current state of the art in security architectures and to determine the feasibility of the work described in TS-7A. The report we publish will be used as background and justification for work we believe is urgently needed in this area. We expect to infer from case studies that traditional bounded security concepts cannot be effectively extended into unbounded domains, leaving a huge gap in software engineering theory that must be filled before we can put even greater reliance on the Internet and the future NII than we already do.

We expect to validate the hypothesis that there is currently a gap in the theoretical foundation needed to perform security analysis and modeling of unbounded systems such as the Internet. Current work in security domain analysis is either entirely focused on isolated networked systems or, instead, makes assumptions to approximate a bounded system in what is actually an unbounded domain.

This work will extend the expertise of the SEI in model-based engineering and architectures and give the software engineering community a seminal source for unbounded domain analysis. This report will also be used as the basis for an essential paradigm shift from making uniformly bounded assumptions about the characteristics of a system to focusing on a more unbounded domain for securing systems such as the Internet.

Customers
The primary customers for this work are DoD researchers concerned with moving applications and systems from closed government networks to open networks like the Internet.

Collaborators
We do not have any collaborators at this time. To the best of our knowledge, no work is currently being done in this area.

Approach
1. Collect the current state-of-the-art security research in domain analysis, nonfunctional attributes, and architectures.
2. Coalesce work in the Security Improvement Tools and Techniques activity area (in particular, the applied security taxonomy work) with our literature search to help us more readily identify gaps and inconsistencies in the literature.
3. Checkpoint 2Q96: Use the results of this feasibility study to determine if TS-7A should proceed.
4. Publish a report documenting the outcome of our study.

TS-6A Description of Internet Architectural and Domain Elements (1996)
We will develop a description of the fundamental architectural and domain elements that comprise the Internet and that may be used to describe any similar unbounded system. We will base this work on incident and vulnerability data from the Incident Handling activity area and will publish this information in a report on Internet security. We will use our knowledge of domain elements, in conjunction with the output of TS-5A, as the foundation for the modeling effort described in TS-7A.

Purpose
The purpose of this work is to abstract knowledge on architectural elements that make up an unbounded domain. This task illustrates the unique position of the SEI to match theoretical security models and architectures with real-world experiences and the abstraction of a large da-

tabase of actual incidents and vulnerabilities covering the past seven years. This information is considered too sensitive to be transferred to any research effort outside the SEI because the SEI, with Trustworthy Systems, is the only research body able to gather and analyze this information.

Customers

The primary customers for this work will be government organizations concerned with Internet security standards, along with researchers and technology transition agents involved with developing the security technologies and practices on the Internet and NII.

Collaborators

We will collaborate with other security and response teams.

Approach

1. Analyze CERT incident and vulnerability data, and abstract domain and architectural elements from case studies of real-world security breaches.
2. Use the domain and architectural elements to characterize the Internet as an example of an unbounded domain.
3. Use this analysis to validate and/or refute work in the current literature, as well as generate new ideas and avenues for contributing to later model development.
4. Checkpoint 4Q96: Use the defined domain elements, along with the feasibility study in TS-5A, to determine if TS-7A should proceed.

TS-7A Software Security Models and Language Description (1996-1997)

Based on the results of the feasibility study in TS-5A, we will create a predictive security model of unbounded systems in general and the Internet in particular. We will disseminate this work in the form of a report that contains a detailed description of a predictive model for understanding and evaluating security attributes in an unbounded system.

Purpose

The purpose of this work is to provide a detailed description of a predictive model for understanding and evaluating security attributes in an unbounded system. The predictive nature of the model will allow us to derive security characteristics from the specific architectural descriptions of model elements in a simulation of an unbounded system (such as the Internet). The model (or modeling technology) will be a step toward creating more secure architectures for the Internet, the NII, and future unbounded systems.

The use of firewall technology has been a cornerstone in the protection and fight against intruders. We believe firewall technology will not be sufficient to guard against security threats in the future networking environment. To address the changing environment of users, new network services, and emerging technologies, we will focus on the development of a new security paradigm to replace firewall protection.

Customers

Our customers are the DoD and other government agencies, designers and administrators of specific portions of an unbounded system, and the network community as a whole.

Collaborators

We do not have collaborators for this activity.

Approach

1. Abstract our current vulnerability records into classes of architectural and implementation practices that lead to specific vulnerabilities.
2. Link this analysis with an analysis of intruder behavior to help determine the real-world security impact of specific architectural and implementation practices.
3. Checkpoint 2Q97: Use the security impact of these practices as a basis for determining if a predictive model can be accomplished in a reasonable time.
4. Create a predictive model based on the abstraction and classification of vulnerabilities and their links to real-world intrusions; extrapolate behavior and system vulnerabilities for future systems.

TS-8A Software Engineering Framework for Trustworthy System Development (1996-1997)

In this work, we will identify key security tradeoff decisions that must be made during the design and development of software. The result will be a description of an engineering technique that will identify the key decision points that must be made while defining the architecture of a product. This technique, when used in conjunction with a process model such as the trusted capability maturity model (T-CMM), will enable software designers and developers to easily evaluate the impact of security tradeoffs.

Purpose

The purpose of this work is to develop a security engineering framework by applying the lessons learned from the experiences of the Disciplined Engineering staff in developing engineering frameworks for performance engineering and dependability engineering. As a result, we will demonstrate that the engineering framework approach can be used to codify best practice in areas of software engineering other than those of the original two frameworks; that is, provide a proof of concept.

Customers

The customers for this work are software engineering practitioners and security professionals. Standards communities are also potential customers.

Collaborators

We do not have external collaborators for this activity.

Approach

This work will be performed in close collaboration between Trustworthy Systems and Disciplined Engineering. The two components of the work are codification of security practice and development of a guide for developing engineering frameworks.

1. Abstract the security practice knowledge from expertise in Trustworthy Systems.
2. Collaborate with the Disciplined Engineering staff to document the lessons learned in the development of their first two engineering frameworks.
3. Checkpoint 2Q96: Determine from the lessons learned if a guide for creating such frameworks can be produced.
4. Develop a guide for creating such frameworks.
5. Checkpoint 1Q97: Determine if the guide can be used to develop an engineering framework for security.
6. Use the guide to develop an engineering framework for security.

This work is contingent upon the development of the technical engineering frameworks in DE-12B and DE-13B.

4.3.8 Related TO&P Activities

The Trust Technology Maturation activity area is currently funded by a TO&P with ARPA CSTO, the same TO&P as the Incident Handling activity area described in an earlier section. The funding has allowed us to maintain awareness of the current state of the art and practice.

We have listed potential work outputs below; our ability to develop them depends upon the resources that are available after meeting the primary responsibilities of incident handling and vulnerability analysis.

Potential Work Outputs

Set of Tools and Database Schema

The tools and schema will help incident response teams to analyze, coordinate, and abstract vulnerability and incident handling information within their operational environment. The tools will build on the current operational environment of the CERT Coordination Center, adding data abstraction techniques and procedures in order to mature the technology of incident handling. We will then deliver the tools and schema to other incident response teams.

Common Practices Summary

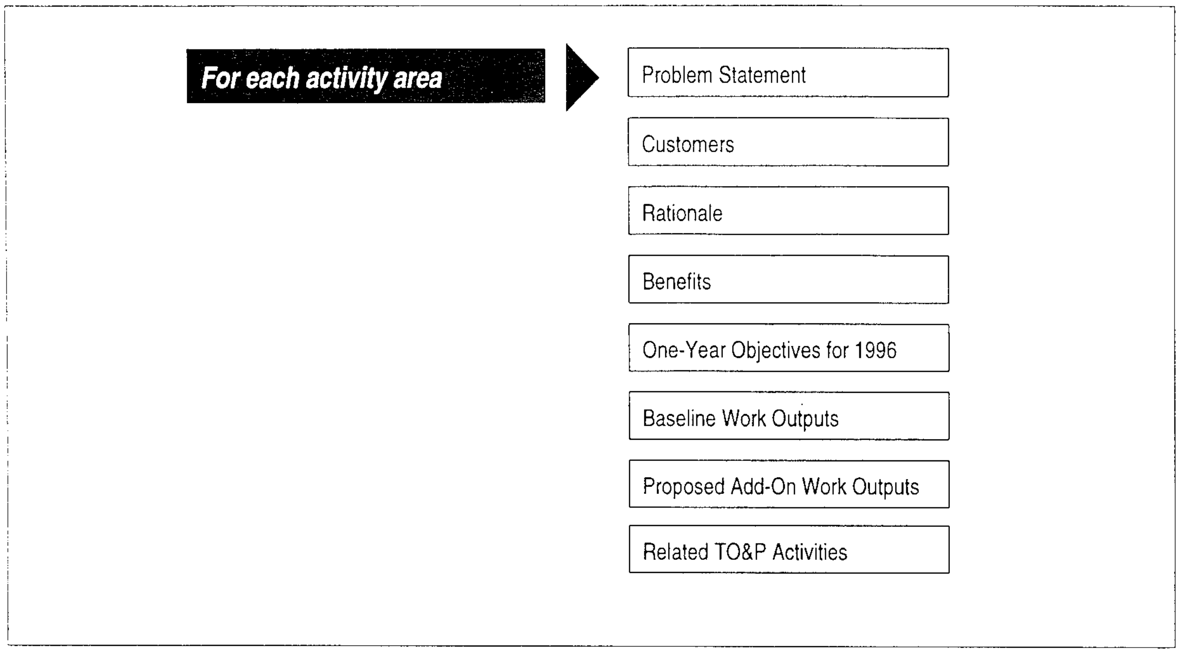
This summary will describe common design and coding practices that lead to the introduction of vulnerabilities in software. To produce the summary, we will apply the tools and techniques described in the previous paragraph to data from CERT incident response, vulnerability analysis, and code analysis.

Volume II
Chapter 5 Table of Contents

5 Professional Infrastructure

The SEI is working to mature the software engineering professional infrastructure because a mature profession substantially improves professional practice and leads to higher quality in software systems. Figure 5-1 shows changes from last year's activities to this year's.

In this chapter, the sections for this impact area are as follows:



5.1 Maturing the Professional Infrastructure

5.1.1 Problem Statement

As described in Section 3.6.3 of Volume I, a profession comprises nine components: initial professional education, accreditation of education, skills development, certification, licensing, professional development (continuing education and training), code of ethics, code of practice, and professional society. Each component can exist at one of several levels of maturity. Presently, the components of the software engineering profession are all at the lowest or "ad hoc" level.

Our solution is to accelerate the maturation of the nine components. However, the components do not mature independently; there are complex dynamic relationships among them. For example, initial professional education prepares an individual for professional practice and certification. Professional development for individuals improves professional practice. A code of practice influences the curriculum for initial professional education. Figure 5-2 shows some of these relationships for the software engineering profession. The figure also illustrates the central premise of this work: the nine components contribute to improved professional practice, which in turn determines the quality of software systems.

The maturation of all the components is a large and lengthy task. As described in Section 3.6.3 of Volume I, the SEI has chosen to work (in 1996 and 1997) on only three problems related to the maturation of selected components.

Code of Practice. Throughout its history, the SEI has been asked repeatedly for straightforward advice on how to "do" software engineering. Some of our most successful efforts, such as the Capability Maturity Model and the Ada Adoption Handbook, have provided some answers to those questions. However, much of that work is at a relatively high level, or it addresses large organizational practices, without giving guidance to individuals on how to do particular tasks. There is a significant need for specific guidance.

There are several ways (mostly ad hoc) of doing many software engineering tasks. It is important for the maturity of the profession that the software community begin to evaluate these methods and practices, identify the better ones, and recommend them to everyone.

In fact, the maturation of the software engineering profession requires that best practices be widely disseminated and used by software engineers. Education and training are part of the answer, but we first need to identify practices that make up the content for education and training.

Professional Development: Infrastructure. Currently software engineers receive an average of 40 hours per year of education and training. In many organizations, this allotment is used in training employees for their immediate tasks. For example, if a project calls for Ada, C++, or Unix, the 40 hours is likely to be spent on this type of training, with no time remaining to keep up with advances in the rapidly evolving field of software engineering. Technology-driven training is woefully inadequate.

At the same time, there is insufficient infrastructure for educators in industry and government for sharing of material, curriculum definition, and the development of return-on-investment data that convinces management to invest more heavily in education.

Prior to 1987, when the SEI sponsored the first SEI Conference on Software Engineering Education (CSEE), there was no continuing opportunity for educators from government, industry, and academia to meet and share their knowledge and experiences. The need for that kind of forum still exists.

Professional Development: Training. Organizations often find, as a result of a CMM or risk appraisal, that training is a weakness. The Capability Maturity Model^{SM} (CMM) level 3 key process area (KPA) on training programs requires that an organization have a training plan and an infrastructure to support it. As organizations mature, the SEI has been getting an increasing

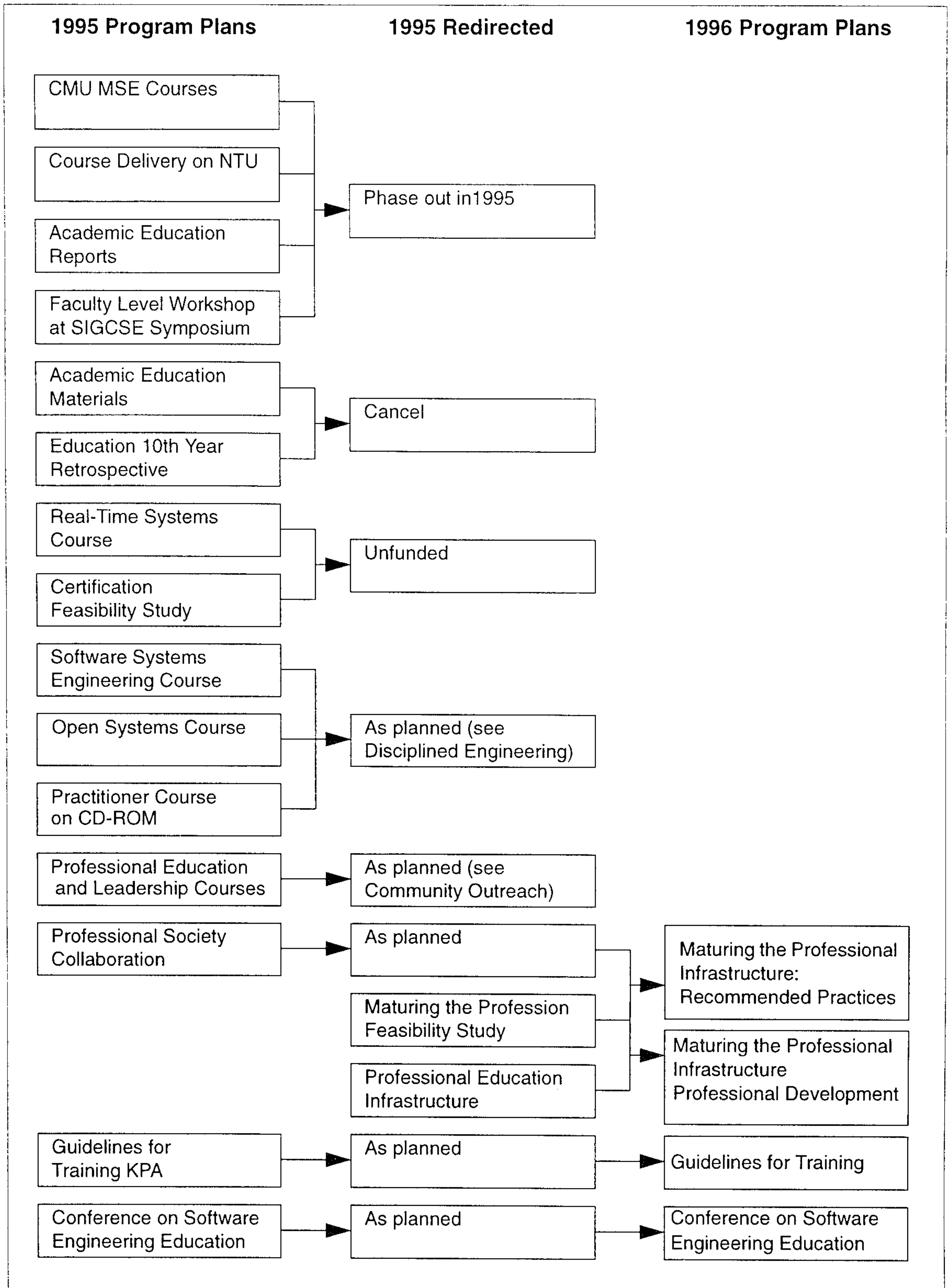

1995 Program Plans
1995 Redirected
1996 Program Plans
CMU MSE Courses
Course Delivery on NTU
Academic Education Reports
Faculty Level Workshop at SIGCSE Symposium
Phase out in1995
Academic Education Materials
Education 10th Year Retrospective
Cancel
Real-Time Systems Course
Certification Feasibility Study
Unfunded
Software Systems Engineering Course
Open Systems Course
Practitioner Course on CD-ROM
As planned (see Disciplined Engineering)
Professional Education and Leadership Courses
As planned (see Community Outreach)
Professional Society Collaboration
As planned
Maturing the Profession Feasibility Study
Professional Education Infrastructure
Maturing the Professional Infrastructure: Recommended Practices
Maturing the Professional Infrastructure Professional Development
Guidelines for Training KPA
As planned
Guidelines for Training
Conference on Software Engineering Education
As planned
Conference on Software Engineering Education

Figure 5-1: Mapping of Professional Infrastructure Activity Areas

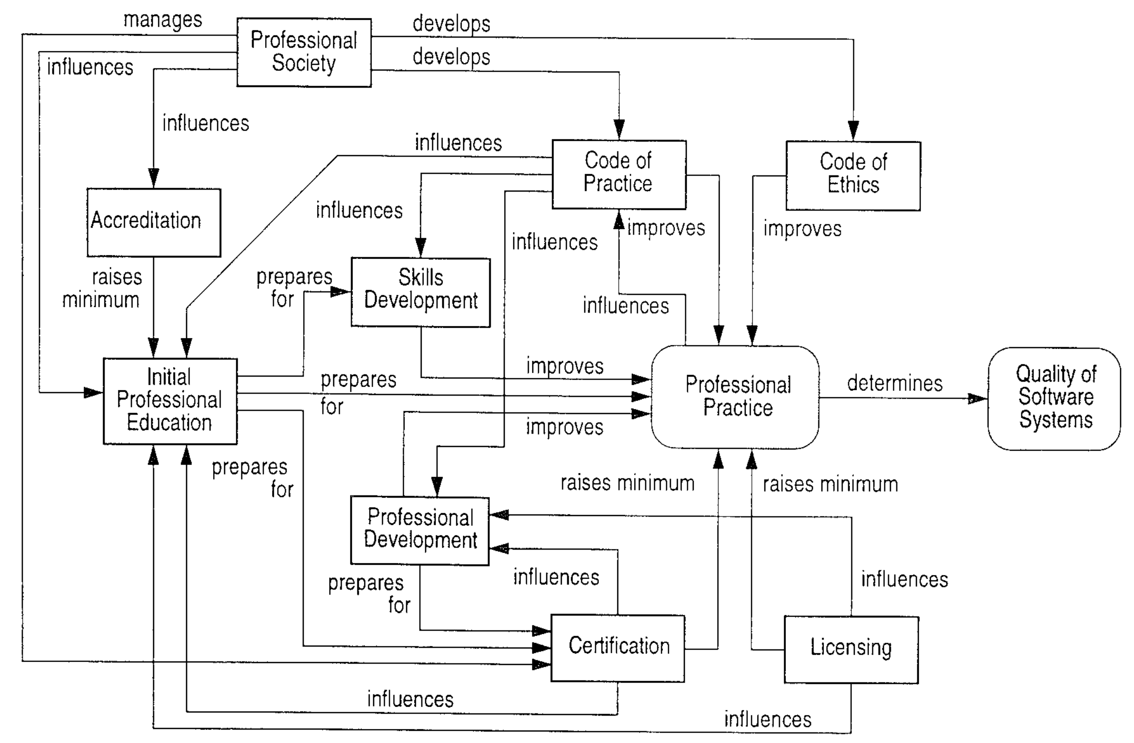


Figure 5-2: Interactions Among Components of the Software Engineering Profession

number of requests for assistance with planning and implementing curricula, recommendations for sources of training, pointers to collaborative efforts to acquire courses, and general guidance on what to teach (needs analysis).

5.1.2 Customers

The entire software community–individual practitioners, managers, executives–are the beneficiaries of a more mature software engineering profession. Indeed society as a whole benefits.

5.1.3 Rationale

Code of Practice. The creation of an initial, partial set of recommended practices will begin to provide guidance to individuals on how to do particular software engineering tasks. The SEI's reputation for providing information and unbiased opinions on software technologies makes us the natural home for this kind of effort. This work, as well as work described earlier in this

Volume, begins to implement one of the recommendations of the 1994 Blue Ribbon Panel: to define and disseminate software practices.

A set of recommended practices accelerates the maturation of several other components of the software engineering profession. In particular, recommended practices are important inputs for designers of software engineering curricula for universities and for continuing education programs. Recommended practices are vital inputs to efforts to define guidelines for accreditation, certification, and licensing.

Professional Development: Infrastructure. The SEI has significant credibility with the software engineering education community because of our previous work. We have the ability to bring together educators from competing organizations and to provide an atmosphere in which the community can work towards common goals for educators in the U.S. software industry, without compromising proprietary corporate information.

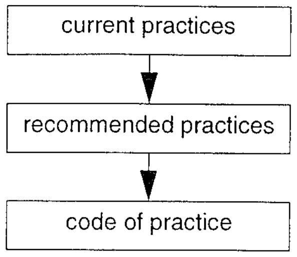
Professional Development: Training. The SEI's customers indicate to us that training is a major client need and that training guidance is necessary for customers committed to process improvement. The 1994 Education Program marketing survey and the evaluation of the 1995 basic funds allocation proposals indicated that there is a demand for the SEI's intervention on software engineering education/training needs. As more organizations achieve CMM level 2 appraisal ratings, they are focusing on the training KPA at level 3, which addresses the establishment of a mature and managed training program for the organization. The SEI's training guidelines add specificity to the interpretation of the CMM by providing organizations with training plan guidance, guidelines for software engineers and their managers for doing training needs analysis, and guidelines for selecting vendors to meet software engineering training needs.

Currently most organizations identify professional development needs through individual practitioners' requests or by project-driven needs for tool-, language-, or method-specific training. The training needs identified are seldom focused on organizational-level issues, such as business strategy, software engineering core competency needs, process improvement goals, accepted career path progression, or software engineering curricula requirements. Only the most mature organizations perform systematic professional education needs analysis and even fewer organizations focus specifically on software engineering needs.

5.1.4 Benefits

A mature profession promotes high standards for professional practice and supports the rapid dissemination of new knowledge. The growth of a mature *software engineering* profession will contribute to substantially improved professional practice and lead to higher quality in software systems. The benefits to the community over time are shown in Figure 5-3.

Maturing all nine components of a profession is too large a task for any one organization. In 1996 the SEI has chosen to focus its resources in the following three areas:



Code of Practice. Not all of software engineering practice is ready for codification, but there are many practices that can now be recommended to the software community. Defining what constitutes a practice (see PI-1B Recommended Practices in Software Engineering) and identifying an initial set of recommended practices are important initial steps along the path from simply documenting current practices in technical reports or research articles toward the creation of a professional code of practice. In addition, a set of recommended practices will not only provide direct guidance to software engineers, but will also have a significant influence on the content of professional education programs in universities, on the guidelines for accreditation of those programs, on professional development goals and curricula, and on possible future criteria for certification and/or licensing of engineers. In fact, it is premature to consider such accreditation guidelines or certification and licensing criteria *without* much more widespread community agreement on recommended practices for software engineers. See Figure 5-3.

Professional Development: Infrastructure. Better professional development for software engineers contributes to improved software practice, and well-defined curricula for professional education contributes to the maturation of the software engineering profession. At the same time, software engineering educators will be recognized as professionals.

The SEI sponsors the Conference on Software Engineering Education. It is the only conference devoted specifically to software engineering education. The CSEE attracts educators, trainers, managers, and administrators from government, industry, and academia. It influences education directions, stimulates new approaches, promotes collaboration, and generates interactive exchanges among all educational stakeholders. Eventually, we expect that as the software engineering profession matures professional societies will assume the sponsorship responsibilities. (See Figure 5-3.)

Professional Development: Training. A mature profession exhibits a commitment to continued professional development. Professional development opportunities should reflect established guidelines for training within recognized career paths. The SEI's guidelines for training of software engineers provide a tested model for designing and implementing training appropriate to the profession and the needs of its practitioners. (See Figure 5-3.)

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 & Beyond	
Professional development: continuing education and training	• Individuals pursue professional development as they determine the need.	• Professional development guidelines (curricula, expenditures per year, etc.) have emerged.	• Nationally published education and training guidelines and curricula exist for career progression.	• There are recognized career paths for software engineers. • It is accepted practice for organizations to invest in professional development (not just in project-specific professional development).
Code of practice	• Some practices are documented in textbooks or company standards.	• An initial set of practices specifically addressing the needs of software engineers is available on the World Wide Web.	• Partial codes of practices are widely recognized, and are regularly reviewed and revised by a continuing body.	• The content of the code of practices strongly influences education, training, certification, and licensing.

Figure 5-3: Trends in Maturing the Professional Infrastructure

5.1.5 One-Year Objectives for 1996

Demonstrate the value and effectiveness of a worldwide-web-based hypertext system to document and deliver recommended practices to practitioners when and where they are needed.

Continue the SEI's participation in national-level policy-setting organizations and committees. Conduct additional workshops on issues that face educators of software engineers in industry and government and take actions identified as a result of the workshops and meetings held in 1995.

Conduct the 1996 Conference on Software Engineering Education, focusing on software engineering as a profession, software engineering curricula, innovative approaches for software engineering courses, industry-academia collaboration, alternative delivery methods, and training and education management.

Complete and disseminate the Software Engineering Education/Training Model for the design and implementation of training. Complete and package software engineering education/training needs analysis tools. Update the *Directory of Industry and University Collaborations with a Focus on Software Engineering Education.*

5.1.6 Baseline Work Outputs

This section describes the baseline work outputs for the activity area related to maturing the professional infrastructure of software engineering. These outputs were approved for 1996 through the basic funds allocation process. Appendix A lists all baseline and proposed add-on work outputs.

PI-1B Recommended Practices in Software Engineering (1996)

This activity results in a prototype hypertext system that exploits worldwide web technologies to make recommended practices available to the software engineering community. This work is the result of a 1995 feasibility study to determine if the SEI can significantly accelerate the maturation of the profession.

Purpose

This activity will demonstrate the value and effectiveness of a worldwide-web-based hypertext system to document and deliver recommended practices to practitioners at their desks when and where they are needed ("just in time").

Customers

All software engineers are customers, as are people involved in software engineering education and training and those developing guidelines for accreditation, certification, or licensing.

Collaborators

The collaborator for this task is the Institute of Electrical and Electronic Engineers / Association for Computing Machinery (IEEE/ACM) Industry Task Force to Establish the Software Engineering Profession.

Approach

Development of a code of practice for software engineering will take many years and involve many individuals and organizations. To build community support for such a broad, long-term, collaborative project, we will build a prototype hypertext system to demonstrate the concept of, and provide a testbed for evaluation of, the worldwide web technology to deliver recommended practices directly to practitioners. More specifically, in 1996 we will take six initial steps.

1. Determine what constitutes a "practice" of software engineering. There are several categories of organizational and individual practices, such as using a particular provably optimal algorithm for a specific class of problem, adhering to an international standard communications protocol, or having a configuration management process in an organization. We will determine how to best represent each category and produce hypertext templates for it.
2. Continue to document the current practices of the software community, which will produce a benchmark against which progress can be measured and produce a candidate set of practices from which the first recommended practices can be chosen.

To accomplish this step, the SEI collaborates with the Industry Task Force of the IEEE Computer Society and the ACM. The task force is working to identify the knowledge, skills, and practices of software professionals using a large international survey. Two SEI staff members are on the task force.

3. Implement a prototype hypertext system using worldwide web technologies to make the practices available to the software community. We have chosen this transition mechanism because no other technology offers the combined benefits of universal accessibility, timely delivery, and ease of maintenance. In fact, because software engineering practices are changing so rapidly, it is only through the use of this kind of technology that creating a meaningful and current set of recommended practices is feasible.
4. Install in the system at least two representative practices from each of the categories identified in step 1. These practices will be selected from recent SEI work. In each case, the presentation of the practice will include such material as a description of the problem that the practice addresses; a description of the practice; a rationale for choosing that practice over others; education and training modules to enable the practitioner to learn to use the practice; and references to supporting information, such as research results, published documents, technologies, and technology vendors.
5. Make a preliminary evaluation of the system. This will include measuring the usage of the system by practitioners, tracking the patterns of navigation to identify how users learn with the system, and interviewing selected users to determine how the system improved their software engineering capabilities.
6. Establish procedures for involving outside experts in the process of selecting and documenting recommended practices. Eventually, we expect that expert external working groups will assume stewardship of the selection of recommended practices in specific technical areas.

During the prototyping and evaluation stages (1996), the practices will reside on the SEI web server, but as stewardship passes to distributed expert groups in later years, the nature of web technology will allow those practices to be distributed as well. The SEI will continue to lead and coordinate the documentation of practices, so that the set of practices, as a whole, will appear coherent and integrated to the end user.

PI-2B Professional Development Infrastructure (1996)

We will publish a report on the status of professional software engineering education that will include discussion of existing curricula, organizational models, return-on-investment data, lessons-learned, and innovative approaches.

Through media such as electronic newsletters, newsgroups, and the worldwide web, we will also enable industry and government educators to exchange information about curricula, course materials, successful ways of teaching specific subjects, and other topics related to professional software engineering education and training.

Purpose

This activity will promote the development of an infrastructure among educators in industry and government that emphasizes sharing of material, curriculum recommendations, and the production of return-on-investment data that convinces management to invest more heavily in education.

Customers

The immediate customers are educators in industry and government organizations. Ultimately, however, practicing software engineers are the customers that the SEI is trying to reach.

Collaborators

Collaborators are the National Software Council and workshop participants.

Approach

The SEI will leverage efforts by participating in national-level councils and committees, such as the National Software Council, to ensure that professional education remains visible at that level. We will also use the efforts of national-level councils to help set the agenda for our work with professional educators. For example, the subjects discussed in our 1995 workshops with professional educators surfaced at the National Software Council Workshop in 1994. We expect that the National Software Summit held in November 1995 will suggest issues for us to address with the education community in 1996 and beyond. We will continue to sponsor workshops with industry and government educators on these issues, in order to remove barriers to continuing education and training.

As a follow-on activity to our 1995 efforts, we will work with educators to develop and publish data showing the return-on-investment provided by professional development courses. The objective is to demonstrate to management the benefits of investing in education.

We will select appropriate communication methods to allow educators to communicate and exchange information about curricula, course materials, successful ways of teaching specific subjects, etc. Candidate mechanisms include electronic newsletters, newsgroups, and the worldwide web. We will encourage and assist professional educators to publish their results and lessons-learned in the interest of information exchange. At present, professional educators (unlike academic educators) are not motivated or encouraged to publish results.

We will publish a report (described above) on the status of professional software engineering education.

We have observed that industry/university partnerships are of mutual benefit to increase the amount of software engineering education delivered to practitioners. As opportunities occur, we will broker such partnerships. (See, for example, the *Directory of Industry and University Collaborations with a Focus on Software Engineering Education* described below.)

PI-3B Guidelines for Training (1996)

To date, little research has been done in needs analysis techniques for the competency- and judgement-based skills that comprise a large measure of the software engineer's expertise. This work will synthesize the contributions of performance analysts and relate them specifically to the field of software engineering.

This activity results in

- **training needs analysis** that will capture an analysis of the education/training needs of software engineering and be disseminated to the software engineering community
- **documentation of training guidelines**, which documents the Software Engineering Education/Training Model for addressing software engineering training needs and the lessons we have learned through its application
- **directory updates** of the *Directory of Industry and University Collaborations with a Focus on Software Engineering Education,* which will be made available in hardcopy, and via File Transfer Protocol (FTP) and the World Wide Web.

Purpose

Software engineering training guidance helps designers of training to focus on the specific needs of the software engineering profession by

- providing a model for designing training that is tailored to the needs of software practitioners
- addressing needs analysis targeted toward the unique career profiles of software practitioners
- highlighting university/industry collaborations to support the design and acquisition of software engineering training

Customers

Customers for software engineering training guidelines are industry, government, and civil sector organizations that develop and maintain software, particularly if the organization is at CMM level 1 or 2 and has limited expertise in software engineering education/training within their training departments.

Collaborators

Collaborators are clients with whom we work to refine the Software Engineering Education/ Training Model and points of contact for industry/university collaborations.

Approach

All the work described in this section is ongoing work begun in 1994 or 1995 and will be completed in 1996.

- **Training needs analysis.** Work begun in 1995 to capture guidelines for analyzing the education/training needs of software engineering must be packaged and disseminated. The guidelines address needs analysis with respect to CMM requirements, broad spectrum knowledge and skill required of software engineers, and more narrow software engineering task-related requirements for practitioners within specific application areas. We will package tools and guidance to address this first step in the design of software engineering training programs.
- **Documentation of training guidelines.** During 1995 the focus is to apply and refine the Software Engineering Education/Training Model through collaborative work with clients from various SEI customer service sectors. During 1996 we will broaden the availability of training guidance by documenting the model for addressing software engineering training needs and the lessons we learned through its application. The documentation of the model, suggested supporting tools and techniques, and lessons-learned by applying the model completes the guidelines for training work.
- **Directory updates.** The initial release of the "Directory of Collaborations with a Focus on Software Engineering Education" was in 1994. Updates were made in 1995, and the directory was made available via FTP and the worldwide web. The directory identifies geographically-focused associations that can provide opportunities to share experience and resources for the development and delivery of cost-effective software engineering education. Because new alliances are being formed and the growing number of the organizations listed in the directory, it needs to be updated periodically to reflect new groups, new leadership, and new direction. Two updates will be completed in 1996.

  The directory is used by universities to promote their work and by industry participants to draw attention to their university involvement. For articles published in the IEEE Computer Society Technical Committee on Software Engineering newsletter and the newly founded journal, Software Process Improvement Forum, the directory was used as the sole source of information on industry-university collaborations.

  Providing the directory is an appropriate role for the SEI, since no one else is documenting the growth of these organizations of pointing them out as resources to the community.

PI-4B Conference on Software Engineering Education (1996)

The Conference of Software Engineering Education (CSEE) includes in-depth tutorials, invited keynote addresses, formal presentations of refereed papers, birds-of-a-feather sessions, and many opportunities for informal discussion.

Purpose

The 1996 Conference on Software Engineering Education focuses on software engineering as a profession, software engineering curricula, innovative approaches for software engineering courses, industry-academia collaboration, alternative delivery methods, and training and education management.

Customers
Customers are educators, trainers, managers, and administrators from government, industry, and academia.

Collaborators
Collaborators include the IEEE Computer Society, which is a cosponsor and a transition partner for the publication and distribution of the proceedings; program committee members from the community at large; the Association for Computing Machinery (ACM); and dozens of referees and contributors.

Approach
The 9th CSEE will be conducted April 22-24, 1996, in Daytona Beach, Florida, with tutorials on April 21. The conference proceedings will be published by IEEE.

5.1.7 Proposed Add-On Work Outputs

This section describes the proposed add-on work outputs for the activity area related to maturing the professional infrastructure of software engineering. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

PI-1A Desktop Hypermedia System for Recommended Software Engineering Practices (1996)

This activity will fully demonstrate to practitioners in government and industry the usability of the prototype system developed in PI-1B Recommended Practices in Software Engineering (1996). We plan to extend the system by populating it more fully with SEI-recommended practices selected from rate-monotonic analysis, the Capability Maturity Model, the Systems Engineering Capability Maturity Model, the Ada Adoption Handbook, and CERT advisories. These practices represent a wide variety (mathematical vs. intuitive, individual vs. organizational, etc.) of the kinds of practices that will ultimately make up the recommended set. They will provide enough content to validate the design of the system and to measure the effectiveness of providing recommended practices “just-in-time” to a practitioner at his or her desk.

Purpose
This activity will demonstrate the value and effectiveness of a worldwide-web-based hypertext system to document and deliver recommended practices to practitioners when and where they are needed.

Customers
All software engineers are customers, as are people involved in software engineering education and training and those developing guidelines for accreditation, certification, or licensing.

Collaborators

Collaborators will be practitioners from government and industry.

Approach

Although the prototype hypertext system [described in PI-1B, Recommended Practices in Software Engineering (1996)] will necessarily contain some practices, the value of documenting and delivering recommended practices via web-based hypertext can only be demonstrated by populating the system with a larger set of practices in easily usable form. Therefore, we propose to identify, organize, describe, and install in the prototype hypertext system the SEI-recommended practices described above. We will then evaluate the effectiveness of using our system to deliver these practices directly to software engineers at their workstations. The evaluation will include tracking the accesses to the system by software practitioners and interviews with selected users.

Furthermore, we must include the best practices from the entire software community. We propose to establish the first expert working groups as part of this add-on work, to help identify recommended practices in a few key areas, to assume responsibility for their stewardship, and to validate and make recommendations to improve our design.

5.1.8 Related TO&P Activities

During 1995 action is being taken to pilot the Software Engineering Education/Training Model with a representative client from each of three sectors: government, industry, and other government agencies. As of this writing, there is a potential for TO&P-funded work in 1995, but it is too early to determine whether there will be 1996 work. If such work is found, it will also contribute to the work on the infrastructure for professional development by providing curriculum and return-on-investment data.

Volume II
Chapter 6 Table of Contents

6 Integrated Transition Strategies and Methods (ITSM)

ITSM work falls in two activity areas. The first area, Software Engineering Transition Practices, focuses on disseminating effective ways of transitioning software engineering technology and methods into use. These transition practices are intended to be used 1) by suppliers of software engineering technologies and processes (so they produce products that are more easily adopted), 2) by adopters (so they can more easily and effectively install improved software engineering practices), and 3) by intermediaries who support suppliers and adopters (so they can be more effective in transitioning improved practices). The SEI itself acts as a supplier and an intermediary to our customers, so these transition practices are used internally as well as externally.

Our second activity area, Collaborative Skills in Software Engineering, focuses on identifying and enhancing the skills software engineers and managers need to collaborate effectively. We focus on collaborative skills because the development and maintenance of software systems is a group process. Activities in this area include dissemination of tools, methods, and practices that increase the ability of groups of software engineers to work together more effectively in applying software engineering practices.

Some of the work reported in the first activity area was described in different sections of last year's 1&5 Year plan, as shown in Figure 6-1. The second activity area is new. It covers work that is proposed to start in 1996.

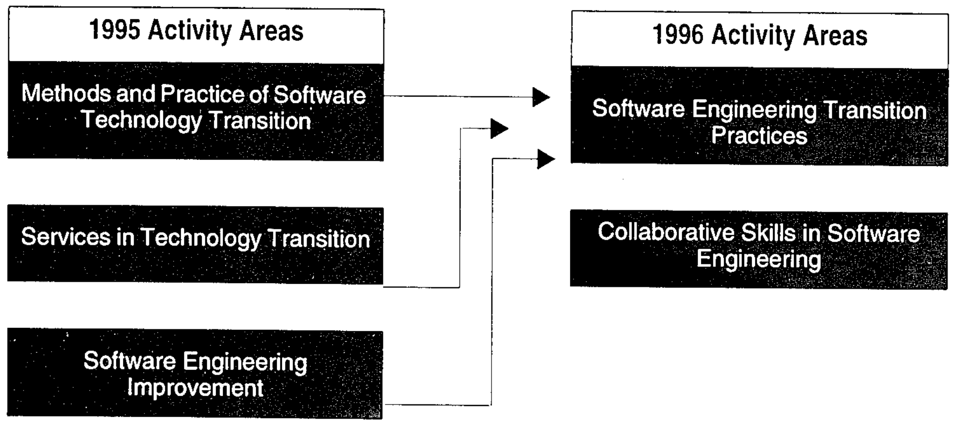
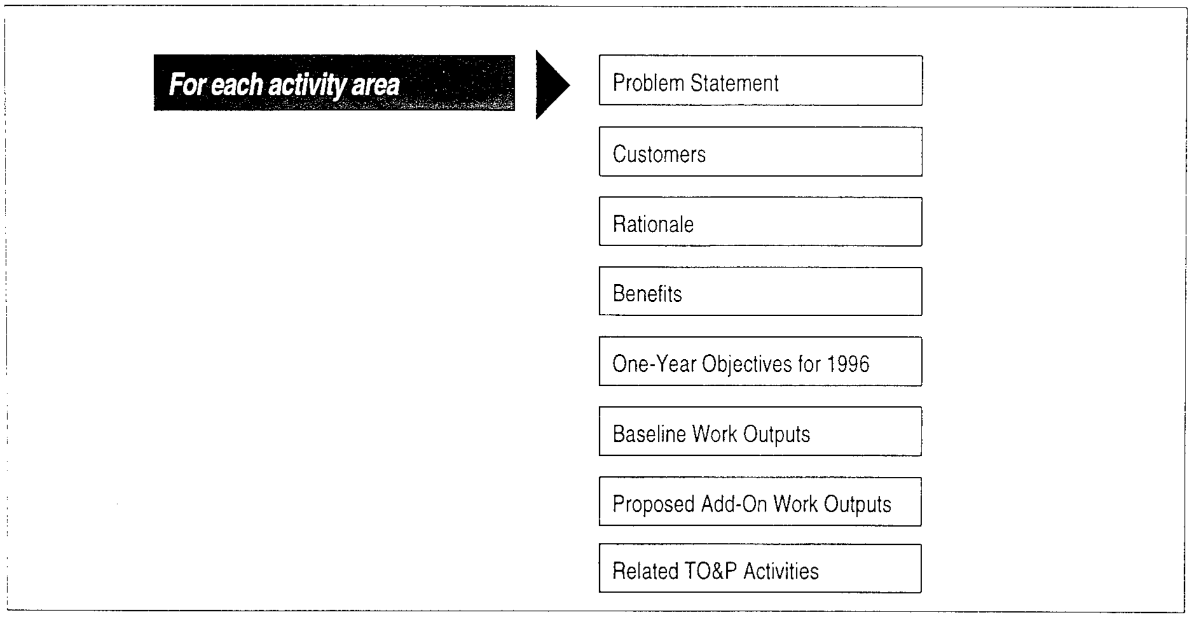

1995 Activity Areas
Methods and Practice of Software Technology Transition
Services in Technology Transition
Software Engineering Improvement
1996 Activity Areas
Software Engineering Transition Practices
Collaborative Skills in Software Engineering

Figure 6-1: Mapping of ITSM Activity Areas

In this chapter, the sections for this impact area are as follows:



6.1 Software Engineering Transition Practices

The SEI collaborates with leading edge organizations to analyze, codify, demonstrate, and disseminate technology transition methods and practices used by the supplier, adopter, and intermediary roles in technology transition. Through direct support to customers, we help organizations initiate and maintain improvements in software development and maintenance practices. We also use our contact with organizations to pilot test and improve transition vehicles and concepts that will later be applied by other organizations.

Software technology transition occurs from the birth of a technology until its retirement. As shown in Figure 6-2, software technology that has been commercially developed and is in use in an organization has most likely been transitioned at least twice, *between* communities respectively concerned with research and development (R&D), new product development, and adoption and implementation. (In addition, the technology undergoes transition as it progresses through its life cycle *within* each community.) Traditionally, these communities have only limited interaction with each other. However, by looking at software technology transition across these established perspectives, we can

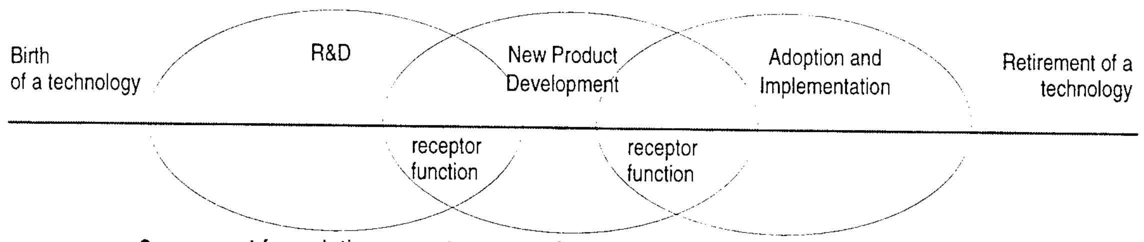
- Identify key leverage points, barriers, and issues in the maturation and transition of software engineering practices.
- Consolidate and build on lessons learned in software technology transition using a common vocabulary and framework.
- Adapt, develop, and demonstrate effective technology transition methods and practices for software technology researchers and product developers (suppliers), adopters, and intermediaries supporting both suppliers and adopters.

Common understanding is particularly important to the development of practical approaches to technology transition. Both SEI personnel and SEI customers must have knowledge and skills in technology transition, based on best practice and the application of proven models, approaches, and methods. By upgrading the software community's capability in this area so technology is used sooner and more effectively, we leverage the role of the SEI and increase its impact.

6.1.1 Problem Statement

When organizations seek to improve their software management processes, their software engineering technology, and the abilities of their personnel to use new software engineering practices effectively, they typically face difficulties in deciding what processes to put in place first, what technologies to adopt, and how to help their staff adopt new ways of developing and maintaining software. Many organizations today know they need to improve their processes and technology, but they do not have effective internal methods for moving their organizations forward.

Lack of knowledge and skills in technology transition is not only a problem for those adopting new software engineering practices. Many technology developers and vendors similarly lack an understanding of the technology transition process. Although technology developers are good at overcoming technical barriers that impede the effectiveness of their technology, they are typically less skilled at understanding the full extent of non-technical barriers faced by potential adopters of their technology. Technology vendors typically understand how to attract customer interest in their products, but they are often less effective in helping their customers install and use these products effectively.



- concept formulation
- development & extension
- enhancement & exploration (internal)
- enhancement & exploration (external)
- (early) popularization

- generating new product ideas
- screening
- concept testing
- business planning
- development
- prototyping
- test marketing/pricing
- product launch

- needs assessment
- selection of candidate products
- evaluation of candidate products
- introduction of selected product to organization
- gathering feedback
- implementation planning
- implementation
- product maintenance
- end user support

Figure 6-2: The Three Domains of Technology Transition

In short, lack of effective understanding of technology transition principles and practices impedes the transition of more effective technology from both the supplier and adopter viewpoints.

6.1.2 Customers

Customers include all those who can benefit from a more predictable and systematic approach to technology "push" or "pull." These include, on the "push" (or supplier) side, software technology researchers and their sponsors and new software product developers such as tool vendors; on the "pull" (or adopter) side, software developers, purchasers of custom software systems, acquisition managers, advanced technology receptor groups, and SEPGs (especially those adopting and introducing non-process software technologies or technologies at CMM level 3 and above). Intermediaries—those who facilitate transactions between "push" and "pull" sides—include consultants, trainers and educators, marketers, incubators, and others developing new businesses. SEI personnel are active in many of these roles and thus can act as surrogates for customers during early pilots of innovative software technology transition methods and practices.

6.1.3 Rationale

The transformation of a technology as it moves from birth in R&D through product development and into widespread use in target markets is a long, difficult, and typically ad hoc process. As an example, it takes more than 40 years to reach the 10% market penetration level of new manufacturing technology in the United States.[1] Corresponding penetration in Japan takes only about 15 years. As an advocate for the software engineering community, the SEI must reduce the time for the deployment of U.S. software technology, as shown in Figure 6-3.

All software organizations need to manage the process of transition to new software technologies. To date, most attention has been paid to choosing the technology, analyzing features of the product, etc. In reality, managing the introduction of a new technology is just as essential as choosing the right technology. According to a March 6, 1995, *Business Week* article, the cost of a technology, such as a PC, is only 10% of the cost to an organization. Support, training, etc., make up 90% of the cost. With systematic introduction practices, we suspect that as much as one-third of this effort can be saved.

6.1.4 Benefits

Software technology and products built with attention to transition are more likely to be successfully used. Change agents using software technology selection criteria based on "transitionability" are more likely to select technologies that can be effectively and efficiently put to

1. Mr. Ted Olsen, Sr. Vice President, National Center for Manufacturing Sciences, in a presentation to the Technology Transfer Committee of the Council of Consortia in January 1993.

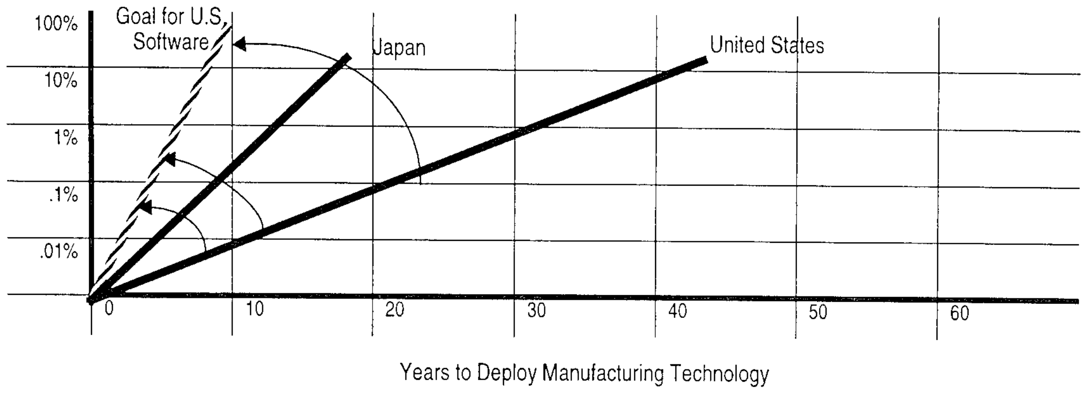


Figure 6-3: SEI Goal of Dramatic Reduction in Deployment Time of Software Technology

use. Effective software technology transition reduces adoption risk, and to be effective, it should be based on the use of models and best practices proven successful not only in software engineering but also in other domains and in leading organizations. Successful practice must be codified and translated into methods and tools.

Expertise exists in the community on how to introduce technologies such as project management methods, peer reviews, and even advanced CASE tools. Often this is undocumented expertise, even when adopting organizations obtain it from consultants. By showing the experts how to codify, package, and systematize the introduction-adoption process, we provide a model of how they can support technology adoption. By showing customers a model of what they should expect from experts, we encourage them to demand more from their suppliers. In this way, we empower the community to create a demand for the types of codified adopter processes and tools that will enable them to be successful in implementing new technologies in their organizations.

Improving the software engineering community's ability to successfully transition software technology will accelerate improvements in the state of the practice of software engineering. A strong capability in software technology transition practice in the software engineering community also

- Boosts our national competitive advantage by reducing the time for the deployment of U.S. software technology.
- Leverages the investment in SEI work in software engineering technology.

Figure 6-4 describes the trends in transition strategies and methods over the next 5-7 years.

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Adopters: Organizations adopting improved SE practices	• Ad hoc for technology improvements; increasingly guided by experience for process improvement	• Systematic guidelines increasingly used for technology adoption by early adopters; early majority is adopting process improvements effectively	• Key transition practices widely known and used by software engineering organizations at level 3 or higher	• Rate of adoption of new technologies by organizations • # of organizations routinely executing key transition practices
Suppliers: Developers or vendors of improved SE technology	• Iterative, risky	• Limited planning and management for transition as part of product development; SE tech. transition plans requested by sponsors	• Extensive planning and management for transition; SE tech transition plans required by sponsors	• Ratio of new SE products that come with explicit and effective transition support matching the key transition practices of adopting organizations
Intermediaries: Organizations supporting technology suppliers or adopters	• Need for transition services not widely recognized; • Few organizations provide transition services	• Role of transition support services is better understood by adopting organizations. • Growing number of companies provide these services	• Transition support services widely recognized as a useful support activity	• # of organizations providing transition support services

Figure 6-4: Trends in Software Engineering Technology Transition Practices

6.1.5 One-Year Objectives for 1996

Provide technology-specific vehicles for technology transition. This work extends previous work on defining "transition packages" for software engineering technologies. It includes defining examples of transition packages for Key Process Areas (KPAs) at level 2 of the CMM, extending a two-part workshop vehicle to include additional technologies, and the adoption of COTS products supporting engineering collaboration.

Provide/update workshops and courses used to support technology transition. This work includes updating existing courses and workshops to reflect changes in the software engineering corporate environment and lessons learned in customer encounters over the past five years. It includes the development of new workshops.

6.1.6 Baseline Work Outputs

There are no baseline work outputs for the activity area related to software engineering transition practices.

6.1.7 Proposed Add-On Work Outputs

This section describes the proposed add-on work outputs for the activity area related to software engineering transition practices. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

IT-1A Transition Packages for Level 2 KPAs (1996-1997)

This work will coalesce the collective learning of the software engineering community about the Acting Phase of the IDEAL Model into KPA Transition Packages. These Packages will codify the best "how to" practice for implementing Key Practice Areas identified by the CMM, saving each organization from having to build its own materials and processes from scratch. At least the following material will be provided:

- a process model of steps in introduction of a KPA and a related process guide that describes how to execute the steps
- templates and examples of pilot and roll out plans for introducing the KPA into one or more organizational units
- a process model of steps for enacting the KPA itself, e.g., for software configuration management, and a related KPA process guide
- education and coaching materials for the process action team sponsor, management steering committee, executive sponsor, and participating project sponsors
- an annotated bibliography
- reprints of standard reference papers
- requirements and specifications for training and communications for all participants
- "sales" information and briefings
- consulting scenarios (how process action teams can support participating projects)
- training selection and customization criteria
- tool selection criteria
- selection criteria for subject matter experts and suppliers of related products and services
- cost/benefit analyses and related products and services

Collaboration in 1995 with two Xerox Corporation software projects resulted in a documented, defined process for software technology introduction (dubbed the Technology Transfer Process, or TXM) and an accompanying Process Guide. This TXM and Process Guide, tailored for each organization, is now being used by Xerox process action teams for introducing technology and processes related to key process areas. Early results show that the second project at Xerox has greatly reduced costs in KPA adoption work due to building on the work outputs of the first project.

Our continuing work with Xerox and technical interchange with other industry and government organizations indicates interest in extending this concept so the TXM and Process Guide become a series of "whole products" for the introduction of KPAs—that is, KPA Transition Packages.[2]

Purpose

The CMM tells organizations *what* to do in software process improvement but not *how*. The *how* falls partly in the Establishing phase but primarily in the Acting phase of the IDEAL model. People are reinventing the implementation of software process improvement as they execute the Acting phase. They are recreating process models of (and project plans for) the introduction of KPA-related changes (including practices, procedures, methods, tools, etc.). They are recreating documents needed for enacting KPAs such as estimating forms, tracking logs, and project plans. Even when there are readily-available sources for these artifacts (e.g., IEEE standards), people want examples and guidance on creating them, to save time and to build on lessons learned elsewhere. In sum, people doing process improvement need "one-stop shopping" for all their KPA transition needs—an authoritative source that identifies all the prerequisite elements of successful KPA implementation, and then either supplies or points to suppliers of these. The Transition Packages produced by this effort will satisfy these needs. Using principles of reuse and customization, organizations will take KPA Transition Packages and translate them for use in their specific context, at a fraction of the expense of re-invention. Creating KPA Transition Packages also serves as a proof of concept for this approach to transitioning software engineering technology: We anticipate the number of suppliers of such packages to grow quickly as a result of this demonstration.

Customers

The KPA Transition Packages produced by this effort will be used by software engineering process groups (SEPGs), process improvement (or action) teams and their sponsors, and anyone responsible for change at a KPA level (as opposed to at an overall organizational improvement level). Suppliers to this community will also use these packages as a model of how to supply the best mix of products and services.

2. Geoffrey Moore (1991). *Crossing the chasm: Marketing and selling technology products to mainstream customers.* Harper Business.

Collaborators

SEI funds will be used only to coordinate codification of KPA transition practices and to prepare final packaging. Eighty percent of the effort will be contributed by the software engineering community, in the form of participants in workshops, Resident Affiliates, in-kind resources, and additional funds. A core group of collaborators, organized as an informal consortium, will contribute or create components of the KPA Transition Packages. Community review will assure relevance and appropriate packaging. Subject matter experts may wish to participate, as may Big Six accounting firms (we anticipate tapping their experience in packaging service-based products).

Approach

In 1996, we will tailor the technology transfer process model for all the KPAs as described and add the additional material as described above. We will develop and evaluate the transition packages we create by working with our collaborators. Finally, in 1997, we will revise the materials and license firms for delivery of the packages; we may also make aspects of the packages available in parallel on the Web or in the Process Asset Library maintained by the Process area.

The success of this effort will initially be evaluated by the degree of interest and participation level we achieve from consortium members, subject matter experts, and other community groups. By mid-1996 we will have the consortium in place and a plan for working together as well as agreement on what baseline data to collect. By the end of 1996 we will have at least one KPA Transition Package drafted and an evaluation plan created. By mid 1997, we will have the remaining Packages drafted and under evaluation by members of the consortium. Final evaluation will be determined by comparing baseline data from the consortium members to data gathered by those members as they use the KPA Transition Packages.

IT-2A Software Technology Transition Workshop Series (1996)

The output of this effort is a software technology transition workshop given in two parts. Part One is designed to present a framework for managing the adoption process by teaching those tasked with introducing technological change how to analyze a software technology transition situation as a problem to solve. The workshop teaches the student to structure the tasks involved in a software technology transition situation in order to manage the transition like a project. Part Two of the workshop is designed and taught by an expert in a specific software technology.

This approach was piloted in 1994-1995 via a Technical Collaboration Agreement with a leading software configuration management tool vendor. Early versions of this workshop, as conference tutorials, were well-received, but were criticized as not providing enough specific "how to" material. We had presented models specific to software, with software examples, but attendees wanted step-by-step guidance on specific situations, such as how to introduce software configuration management. In 1995, Part One materials will be finalized and submitted to the SEI Education and Training Review Board (ETRB). Licensing requirements will also be established, and two to four SEI MTS will be recruited and trained to deliver Part One of the workshop.

This workshop is a companion to the KPA Transition Packages effort (IT-1A). Part Two of the workshop will be coordinated with a particular Transition Package for a particular KPA.

Purpose
Many SEI customers tell us that they need more detailed information on how to proceed with a software process improvement effort. We have provided a general framework for them in the CMM and the IDEAL model, but time and time again they ask "*how* do we do this?" This workshop provides step-by-step instructions in the context of a specific technology.

The long-range goal of this effort is to get many different technology producers to use this two part workshop (along with IT-1A, KPA Transition Packages, for example) as a means of getting technologies adopted faster and more effectively. In essence, we will be providing technology producers examples of good transition practices.

Customers
The workshops are designed to combine general software technology transition expertise with domain-specific transition knowledge in order to teach change agents such as software managers, members of SEPGs, and process action teams how to manage a specific software technology transition situation effectively.

Collaborators
Subject matter experts from specific software engineering technologies ranging from CMM Key Process Areas to advanced technologies such as process-centered environments will be solicited for co-developing Part Two of this workshop and for co-teaching the workshop.

Approach
We will prepare a set of standard requirements for Part Two of the Workshop and solicit participation by domain experts within the community to create many instances of Part Two. Part One will be licensed to outside firms or, in special cases, to individual experts.

This proposal is being presented as new work because of the scale of the effort; we will aggressively seek parallel participation by several experts so that this approach gains extensive visibility.

We will evaluate the success of this effort by the demand for offerings.

IT-3A Adoption of COTS Products Supporting Engineering Collaboration (Feasibility Study) (1996)

The report developed by this effort will analyze and document the growing set of problems faced by software engineering organizations as they seek to select and introduce COTS products such as Lotus Notes and personal conferencing tools that are intended to support their work processes. The report will take advantage of the technology transfer model and process guide co-developed with Xerox (see IT-1A) and will identify any parts of the current model that should be refined or expanded based on its application to this kind of technology.

Purpose

The selection and broad introduction of COTS support tools into an organization typically takes a surprising amount of effort. The rapid growth of a commodity-style computer marketplace has accelerated the rate and changed the focus of technology tool adoption. From few tools with relatively low complexity for a few technical end users, we have moved to inexpensive hardware and an extensive and complex selection of software. Desktop computing is ubiquitous—we cannot assume users have a technical background. As business processes have become more intimately coupled with desktop computing, and collaborative computing such as conferencing or the Web becomes more widespread, end-user support and technology adoption issues will be an important organizational concern. These issues hold whether the organization is adopting personal productivity tools such as word processors, group process tools such as conferencing hardware and software, or organization-wide communication facilitators such as Lotus Notes. Because the work will take advantage of the technology transfer model co-developed with Xerox, the result will help determine the breadth and robustness of the model for use with software engineering technology by applying it to a different kind of technology than what it has been used for up until now.

Customers

Software engineering organizations are using more and more computer-based tools to support collaborative efforts both within their organization and with their suppliers, subcontractors, and customers. Geographically distributed integrated product teams are becoming more common. Selection of appropriate computer-based support tools is becoming an increasingly important issue to support effective software engineering practices. The output of this effort will be useful to the set of organizations facing these issues.

Collaborators

The SEI currently deals with an increasingly large number of organizations in a computer-based collaboration mode. These organizations should be willing to collaborate with this effort in developing suitable guidelines and strategies for adopting collaboration software and other COTS tools that support distributed collaborative efforts.

Approach

The work will examine actual examples drawn from SEI experience and examples documented by organizations such as the Gartner Group who are focused on helping computing services organizations. After analyzing the data, we will determine to what extent the existing software technology transfer model and process guide can be adapted and used as a model for COTS support tool adoption.

IT-4A Update of the "Managing Technological Change" Course (1996)

This course teaches basic concepts of organizational change management and the use of assessment and planning instruments that prepare individuals to understand the specific organizational issues that may be faced when introducing new technology into a workplace. The topics covered in this course include:

- assessing the skills and motivation of key stakeholders authorizing and reinforcing the change
- identifying the potential for, and sources of, resistance during implementation of a change from target groups affected by the change
- determining the potential for resistance from the current values, behaviors, and "unwritten rules" in the organization's culture
- practical strategies and tactics to drive the change through an organization while building commitment
- a common language for talking about change

The course is hands-on and teaches a structured approach to planning for the organizational and human aspects of the technology transfer.

Purpose

"Managing Technological Change" is a course that has been given to over 1700 executives and managers since 1990. It supports the IDEAL model for process improvement. The course was primarily developed in 1989, and much has changed in the U.S. workplace since then. The specific examples, scenarios, and suggested methods given in the course need to be updated to reflect the decentralization, downsizing, and flattened organizations that exist today since it is not effective to use examples that are widely divergent from the organizational situations faced by course attendees. The course also needs to be updated with examples drawn from experiences in actually introducing software engineering changes in organizations over the past five years.

Customers

This course is aimed at technical practitioners who will be participating in and facilitating technology transition in their organizations and projects. Such individuals have responsibility for introducing new technology to their organizations but are typically unaware of the complete set of problems that may need to be addressed and effective solutions for dealing with these problems.

Collaborators

The costs of updating this course are expected to be recovered from those people who take the updated version. In addition, versions of the revised course will be piloted with selected customers before the update is finalized.

Approach

The people responsible for giving the course over the past five years plus the accumulated comments of those who have taken the course provide the basic body of information needed to understand what revisions are needed. Since the people responsible for giving the course in the past will be involved in revising the course, the effort required to understand the scope of the revision is minimal.

The course will be revised with the guidance of the Education and Training Review Board, which is the internal SEI body responsible for ensuring the quality of SEI training materials. Pilot versions of the revised course will be presented to selected customers before the update is finalized.

IT-5A Extension of the "Consulting Skills Workshop" for Software Engineers (1996)

This Workshop is currently designed to complement the IDEAL approach to integrated software process improvement programs. The revised workshop will make additional software engineering case studies available that are relevant to different software engineering audiences, e.g., technology providers.

The content of the workshop is built around a six-stage consulting model that provides entry and exit criteria as well as checklists for each stage. Participants learn techniques and methods to use in everyday work, such as forming collaborative working relationships, negotiating roles and expectations, collecting and using data effectively throughout the consultation process, and handling difficult situations that occur when circumstances change in the organization.

Purpose

The Consulting Skills Workshop was developed in 1991 and has been attended by over 800 professionals responsible for change efforts in their organization. The workshop uses hands-on exercises and case studies that need to be updated and extended to make the workshop useful to different audiences, e.g., technology providers. In addition, based on student requests, additional modules need to be provided in the workshop.

Customers

This workshop is intended for individuals responsible for managing expectations and ensuring cooperation during periods of change within their software engineering organizations. Both managers (potential clients of a change agent) and change agents (consultants) need the strategies provided by this workshop for creating and sustaining effective working relationships under stressful conditions.

Collaborators

The costs of updating this workshop are expected to be recovered from those people who take the updated workshop.

Approach

The case study used in the current workshop will be updated to be consistent with CMMv1.1 and will take into account potential changes planned for CMMv2.0 to the extent this information is available. Additional software engineering case studies will be provided for use with audiences not focused on process improvement. The exercises used in the workshop will be revised based on the experience that has been gained in giving the workshop more than forty times. Finally, based on student requests, the following new workshop modules will be developed and piloted:

- exercises and practice in dealing with resistance to change
- application of interpersonal skills to software engineering situations

Since the people responsible for giving the workshop will be involved in revising the workshop, the effort required to understand the scope of the revision is minimal. The revision will be performed under the guidance and review of the Education and Training Review Board, which is the internal SEI body responsible for ensuring the quality of SEI training materials. Pilot versions of the revised workshop will be presented to selected customers before the update is finalized.

IT-6A Software Strategic Planning Workshop (1996)

For a given enterprise, the output of this workshop is to define the role of software appropriately in the enterprise's overall goals and strategies. If software is indeed a key "enabling" technology for the enterprise to reach its overall goals, then it is critical that software issues be included in an enterprise's overall strategies. This workshop helps participants to define appropriate software goals and strategies at the corporate level of an enterprise, thereby laying the groundwork for improvement of an organization's software engineering practices.

Purpose

Funding for technology improvement and management follow-through only happen when high-level management understands how software engineering deficiencies can impair an enterprise's ability to achieve its goals, which nominally, may have little to do with software. In such organizations, the effort needed to maintain and improve software engineering practices won't be provided unless software issues are appropriately embodied in the enterprise's overall corporate strategies.

This workshop is designed to stimulate enterprise strategic planners and corporate management to develop an accurate understanding of the strategic business potential of improved software engineering practices to their enterprise. (We understand that software is not strategically important to *every* enterprise, but for many organizations, software is often more critical than top-level management understands.) The Workshop is designed to facilitate the inclusion of software engineering issues in the overall enterprise strategic plan when software is legitimately a key factor in achieving success. As a consequence of this workshop, organizations that should be exploiting improved software engineering practices will become more proactive in establishing and supporting software engineering transition initiatives. As a consequence, it will become easier to transition improved software engineering practices.

Customers

The intended customers are high-level managers in an organization that has a software component, plus the software managers.

Collaborators

Many organizations are willing to participate in and to pilot this Workshop.

Approach

Based on our experience with a variety of software organizations, we will develop workshop materials that are appropriate for introducing software issues into enterprise-level strategic planning. We will then pilot the workshop and make appropriate revisions. The workshop will be developed under the guidance and review of the Education and Training Review Board, the internal SEI body responsible for ensuring the quality of SEI training materials.

6.1.8 Related TO&P Activities

Our work on developing KPA Transition Packages for Level 2 KPAs requires TO&P support from organizations that want to install these KPAs. The extension of the two-day transition workshop to cover additional technologies requires collaboration from subject matter experts to support Part Two of the workshop. We will seek TO&P support for preparation and pilot delivery of the updated and new courses and other workshops proposed for development.

6.2 Collaborative Skills in Software Engineering

6.2.1 Problem Statement

The U.S. software industry, in order to remain competitive, is faced with developing increasingly larger and more complex software systems, much more quickly and with higher quality and lower cost. In addition, U.S. industry is increasingly looking toward collaborative teams to increase worker involvement, improve quality and productivity and help flatten, downsize, and decentralize the organization. This trend has been accelerated by the influence and success of the Japanese and their widespread use of teams.The team approach is also being applied to the development and maintenance of software. Indeed we see many software engineering organizations trying to incorporate "team based" concepts of collaborative software engineering into their current software practice.

Many of the barriers to successfully implementing team-based collaborative software engineering practice trace directly to practitioners' and managers' lack of understanding and training in the interaction skills required to successfully enact such collaborative activities as requirements elicitation, project management, and peer reviews. This skill deficit affects any work activity that involves two or more people, and is particularly evident in team type work (e.g., system test) and group interactions (e.g., meetings). While software managers and practitioners often recognize the problem, they typically cannot identify or implement effective solutions.

The problem with addressing interaction skills with software practitioners is that these skills are generally not perceived to be related to the participants' technical work or successful project performance and outcomes. Therefore, the skills are not practiced or incorporated to become new and more effective behavior. Perhaps the most significant barrier to skill acquisition, however, is that methods for dealing with "people issues" often do not resonate for most software engineers or their managers. While generic materials and training in these skills are available, almost nothing is tailored to the engineering frame of reference in order to make the concepts underlying effective interpersonal behavior relevant to engineers' daily technical work and project success.

6.2.2 Customers

The outputs in this activity area are intended for use by software engineering practitioners and managers who find that interactions with their peers are frustrating because it is difficult to identify, discuss, and resolve differences in viewpoints of how to proceed in addressing software engineering issues. We are also focused on those members of the software engineering community who see the strengths in using software engineering teams, but are finding it difficult to get these teams to work together collaboratively and productively.

6.2.3 Rationale

The SEI has always been engaged in field work with the software community, and we have had extensive first-hand experience in identifying and analyzing the barriers to software engineering improvement. While the assessment and evaluation methods related to the CMM for software are probably the most widely known activities, the SEI also has established relationships with a number of customers to address a wide range of problems that affect their ability to produce quality software on time and within budget.

The most consistent transition barrier identified in our field work has been generally referred to as "people problems." When analyzed further, the problems can be characterized as an inability of groups to effectively capitalize on the cumulative talent and technical skill resident in the participating individuals because they have no commonly understood, accepted, or enacted ground rules for working together on their shared technical tasks.

The reputation of the Software Engineering Institute for providing useful materials to engineers makes it an attractive and credible vehicle for packaging and disseminating concepts critical to disciplined software engineering practice that might otherwise be viewed as suspect. Thus, the SEI is in a prime position to transition improvements in collaborative skills into the software engineering community.

6.2.4 Benefits

Given a trend toward the use of teams in software engineering, as well as in other areas of business, the benefits of this work are to show software engineers how to establish effective teams by developing and applying appropriate collaboration skills.

Figure 6-5 describes the trends in collaborative software engineering skills over the next 5-7 years.

Key Items	State of practice as of:			Impact/Metrics
	1996	1998-1999	2000 and Beyond	
Adopters: Organizations adopting improved SE practices	• A few leading organizations are developing collaborative software engineering skills	• Leading organization prove effectiveness of collaborative software engineering. Majority start migrating to these methods.	• Collaborative skills widely acknowledged and used as essential to effective software engineering. Tools and methods rely on and encourage use of these skills.	• # of teams using collaborative skills • # of tools depending on existence of such skills
Suppliers: Developers or vendors of tools supporting collaborative software development	• Minimal investment in tools supporting collaborative infrastructure.	• Some suppliers of software engineering technology incorporate mechanisms to support collaborative software engineering.	• Extensive supply of tools, methods, and supporting infrastructure relying on collaborative software engineering techniques and skills.	• # of tools depending on existence of specific collaborative skills
Intermediaries: Organizations supporting technology suppliers or adopters	• Ignore need for collaborative skills and tools	• Provide increased training in collaborative skills and use of tools	• Training in collaborative skills routinely provided to software engineering organizations	• # of training courses given on collaborative methods and skills

Figure 6-5: Trends in Collaborative Skills in Software Engineering

6.2.5 One-Year Objectives for 1996

Establish the value of a focus on collaborative skills in software engineering. Given that this is a new activity area for 1996, the essential goal will be to demonstrate that paying attention to these issues has a positive impact on the practice of software engineering by interactions with specific organizations outside the SEI.

6.2.6 Baseline Work Outputs

None. All work to be performed in 1996 is proposed new work.

6.2.7 Proposed Add-On Work Outputs

This section describes the proposed add-on work outputs for the activity area related to collaborative skills in software engineering. The SEI is asking selected members of the software engineering community to evaluate the relative attractiveness of these add-on proposals as possible elements of the final 1996 basic program, as funding permits. Appendix A lists all baseline and proposed add-on work outputs.

IT-7A Applying Conflict Resolution Skills in Software Development (Feasibility Study) (1996)

The first part of this effort is a feasibility study that will investigate the need for and benefits of training software engineers in conflict resolution skills. The assumption underlying this study is that conflicting goals and objectives are a normal part of the software development process—conflict is unavoidable. Handling this conflict in a constructive way is critical to increasing process maturity and project success.

If the feasibility study demonstrates that there is value in having the SEI develop a workshop dealing with conflict resolution in software engineering, we will develop such a workshop. Participants in the workshop will learn a systematic approach for surfacing the root cause issues that are in conflict and methods for resolving conflict between individuals as well as between the individual and the organization. Exercises in everyday software activities such as software requirements elicitation, peer reviews, project planning and managing customer expectations will be used as a vehicle for participants to gain skills in surfacing the key issues and experience in using conflict resolution methods.

Purpose

In training and working with over 3000 software professionals, trying to improve their organizations' software capabilities, a consistent theme has been how to deal with conflict within the organization. It is clear to these professionals that when developing software, difficult issues and conflicts need to be constructively surfaced and resolved. They are very concerned because in many cases there has not been a history of successful and efficient conflict resolution within their organizations. As a result there have been many requests for SEI to include training in this area. This workshop is designed to provide an effective method for surfacing and resolving conflict in the software environment.

Customers
The workshop is designed for software professionals who require skills in interpersonal communication, negotiation, and conflict resolution. This workshop is important for those managers sponsoring software improvement efforts as well as change agents, managers, and technical practitioners affected by or facilitating the change that surrounds the adoption of new software processes and technology. The workshop is also for those seeking to improve their effectiveness in executing software engineering processes that require the identification and resolution of conflicting views of a situation, e.g., requirements analysis, design reviews, code inspections, risk analysis, etc.

Collaborators
This work will include extensive client input and review. We will examine software engineering organizations to see to what extent they already provide training in conflict resolution skills and to what extent they find such training valuable. If we find a need for the workshop, pilot presentations of the workshop will be delivered and evaluated before finalizing the workshop.

Approach
The first part of this effort is a feasibility study that will examine the extent to which software engineering organizations currently recognize the need for training in conflict resolution skills and the extent to which this need is widely recognized and supported by existing elements of the software engineering community. If it is shown that highly successful organizations have recognized the need for such training but that these benefits are not widely recognized or accepted by the software engineering community, then we will develop and execute an action plan for increasing the awareness of the benefits of using these skills and if necessary, will develop training materials that can be used to demonstrate the value of such training.

IT-8A Human Interaction Capability (HIC) Framework (Feasibility Study) (1996)

This work will provide techniques for use by an audience that traditionally has not taken advantage of concepts drawn from cognitive psychology, group psychology and family therapy, all of which are well-understood disciplines. The work will apply these concepts to develop the skills needed to enact everyday engineering practices. To the extent that existing materials can be tailored to accelerate application of the HIC framework, they will be employed.

Eight specific skills, including listening, decision-making, negotiation and conflict resolution, form the building blocks for increasing interpersonal capability. These skills then will be directly mapped to familiar software engineering tasks, such as project planning and reviews. The HIC framework provides the overall structure for characterizing the capability of individuals to interact effectively in groups, thus creating a roadmap for improving interpersonal skills, which in turn will increase the likelihood of successfully enacting engineering practices that involve more than one person.

Purpose

The Human Interaction Capability framework will contribute to software engineering improvement by transitioning a set of human interaction skills to software engineers and managers that will enable them to perform their technical work more effectively and reduce their risks in meeting cost, quality and schedule goals. These skills will be directly linked to enactment of key process areas in the SEI's several Capability Maturity Models, and in the Personal Software Process (PSP), and will also be related to software engineering activities in general.

Customers

The intended audience for the HIC framework and materials will be practitioners and managers engaged in software engineering.

Collaborators

The HIC development effort is closely related to the several SEI projects developing CMMs and the PSP. The HIC developers are working with the SEI maturity model integration effort, the individual CMM projects, and the PSP project to ensure that appropriate pointers to HIC concepts are included their work and that the HIC effort develops solutions which support enactment of all the CMMs and the PSP.

The HIC framework is most closely aligned with the People CMM (PCMM), complementing the top-down approach of the PCMM by offering a bottom-up approach to the people dimension of software improvement. It is analogous to the PSP providing the bottom-up approach to the process dimension, complementing the top-down approach to software improvement of the original CMM. The consensus regarding the HIC among the related SEI projects is that the capabilities identified by the HIC are foundational to successfully enacting the practices identified in all the CMMs and in the PSP.

Approach

Preliminary thought was given to the HIC framework in 1994 in conjunction with capturing and disseminating lessons learned about software improvement for two workshop tutorials. This work will enable us to continue initial research already conducted, analyzing the data already collected, and will devise a robust approach to developing and testing the framework. These efforts will be documented in a feasibility report that will provide the basis for developing the work further. The framework initially developed for the tutorials will be tested against the results of the data analysis and literature review to create a prototype framework. Initial specifications for collateral materials to support application of the framework by software engineers and managers will be developed.

The first set of activities to develop the framework are planned and have been implemented at a sufficient level to encourage further refinement, leading to additional insight into the key issues that the framework needs to address. The outcome of this effort will lead to a decision about whether or not to proceed with further development. This decision will be based on the findings of the initial data analysis and literature review.

6.2.8 Related TO&P Activities

Community interest expressed at two 1994 tutorials indicates strong potential for securing external funding and cooperation in conducting the necessary field tests of the HIC materials as they are developed. We also need support for pilot presentations of the Conflict Resolution Workshop.

Volume II
Chapter 7 Table of Contents

7 Community Outreach

To satisfy its technology transition mission, the Software Engineering Institute (SEI) engages in a large amount of customer interactions with the software community. Some of these technology transition activities are funded by customers through cost recovery income; for example, course tuition, conference registration fees, and funding of certain technical collaboration efforts. There are also activities that provide general information and facilitate customer interactions. These outreach activities, which were also included in last year's plan, *SEI Program Plans: 1995-1999*, are described in this section.

7.1 Baseline Work Outputs

CO-1B Impact of Software Engineering Practices (1996-1997)

In its second year in 1996, this activity is an ongoing effort to identify and quantify the impact of software engineering practices on quality and productivity in the software industry. Known in last year's plan, *SEI Program Plans: 1995-1999,* as the Value of the SEI Study, the name of this activity has been changed to emphasize the broader goal of understanding the impact of software engineering practices, regardless of who champions them. Data and analyses focus on improvements in product quality, cost, schedule, and business value, as well as the SEI's success in transitioning technologies to the broader software community. The audience for the work includes the SEI's sponsors, customers, and the broader software engineering community, as well as the SEI's management and technical staff.

Beginning in 1995, work processes are being defined and refined to routinely capture, analyze, and act upon data about the impact of software engineering practices. The activity has focused initially on work that has high visibility within the SEI and the software engineering community. Eventually, the activity will address all impact area and sector efforts that deliver outputs to external customers. The resources for the activity will be earmarked under normal work processes. Because work of this kind is not yet common in software engineering, a separate initiative is necessary at this time to mature and routinize those work processes.

In 1996 we will refine the methods and processes developed in 1995. Additional analyses will be done of the impact on customers who work directly with the SEI. Work in 1996 also will capture and analyze data from customers who are engaged through the SEI's transition partners. Additional data gathering and analysis will be conducted with occasional customers and users. An intensive effort will be made in 1996 to improve the quality of sampling information necessary to enable more accurate studies of current and potential customers, as well as broad-based market analyses.

As standard methods and processes for data collection, management, and analysis are refined and routinely used, available resources can be directed towards in-depth, applied research that moves beyond demonstration of current value, and provides more detailed evidence aimed at improvement. Analyses in collaboration with selected SEI impact areas and customer sectors will be conducted in 1996.

CO-2B Quarterly Update / Summary of Technical Operations (1996-2000)

The Quarterly Update (QU) is produced three times per year according to contract requirements. The fourth issue is the Summary of Technical Operations (STO), a retrospective on the year's efforts. Taken together, these four documents provide status information on the SEI technical program.

The QU (CDRL A005) and STO (CDRL A003) is delivered to the SEI joint program office (JPO) in fulfillment of contract data requirements list. Each report highlights the progress made toward milestones specified in the operational plans for all technical units of the SEI. In addition to technical tasks, the reports cover the SEI's participation in significant events that result in new technical directions or potential collaborations with newly identified customers.

To better use resources while reaching a broader customer base, a summary of each quarter's most salient activities and accomplishments will appear as part of the SEI publication *Bridge*. Interest in the SEI's progress reports has grown over the past several years, especially among former affiliates and others who have maintained long-term relationships with the SEI. By combining the Quarterly Update and *Bridge*, the SEI will provide both progress statements and features describing future events and opportunities for involvement in SEI technical offerings. This approach permits improved financial and information management practices for the SEI.

CO-3B Resident Affiliate Program (1996-2000)

The administration of the Resident Affiliate Program is an important, ongoing community outreach activity.

Through the Resident Affiliate Program, U.S. industry and government organizations may sponsor resident affiliates at the SEI to work on technical projects of interest both to the SEI and the sponsoring organization. Project assignments for resident affiliates are decided by assessing how to make the best use of the resident affiliate's particular expertise while meeting the mutual goals and objectives of the SEI and the sponsoring organization.

The program also includes coordination with past resident affiliates. Alumni resident affiliates are considered to be important advocates for the SEI, and the SEI expends a great deal of effort to continue these relationships after the resident affiliates leave:

- Resident affiliates are able to keep their SEI computing accounts and are granted permanent subscriber benefits at no charge.

- The resident affiliate manager keeps in touch with alumni resident affiliates through regular e-mail messages and mailings.
- An annual resident affiliate luncheon is held each year at the SEI symposium. At this function, current and alumni resident affiliates, resident affiliate sponsors, and the SEI management team gather for an information sharing session.

Resident affiliates dedicate a portion of their time to transferring their project's findings and the results of other SEI work to their sponsoring organizations. Information gathering in support of this effort is accomplished both formally and informally. Monthly resident affiliate meetings provide the opportunity to hear presentations on many of the SEI's projects and to discuss these with project members. Resident affiliates also have access to electronic bulletin boards, information seminars, the SEI library, SEI technical reports, and SEI staff.

While they are on site, resident affiliates are also encouraged to take advantage of the SEI's training courses. Resident affiliates typically take at least two courses: Managing Technological Change and Consulting Skills Workshop. These courses assist them in planning their technology transition activities. Each month, various training courses and workshops are offered on-site at the SEI, and resident affiliates have the opportunity to participate in many of these offerings during their tenure.

Organizations that have participated in the Resident Affiliates Program are listed in Appendix B.

CO-4B SPIN Coordination (1996-2000)

The primary role of the SEI is to disseminate information from existing Software Process Improvement Network (SPIN) organizations to groups of people in common geographical locations who are interested in starting new SPIN groups. In this effort, the SEI will

- Maintain a directory of all currently active SPINs and points of contact in areas where interest has been expressed in forming a new SPIN. Both the *SPIN Directory* and the SPIN start-up information will be maintained and updated, and information will be written and distributed on a regular basis. Information is currently distributed on an almost daily basis, either electronically or by other means. By creating, maintaining, and distributing the SPIN Directory, the SEI is able to connect many software professionals with forming or existing SPINs.
- Maintain a database of people interested in starting SPINs. In this way, the SEI is often able to put together core groups of people who can pool their talents and resources to start up a new SPIN.
- Maintain an email alias used to disseminate announcements of interest to the network.
- Distribute SPIN start-up information on forming SPIN organizations to anyone interested in forming a SPIN in their area.

The SEI serves as a central point of contact for all SPINs, keeping all involved informed of what's going on elsewhere in the growing network of organizations.

A list of the current SPIN organizations is in Appendix C.

CO-5B SEPG Conference (1996-2000)

The Software Engineering Process Group (SEPG) annual conference provides a forum for representatives from all segments of the software community to meet and review the state of software engineering process improvement. Unlike other conferences on software engineering, the SEPG Conference focuses solely on process related issues and approaches to improvement.

In its collaborative role, the SEI provides the full logistical sufpport to enable the SEPG Conference, drawing on its experience in event management, marketing, publishing, and media. In addition, the SEI provides the conference co-chair and representatives to all of the major committees responsible for the planning and execution of the conference.

The SEI issues a call for proposals for the conference two years in advance of the actual conference. Proposals are submitted by all interested regional SPIN groups. The SEI evaluates the proposals based on overall ability to deliver a high-quality technical program at reasonable cost to attendees. Although one logistical objective is to hold the conference in various regions of the nation to achieve broader representation, such decisions are secondary to the primary goal of achieving a solid technical program that can be executed to the best advantage of the attendees. The growth in attendance at this conference—from 46 attendees in 1988 to over 1300 in 1995—suggests that the collaborations between the SEI and the host SPIN groups have created a legacy of satisfaction among customers.

Appendix A Baseline and Add-On Work Outputs

Output Name		Basic		Projected TO&P[1]	Dependencies[2]	Page
		Baseline	Add-on			
SP-1B	CMM Version 2 (1996-1998)	■			SP-2B, SP-3B, SP-12B	II-15
SP-2B	Maturity Model Integration Framework (1996-1997)	■			SP-3B	II-16
SP-3B	SPICE Product Suite (1996-1997)	■				II-17
SP-4B	Community Involvement (1996-2000)	■			SP-1B,SP-2B, SP-5B, SP-6B, SP-7B, SP-8B, SP-3A, SP-6A, SP-7A, SP-9A, SP-10A, SP-12A, SP-14A	II-29
SP-5B	CMM Appraisal Framework (CAF) (1996-1998)	■			SP-1B, SP-3B, SP-4B, SP-3A	II-30
SP-6B	CMM-Based Appraisal for Internal Process Improvement (CBA IPI) (1996-1998)	■			SP-1B, SP-4B, SP-5B, SP-7A, SP-13A	II-30
SP-7B	CMM Validity Studies (1996-1998)	■			SP-1B, SP-4B, SP-12B, SP-13A, SP-14A	II-32
SP-8B	CMM-Based Appraisal for Software Capability Evaluations (CBA SCE) (1996-1998)	■		■		II-33
SP-9B	Method for Defining Software Processes (1996)	■			SP-4A	II-34
SP-10B	Product Specification for Process Guides (1996)	■				II-35
SP-11B	Software Measurement Handbook (1995-1996)	■			SP-1B, SP-4B, SP-12A, SP-14A	II-56
SP-12B	Process Appraisal Information System (PAIS) (1995-1997)	■			SP-4B, SP-6B	II-57
SP-1A	Integrated Product Development (IPD) Framework (1996)		■		SP-1B, SP-2B, SP-3A, IT-5A	II-18
SP-2A	Assessment of Highly Effective Software Development Teams and Environments (Feasibility Study) (1996)		■			II-20

Figure A-1: Baseline and Add-On Work Outputs for Software Process

1. The Projected TO&P Column for an output is marked with a black rectangle if there is more than a 50% chance that TO&P funds will be available to augment basic funds work on this output.

2. An output in the Output Name column has dependencies if progress on that output depends on the approval of outputs listed in the Dependencies column.

Output Name		Basic		Projected TO&P[1]	Dependencies[2]	Page
		Baseline	Add-on			
SP-3A	SE-CMM Version 2.0 (1996-1997)		▮			II-21
SP-4A	Advanced Process Definition Reports (1996-1997)		▮		SP-9B, SP-4A	II-36
SP-5A	Method for Defining Software Processes, Version 2.0 (1996-1997)		▮			II-37
SP-6A	Measuring the Impacts of the Personal Software Process (PSP) (1996-1997)		▮	▮	SP-4B, SP-9A	II-38
SP-7A	IDEAL Model Products (1996-1997)		▮		SP-4B, SP-6B, SP-8B, SP-9B, SP-11B, SP-1A	II-39
SP-8A	Applying Lessons Learned for Developing Software Process Improvement Teams (1996)		▮			II-40
SP-9A	Addressing Software Process Transition Barriers (1996-1998)		▮		SP-4B, SP-7B, SP-12B, SP-6A, SP-10A, SP-12A, SP-14A	II-41
SP-10A	Process Value Method (PVM) for Higher Maturity Organizations (1996-1997)		▮			II-42
SP-11A	Applied Process Research Collaborations (1996-1997)		▮		SP-4B, SP-12B, SP-14A, RM-9A	II-44
SP-12A	Information on CMM-Compliant Practices (1996-1998)		▮			II-45
SP-13A	Software Process Improvement for Small Organizations (1996-1997)		▮		SP-4B, SP-6A, SP-9A	II-46
SP-14A	Software Engineering Information Repository (1995-1997)		▮		SP-4B, SP-12B, SP-9A, RM-2B, RM-3A, SP-12A, IT-6A	II-58
SP-15A	Understanding Organizational Conditions Leading to Successful Software Process Improvement (1996)		▮			II-60

Figure A-1: Baseline and Add-On Work Outputs for Software Process (Continued)

1. The Projected TO&P Column for an output is marked with a black rectangle if there is more than a 50% chance that TO&P funds will be available to augment basic funds work on this output.

2. An output in the Output Name column has dependencies if progress on that output depends on the approval of outputs listed in the Dependencies column.

Output Name		Basic		Projected TO&P[1]	Dependencies[2]	Page
		Baseline	Add-on			
RM-1B	Software Acquisition Maturity Model (1996)	■		■		II-69
RM-2B	Software Engineering Risk Repository (1996-1998)	■				II-85
RM-3B	Knowledge Summarization, Analysis, and Visualization (1996)	■				II-87
RM-4B	Software Engineering Information Capture (1996-1997)	■				II-88
RM-1A	Software Acquisition Improvement Framework (1996-1997)		■		RM-1B	II-71
RM-2A	Software Acquisition Maturity Model Guidebooks (1996-1997)		■	■	RM-1B	II-73
RM-3A	Software Risk Metrics (1996-1998)		■	■	SP-11B	II-78
RM-4A	Risk Cost Model (1996-1997)		■	■		II-79

Figure A-2: Baseline and Add-On Work Outputs for Risk Management

1. The Projected TO&P Column for an output is marked with a black rectangle if there is more than a 50% chance that TO&P funds will be available to augment basic funds work on this output.

2. An output in the Output Name column has dependencies if progress on that output depends on the approval of outputs listed in the Dependencies column.

Output Name		Basic		Projected TO&P[1]	Dependencies[2]	Page
		Baseline	Add-on			
DE-1B	Business Strategies for Model-Based Software Engineering (MBSE) (1996)	▮		▮	DE-6B	II-101
DE-2B	Domain Engineering Guidebook (1996)	▮		▮		II-102
DE-3B	Report on the State of Program-Understanding Technology (1996)	▮			DE-1A	II-103
DE-4B	Domain Analysis for System Understanding (1996)	▮		▮	DE-15B	II-104
DE-5B	Open Systems Standards (1996)	▮		▮		II-116
DE-6B	Open Systems Evidence Module (1996)	▮		▮	DE-5B	II-117
DE-7B	Roadmap for Environment (Integration) Technology (1996)	▮		▮		II-117
DE-8B	Software Engineering Environments Technology Evaluation, Integration, and Measurement (STEIM) (1996-1997)	▮			DE-7B	II-118
DE-9B	Dependable Real-Time Systems Handbook (1996-1997)	▮			DE-10B	II-119
DE-10B	Airborne Radar Study (MITRE) (1996)	▮				II-120
DE-11B	Report on State of Practice in Process-Centered Environments (1996)	▮				II-121
DE-12B	Quality Attribute Engineering Framework (1996-1997)	▮			DE-13B, DE-14B	II-138
DE-13B	Performance Engineering Framework (1996)	▮			DE-14B	II-138
DE-14B	Assessment of Architecture Evaluation Practice (1996-1997)	▮				II-140
DE-15B	Evaluation of Architecture Representation Technology (1996)	▮				II-141

Figure A-3: Baseline and Add-On Work Outputs for Disciplined Engineering

1. The Projected TO&P Column for an output is marked with a black rectangle if there is more than a 50% chance that TO&P funds will be available to augment basic funds work on this output.

2. An output in the Output Name column has dependencies if progress on that output depends on the approval of outputs listed in the Dependencies column.

Output Name		Basic		Projected TO&P[1]	Dependencies[2]	Page
		Baseline	Add-on			
DE-1A	Domain Analysis and System Building (1996-1997)		■	■	DE-2B, DE-12B, DE-14B	II-105
DE-2A	Disciplined Evolution of Legacy Systems (1996-1997)		■		DE-3B, DE-14B	II-107
DE-3A	Use of SEI Repository for Developing Case Studies in Reengineering (Feasibility Study) (1996)		■		SP-15A, RM-2B	II-109
DE-4A	Impact of Object-Oriented Technology (Feasibility Study) (1996)		■		DE-2B, DE-14B, DE-1A	II-110
DE-5A	Lightweight CASE Integration for Architecture-Based Evolutionary Design (1996-1997)		■		DE-8B	II-122
DE-6A	Affordable Use of COTS Components (1996-1997)		■		DE-6B, DE-8B, DE-5A, DE-7A	II-124
DE-7A	The Evolution of Distributed Mission-Critical Systems (1996-1997)		■	■	DE-9B	II-125
DE-8A	The Evolution of Discrete Sequential Control Systems (1996)		■			II-126
DE-9A	Risk Management for Open Systems (1996)		■		DE-6B	II-127
DE-10A	Open Systems Assessment Instrument (1996-1997)		■		DE-6B	II-128
DE-11A	Open Systems Interactive (1996)		■			II-129
DE-12A	Evaluation and Application of Software Process Modeling Technology (1996-1997)		■		DE-11B, SP-1A, SP-3A	II-130
DE-13A	Guidebook for Addressing Architectural Mismatch (1996-1997)		■		DE-14B, DE-15B	II-142

Figure A-3: Baseline and Add-On Work Outputs for Disciplined Engineering (Continued)

1. The Projected TO&P Column for an output is marked with a black rectangle if there is more than a 50% chance that TO&P funds will be available to augment basic funds work on this output.

2. An output in the Output Name column has dependencies if progress on that output depends on the approval of outputs listed in the Dependencies column.

Output Name		Basic		Projected TO&P[1]	Dependencies[2]	Page
		Baseline	Add-on			
TS-1A	Security Risk Taxonomy (1996-1997)		▮			II-157
TS-2A	Security Risk Evaluation Methodology (1996-1997)		▮			II-158
TS-3A	Security Analysis Toolkit (1996-1998)		▮			II-159
TS-4A	Security Guidelines for Open Systems Acquisition (1996-1997)		▮			II-160
TS-5A	Description of the State of Security Architectures of Unbounded Domains (1996)		▮			II-166
TS-6A	Description of Internet Architectural and Domain Elements (1996)		▮			II-167
TS-7A	Software Security Models and Language Description (1996-1997)		▮		TS-5A, TS-6A	II-168
TS-8A	Software Engineering Framework for Trustworthy System Development (1996-1997)		▮		DE-12B, DE-13B	II-169

Figure A-4: Add-On Work Outputs for Trustworthy Systems

1. The Projected TO&P Column for an output is marked with a black rectangle if there is more than a 50% chance that TO&P funds will be available to augment basic funds work on this output.

2. An output in the Output Name column has dependencies if progress on that output depends on the approval of outputs listed in the Dependencies column.

Output Name		Basic		Projected TO&P[1]	Dependencies[2]	Page
		Baseline	Add-on			
PI-1B	Recommended Practices in Software Engineering (1996)	■				II-180
PI-2B	Professional Development Infrastructure (1996)	■				II-181
PI-3B	Guidelines for Training (1996)	■				II-183
PI-4B	Conference on Software Engineering Education (1996)	■				II-184
PI-1A	Desktop Hypermedia System for Recommended Software Engineering Practices (1996)		■			II-185

Figure A-5: Baseline and Add-On Work Outputs for Professional Infrastructure

1. The Projected TO&P Column for an output is marked with a black rectangle if there is more than a 50% chance that TO&P funds will be available to augment basic funds work on this output.
2. An output in the Output Name column has dependencies if progress on that output depends on the approval of outputs listed in the Dependencies column.

Output Name		Basic		Projected TO&P[1]	Dependencies[2]	Page
		Baseline	Add-on			
IT-1A	Transition Packages for Level 2 KPAs (1996-1997)		■			II-195
IT-2A	Software Technology Transition Workshop Series (1996)		■			II-197
IT-3A	Adoption of COTS Products Supporting Engineering Collaboration (Feasibility Study) (1996)		■			II-198
IT-4A	Update of the "Managing Technological Change" Course (1996)		■			II-200
IT-5A	Extension of the "Consulting Skills Workshop" for Software Engineers (1996)		■			II-201
IT-6A	Software Strategic Planning Workshop (1996)		■			II-202
IT-7A	Applying Conflict Resolution Skills in Software Development (Feasibility Study) (1996)		■			II-206
IT-8A	Human Interaction Capability (HIC) Framework (Feasibility Study) (1996)		■			II-207

Figure A-6: Add-On Work Outputs for Integrated Transition Strategies and Methods

1. The Projected TO&P Column for an output is marked with a black rectangle if there is more than a 50% chance that TO&P funds will be available to augment basic funds work on this output.
2. An output in the Output Name column has dependencies if progress on that output depends on the approval of outputs listed in the Dependencies column.

Output Name		Basic		Projected TO&P[1]	Dependencies[2]	Page
		Baseline	Add-on			
CO-1B	Impact of Software Engineering Practices (1996-1997)	▮				II-213
CO-2B	Quarterly Update / Summary of Technical Operations (1996-2000)	▮				II-214
CO-3B	Resident Affiliate Program (1996-2000)	▮				II-214
CO-4B	SPIN Coordination (1996-2000)	▮				II-215
CO-5B	SEPG Conference (1996-2000)	▮				II-216

Figure A-7: Baseline Work Outputs for Community Outreach

1. The Projected TO&P Column for an output is marked with a black rectangle if there is more than a 50% chance that TO&P funds will be available to augment basic funds work on this output.

2. An output in the Output Name column has dependencies if progress on that output depends on the approval of outputs listed in the Dependencies column.

Appendix B Resident Affiliates

	Organization	Total (as of 07/13/95)
Industry	AT&T Bell Labs	1
	Bell Northern Research, Inc.	1*
	Boeing	1
	Computer Sciences Corporation	6*
	Data General Corporation	1*
	GE Aerospace	2
	General Dynamics	1
	GTE Government Systems	4*
	Hughes Aircraft Company	5
	Lockheed Missiles & Space Co., Inc.	1
	Loral Federal Systems	6*
	Pacific Bell	1
	Process, Inc.	1
	Raytheon Company	1
	SEMATECH	1
	Siemens Corporate Research	1
	SYSCON Corporation	1
	TeleSoft	1
	Texas Instruments	3*
	Unisys	3
	Westinghouse Electric Corporation	3
	Wilcox Electric	1

	Organization	Total (as of 07/13/95)
Government	Coastal Systems Station	1
	Naval Air Development Center	2
	Naval Air Warfare Center	1
	Naval Ocean Systems Center	3
	Naval Surface Warfare Center	4*
	Naval Undersea Warfare Center, Division Newport	1*
	Naval Undersea Warfare Engineering Station	2
	Naval Weapons Center	2
	Communications-Electronics Command	6
	United States Military Academy	1
	Air Combat Command	1
	Air Force Institute of Technology	5
	Air Logistics Center	1
	Electronic Systems Center	3*
	Space Command	1
	Standard Systems Center	1
	Department of Defense	11*
	Nuclear Regulatory Commission	1*

* indicates current resident affiliate

Appendix C Active SPIN Groups

Domestic SPIN Groups[1]			
Alabama	Birmingham	Massachusetts	Boston
	Huntsville	Missouri	Kansas City
	Montgomery		St. Louis
Arizona	Phoenix	Nebraska	Omaha Area
	Tucson	New Jersey	North Jersey
California	Bay Area	New Mexico	Albuquerque
	Los Angeles	New York	Hudson Valley
	Sacramento Valley	Ohio	Northeast Ohio
	Silicon Valley	Oklahoma	Stillwater
	Southern California	Pennsylvania	Pittsburgh
Connecticut	Fairfield	Texas	Austin
Colorado	Front Range Area		Dallas/Ft. Worth
District of Columbia	Washington DC		Houston
Florida	Central Florida	Virginia	Hampton Roads
Georgia	Atlanta	Nebraska	Omaha Area
Illinois	Chicago		

1. As of 07/13/95

International SPIN Groups[2]			
Australia	Adelaide	Israel	Tel Aviv
	Canberra	Italy	Valenzano
	Melbourne		Torino
	Perth	Korea	Seoul
	Sydney	Netherlands	Eindhoven
Canada	Montreal	Spain	Madrid
France	Les Clayes sous Bois		Bizkaia
Hong Kong	Kowloon	Sweden	Stockholm
India	Bangalore	United Kingdom	Guildford
Ireland	Dublin		Bath

2. As of 07/13/95

Appendix D Technical Collaboration Agreements

Technical Collaboration Agreements[1]	
Allied Signal Aerospace	Motorola
Applied Software Engineering Center, Canada	Robbins-Gioia, Inc.
Citibank	Siemens
Citicorp	Student Loan Marketing Association (SALLIE MAE)
Ford/ACD	University of Southern California Center for Software Engineering
Hughes Aircraft Company	Xerox
Loral Federal Systems	

1. As of 07/13/95

Appendix E Advisory Groups

The Software Process Advisory Board oversees process outputs, services, and their supporting projects. It provides on-going advice concerning current and future strategic directions of the Process impact area. Board meetings are held twice a year, with interim contact on specific subjects of relevance to the Board's charter. Board members were carefully chosen for their expertise and experience; they are

- two members from the Department of Defense (DoD)
- two members from the DoD contractor community
- one member from industry (a non-DoD contractor)
- two members from academia
- one member from Carnegie Mellon University (CMU)
- three former Software Engineering Institute (SEI) Process directors

The Software Process Measurement Steering Committee (MSC) is composed of 20 leaders in software measurement and management from industry, academia, government, and the SEI.

The Measurement Steering Committee (MSC) is assembled to provide technical input to the SEI Software Engineering Measurement (SEM) team. The MSC will assist the SEI SEM team in

- identifying measurement needs
- reviewing measurement products and services
- assessing or reviewing status, progress, activities, and goals
- identifying and planning future strategic directions

The MSC will also advise and assist the SEI in the transition of measurement and quantitative analyses into the (national) software engineering community.

The Capability Maturity Model (CMM) Advisory Board (CAB) is the formal mechanism for public involvement in building the CMM and CMM-based products since 1989. The CAB's purpose is to function as a standing committee that formally provides user input on CMM activity area objectives and releases and produces recommendations to which the CMM area can respond.

The mission of the CAB is to

- independently review proposed enhancements to the CMM and related products before their release
- provide written recommendations, based on these reviews, with the objective of furthering the SEI's mission and the satisfaction of our user community
- advise on CMM product objectives and development plans
- facilitate communication between the CMM activity area and the user community

The Systems Engineering Capability Maturity Model (SE-CMM) Steering Group oversees the Systems Engineering CMM. This group consists of six industrial participants, the SEI, ex-officio members from the National Council on Systems Engineering, and representatives from the U.S. government. The group meets four times a year, with interim contacts as needed. It has release authority over project work products and provides strategic direction on maintenance and expansion of the SE-CMM.

The People CMM (P-CMM) Advisory Board assists the P-CMM group in achieving its mission and strategic national objectives. The P-CMM Advisory Board brings together leading experts in (1) human resources management of software engineering organizations, (2) process improvement in software engineering and information systems organizations, and (3) the management of software engineering and information systems organizations, to represent industry and government's interests in the P-CMM. The board's 17 members represent industry, as well as civilian and defense agencies of the U.S. government.

Responsibilities of the P-CMM Advisory Board include the following:

- Advise the P-CMM group on the structure and content of the P-CMM.
- Advise the group on the readiness of the P-CMM for public release.
- Advise the group on appropriate release strategies for P-CMM products and services.
- Advise the group on the practicality of its plans.
- Identify reviewers for the P-CMM.
- Identify pilot sites for applying the P-CMM.
- Evaluate the group's performance and outcomes.
- Raise national interest and awareness in the P-CMM

The Disciplined Engineering Advisory Board provides an external perspective on the long-term goals and strategies of Disciplined Engineering. The Advisory Board is exposed to the strategic goals of this impact area and the plans that support the attainment of these goals.

The purpose of the board is to

- provide independent thinking and technically sound counsel regarding software technology and its reflection in software engineering practice, and regarding the most appropriate role of the Disciplined Engineering area of the SEI
- provide concrete advice and recommendations on the goals, strategies, and plans of the SEI's Disciplined Engineering area, including the relationship between the parts of the area
- represent customer needs and interests for each board member's segment of the software community
- provide advice and recommendations regarding the relationship of Disciplined Engineering with other SEI and external activities
- act as external advocates for Disciplined Engineering

The board consists of eight members selected by the SEI: three members each from government and industry, and two members from academia. The terms are staggered and last three years. The board meets three times a year.

The Trustworthy Systems Advisory Board is made up of government, industry, and academic leaders who have special information system security needs and insights. The Board, which meets twice a year, reviews the goals, strategies, objectives, and plans of the Trustworthy Systems impact area to help the area make the most effective use of its resources in achieving its goal: raising the security level of operational information systems.

The Risk Advisory Board assists the Risk impact area in achieving its goals. The Board brings together leading experts in (1) managing risk in software engineering and information systems organizations, and (2) managing system acquisitions. The members represent industry and government's interests in the Risk area's goals and strategies. The Board's seven members represent industry and the DoD. Responsibilities of the Advisory Board include the following:

- Help the Risk area understand the needs and perspectives of the acquisition and development community.
- Advise the Risk area on its future direction.
- Provide Risk with feedback on (1) its project objectives and plans and (2) its outputs and services.
- Promote risk management and acquisition improvement awareness and advocacy.

The Education Advisory Board, once advising the SEI on its former Educational Products Program, now advises the SEI on the Maturing the Professional Infrastructure activity area and other activities related to education. The board provides concrete advice and recommendations to those involved in educational endeavors, the SEI's long-term goals and strategies related to education, and near-term plans. The board meets twice a year and consists of six members: two each from academia, government, and industry. Each member represents the customer needs and interests of his or her community.

List of Acronyms

ADL	architecture description language
AFB	Air Force Base
ARMSS	Acquisition Risk Management Source Selection
ARPA	Advanced Research Projects Agency
ART	Advanced Real Time
ASC/YT	Aeronautical Systems Command
ASC	Aeronautical Systems Center
AWACS	Airborne Warning and Control System
C3I	command, control, communications, and intelligence
CAE	conjecture, analysis, evaluate
CAF	CMM Appraisal Framework
CARDS	Comprehensive Approach for Reusable Defense Software
CASE	computer-aided software engineering
CBA	CMM-based appraisal
CBA SCE	CMM-Based Appraisal for Software Capability Evaluation
CBA/IPI	CBA Internal Process Improvement
CECOM	Communications-Electronics Command
CMM	Capability Maturity Model
CORBA	common object request broker architecture
COTS	commercial off-the-shelf software
CRADA	Cooperative Research and Development Agreement
CSC	Computer Sciences Corporation
CSEE	Conference on Software Engineering Education
CSL	Computer Science Laboratory
CSTO	Computer Systems Technology Office
DE	Disciplined Engineering
DEC	Digital Equipment Corporation
DISA	Defense Information Systems Agency
DMA	Defense Mapping Agency
DSSA	domain-specific software architectures

DoD	Department of Defense
EDCS	Evolutionary Design of Complex Software
EPRI	Electrical Power Research Institute
ESC	Electronic Systems Center
ETRB	Education and Training Review Board
FAA	Federal Aviation Administration
FEAST	Feedback, Evolution, and Software Technology
FODA	Feature-Oriented Domain Analysis
FTP	File Transfer Protocol
HIC	Human Interaction Capability
IEEE	Institute of Electrical and Electronic Engineers
IP	interim profiles
IPD	Integrated Product Development
IPI	internal process improvement
IPPD	Integrated Product and Process Development
ISO/IEC	International Organization for Standardization and International Electrotechnical Commission
ITSM	Integrated Transition Strategies and Methods
JAST	Joint Advanced Strike Technology
JLC	Joint Logistics Commanders
JPO	joint program office
KPA	key practice area
K-SAV	knowledge summarization, analysis, and visualization
MBSE	model-based software engineering
MCC	Microelectronics and Computer Technology Corporation
MEL	Manufacturing Engineering Laboratory
MICOM	Missile Command
MIT	Massachusetts Institute of Technology
ML	maturity level
NASA	National Aeronautics and Space Administration
NAVAIR	Naval Air Systems
NAVOCEANO	Naval Oceanographic Office

NGCR	Next Generation Computing Resources
NII	National Information Infrastructure
NIST	National Institute of Standards and Technology
NOAA	National Oceanographic and Atmospheric Sciences Agency
NSA	National Security Agency
NSF	National Science Foundation
NUWC	Naval Underwater Warfare Center
OASD	Office of the Secretary of Defense
OOPSLA	Object-Oriented Programming Systems, Languages and Applications
ORB	object request broker
OSD	Office of the Secretary of Defense
OSJTF	Office of the Under Secretary of Defense's Open System Joint Task Force
PAIS	Process Appraisal Information System
PAL	process asset library
PAT	process action teams
P-CMM	People Capability Maturity Model
PDSS	post-deployment software support
PEL	process example library
PEO	program executive officers
PI	process improvement
PRISM	Portable Reusable Integrated Software Modules
PSP	Personal Software Process
PVM	Process Value Method
QU	Quarterly Update
R&D	research and development
ROAM	Reuse Opportunity Analysis Method
ROI	return on investment
SAAM	Software Architecture Analysis Method
SAMM	Software Acquisition Maturity Model
SCE	Software Capability Evaluation
SE-CMM	Systems Engineering Capability Maturity Model
SEIR	Software Engineering Information Repository

SEPG	software engineering process group
SERR	Software Engineering Risk Repository
SISTO	Software and Intelligent Systems Technology Office
SMC	Space and Missile Center
SPA	software process appraisal
SPAWAR	Space Warfare Systems
SPC	Software Productivity Consortium
SPI	software process improvement
SPICE	Software Process Improvement Capability dEtermination
SPIN	software process improvement network
SPMT	software process modeling technology
SRE	Software Risk Evaluation
SRM	Software Risk Metrics
STARS	Software Technology for Adaptable, Reliable Systems
STEIM	Software Engineering Environments Technology Evaluation, Integration, and Measurement
STO	Summary of Technical Operations
STSC	Software Technology Support Center
TACOM	Tank-Automotive Command
T-CMM	Trusted CMM
TO&P	technical objectives and plans
TXM	Technology Transfer Model
USC	University of Southern California
USC-CSE	University of Southern California-Center for Software Engineering

UNLIMITED, UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

1a. REPORT SECURITY CLASSIFICATION Unclassified	1b. RESTRICTIVE MARKINGS None
2a. SECURITY CLASSIFICATION AUTHORITY N/A 2b. DECLASSIFICATION/DOWNGRADING SCHEDULE N/A	3. DISTRIBUTION/AVAILABILITY OF REPORT Approved for Public Release Distribution Unlimited
4. PERFORMING ORGANIZATION REPORT NUMBER(S) CMU/SEI-95-SR-027	5. MONITORING ORGANIZATION REPORT NUMBER(S)
6a. NAME OF PERFORMING ORGANIZATION Software Engineering Institute 6b. OFFICE SYMBOL (if applicable) SEI	7a. NAME OF MONITORING ORGANIZATION SEI Joint Program Office
6c. ADDRESS (city, state, and zip code) Carnegie Mellon University Pittsburgh PA 15213	7b. ADDRESS (city, state, and zip code) HQ ESC/ENS 5 Eglin Street Hanscom AFB, MA 01731-2116
8a. NAME OFFUNDING/SPONSORING ORGANIZATION SEI Joint Program Office 8b. OFFICE SYMBOL (if applicable) ESC/ENS	9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER F19628-95-C-0003
8c. ADDRESS (city, state, and zip code)) Carnegie Mellon University Pittsburgh PA 15213	10. SOURCE OF FUNDING NOS. PROGRAM ELEMENT NO 63756E; PROJECT NO. N/A; TASK NO N/A; WORK UNIT NO. N/A

11. TITLE (Include Security Classification)
SEI Program Plans: 1996-2000

12. PERSONAL AUTHOR(S)

13a. TYPE OF REPORT	13b. TIME COVERED	14. DATE OF REPORT (year, month, day)	15. PAGE COUNT
Final	FROM TO	January 1996	254

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB. GR.

18. SUBJECT TERMS (continue on reverse of necessary and identify by block number)
program plans, technical program, strategy, one-year plans, five-year plan, technology transition, outputs, software process, risk management, discipline engineering, trustworthy systems, professional infrastructure, integrated transition strategies and methods

19. ABSTRACT (continue on reverse if necessary and identify by block number)
This document, which is in two volumes, presents the Software Engineering Institute (SEI) strategy and one-year implementation plan for calendar year (CY) 1996, together with the SEI five-year program plans. Volume I describes the five-year strategic plan, and Volume II describes the one-year tactical plan. This document was written in early 1995 and was delivered to our sponsor (ARPA/ESC) as a contract deliverable in July 1995. As such, it was a draft plan; its execution depends primarily on approved resource allocations. The planning starts long before the Congress completes its budget authorization and appropriation. Historically, circumstances such as changing customer needs and shifting resource allocations have made it necessary to change our plans.

20. DISTRIBUTION/AVAILABILITY OF ABSTRACT UNCLASSIFIED/UNLIMITED ■ SAME AS RPT ☐ DTIC USERS ■	21. ABSTRACT SECURITY CLASSIFICATION Unclassified, Unlimited Distribution	
22a. NAME OF RESPONSIBLE INDIVIDUAL Thomas R. Miller, Lt Col, USAF	22b. TELEPHONE NUMBER (include area code) (412) 268-7631	22c. OFFICE SYMBOL ESC/ENS (SEI)

DD FORM 1473, 83 APR EDITION of 1 JAN 73 IS OBSOLETE

UNLIMITED, UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

ABSTRACT — continued from page one, block 19