ANITA BORG INSTITUTE
GRACE HOPPER CELEBRATION
OF WOMEN IN COMPUTING

2014

ANITA BORG INSTITUTE
WOMEN TRANSFORMING TECHNOLOGY

Association for Computing Machinery

#GHC14

Trends and New Directions in Software Architecture

Linda Northrop
Chief Scientist, Software Solutions Division
SEI Fellow
Software Engineering Institute (SEI)
Carnegie Mellon University
October 10, 2014

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
10 OCT 2014	**N/A**	**-**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
Trends and New Directions in Software Architecture	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
Linda Northrop	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)	8. PERFORMING ORGANIZATION REPORT NUMBER
Software Engineering Institute Carnegie Mellon University Pittsburgh, PA 15213	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release, distribution unlimited

13. SUPPLEMENTARY NOTES

FINAL REPORT V1.1 TATRC BIG DATA INVESTIGATION FINAL REPORT, The original document contains color images.

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			
unclassified	**unclassified**	**unclassified**	**SAR**	**55**	

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

DM-0001699

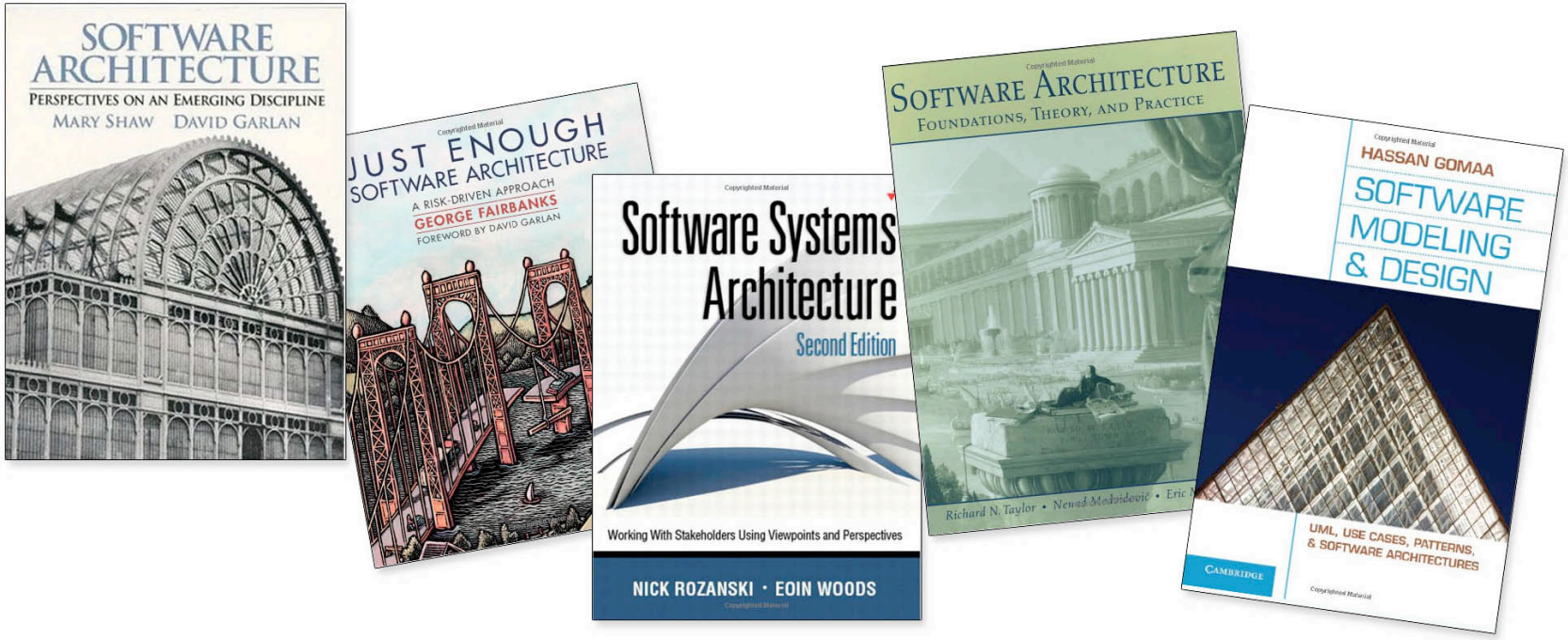
Software Architecture

- The quality and longevity of a software-reliant system is largely determined by its architecture.
- Recent US studies identify architectural issues as a systemic cause of software problems in government systems (OSD, NASA, NDIA, National Research Council).

Architecture is of enduring importance because it is the right abstraction for performing ongoing analyses throughout a system's lifetime.



Software Architecture Thinking



- High-level system design providing system-level abstractions and quality attributes, which help in managing complexity
- Makes engineering tradeoffs explicit

Quality Attributes

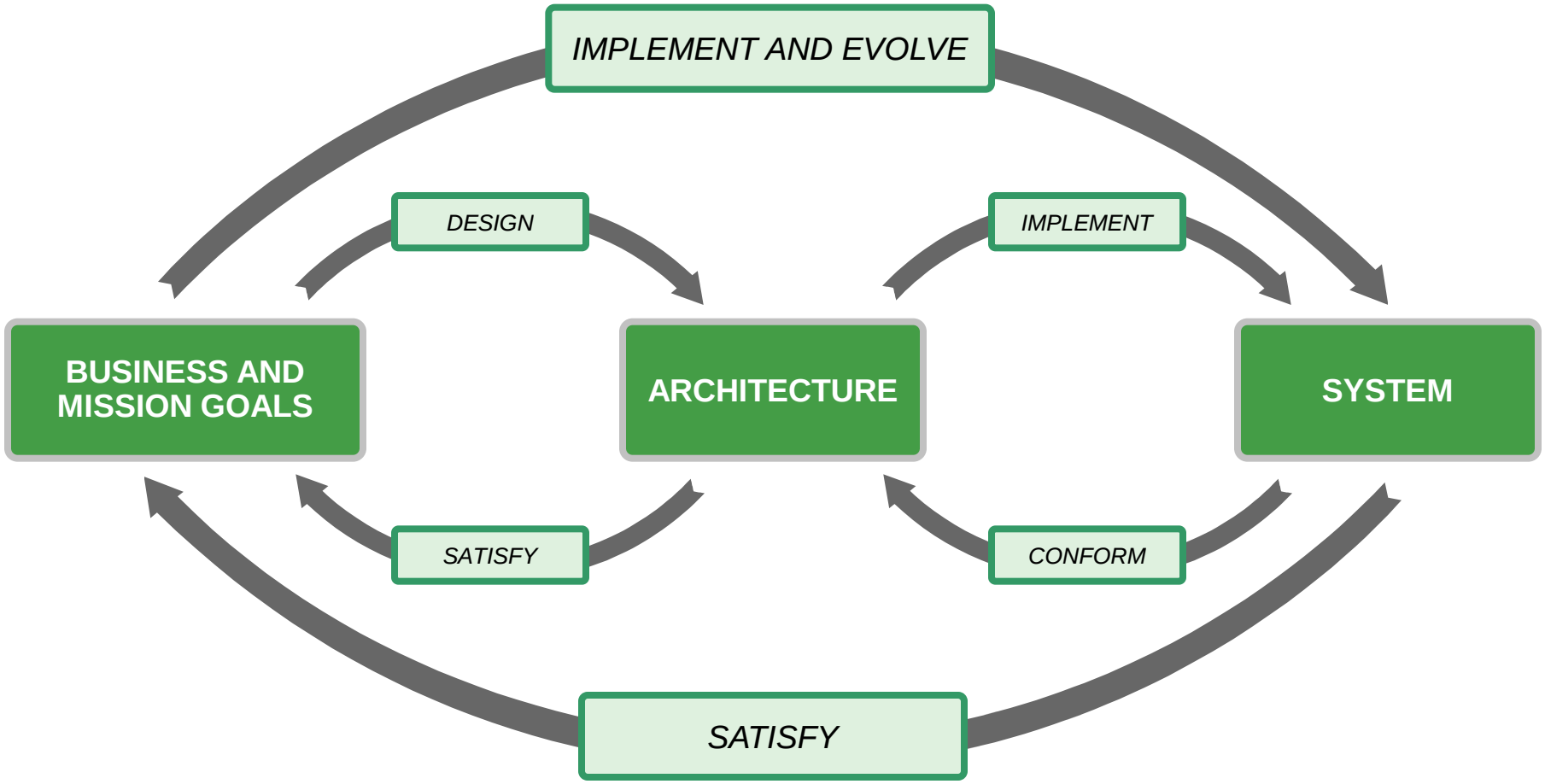
Quality attributes

- properties of work products or goods by which stakeholders judge their quality
- stem from business and mission goals.
- need to be characterized in a system-specific way.

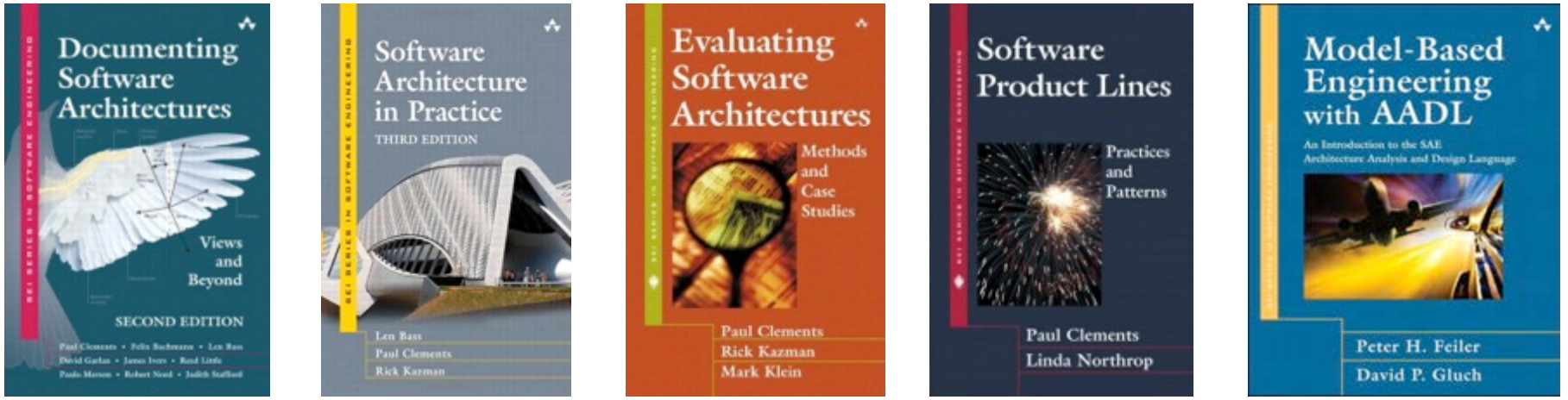
Quality attributes include

- Performance
- Availability
- Interoperability
- Modifiability
- Usability
- Security
- Etc.

Central Role of Architecture



Our View: Architecture-Centric Engineering



- *explicitly focus on quality attributes*
- *directly link to business and mission goals*
- *explicitly involve system stakeholders*
- *be grounded in state-of-the-art quality attribute models and reasoning frameworks*

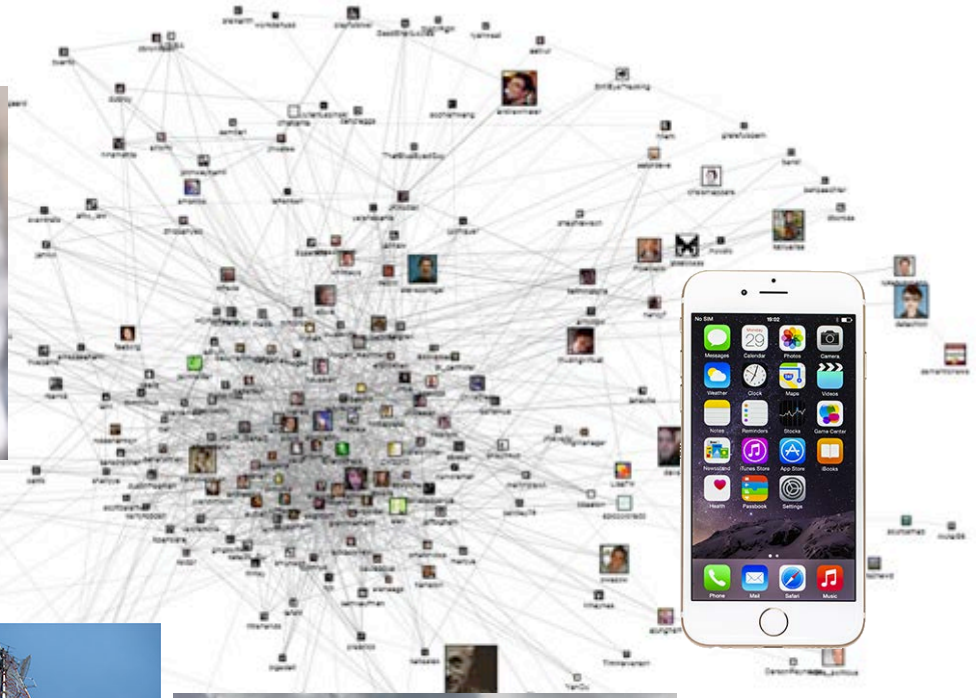
Advancements Over the Years

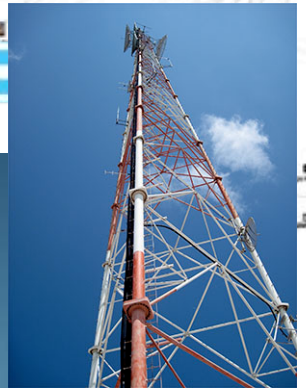
- Architectural patterns
- Component-based
- Company specific product lines
- Model-based
- Frameworks and platforms
- Standard interfaces

What HAS Changed?

- Increased connectivity
- scale and complexity
- decentralization and distribution
- “big data”
- increased operational tempo
- mismatched ecosystem tempos
- vulnerability
- collective action
- disruptive and emerging technologies

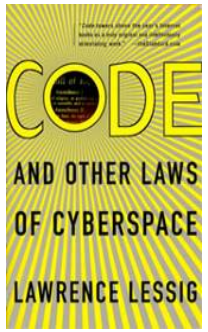
Technology Trends



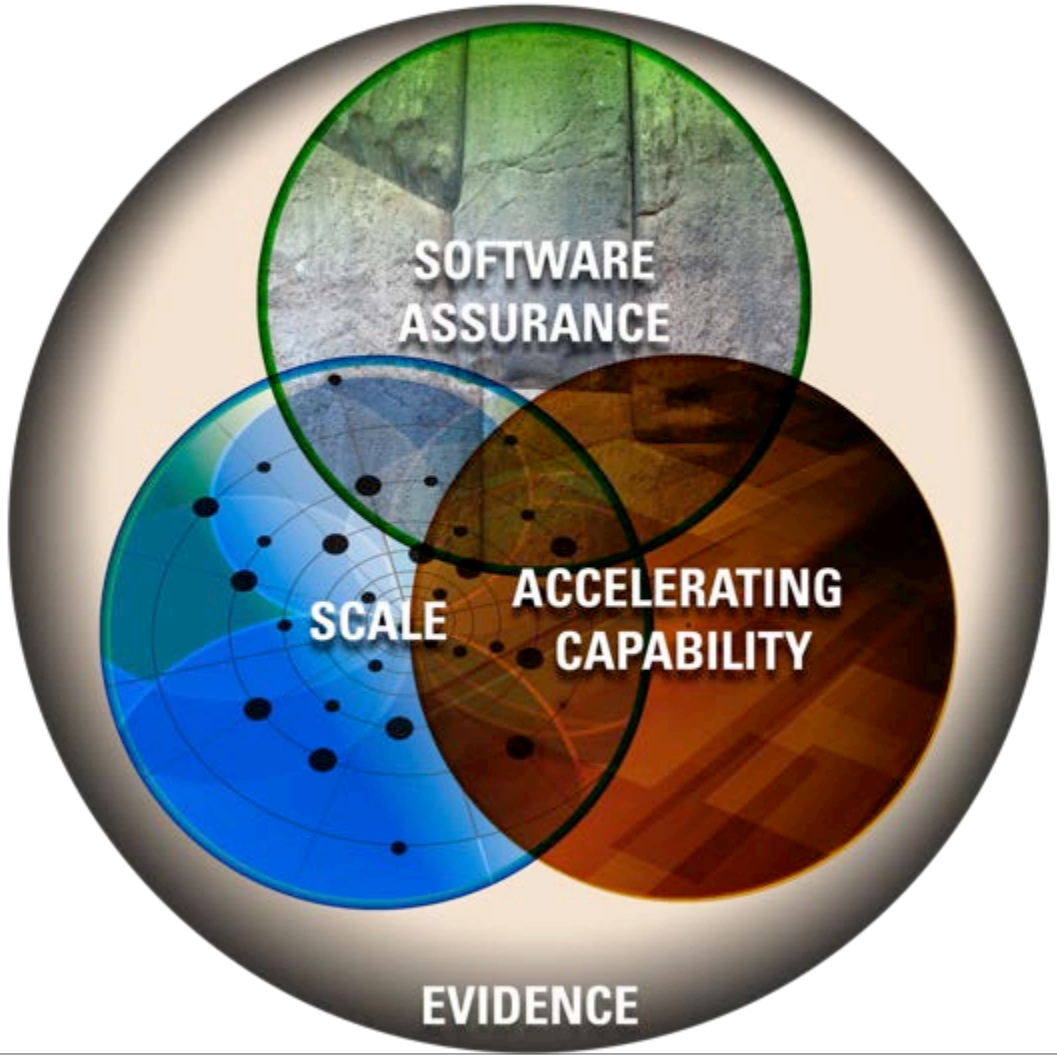


Software Development Trends

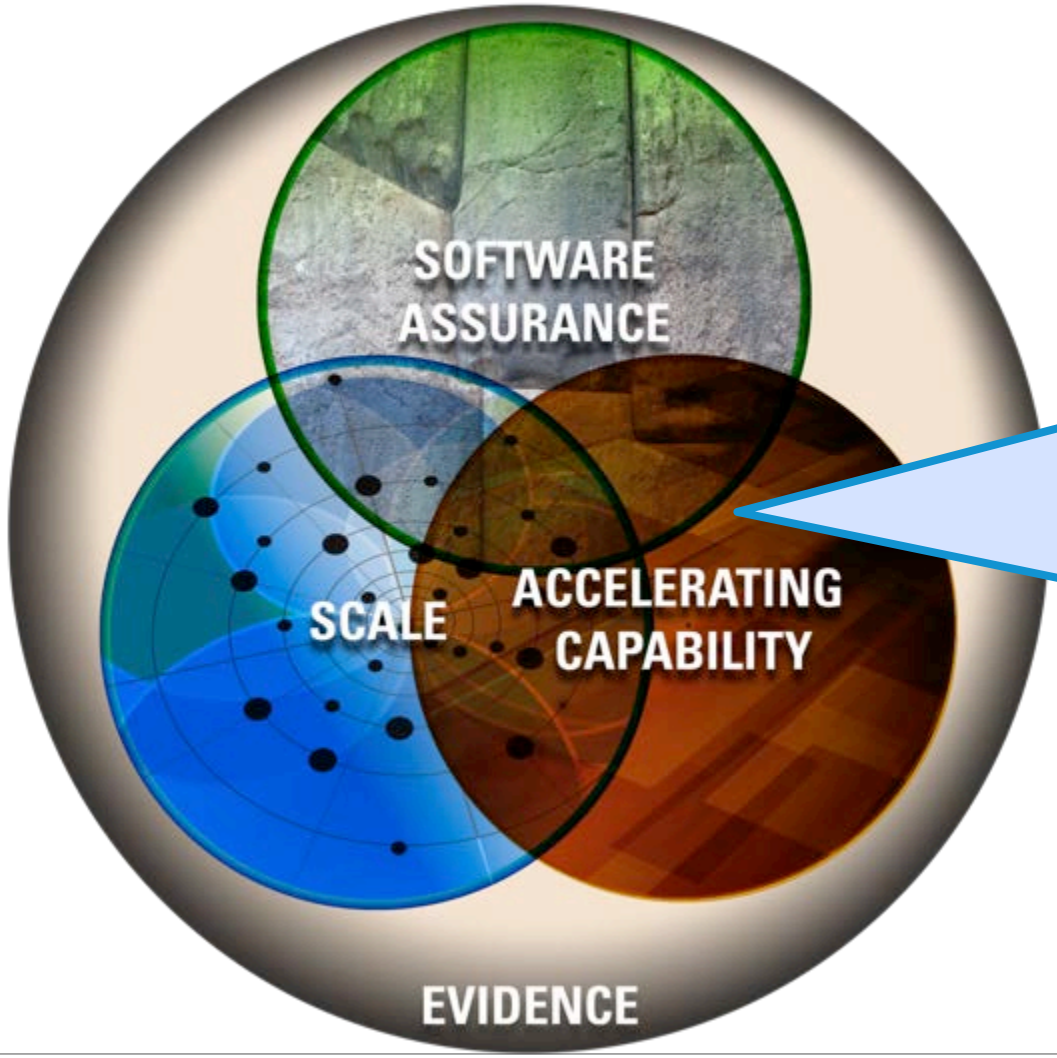
- Application frameworks
- Open source
- Cloud strategies
- NoSQL
- Machine Learning
- MDD
- Incremental approaches
- Dashboards
- Distributed development environments
- DevOps



Technical Challenges



The Intersection and Architecture



At the intersections there are difficult tradeoffs to be made in structure, process, time, cost, and assurance.

Architecture is the enabler for tradeoff analyses

Architecture and Accelerated Capability

How much architecture design is enough?

Can architecture design be done incrementally?

There is a difference between being agile and doing agile.

Agility is enabled by architecture – not stifled by it.

Managing technical debt is key.

Managing Technical Debt*

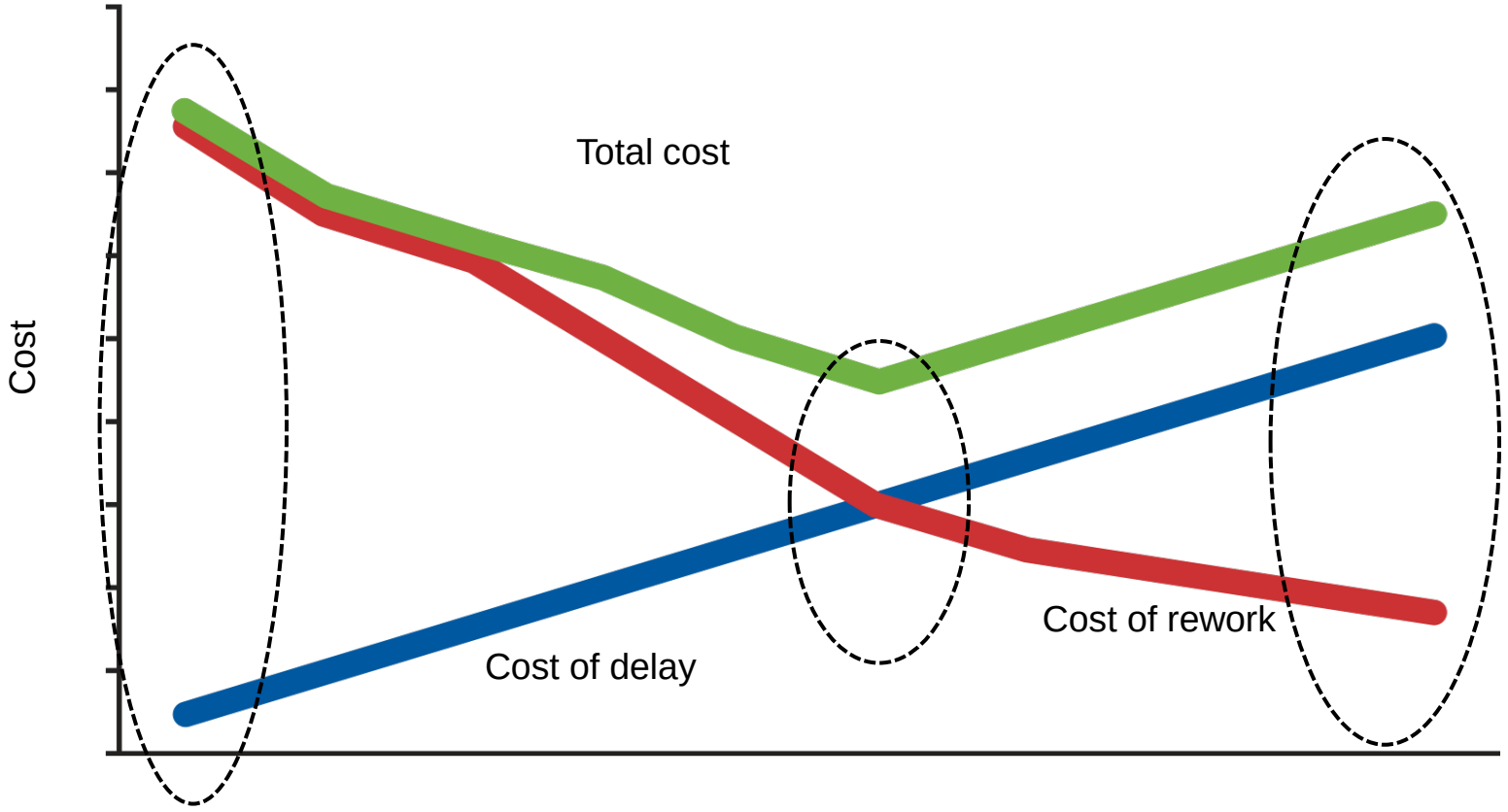
A design or construction approach that's expedient in the short term but that creates a technical context that increases complexity and cost in the long term.

Some examples include:

- Continuing to build on a foundation of poor quality legacy code
- Prototype that turns into production code
- Increasing use of "bad patches," which increases number of related systems that must be changed in parallel

* Term first used by Cunningham, W. 1992. *The WyCash Portfolio Management System*. OOPSLA '92 Experience Report. http://c2.com/doc/oopsla92.html.

Hitting the Sweet Spot



Architecture Increments

Technical Debt Landscape

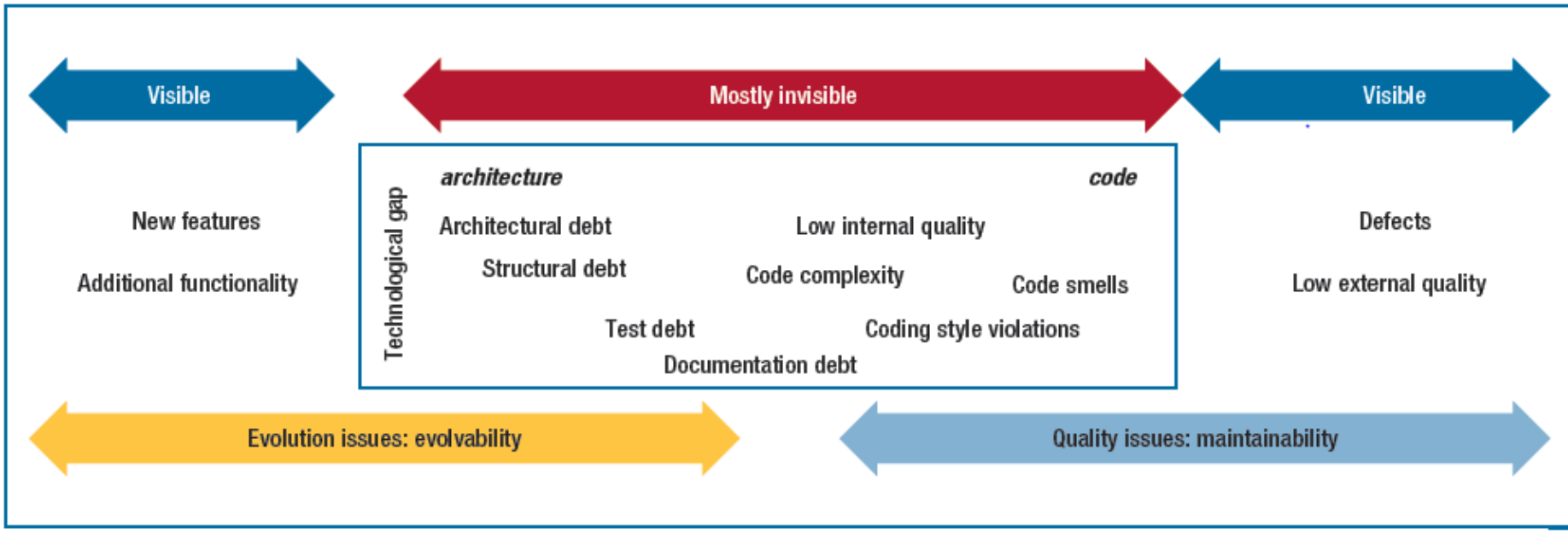


FIGURE 1. The technical debt landscape. On the left, evolution or its challenges; on the right, quality issues, both internal and external.

"invisible results of past decisions about software that negatively affect its future…deferred investment opportunities or poorly managed risks"

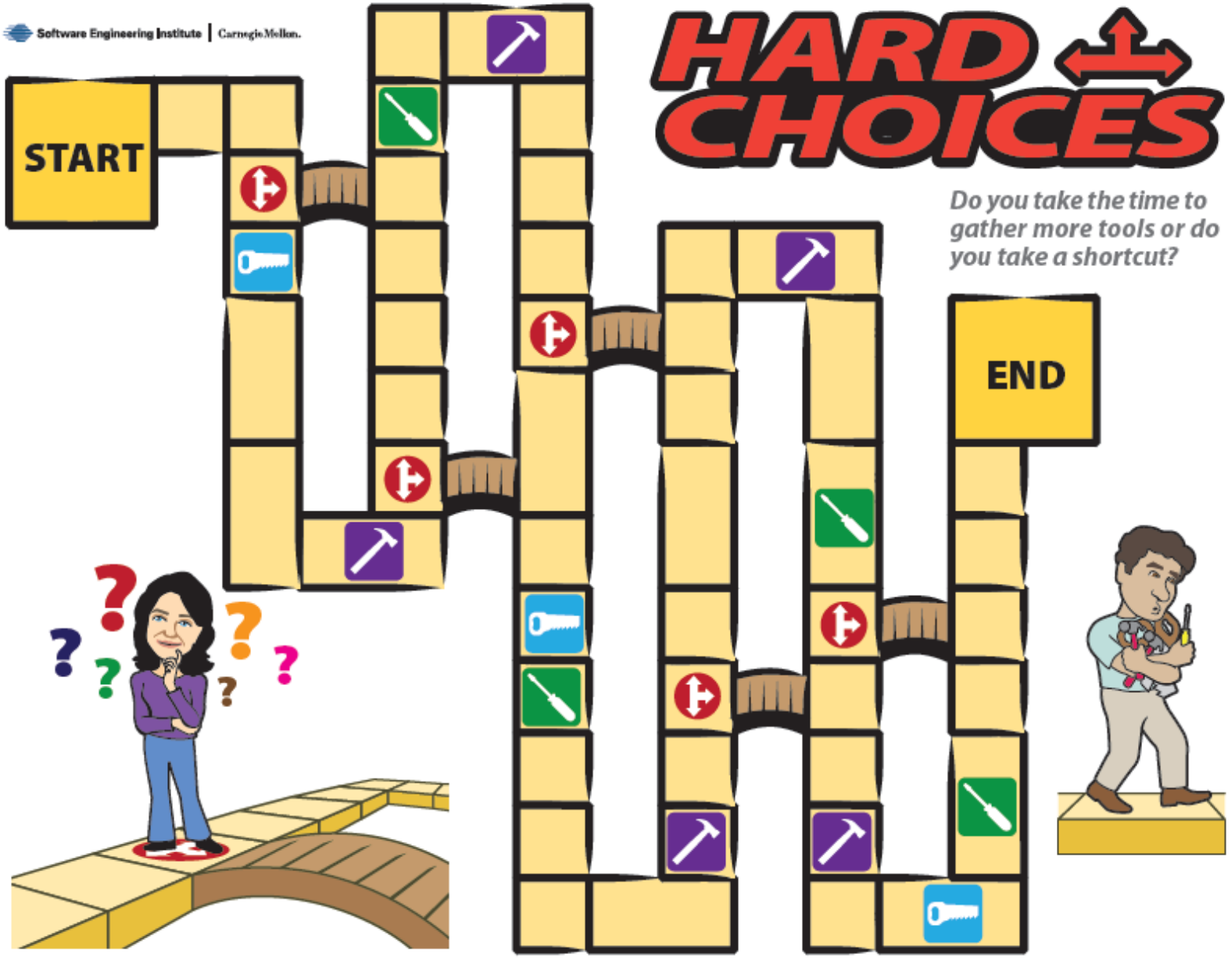
Kruchten, P. Nord, R.L., Ozkaya, I. 2012. Technical Debt: From Metaphor to Theory and Practice, IEEE Software, 29(6), Nov/Dec 2012.

Making Hard Choices about Technical Debt

In the quest to become market leader, players race to release a quality product to the marketplace.

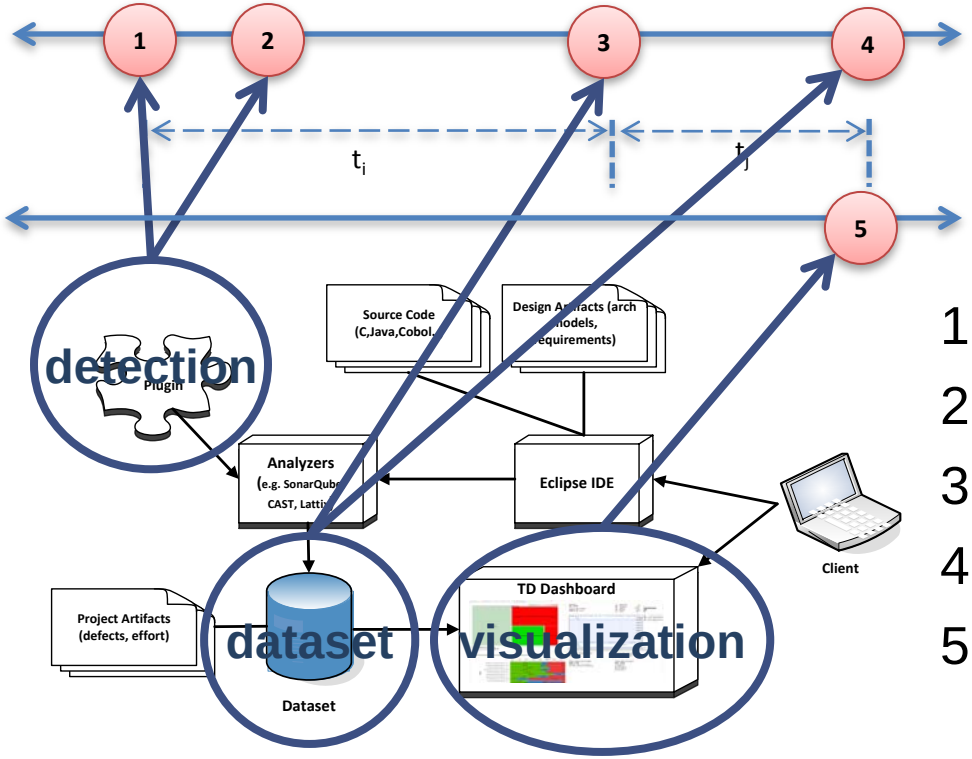
The Hard Choices game is a simulation of the software development cycle meant to communicate the concepts of uncertainty, risk, options, and technical debt.

Hard Choices Strategy Game to Communicate Value of Architecture Thinking game downloadable from http://www.sei.cmu.edu/architecture/tools/hardchoices/.


Software Engineering Institute | Carnegie Mellon.
HARD CHOICES
START
END
Do you take the time to gather more tools or do you take a shortcut?

Current Research

What code and design indicators that correlate well with project measures allow us to manage technical debt?



1. time technical debt is incurred
2. time technical debt is recognized
3. time to plan and re-architect
4. time until debt is actually paid-off
5. continuous monitoring

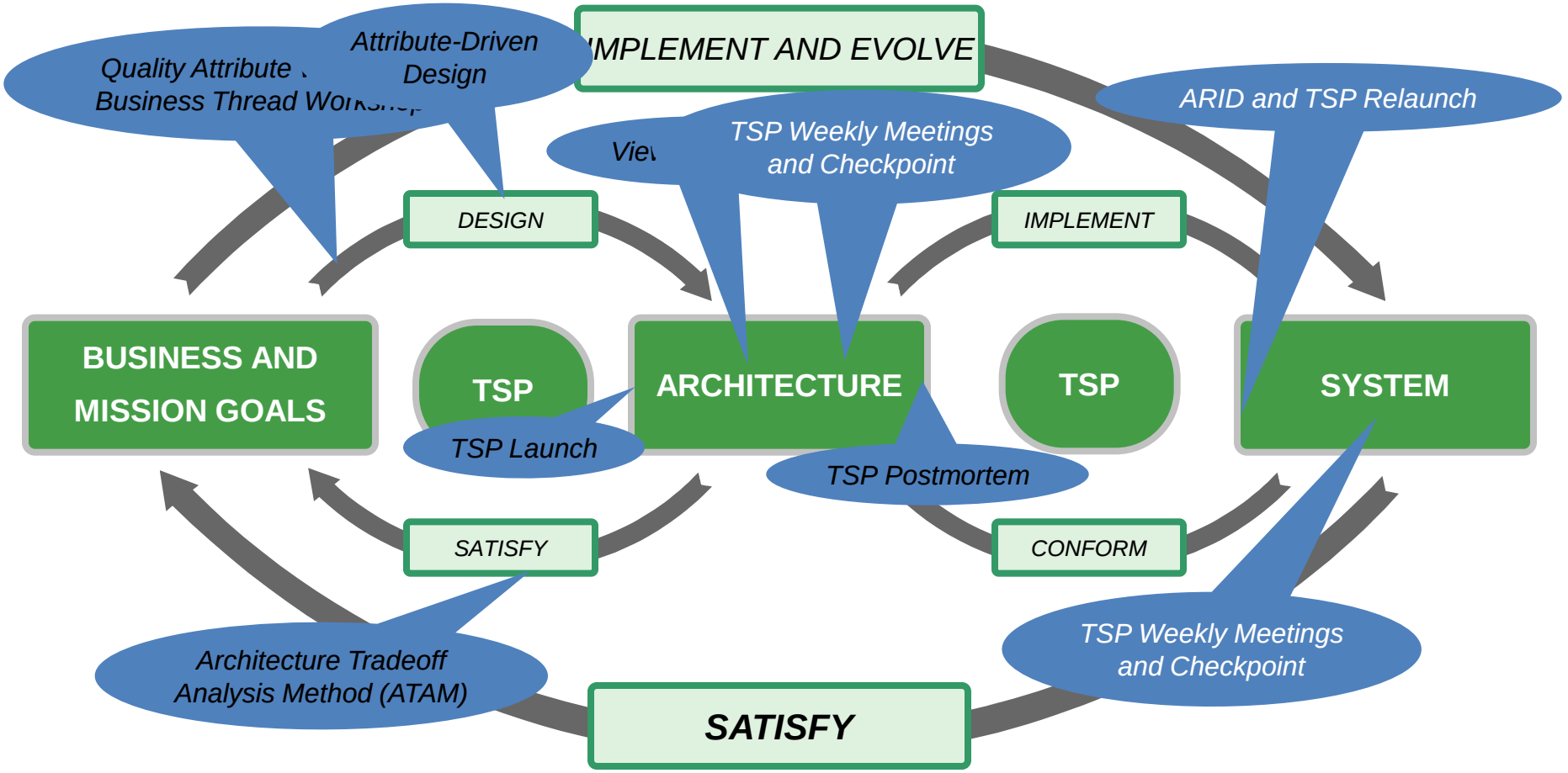
Architecture Done Incrementally

- Bolsa Mexicana de Valores (BMV) operates the Mexican Financial Markets on behalf of the Mexican government.
- Bursatec is the technology arm of the BMV.
- BMV desired a new stock trading engine to drive the market.
- BMV performed a build vs. buy analysis and determined that Bursatec would replace their three existing trading engines with one in-house developed system.

Bursatec committed to deliver a trading engine in 8-10 quarters.

- High performing
- Reliable and of high quality
- Scalable

Approach

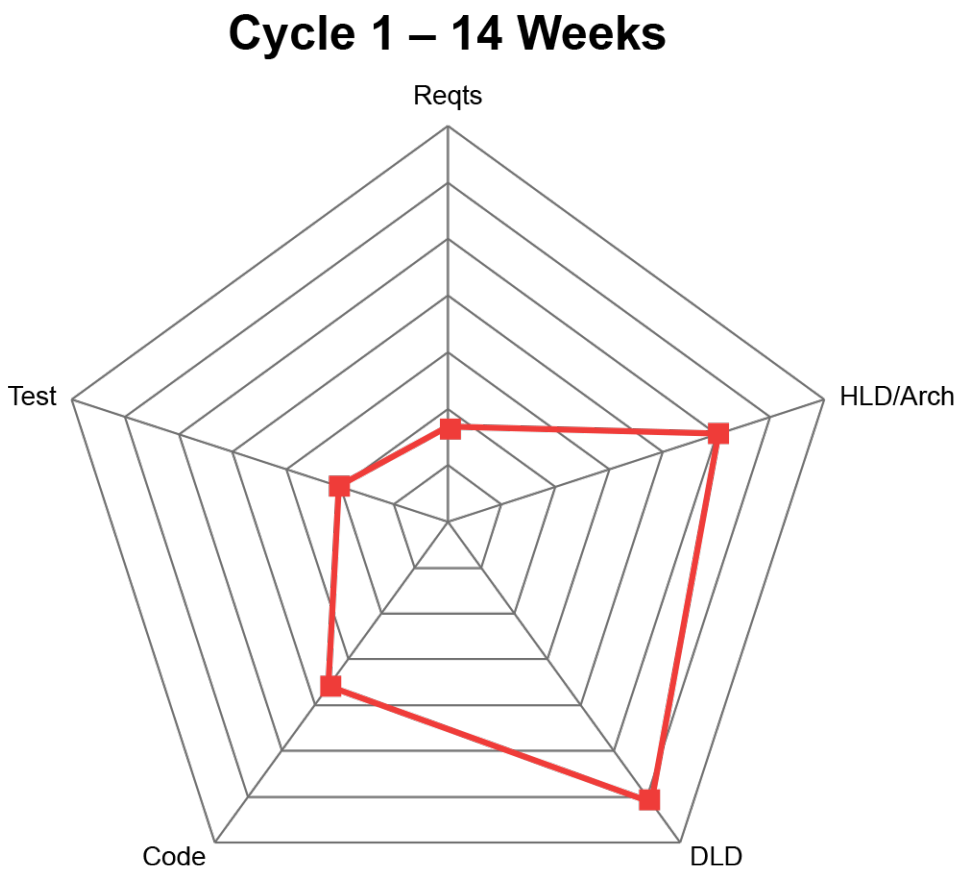


Team Software Process (TSP) and Architecture-Centric Engineering (ACE)

Project Challenges

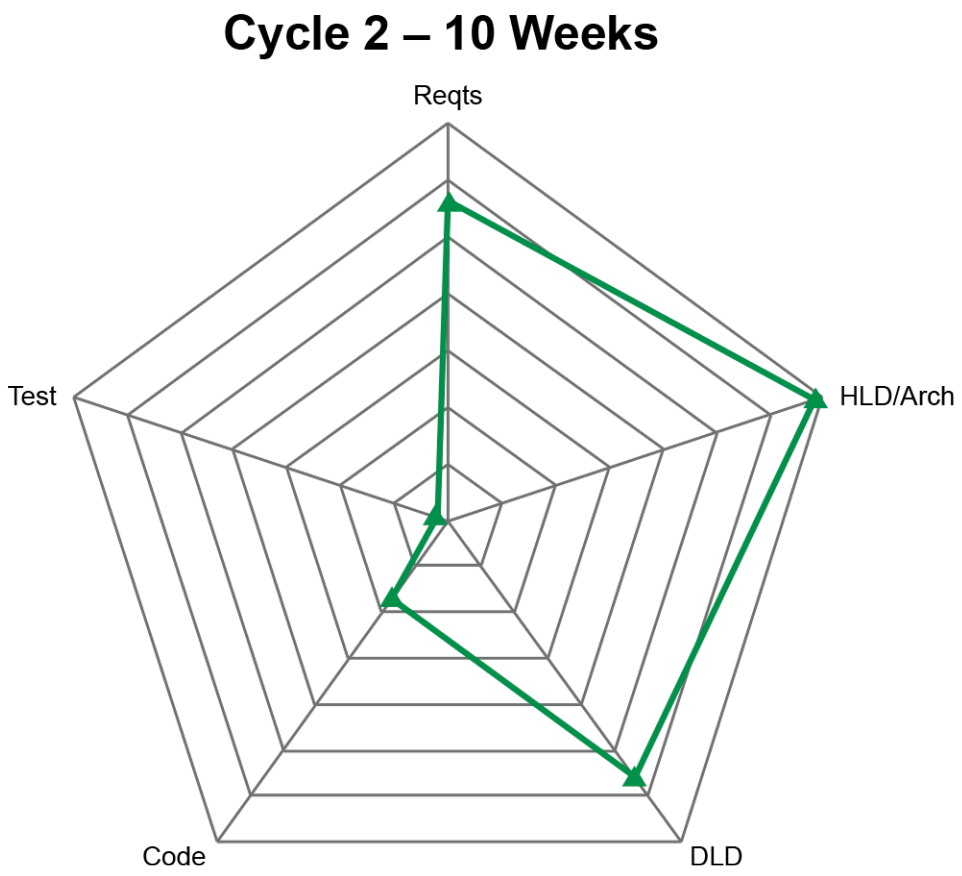
- Measuring, planning, estimating, and tracking architectural design activities
- Integrating architectural design activities with iterative/incremental development models and TSP
- Improving the as-practiced fidelity of the architecture development process
- Measuring the benefits and ROI for architecture practices

Effort in Percent over Cycles – 1



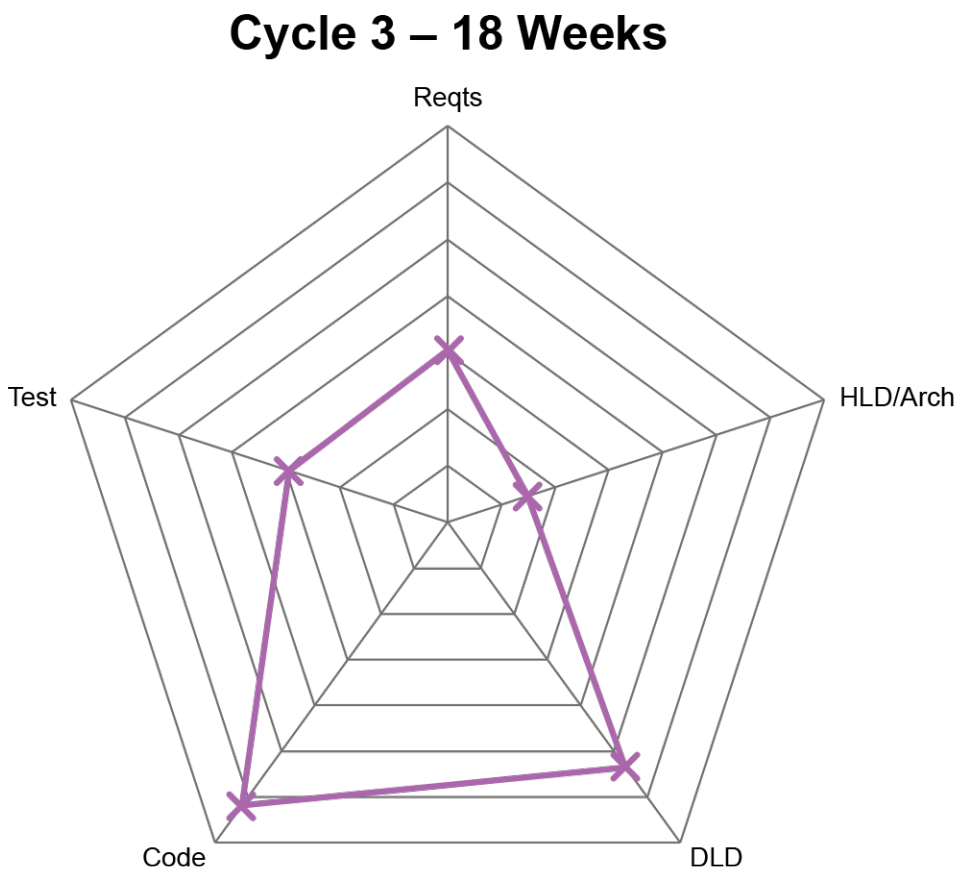
Reqts: Requirements
HLD/Arch: High level Design / Architecture
DLD: Detailed Design (UML)
Code: Coding (no detailed design)
Test: Testing

Effort in Percent over Cycles – 2



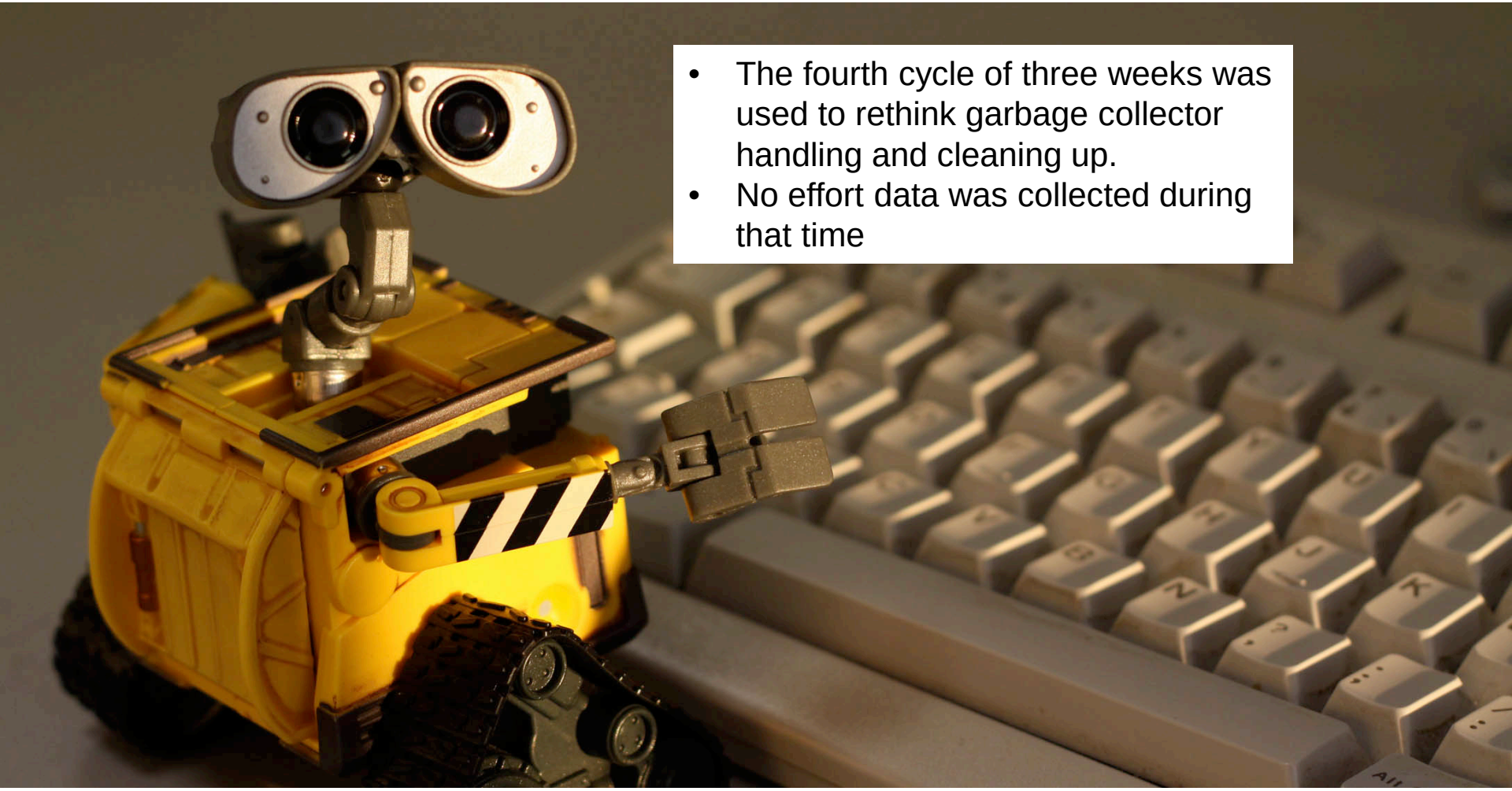
Reqts: Requirements
HLD/Arch: High level Design / Architecture
DLD: Detailed Design (UML)
Code: Coding (no detailed design)
Test: Testing

Effort in Percent over Cycles – 3



Reqts: Requirements
HLD/Arch: High level Design / Architecture
DLD: Detailed Design (UML)
Code: Coding (no detailed design)
Test: Testing

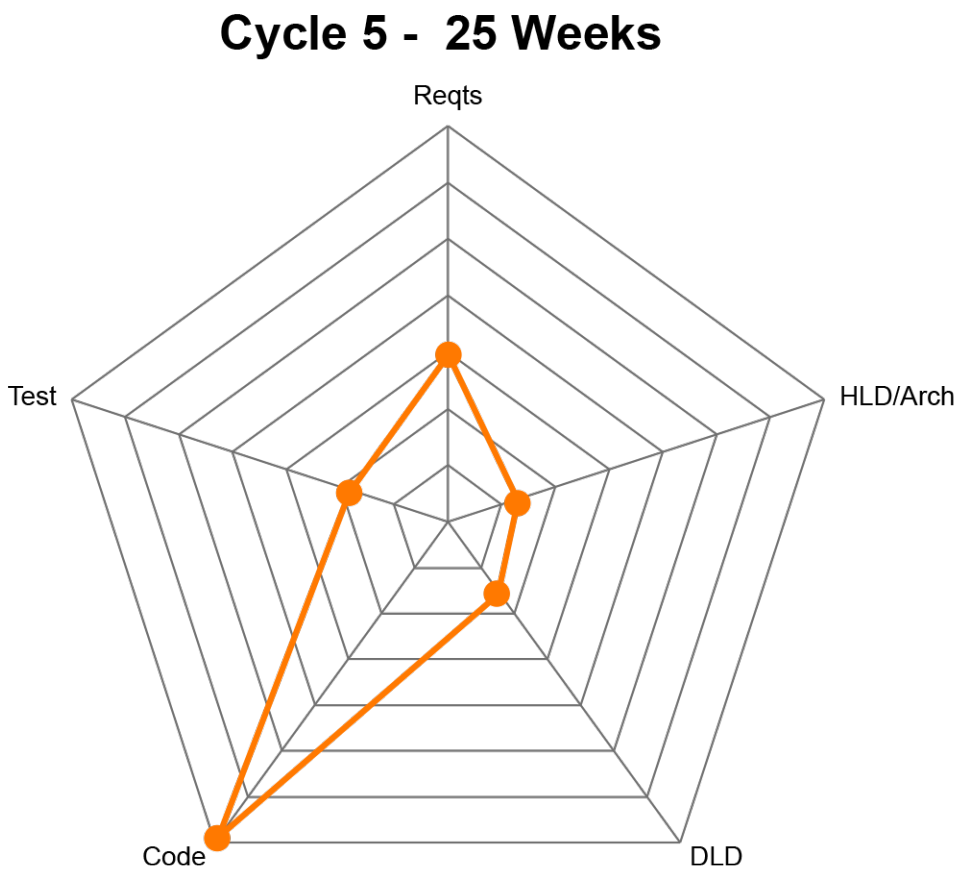
Effort in Percent over Cycles – 4



- The fourth cycle of three weeks was used to rethink garbage collector handling and cleaning up.
- No effort data was collected during that time

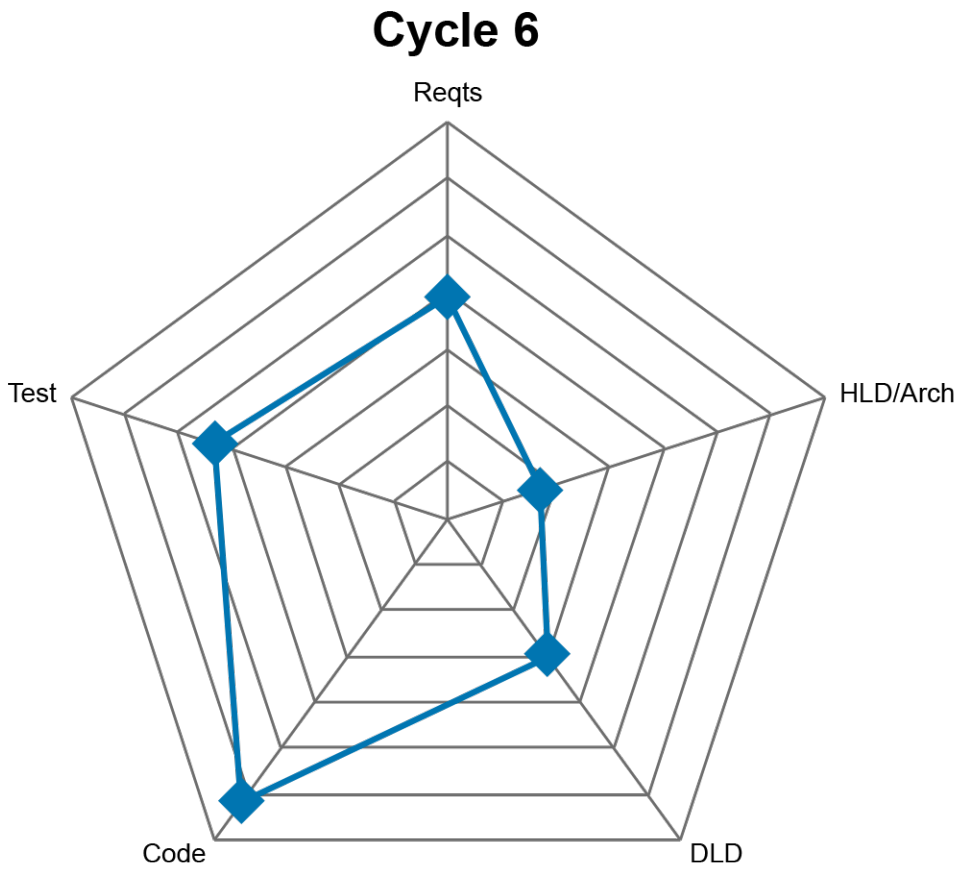
https://www.flickr.com/photos/arthur-caranta/

Effort in Percent over Cycles – 5



Reqts: Requirements
HLD/Arch: High level Design / Architecture
DLD: Detailed Design (UML)
Code: Coding (no detailed design)
Test: Testing

Effort in Percent over Cycles – 5



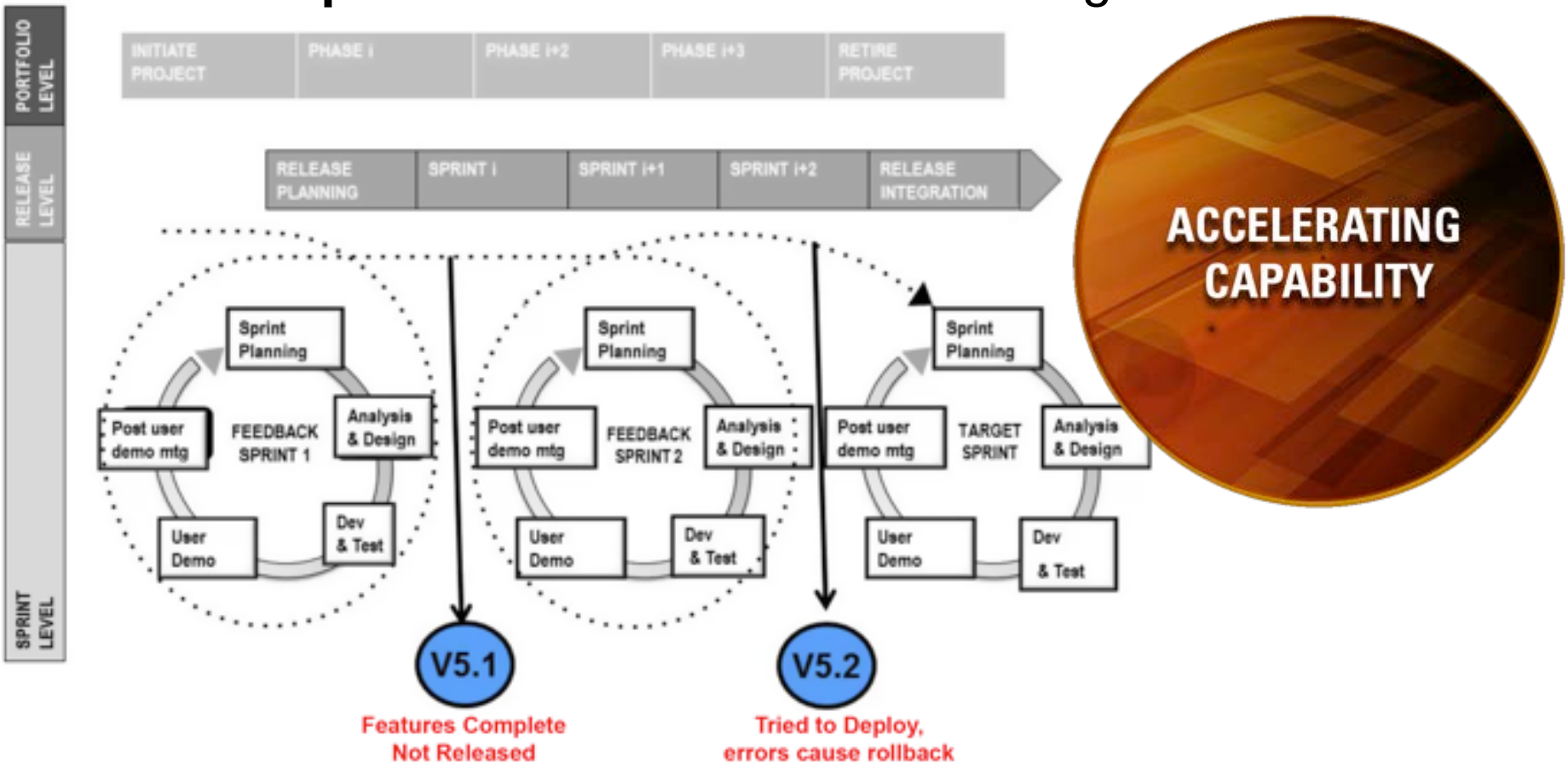
Reqts: Requirements
HLD/Arch: High level Design / Architecture
DLD: Detailed Design (UML)
Code: Coding (no detailed design)
Test: Testing

Results

Results	Target	Actual
Latency	1ms	0.1ms
Throughput (transactions per second)	1,000	200,000
Schedule (months)	18	17
Quality (defects/KLOC found during validation testing)	0.25	0.1

Deployment Challenges

The **DevOps** movement continues what Agile started.



DevOps: State of the Practice

Focus is on

- Culture and teaming
- Process and practices
- Value Stream Mapping
- Continuous Delivery practices
- Lean Thinking
- Tooling, automation and measurement
  - Tooling to automate repetitive tasks
  - Static analysis automation for monitoring architectural health
  - Performance dashboards

DevOps and Architecture

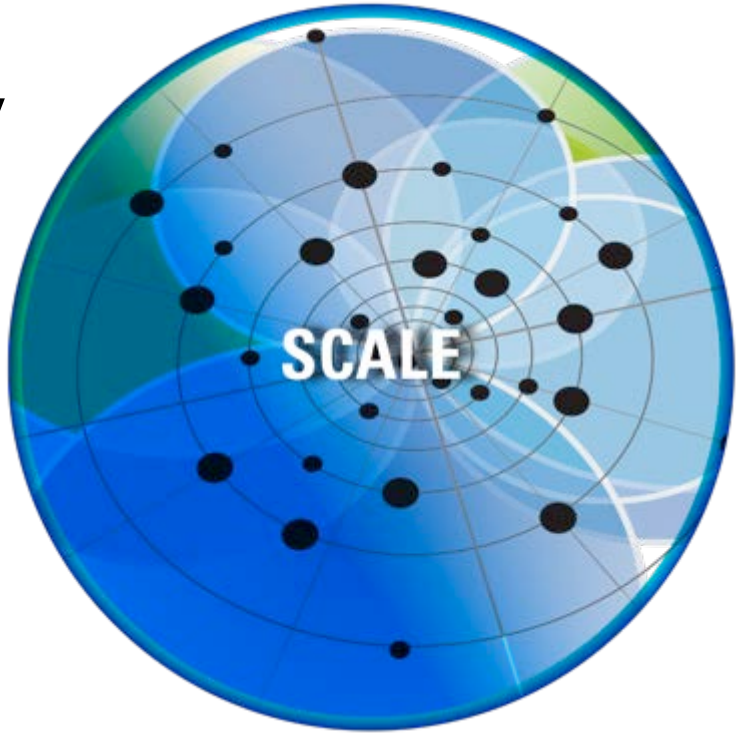
Design decisions that involve deployment-related limitations can blindside teams.

DevOps Tips

- Don’t let designing for deployability be an afterthought
- Establish monitoring mechanisms
- Leverage measurable deployability quality attribute
- Align design with concrete requirements and response measures
- Use design abstractions to reason about implications of design options and trade-offs
- Consider design tactics that promote modifiability, testability, and operational resilience

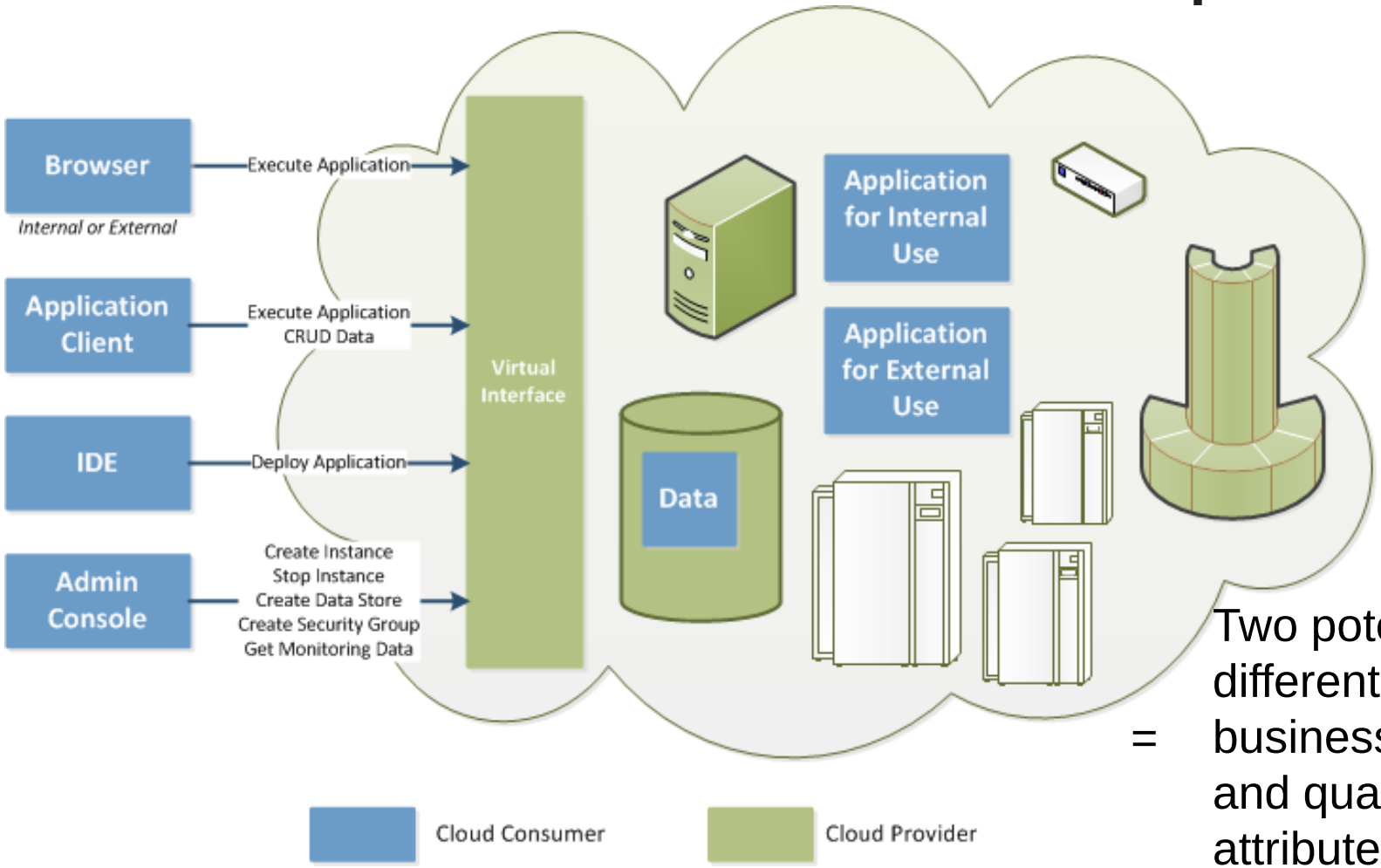
Scale and Architecture

- Cloud strategies
- Cloud strategies for mobility
- Big data



“Scale Changes Everything”

Two Perspectives of Software Architecture in Cloud Computing



= Two potentially different sets of business goals and quality attributes

Cloud Computing and Architecting

- SLAs cannot prevent failures.
- In cloud environments
  - Cloud consumers have to design and architect systems to account for lack of full control over important quality attributes
  - Cloud providers have to design and architect infrastructures and systems that provide the most efficient way to manage resources and keep promises made in SLAs

Mobile Device Trends

Architecture Trends: Cyber-Foraging

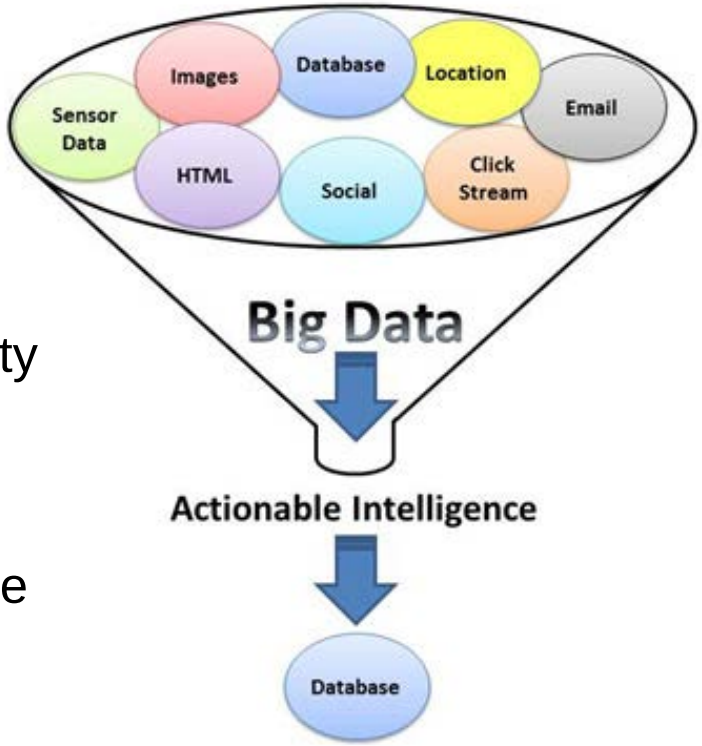
- Edge Computing
- Using external resource-rich surrogates to augment the capabilities of resource-limited devices
  - Code/Computation Offload
  - Data Staging
- Industry is starting to build on this concept to improve mobile user experience and decrease network traffic
- Our research: cloudlet-based cyber-foraging – brings the cloud closer to the user

Nokia Siemens Networks
Liquid Applications

Cisco Systems
Fog Computing

Big Data Systems

- Comprise two very distinct but related technological thrusts
  - Data analytics
  - Infrastructure for storage and processing
- Analytics is typically a massive data reduction exercise – Data to Decisions
  - Input: high volume, low information density
  - Output: Low volume, high information density
- Computation infrastructure necessary to ensure the analytics are
  - Fast
  - Scalable
  - Secure
  - Easy to use

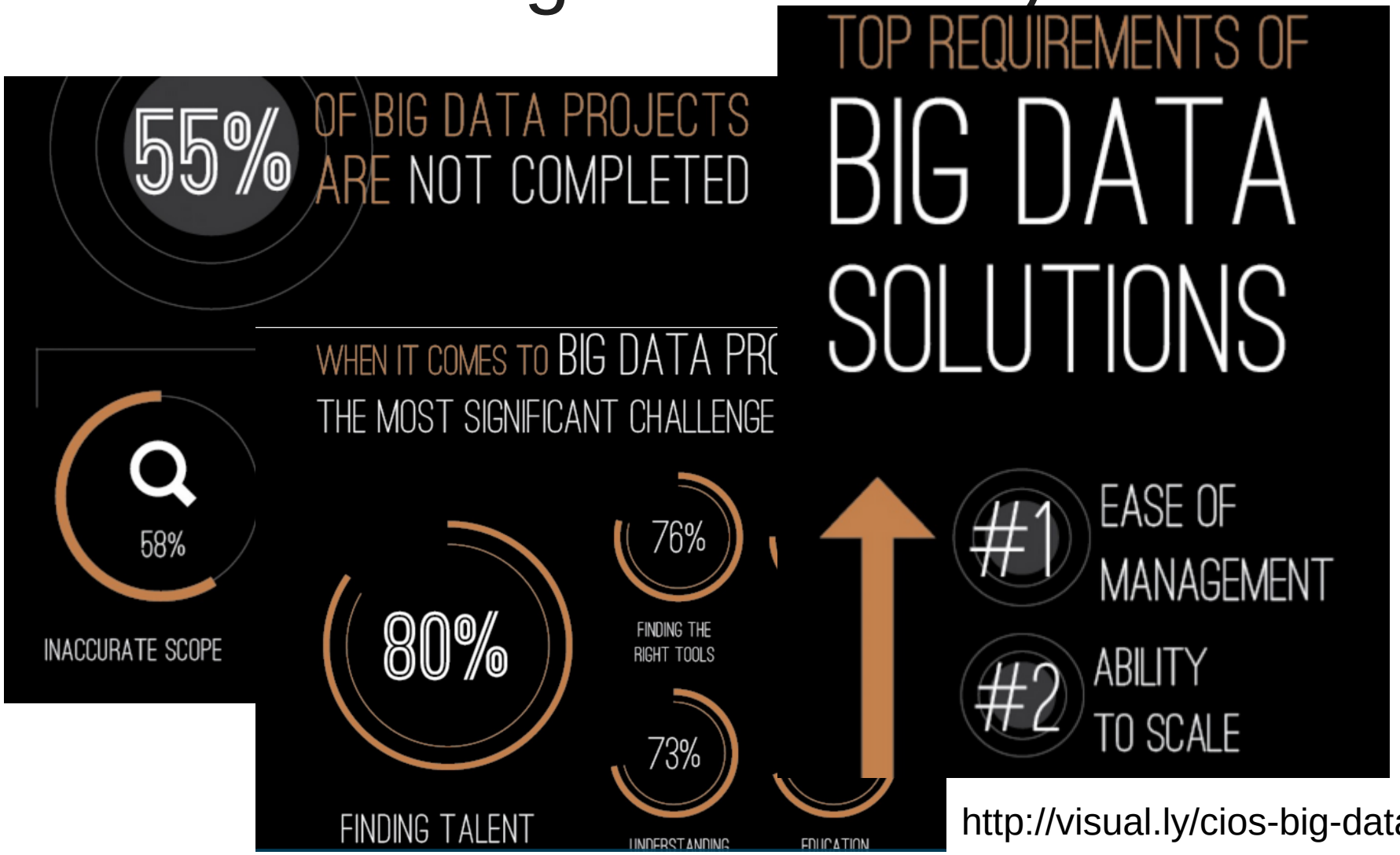


Big Data – State of the practice
"The problem is not solved"

Building scalable, assured big data systems is hard

Building scalable, assured big data systems is expensive

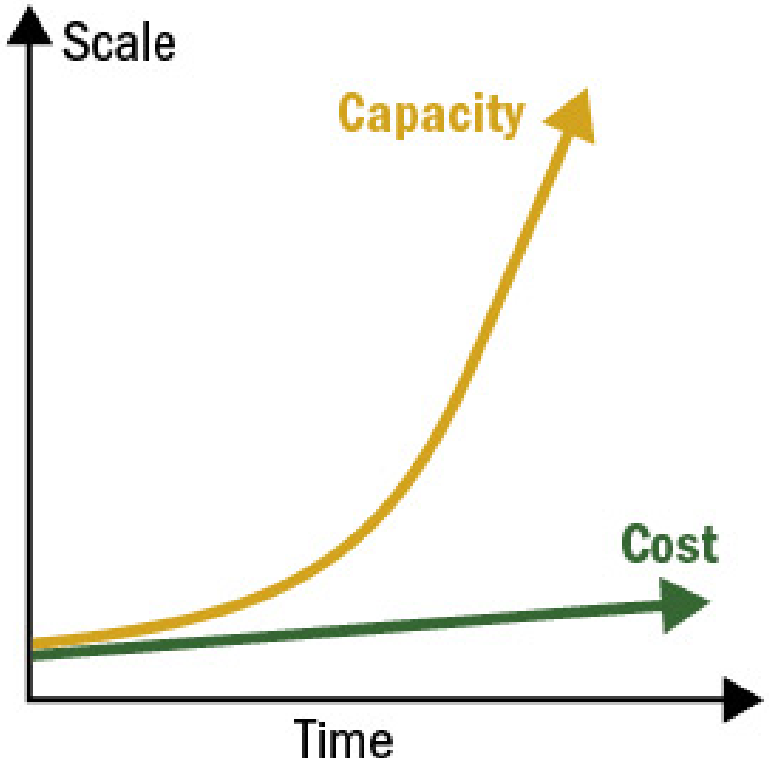
Big Data Survey



http://visual.ly/cios-big-data

Architecture and Big Data

- System costs must grow more slowly than system capacity
- Approaches
  - Scalable software architectures
  - Scalable software technologies
  - Scalable execution platforms
- Scalability reduces as implementation complexity grows
- NoSQL Models are not created equal



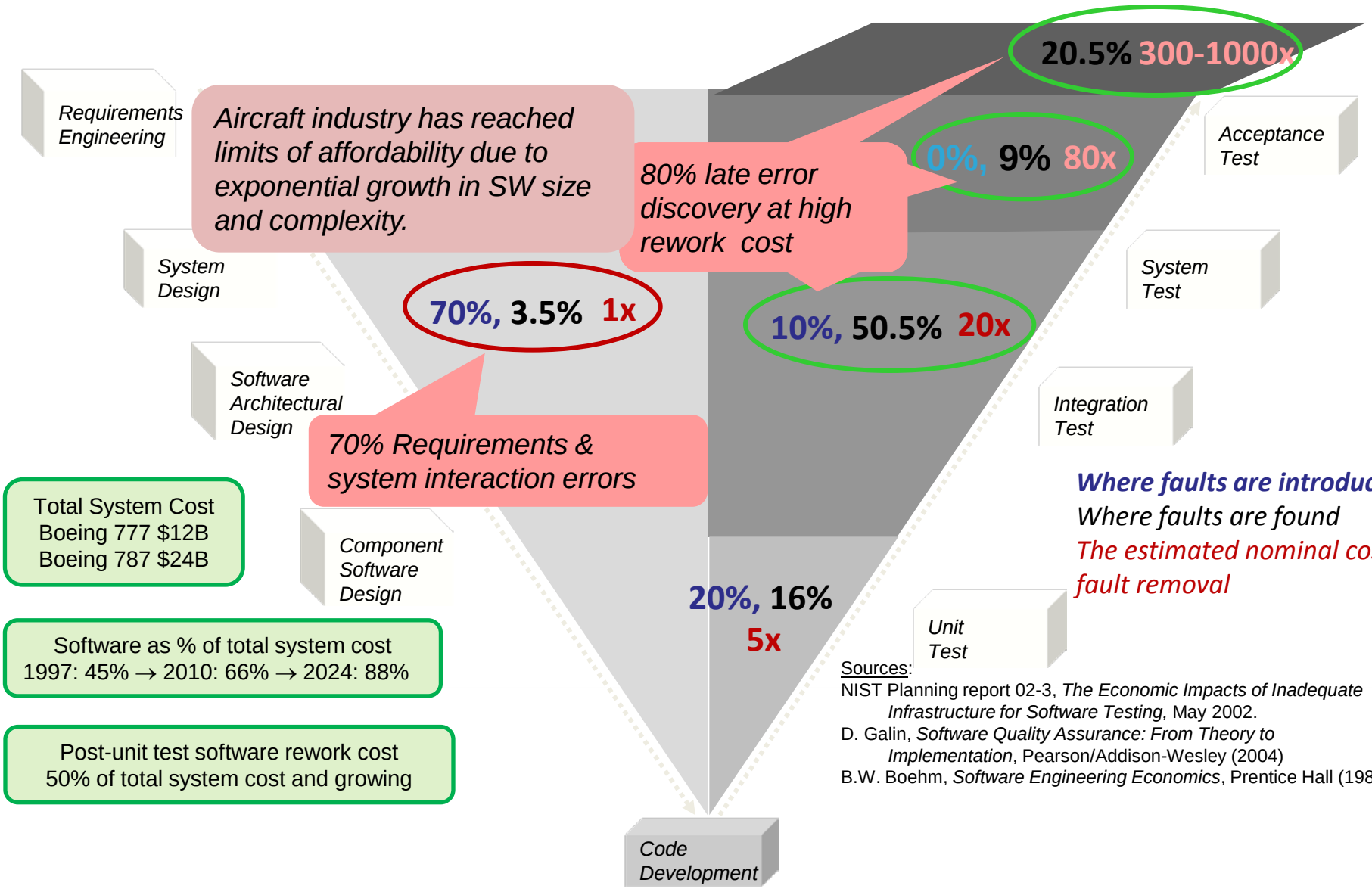
Our Current Research

- Lightweight Evaluation and Architecture Prototyping for Big Data (LEAP4BD)
- QuABase: A Knowledge Base for Big Data System Design
  - Semantics-based knowledge model
    - General model of software architecture knowledge
    - Populated with specific big data architecture knowledge
  - Dynamic, generated, and queryable content
  - Knowledge Visualization

Software Assurance and Architecture

In safety critical systems more is needed.

High Fault Leakage Drives Major Increase in Rework Cost



Total System Cost
Boeing 777 $12B
Boeing 787 $24B

Software as % of total system cost
1997: 45% → 2010: 66% → 2024: 88%

Post-unit test software rework cost
50% of total system cost and growing

Where faults are introduced
Where faults are found
The estimated nominal cost for fault removal

Sources:
NIST Planning report 02-3, *The Economic Impacts of Inadequate Infrastructure for Software Testing,* May 2002.
D. Galin, *Software Quality Assurance: From Theory to Implementation*, Pearson/Addison-Wesley (2004)
B.W. Boehm, *Software Engineering Economics*, Prentice Hall (1981)

Architectural Models

- capture architecture in a form amenable to analysis
- range from informal (e.g., visio diagrams) to formal (e.g., with precisely defined execution semantics)

SAE Architecture Analysis & Design Language (AADL) Standard Suite (AS-5506 series)

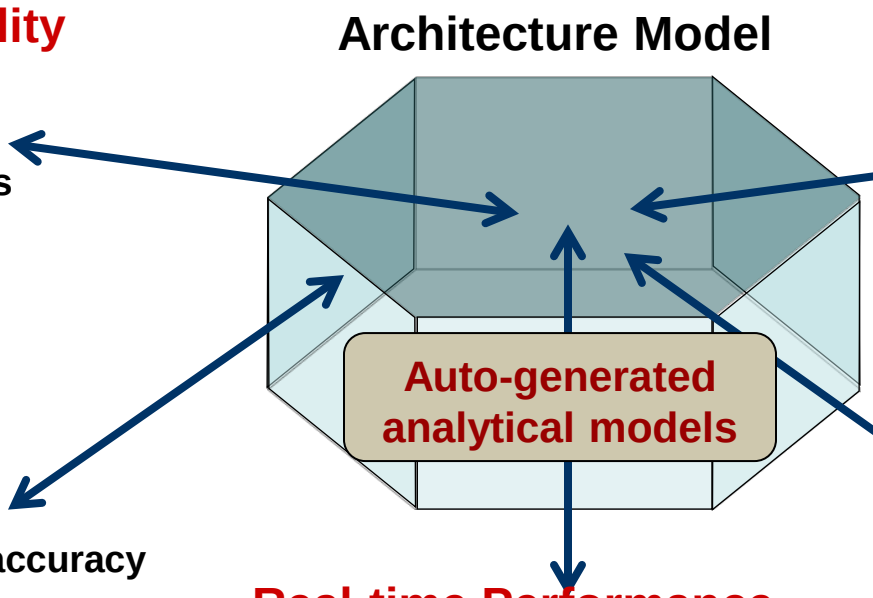
- Core AADL language standard (V2.1-Sep 2012, V1-Nov 2004)
  - Strongly typed language with well-defined semantics
  - Textual and graphical notation
  - Standardized XMI interchange format

Standardized AADL Extensions

- Error Model language for safety, reliability, security analysis
- ARINC653 extension for partitioned architectures
- Behavior Specification Language for modes and interaction behavior
- Data Modeling extension for interfacing with data models (UML, ASN.1, …)

Architecture-Centric Quality Attribute Analyses

Single Annotated Architecture Model Addresses Impact Across Operational Quality Attributes



Safety Reliability
- MTBF
- FMEA
- Hazard Analysis

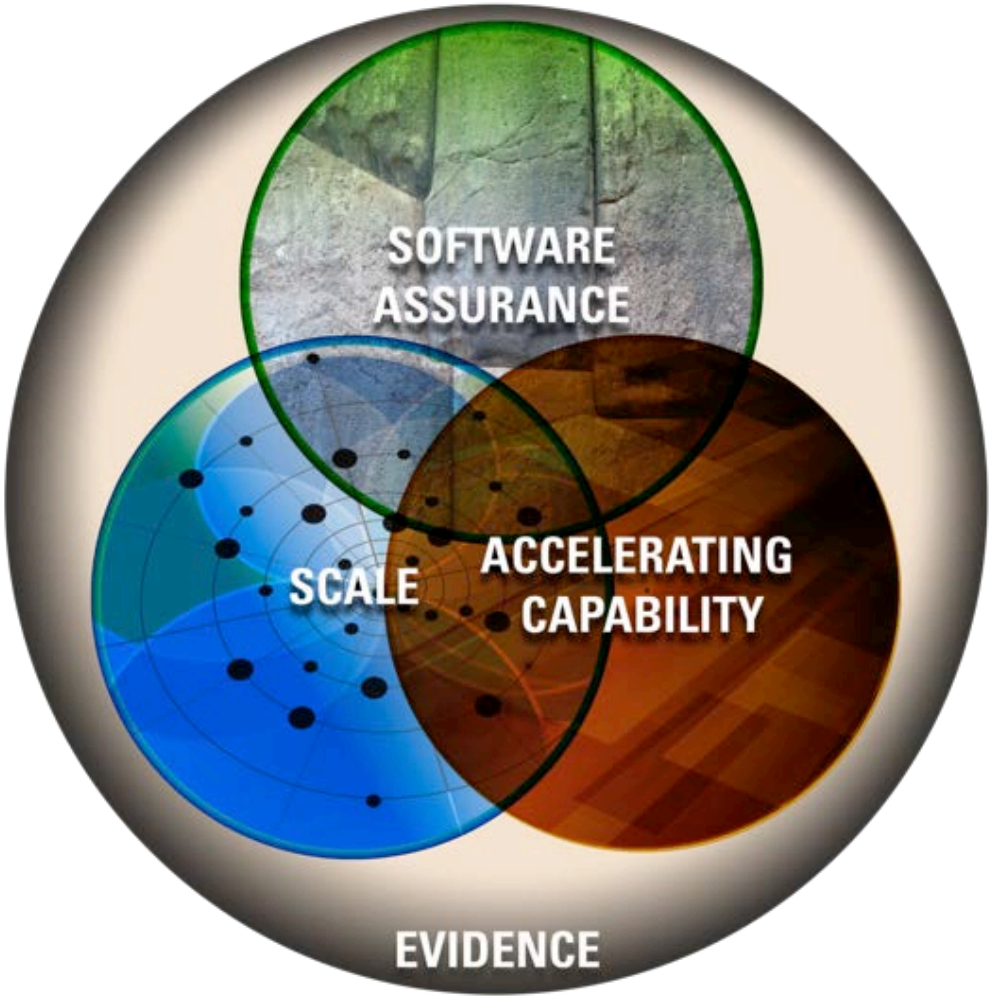
Security
- Intrusion
- Integrity
- Confidentiality

Data Quality
- Data precision/accuracy
- Temporal correctness
- Confidence

Real-time Performance
- Execution time/deadline
- Deadlock/starvation
- Latency

Resource Consumption
- Bandwidth
- CPU time
- Power consumption

Conclusion



- Foundational software architecture principles persist.
- Change brings new challenges.
- Software architecture practices and research are key to meeting new challenges.
- Much remains to be done.

This is the Work of Many

At the SEI

- Felix Bachmann
- Stephany Bellomo
- Peter Feiler
- Ian Gorton
- James Ivers
- Rick Kazman
- John Klein
- Mark Klein
- Grace Lewis
- Ipek Ozkaya
- Rod Nord
- And many more…

Thanks, Grace!

https://www.flickr.com/photos/expertinfantry/

Contact Information

Linda Northrop
SEI Fellow
Chief Scientist
Software Solutions Division
Telephone: 412-268-7638
Email: lmn@sei.cmu.edu
Website: http://www.sei.cmu.edu/architecture

U.S. Mail:
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213-3890

SEI Fax: 412-268-5758

More Information

http://blog.sei.cmu.edu/

Got Feedback?

★ Rate and Review the session using the GHC Mobile App

To download visit www.gracehopper.org