Complexity and Software: How to Meet the Challenge

NDIA CMMI Technology Conference

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

Paul Nielsen
November 15, 2011
Overview

Complexity
Software
How to Meet the Challenge
Summary
Complex Systems

• Complex systems are already woven tightly into our everyday lives
• 21st century systems will be increasingly complex
• How failing to understand them, their risks, and their management challenges poses a 21st-century hazard
The Rise of Complexity

- Scale
- Interconnectedness
- Autonomy
- Time criticality
- Security
- Safety
- Regulation
Complexity Brings Rewards – and Perils

Bad things happen when we …

• Misdesign
• Mismanage

… complex systems

Average person does not know or care about root cause of problem in system complexity
Examples: Good … and Bad
An Interconnected Society

The Internet, 1969

THE ARPA NETWORK
DEC 1969
Y Wrks

The Internet, Today
Autonomous Systems
Human System Interaction
Complex Systems at the SEI

The SEI is at the nexus of systems and complexity:

- We study them side-by-side
- For 25 years, we’ve been helping engineers design and manage software systems
- It’s our job to “ring the bell” on the importance of managing complexity

We also appreciate risk and the importance of managing it

- Process improvement
- Risk management and resilience
- Architectural approaches
- Multi-view models
Software is Everywhere
Software is Important

Manufacturing

Finance

Space

Engineering
Software is Increasingly Complex
Software Connects Us
Software is Becoming More Personal

A Super Smart Grid
“Software is Eating the World”

Marc Andreessen essay, August 20, 2011

• Software is eating the world … it is everywhere
• “A trend I've observed, one that makes me optimistic about the future growth of the American and world economies … “
• “More and more major businesses and industries are being run on software and delivered as online services—from movies to agriculture to national defense.”

[Image of Marc Andreessen and a newspaper article]
How to Handle Complexity

Models, Process and Process Improvement
Architecture
Risk Management and Resiliency
Evolution and Disruption
People
Process Improvement, People, and Models

Process Improvement helps us standardize, examine and improve how we work

- Process improvement also lets us learn from others and ourselves

Models help us simplify complex systems and give us insightful views

Process improvement and models are tools for people, not their masters

- People design, develop and operate systems
Why Use CMMI?

Makes you more competitive
Improves your quality
Links you with a global market
Scales to your needs
Helps you prioritize
The Key Ideas of CMMI

Elements:
- know your customer
- know your company
- focus on quality
- commit to continuous improvement
- make decisions based on data

Benefits:
- deliver quality products
- fewer delivered defects
- less rework
- meet schedules
- reduce costs profits
- quicker, more agile cycles
- more business
TSP – Software Engineering Best Practice

Scalable…small to large projects
Supports broad portfolio of work
Best standard practice

Best development practices by size of application\(^1\)

## U.S. SOFTWARE PERFORMANCE LEVELS

<table>
<thead>
<tr>
<th>PROJECT MANAGEMENT</th>
<th>TECHNICAL STAFFS</th>
<th>SOFTWARE USERS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sizing</td>
<td>Fair</td>
<td>Requirements</td>
</tr>
<tr>
<td>Estimating</td>
<td>Poor</td>
<td>Design</td>
</tr>
<tr>
<td>Planning</td>
<td>Fair</td>
<td>Coding</td>
</tr>
<tr>
<td>Tracking</td>
<td>Poor</td>
<td>Reviews</td>
</tr>
<tr>
<td>Measuring</td>
<td>Poor</td>
<td>Testing</td>
</tr>
<tr>
<td>Overall</td>
<td>Poor</td>
<td>Good</td>
</tr>
</tbody>
</table>

**Conclusion:** U. S. technical skills are better than U. S. management skills. Project management and quality are frequent problem areas.

From *Team Software Process (TSP) in Context*, by Capers Jones
Accelerated Improvement Method (AIM)

Integrates and Leverages SEI Technologies
AIM – Predictably Improves Software Engineering Practice

Summary of field trial results

- rapid deployment – 15 months
- tactical project-focused approach
- supports implementation in-progress on critical software engineering projects
- predictable improvement timeline and cost
- faster, higher ROI addresses “no-time” and “can’t afford” barriers to change
- same TSP performance benefits

<table>
<thead>
<tr>
<th>AIM Results</th>
<th>CGI Results</th>
<th>Typical Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMMI-DEV ML1 to ML3 (months)</td>
<td>15</td>
<td>36 to 48</td>
</tr>
<tr>
<td>Schedule Variance</td>
<td>&lt;10%</td>
<td>25% to 100%</td>
</tr>
<tr>
<td>Effort Variance</td>
<td>7%</td>
<td>25% to 100%</td>
</tr>
<tr>
<td>Productivity</td>
<td>up 35%</td>
<td></td>
</tr>
<tr>
<td>Quality (defects)</td>
<td>down 50%</td>
<td></td>
</tr>
</tbody>
</table>
Architecture is Important

The quality and longevity of a software-reliant system is largely determined by its architecture.

In recent studies by OSD, the National Research Council, NASA, and the NDIA, architectural issues are identified as a systemic cause of software problems in DoD systems.
Why is Software Architecture Important?

- Represents *earliest* design decisions:
  - hardest to change
  - most critical to get right
  - communication vehicle among stakeholders

- *First* design artifact addressing:
  - performance
  - modifiability
  - reliability
  - security

- Key to systematic *reuse*:
  - transferable, reusable abstraction

- Key to system *evolution*:
  - manage future uncertainty
  - assure cost-effective agility

The right architecture paves the way for system success. The wrong architecture usually spells some form of disaster.
Trends Leading to Architecture Challenges

Scale and complexity

Increased operational tempo

Decentralization and distribution

Disruptive technologies
Architecture Challenges Extend Across a Spectrum of System Types and Scale

Challenges include:

- determining how to structure and adapt systems at all scales
- managing interactions among these types of systems
- assuring software-reliant capabilities that are sufficiently reliable, secure, responsive, and adaptable to change

**Predict and control behavior**

- Stand-alone Systems
  - software applications

- Embedded Systems
  - software embedded in hardware devices

- Cyber-Physical Systems
  - mutually dependent computational systems and physical processes

- Software Product Lines
  - families of similar systems

- Systems of Systems
  - federations of independent systems

- Ultra-Large-Scale Systems
  - webs of software-reliant systems, people, economies, and cultures

**Assure and bound behavior**

Coupling to organizational structure and practices increases
Quality Attribute Requirements

Quality attributes include

• performance
• availability
• interoperability
• modifiability
• evolvability
• usability
• security
• etc.

Quality attribute requirements stem from business and mission goals. Key quality attributes need to be characterized in a system-specific way. Otherwise, they are not operational.
Key Principles of Resiliency

Resilience is the ability to provide and maintain an acceptable level of service in the face of faults and challenges to normal operation.

- security “built in”
- failure scenarios understood, planned for
- redundancy is provided for in key areas
- capability remains available under adverse conditions

At SEI, both organizational and software:

- Resilience Maturity Model (RMM)
- Security Quality Requirements Engineering (SQUARE)
- Current blog series topic (http://blog.sei.cmu.edu/)
Continuity

A key aim of resiliency (and managing operational risk)

Business Functions:

• Developing and executing continuity plans, recovery plans, and restoration plans

IT Function:

• Developing, implementing, and managing processes to deliver IT services and manage IT infrastructures
Resiliency Maturity Model

What is CERT-RMM?

CERT-RMM is a maturity model for managing and improving operational resilience.

- Guides implementation and management of operational resilience activities
- Converges key operational risk management activities: security, business continuity/disaster recovery, and IT operations
- Defines maturity through capability levels (like CMMI)
- Improves confidence in how an organization responds in times of operational stress
Key Ideas in Software Engineering -1

Critical Code—Software Practice and Research

1. There is a rapid growth in the strategic significance of software for DoD
   - DoD needs to actively address its software producibility needs
   - DoD cannot rely on industry alone to address software challenges for defense

2. Iterative engineering of innovative software can be successfully managed
   - Apply advanced technologies and practice for iterative incremental development of software intensive systems
   - Update earned value models and practices to support management process

3. There is insufficient DoD-aligned software experience
   - DoD needs to be a smart software customer
Key Ideas in Software Engineering -2

4. Assert DoD architectural leadership
   - In highly complex systems, architecture decisions may need to dominate functional capability choices

5. Adopt a strategic approach to software assurance
   - Integrate preventive practices into development to support ongoing creation of evidence in support of assurance

6. It is essential to reinvigorate DoD software engineering research
   - NITRD data reveal the extent of the S&T disengagement
   - Apply appropriate criteria in identifying goals for research programs
   - Focus research effort on identified goals in seven technological areas
Recommendations -1

Summary of recommendations
Improve critical areas of current practice

– Process and measurement
  • Enable incremental iterative development at arm's length

– Architecture
  • Enable architecture leadership, interlinking, flexibility

– Assurance and security
  • Enable high assurance at scale with rich supply chains
Recommendations -2

Undertake research to support the critical areas of practice

1. Architecture modeling and architectural analysis
2. Validation, verification, and analysis of design and code
3. Process support and economic models for assurance
4. Requirements
5. Language, modeling, code and tools
6. Cyber-physical systems
7. Human-system interaction
People
Summary

Software has increasing impact
Software challenges are growing
Software engineering provides useful tools
Great people are crucial
# Contact Information Slide Format

**Presenter / Point of Contact**
- **Title**
- **Program or Initiative**
- **Telephone:** +1 412-268-5800
- **Email:** info@sei.cmu.edu

**Web**
- www.sei.cmu.edu
- www.sei.cmu.edu/contact.cfm

**U.S. Mail**
- Software Engineering Institute
- Customer Relations
- 4500 Fifth Avenue
- Pittsburgh, PA 15213-2612
- USA

**Customer Relations**
- **Email:** info@sei.cmu.edu
- **Telephone:** +1 412-268-5800
- **SEI Phone:** +1 412-268-5800
- **SEI Fax:** +1 412-268-6257
NO WARRANTY

THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

Use of any trademarks in this presentation is not intended in any way to infringe on the rights of the trademark holder.

This Presentation may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.

This work was created in the performance of Federal Government Contract Number FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. The Government of the United States has a royalty-free government-purpose license to use, duplicate, or disclose the work, in whole or in part and in any manner, and to have or permit others to do so, for government purposes pursuant to the copyright license under the clause at 252.227-7013.