SERC WRT-1016: Reducing Total Cost of Ownership and Schedule

Sponsor: OUSD(R&E)

Dr. Barry Boehm, USC

12th Annual SERC Sponsor Research Review
HOSTED VIRTUALLY ON: November 18, 2020
www.sercuarc.org
• The long-term transition goal for the project is to enable complex system projects to reduce their total ownership costs by using better methods, processes, and tools for satisfying their non-functional (quality) requirements (NFRs). Compared with functional requirements, the NFRs are more difficult to specify and are given lower priority during system definition than the FRs. They are often in conflict with each other. As one example on a major DoD project, the Security team proposed to minimize their vulnerability profile by having a single copy of the database and a single-agent key distribution system, only to have the Reliability team point out that they would be creating two serious points of failure.
Among the potential tools and guides resulting from the research are major extensions of the tools developed in the previous SERC System Qualities Ontology, Tradespace, and Affordability (SQOTA) project. For example, the System Qualities Understanding by Analysis of Abundant Data (SQUAAD) tool for large-scale life-cycle analysis of software technical debt was successfully used on a Navy application, and scaled up to analyze the technical debt histories of 1.3 billion lines of code across three large companies and 15 years.
We are also developing an extension of the COCOMO cost model for estimating the cost of security-critical software. Its current definition was presented at our recent annual COCOMO Forum, at which attendees from the Navy, NASA-JPL, Aerospace Corp., the SEI, and several aerospace companies were interested in participating in its definition and calibration. USC-CSSE and the CAST organization are co-sponsors of the Consortium for Information System Quality, which has strong support from DHS, DoD, and other federal organizations, and we participate in CISQ events involving information system security cost estimation.
• We have used our Systems and Software Qualities Ontology as a basis for reorganizing the Qualities section of the Specialty Engineering Part 6 of the Systems Engineering Body of Knowledge.

• We are also working on an update of our System Qualities Synergies and Conflicts matrix for industry and government organizations to evaluate side effects of system qualities changes.
Large Scale Mission Software Data Exploitation: SQUAAD

Dr. Pooyan Behnamghader  
Dr. Barry Boehm

Center for Systems and Software Engineering  
University of Southern California

Aerospace Ground Systems Architecture Workshop 2020
Introduction
  ○ Applying large scale data analytics to Ground System software.

Mining a Software Repository.
  ○ Importance of Commit Level Software Evolution Analysis.

Software Quality Metrics and Their Inference
  ○ Technical Debt, Size, Code Quality, and Security

Commit-Impact Analysis
  ○ Targeting a Software Module and Understanding the Impact of Every Developer.

Software Quality Understanding by Analysis of Abundant Data (SQUAAD)
  ○ Open Source Case Study.
  ○ Private Cloud for DoD Affiliated Organizations.

Demo, Discussions and Conclusions
Software defects costed $1.1 trillion of TD

Each Java LOC has $5.42 of TD

Software defects affected 4.4 billion people

75% of systems lifecycle cost is spent on software maintenance

Slide Credit: Reem Alfayez at al., USC CSSE, COCOMO Forum 2018
A Large Scale Analysis

Experiment's Scale

<table>
<thead>
<tr>
<th>Org.</th>
<th>Timespan</th>
<th>Sys.</th>
<th>Impactful</th>
<th>LOC (MS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apache</td>
<td>01/02-02/18</td>
<td>38</td>
<td>22627</td>
<td>734</td>
</tr>
<tr>
<td>Google</td>
<td>08/08-01/18</td>
<td>18</td>
<td>11527</td>
<td>760</td>
</tr>
<tr>
<td>Netflix</td>
<td>05/11-01/18</td>
<td>12</td>
<td>3684</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>01/02-02/18</td>
<td>68</td>
<td>37838</td>
<td>1531</td>
</tr>
</tbody>
</table>
## Metrics

### Multi-Perspective Analysis of Software Quality Evolution

<table>
<thead>
<tr>
<th>3 Groups</th>
<th>9 Metrics</th>
<th>Source/Bytecode Analysis</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Group</strong></td>
<td><strong>Abbr.</strong></td>
<td><strong>Tool</strong></td>
<td><strong>Description</strong></td>
</tr>
<tr>
<td>Basic</td>
<td>LC</td>
<td>SonarQube</td>
<td>Physical Lines excl. Whitespaces/Comments</td>
</tr>
<tr>
<td></td>
<td>FN</td>
<td>SonarQube</td>
<td>Functions</td>
</tr>
<tr>
<td></td>
<td>CS</td>
<td>FindBugs</td>
<td>Classes</td>
</tr>
<tr>
<td>Code Quality</td>
<td>CX</td>
<td>SonarQube</td>
<td>Complexity (Number of Paths)</td>
</tr>
<tr>
<td></td>
<td>SM</td>
<td>SonarQube</td>
<td>Code Smells</td>
</tr>
<tr>
<td>Security</td>
<td>VL</td>
<td>SonarQube</td>
<td>Vulnerabilities</td>
</tr>
<tr>
<td></td>
<td>SG</td>
<td>PMD</td>
<td>Security Guidelines</td>
</tr>
<tr>
<td></td>
<td>FG</td>
<td>FindBugs</td>
<td>Malicious Code, Security</td>
</tr>
</tbody>
</table>
Valuable Dataset

Commit History Over a Period of 9 Years
➢ An automated cloud-based infrastructure to
  o Retrieve a subject system’s information from various sources (e.g., commit history and issue repository).
  o Distribute hundreds of relevant revisions on multiple cloud instances, efficiently compile each revision, and run static/dynamic programming analysis techniques on it.
  o Collect and interpret the artifacts generated by programming analysis techniques to extract quality attributes or calculate change.

➢ A set of statistical analysis techniques tailored for understanding software quality evolution.
  o Simple statistics, such as frequency of code smell introduction or correlation between two quality attributes.
  o Machine learning techniques, such as clustering developers based on their impact.
Conclusions

- A Novel Mining Software Repositories Approach
- A High Maximum Compilation over Commit History
- A Better Understanding of Life Cycle Software Qualities
- A Large Scale Multi Perspective Analysis of Software Quality Evolution

Software Quality Understanding by Analysis of Abundant Data (SQUAAD)
CISQ Webinar

Consortium for Information and Software Quality

Costs of Secure Software Development: COSECMO 2021

Elaine Venson
Barry Boehm

Initially Presented: September 22, 2020
• Cost-effectiveness of secure software development

• Sources of cost in secure software development
  — Security practices
  — Security controls

• Security practices application survey

• Models for costing secure software development

• Open issues and opportunities

• Next steps
• Based on the US National Vulnerabilities DB (NVD) with more than 85K publicly reported vulnerabilities (2015)

• 2015

• Cost-effectiveness of secure software development

• Sources of cost in secure software development
  – Security practices
  – Security controls/features

• Security practices application survey

• Models for costing secure software development

• Open issues and opportunities

• Next steps
### Sources of Cost (from literature)

<table>
<thead>
<tr>
<th>Source</th>
<th>Papers</th>
<th>Source</th>
<th>Papers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perform Security Review</td>
<td>21</td>
<td>Perform Security Training</td>
<td>6</td>
</tr>
<tr>
<td>Apply Threat Modeling</td>
<td>18</td>
<td>Improve Development Process</td>
<td>5</td>
</tr>
<tr>
<td>Perform Security Testing</td>
<td>16</td>
<td>Perform Penetration Testing</td>
<td>5</td>
</tr>
<tr>
<td>Apply Security Requirements</td>
<td>11</td>
<td>Achieve Security Level</td>
<td>3</td>
</tr>
<tr>
<td>Apply Security Tooling</td>
<td>11</td>
<td>Document Technical Stack</td>
<td>3</td>
</tr>
<tr>
<td>Implement Countermeasures</td>
<td>9</td>
<td>Security Experts, Security Groups, Security</td>
<td>3</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Master</td>
<td></td>
</tr>
<tr>
<td>Fix Vulnerabilities</td>
<td>9</td>
<td>Track Vulnerabilities</td>
<td>3</td>
</tr>
<tr>
<td>Apply Secure Coding Standards</td>
<td>8</td>
<td>Functional Features</td>
<td>2</td>
</tr>
<tr>
<td>Apply Data Classifications Scheme</td>
<td>7</td>
<td>Hardening Procedures</td>
<td>2</td>
</tr>
<tr>
<td>Publish Operations Guide</td>
<td>7</td>
<td>Security by Design Paradigm</td>
<td>1</td>
</tr>
</tbody>
</table>

#### SWSec Practices

Developing secure software

Goals

Build-in security to preserve assets (CIA)

Requirements

Functional

Features, controls, components

Non-functional

Security practices (threat modeling, pen-testing)

Development

Measurement

Lines of code, functions points, objective points

Levels of application (scope and rigor)
- Participants of the Software Security group on LinkedIn
- 110 complete responses
- 29 countries

![Pie chart showing role/position distribution]

- Security expert: 37%
- Management (e.g., Area manager): 13%
- Software developer: 16%
- Project leader in the development: 16%
- Member of the security group: 16%
- Security tester: 12%
- Other: 5%
- Security expert: 1%
Which security practices does your organization apply during software development? (select all that apply)

- Secure Coding Standards
- Security Tooling
- Track Vulnerabilities
- Security Requirements
- Security Testing
- Development Process Improvement

- Document Technical Stack
- Security Review
- Threat Modeling
- Penetration Testing
- Security Training
- Data Classification Scheme
- Publish Operations Guide
Practices Usage

- Apply Secure Coding Standards
- Apply Security Tooling
- Track Vulnerabilities
- Apply Security Requirements
- Perform Security Testing
- Improve Development Process
- Document Technical Stack
- Perform Security Review
- Apply Threat Modeling
- Apply Data Classification Scheme
- Perform Penetration Testing
- Publish Operations Guide
- Perform Security Training

0% 10% 20% 30% 40% 50% 60% 70% 80% 90%

- Daily
- Weekly
- Monthly
By Development Type

- New development
- Enhancement
- Migration
- Re-development
- Other
“Getting people to truly stop, and understand 100% why the best practices are needed, can be a challenge - when people get focused on delivery dates. Once you explain the ‘What could happen...’ - it tends to sink in.”

“Always people considered security as feature to add after business logic and programming are finished so it happens to delay the project a lot.”

“Convincing project manager to incorporate security related time and effort.”
How is effort for software security estimated in your organization? (select one)

- Ad-hoc
- Expert opinion
- Analogy-based
- Model/parametric
- Other
- NA
• Cost-effectiveness of secure software development

• Sources of cost in secure software development
  – Security practices
  – Security controls

• Security practices application survey

• Models for costing secure software development

• Open issues and opportunities

• Next steps
## Approaches to Estimating Costs of SWSec

<table>
<thead>
<tr>
<th>Approach</th>
<th>Additional Cost</th>
<th>Productivity Range</th>
<th>Source</th>
<th>Validation</th>
</tr>
</thead>
<tbody>
<tr>
<td>COCOMO II security extension [Reifer 2003]</td>
<td>0.94 (Low) 1.02 (Nominal) 1.27 (High) 1.43 (Very High) 1.75 (Extra High)</td>
<td>1.86</td>
<td>Expert estimation</td>
<td>Not validated</td>
</tr>
<tr>
<td>COSECMO [Colbert 2008]</td>
<td>0% (Nominal) 20% to 80% (EAL 3 - High) 50 to 200% (EAL 4 - Very High) 125% to 500% (EAL 5 - Extra High) 313% to 1250% (EAL 6 - Super High) 781% to 3125% (EAL 7 - Ultra High)</td>
<td>31.25</td>
<td>Expert estimation with two inputs provided by a Commercial Company</td>
<td>Not validated</td>
</tr>
<tr>
<td>Weapon systems cost model (COCOMO II based) [Lee 2014]</td>
<td>1.0 (Low or Nominal) 1.87 (High)</td>
<td>1.87</td>
<td>Expert estimation and 73 data points</td>
<td>Cross validation</td>
</tr>
<tr>
<td>Secure OS software cost model (COCOMO II based) [Yang 2015]</td>
<td>1 (Nominal) 1.25 to 1.5 (High) 1.75 to 2.0 (Very High) 2.0 to 2.75 (Extra High) 3.0 to 3.75 (Super High)</td>
<td>3.75</td>
<td>Expert estimation</td>
<td>Case study</td>
</tr>
<tr>
<td>FPA security extension (GSC) [Abdullah 2010]</td>
<td>0 to 5% increase in the function points size of the project</td>
<td>1.05</td>
<td>Practices from survey with developers</td>
<td>Not validated</td>
</tr>
</tbody>
</table>
• Cost-effectiveness of secure software development

• Sources of cost in secure software development
  — Security practices
  — Security controls

• Security practices application survey

• Models for costing secure software development

• Open issues and opportunities

• Next steps
• Framework focused on product *certification*

• Used for security *benchmark* of IT products

• Certification is expensive and take time

• EALs are defined around the depth and rigor of design, tests and reviews of security features

• Not developed for secure software development in general

• Opportunity to develop a rating scale, based on security practices, that captures the current secure software development scenario
• No model has been properly validated with industry data

• COCOMO III initiative to collect data from industry

• Open source software repositories

• Involvement of the communities of security experts and estimation experts
• Cost-effectiveness of secure software development

• Sources of cost in secure software development
  — Security practices
  — Security controls

• Security practices application survey

• Models for costing secure software development

• Open issues and opportunities

• Next steps
Scale Development

Item Development
1. Identification of domain and item generation
2. Content validity

Scale Development
3. Scale points description
4. Item reduction
5. Pre-testing scale
6. Sampling and data collection

Scale Evaluation
7. Tests of reliability
Tests of validity

Modeling Methodology

1. Analyze Existing Literature
2. Perform Behavioral Analysis
3. Determine Form of the Model; Identify relative significance of parameters
4. Perform Expert-Judgement, Delphi Assessment
5. Gather Project Data
6. Determine Bayesian A Posteriori Update
7. Gather More Data, Refine Model
Proposed Model Form

\[ PM = A \cdot \text{Size}^E \cdot SECU \cdot \prod_{i=1}^{n} EM_i \]

SECU: Effort multiplier for secure software development level

\[ \text{Size} = \text{Functional Size} + \text{Security Features Size} \]

OR

\[ \text{Size} = \text{Functional Size} \cdot SSF \]

SSF: Security Size Factor for security level

Build-in security to preserve assets (CIA)

Functional
- Features, controls, components

Non-functional
- Security practices (threat modeling, pen-testing)
- Levels of application (scope and rigor)

Lines of code, functions points, objective points
Data Collection

Security experts estimates for the security parameter

Estimation experts estimates for the security parameter

Wideband Delphi

Industry

Projects’ Data ➔ Manual Data Collection Form

OSS

Projects’ Data ➔ Automated Data Collection

Projects’ Data ➔ Survey


