SQOTA: 8-University Project; 3 Primary Activities

• SQ Ontology and Tradespace
  – Means-Ends Hierarchy and Tradespace Relations: USC
  – Changeability Semantic Ontology: MIT
  – Formal Representation and Analysis: U. Virginia

• SQ Models, Methods, Processes, and Tools (MMPT’s)
  – Model-Driven, ISR Drones Coordination: AFIT, NPS; USC
  – MOSA and Set-Based Design: Wayne State, Penn State
  – Vehicle Tradespace Tool Framework and Family: GTRI
  – Maintainability and Technical Debt Toolset: USC

• Next-Generation Cost Estimation Models
  – COCOMO III, COSYSMO 3.0: USC, NPS
  – System and Cost Model Integration: NPS, GTRI, USC
Why Emphasize Software Maintainability?

- General software trends and maintainability implications
- Key role of Maintainability among software qualities
- Growing risks for software-intensive systems
- Growing software technical debt

• Tools for improving maintainability
  - Maintainability, technical debt assessment tools
  - Software Maintainability Readiness Framework (SMRF)
    • Non-technical sources of technical debt

• Conclusions
Problem and Opportunity (%O&M costs)

- **US Government IT**: ~75%; $59 Billion [GAO 2015]

- **Hardware [Redman 2008]**
  - 12% -- Missiles (average)
  - 60% -- Ships (average)
  - 78% -- Aircraft (F-16)
  - 84% -- Ground vehicles (Bradley)

- **Software [Koskinen 2010]**
  - 75-90% -- Business, Command-Control
  - 50-80% -- Complex platforms as above
  - 10-30% -- Simple embedded software
Some Surprise-Free Software Trends with Maintainability Implications

• More, larger, more complex software systems
  – Internets of things, self-driving vehicles, ...
• Increasing speed of change
• Increasing need for software dependability
  – Particularly cyber security
• Increasing software autonomy
  – Principle of Human Primacy in microseconds?
• Increasing data capture, data analytics
• Increasing legacy software, evolution challenges
  – Mounting technical debt
Systems and Software Ontology: Dependability, Changeability, and Resilience

Reliability
- Defect Freedom
- Survivability
- Fault Tolerance

Complete
- Robustness
- Self-Repairability

Partial
- Graceful Degradation

Repairability
- Choices of Security, Safety

Maintainability
- Testability

Test Plans, Coverage
- Test Scenarios, Data
- Test Drivers, Oracles

Test Software Qualities

Changeability

Resilience

Modifiability

Dependability, Availability

Testability, Diagnosability, etc.
Investing in Reliability vs. Maintainability

• Baseline: System with 10,000 hours MTBF, 4 days MTTR
  – Availability = 10,000 / (10,000 + 96) = 0.9905

• A. Higher Reliability: 100,000 hour Mean Time Between Failures
  – 4 days Mean Time to Repair

• B. Higher Maintainability: 10,000 hour MTBF
  – 4 hours Mean Time to Repair

• Compare on Availability = MTBF / (MTBF + MTTR)

• A. Availability = 100,000 / (100,000 + 96) = 0.9990

• B. Availability = 10,000 / (10,000 + 4) = 0.9996
Average Change Processing Time: Two Complex Systems of Systems

Average workdays to process changes

Incompatible with turning within adversary’s OODA loop

11-8-2017
What is Technical Debt (TD)?

• TD: Delayed technical work or rework that is incurred when short-cuts are taken or short-term needs are addressed first
  – The later you pay for it, the more it costs (interest on debt)

• Global Information Technology Technical Debt [Gartner 2010]
  – 2010: Over $500 Billion; By 2015: Over $1 Trillion

• TD as Investment
  – Competing for first-to-market
  – Risk assessment: Build-upon prototype of key elements
  – Rapid fielding of defenses from terrorist threats

• TD as Lack of Foresight
  – Overfocus on Development vs. Life Cycle
  – Skimping on Systems Engineering
  – Hyper-Agile Development: Easiest-First increments
    • Neglecting Rainy-Day Use Cases, Non-Functional Requirements
Outline

• Why Emphasize Software Maintainability?
  – General software trends and maintainability implications
  – Key role of Maintainability among software qualities
  – Growing risks for software-intensive systems
  – Growing software technical debt

→ Tools for improving Maintainability
  – Maintainability, technical debt assessment tools
  – Software Maintainability Readiness Framework (SMRF)
    • Non-technical sources of technical debt

• Conclusions
<table>
<thead>
<tr>
<th>Factor</th>
<th>Very Low</th>
<th>Low</th>
<th>Nominal</th>
<th>High</th>
<th>Very High</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Structure</strong></td>
<td>Very low cohesion, high coupling, spaghetti code.</td>
<td>Moderately low cohesion, high coupling.</td>
<td>Reasonably well structured; some weak areas.</td>
<td>High cohesion, low coupling.</td>
<td>Strong modularity, information hiding in data/control structures.</td>
</tr>
<tr>
<td><strong>Application Clarity</strong></td>
<td>No Match between program and application.</td>
<td>Some correlation between program and application.</td>
<td>Moderate correlation between program and application.</td>
<td>Good correlation between program and application.</td>
<td>Clear match between program and application worldviews.</td>
</tr>
<tr>
<td><strong>Self-Descriptiveness</strong></td>
<td>Obscure code; documentation missing, obscure or obsolete.</td>
<td>Some code commentary and headers; some useful documentation.</td>
<td>Moderate level of code commentary, headers, documentation.</td>
<td>Good code commentary and headers; useful documentation; some weak areas.</td>
<td>Self-descriptive code; documentation up-to-date, well organized, with design rationale.</td>
</tr>
</tbody>
</table>

**Effort Penalty**
- 50%  
- 40%  
- 30%  
- 20%  
- 10%
Controlled Experiment Setup

- **Participants**: six graduate students
  - Students were recruited and selected from 38 applicants through two rounds of interviews.
  - They were not paid through the experiment period.
  - They demonstrated strong programming skills in Java and PHP.

- **Tasks**: maintenance tasks
  - Students performed maintenance tasks (bug fixing or new feature requests implementation) on 11 projects

- **Collected metrics**:
  - Developers: Overall industry experience, OSS experience
  - Tasks: Task difficulty, average time spent on task, task completion
  - COCOMO II SU factors: Factor rating and rationale
Correlation between SU factors and maintenance effort

- All three COCOMO II SU Factors showed strong negative relationship with maintenance effort (less effort spent on maintenance tasks).
- Automated analysis techniques (Maintainability Index, Complexity, Code Smells, Vulnerabilities) showed weaker correlation with maintenance effort, but stronger identification of problem-area parts of the code.
- Both human and automated methods are valuable. The automated methods are best at identifying parts of the code on which to apply the more labor-intensive human methods.
Large-Scale Maintainability Data Analytics

• Two approaches
  — Commit-absolute: absolute value of quality attributes in each data point
  — Commit-impact: impact of commits on different quality attributes

• Primary quality attributes
  — Code Smells, Security Vulnerabilities, Technical Debt histories

• Scale: 38 Apache programs, 586 million SLOC, 19580 commits across 15 years
Cumulative Technical Debt by Committer

commit-absolute: code_smells (sonarqube_system_uni) in apache-phoenix
Technical Debt Increase, Decrease by Committer
# Technical Debt Tools Used

## TABLE 1. QUALITY METRICS

<table>
<thead>
<tr>
<th>Group</th>
<th>Abbr.</th>
<th>Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Basic</strong></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><strong>Code Quality</strong></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><strong>Security</strong></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>
### Percent Technical Debt by Tool Used

#### TABLE 5. The percentage of $\text{const}(X) \cap \text{change}(Y)$ to $\text{const}(X)$

<table>
<thead>
<tr>
<th>Const</th>
<th>Basic</th>
<th>Change</th>
<th>Code Quality</th>
<th>Security</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LC</td>
<td>FN</td>
<td>CS</td>
<td>CX</td>
</tr>
<tr>
<td>LC</td>
<td>-</td>
<td>1.2</td>
<td>0.4</td>
<td>5.5</td>
</tr>
<tr>
<td>FN</td>
<td>55.1</td>
<td>-</td>
<td>2.0</td>
<td>29.6</td>
</tr>
<tr>
<td>CS</td>
<td>65.1</td>
<td>24.5</td>
<td>-</td>
<td>45.7</td>
</tr>
<tr>
<td>CX</td>
<td>40.1</td>
<td>1.8</td>
<td>1.6</td>
<td>-</td>
</tr>
<tr>
<td>SM</td>
<td>52.6</td>
<td>17.1</td>
<td>3.9</td>
<td>33.5</td>
</tr>
<tr>
<td>PD</td>
<td>37.9</td>
<td>6.5</td>
<td>1.7</td>
<td>17.7</td>
</tr>
<tr>
<td>VL</td>
<td>69.1</td>
<td>32.8</td>
<td>13.8</td>
<td>51.5</td>
</tr>
<tr>
<td>SG</td>
<td>70.2</td>
<td>34.5</td>
<td>15.5</td>
<td>53.1</td>
</tr>
<tr>
<td>FG</td>
<td>70.1</td>
<td>34.4</td>
<td>15.2</td>
<td>53.0</td>
</tr>
</tbody>
</table>
• Why Emphasize Software Maintainability?
  – General software trends and maintainability implications
  – Key role of Maintainability among software qualities
  – Growing risks for software-intensive systems
  – Growing software technical debt

• Tools for improving maintainability
  – Maintainability, technical debt assessment tools
    → Software Maintainability Readiness Framework (SMRF)
      • Non-technical sources of technical debt

• Conclusions
1. Separate organizations and budgets for systems and software acquisition and maintenance
2. Overconcern with the Voice of the Customer
3. The Conspiracy of Optimism
4. Inadequate system engineering resources
5. Hasty contracting that focuses on fixed operational requirements
6. CAIV-limited system requirements
7. Brittle, point-solution architectures
8. The Vicious Circle
9. Stovepipe systems
10. Over-extreme forms of agile development
2. Overconcern with the Voice of the Customer/User
Bank of America Master Net

<table>
<thead>
<tr>
<th>Users</th>
<th>Acquirers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Many features</td>
<td>Mission cost/effectiveness</td>
</tr>
<tr>
<td>Changeable requirements</td>
<td>Limited development budget, schedule</td>
</tr>
<tr>
<td>Applications compatibility</td>
<td>Government standards compliance</td>
</tr>
<tr>
<td>High levels of service</td>
<td>Political correctness</td>
</tr>
<tr>
<td>Voice in acquisition</td>
<td>Development visibility and control</td>
</tr>
<tr>
<td>Flexible contract</td>
<td>Rigorous contact</td>
</tr>
<tr>
<td>Early availability</td>
<td></td>
</tr>
</tbody>
</table>

**Maintainers**
- Ease of transition
- Ease of maintenance
- Applications compatibility
- Voice in acquisition

**Developers**
- Flexible contract
- Ease of meeting budget and schedule
- Stable requirements
- Freedom of choice: process
- Freedom of choice: team
- Freedom of choice: COTS/reuse

PC: Process
PD: Product
PP: Property
S: Success
3. The Conspiracy of Optimism
Take the lower branch of the Cone of Uncertainty

Aerospace America, 1/2016

F-22
187 A/C
$79B

F-22
750 A/C
$26B
4. Inadequate system engineering resources

---

**Sweet Spot Drivers:**
- Rapid Change: leftward
- High Assurance: rightward

---
5. Hasty contracting that focuses on fixed operational requirements
   5. Fixed price contract to minimum-cost, technically-acceptable bidder

6. CAIV-limited system requirements
   5. Below-threshold capabilities dropped from RFP, losing evolution insight

7. Brittle, point-solution architectures
   5. Result of 5,6. Need set-based design

8. The Vicious Circle
   5. Maintainers: We wish we could participate in the acquisition process, but we’re overloaded in fixing TD problems from similar previous acquisitions

9. Stovepipe systems
   5. Sharing common elements means they are upgraded once vs. N times

10. Over-extreme forms of agile development
    5. Easiest-first increments; Sunny-day use cases; Defer quality requirements
# Software-Intensive Systems Maintainability Readiness Levels

<table>
<thead>
<tr>
<th>SMR Level</th>
<th>OpCon, Contracting: Missions, Scenarios, Resources, Incentives</th>
<th>Personnel Capabilities and Participation</th>
<th>Enabling Methods, Processes, and Tools (MPTs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>5 years of successful maintenance operations, including outcome-based incentives, adaptation to new technologies, missions, and stakeholders</td>
<td>In addition, creating incentives for continuing effective maintainability, performance on long-duration projects</td>
<td>Evidence of improvements in innovative O&amp;M MPTs based on ongoing O&amp;M experience</td>
</tr>
<tr>
<td>8</td>
<td>One year of successful maintenance operations, including outcome-based incentives, refinements of OpCon.</td>
<td>Stimulating and applying People CMM Level 5 maintainability practices in continuous improvement and innovation in such technology areas as smart systems, use of multicore processors, and 3-D printing</td>
<td>Evidence of MPT improvements based on ongoing refinement, and extensions of ongoing evaluation, initial O&amp;M MPTs.</td>
</tr>
<tr>
<td>7</td>
<td>System passes Maintainability Readiness Review with evidence of viable OpCon, Contracting, Logistics, Resources, Incentives, personnel capabilities, enabling MPTs</td>
<td>Achieving advanced People CMM Level 4 maintainability capabilities such as empowered work groups, mentoring, quantitative performance management and competency-based assets, particularly across key domains.</td>
<td>Advanced, integrated, tested, and exercised full-LC MBS&amp;SE MPTs and Maintainability-other-SQ tradespace analysis.</td>
</tr>
<tr>
<td>6</td>
<td>Mostly-elaborated maintainability OpCon. with roles, responsibilities, workflows, logistics management plans with budgets, schedules, resources, staffing, infrastructure and enabling MPT choices, V&amp;V and review procedures.</td>
<td>Achieving basic People CMM levels 2 and 3 maintainability practices such as maintainability work environment, competency and career development, and performance management especially in such key areas such as V&amp;V, identification &amp; reduction of technical debt.</td>
<td>Advanced O&amp;M MPT capabilities identified for use, particularly for O&amp;M MPTs and SW/SE MPTs identified for use. Basic MPTs for tradespace analysis among maintainability &amp; other SQs, including TCO being used.</td>
</tr>
<tr>
<td>5</td>
<td>Convergence, involvement of main maintainability success-critical stakeholders. Some maintainability use cases defined. Rough maintainability OpCon, other success-critical stakeholders, staffing, resource estimates. Preparation for NDI and outsource selections.</td>
<td>In addition, independent maintainability experts participate in project evidence-based decision reviews, identify potential maintainability conflicts with other SQs.</td>
<td>Automated O&amp;M MPT capabilities identified for use: Model-Based SW/SE, TCO analysis support. Basic O&amp;M MPT capabilities for modification, repair and V&amp;V; some initial use.</td>
</tr>
<tr>
<td>4</td>
<td>Artifacts focused on missions. Primary maintenance options determined, Early involvement of maintainability success-critical stakeholders in elaborating and evaluating maintenance options.</td>
<td>Critical mass of maintainability SysEs with mission SysE capability, coverage of full M-SysE skills areas, representation of maintainability success-critical-stakeholder organizations.</td>
<td>Advanced full-lifecycle (full-LC) O&amp;M MPTs and SW/SE MPTs identified for use. Basic MPTs for tradespace analysis among maintainability &amp; other SQs, including TCO being used.</td>
</tr>
<tr>
<td>2</td>
<td>Mission evolution directions and maintainability implications explored. Some mission use cases defined, some O&amp;M options explored.</td>
<td>Highly maintainability-capable SysEs included in Early SysE team.</td>
<td>Initial exploration of O&amp;M MPT options.</td>
</tr>
<tr>
<td>1</td>
<td>Focus on mission opportunities, needs. Maintainability not yet considered</td>
<td>Awareness of needs for early expertise for maintainability, concurrent engg'g, O&amp;M integration, Life Cycle cost estimation</td>
<td>Focus on O&amp;M MPT options considered.</td>
</tr>
</tbody>
</table>
### Software-Intensive Systems Maintainability Readiness Levels

<table>
<thead>
<tr>
<th>SMR Level</th>
<th>OpCon, Contracting: Missions, Scenarios, Resources, Incentives</th>
<th>Personnel Capabilities and Participation</th>
<th>Enabling Methods, Processes, and Tools (MPTs)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>7</strong></td>
<td>System passes Maintainability Readiness Review with evidence of viable OpCon, Contracting, Logistics, Resources, Incentives, personnel capabilities, enabling MPTs</td>
<td>Achieving advanced People CMM Level 4 maintainability capabilities such as empowered work groups, mentoring, quantitative performance management and competency-based assets, particularly across key domains.</td>
<td>Advanced, integrated, tested, and exercised full-LC MBS&amp;SE MPTs and Maintainability-other-SQ tradespace analysis</td>
</tr>
<tr>
<td><strong>6</strong></td>
<td>Mostly-elaborated maintainability OpCon. with roles, responsibilities, workflows, logistics management plans with budgets, schedules, resources, staffing, infrastructure and enabling MPT choices, V&amp;V and review procedures.</td>
<td>Achieving basic People CMM levels 2 and 3 maintainability practices such as maintainability work environment, competency and career development, and performance management especially in such key areas such as V&amp;V, identification &amp; reduction of technical debt.</td>
<td>Advanced, integrated, tested full-LC Model-Based Software &amp; Systems (MBS&amp;SE) MPTs and Maintainability-other-SQ tradespace analysis tools identified for use, and being individually used and integrated.</td>
</tr>
<tr>
<td><strong>5</strong></td>
<td>Convergence, involvement of main maintainability success-critical stakeholders. Some maintainability use cases defined. Rough maintainability OpCon, other success-critical stakeholders, staffing, resource estimates. Preparation for NDI and outsource selections.</td>
<td>In addition, independent maintainability experts participate in project evidence-based decision reviews, identify potential maintainability conflicts with other SQs</td>
<td>Advanced full-lifecycle (full-LC) O&amp;M MPTs and SW/SE MPTs identified for use. Basic MPTs for tradespace analysis among maintainability &amp; other SQs, including TCO being used.</td>
</tr>
</tbody>
</table>
Conclusions

• Software trends point toward higher percentage of Total Ownership Costs going into Maintenance
  – Maintainability is also critical for software Availability, Speed of Adaptation to threats and opportunities

• Tools are becoming available for assessing and improving Maintainability and reducing Technical Debt
  – CAST, SonarQube, Maintainability Index, PMD, FindBugs, Software Understanding
  – Large-scale Maintainability, Technical Debt data analytics

• Most of technical debt comes from non-technical sources
  – Assessable via Software Maintenance Readiness Framework (SMRF)
Bottom Line: Software Maintainability

- **Ever more success-critical**
  - Rapid response to threats, opportunities

- **Ever more difficult**
  - More severe threats; needs to interoperate

- **Ever more expensive**
  - More change, complexity, safe autonomy

- **Ever more constrained by legacy systems**
Backup Charts
### Software-Intensive Systems Maintainability Readiness Levels

<table>
<thead>
<tr>
<th>SMR Level</th>
<th>OpCon, Contracting: Missions, Scenarios, Resources, Incentives</th>
<th>Personnel Capabilities and Participation</th>
<th>Enabling Methods, Processes, and Tools (MPTs)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>2</strong></td>
<td>Mission evolution directions and maintainability implications explored. Some mission use cases defined, some O&amp;M options explored.</td>
<td>Highly maintainability-capable SysEs included in Early SysE team.</td>
<td>Initial exploration of O&amp;M MPT options</td>
</tr>
<tr>
<td><strong>1</strong></td>
<td>Focus on mission opportunities, needs. Maintainability not yet considered</td>
<td>Awareness of needs for early expertise for maintainability. concurrent engr'g, O&amp;M integration, Life Cycle cost estimation</td>
<td>Focus on O&amp;M MPT options considered</td>
</tr>
</tbody>
</table>
# SIS Maintainability Readiness Levels 3-5

## Software-Intensive Systems Maintainability Readiness Levels

<table>
<thead>
<tr>
<th>SMR Level</th>
<th>OpCon, Contracting: Missions, Scenarios, Resources, Incentives</th>
<th>Personnel Capabilities and Participation</th>
<th>Enabling Methods, Processes, and Tools (MPTs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>5</td>
<td>Convergence, involvement of main maintainability success-critical stakeholders. Some maintainability use cases defined. Rough maintainability OpCon, other success-critical stakeholders, staffing, resource estimates. Preparation for NDI and outsource selections.</td>
<td>In addition, independent maintainability experts participate in project evidence-based decision reviews, identify potential maintainability conflicts with other SQs</td>
<td>Advanced full-lifecycle (full-LC) O&amp;M MPTs and SW/SE MPTs identified for use. Basic MPTs for tradespace analysis among maintainability &amp; other SQs, including TCO being used.</td>
</tr>
<tr>
<td>4</td>
<td>Artifacts focused on missions. Primary maintenance options determined, Early involvement of maintainability success-critical stakeholders in elaborating and evaluating maintenance options.</td>
<td>Critical mass of maintainability SysEs with mission SysE capability, coverage of full M-SysE.skills areas, representation of maintainability success-critical-stakeholder organizations.</td>
<td>Advanced O&amp;M MPT capabilities identified for use: Model-Based SW/SE, TCO analysis support. Basic O&amp;M MPT capabilities for modification, repair and V&amp;V: some initial use.</td>
</tr>
</tbody>
</table>
# Software-Intensive Systems Maintainability Readiness Levels

<table>
<thead>
<tr>
<th>SMR Level</th>
<th>OpCon, Contracting: Missions, Scenarios, Resources, Incentives</th>
<th>Personnel Capabilities and Participation</th>
<th>Enabling Methods, Processes, and Tools (MPTs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>9</td>
<td>5 years of successful maintenance operations, including outcome-based incentives, adaptation to new technologies, missions, and stakeholders</td>
<td>In addition, creating incentives for continuing effective maintainability, performance on long-duration projects</td>
<td>Evidence of improvements in innovative O&amp;M MPTs based on ongoing O&amp;M experience</td>
</tr>
<tr>
<td>8</td>
<td>One year of successful maintenance operations, including outcome-based incentives, refinements of OpCon.</td>
<td>Stimulating and applying People CMM Level 5 maintainability practices in continuous improvement and innovation in such technology areas as smart systems, use of multicore processors, and 3-D printing</td>
<td>Evidence of MPT improvements based on ongoing refinement, and extensions of ongoing evaluation, initial O&amp;M MPTs.</td>
</tr>
<tr>
<td>7</td>
<td>System passes Maintainability Readiness Review with evidence of viable OpCon, Contracting, Logistics, Resources, Incentives, personnel capabilities, enabling MPTs</td>
<td>Achieving advanced People CMM Level 4 maintainability capabilities such as empowered work groups, mentoring, quantitative performance management and competency-based assets, particularly across key domains.</td>
<td>Advanced, integrated, tested, and exercised full-LC MBS&amp;SE MPTs and Maintainability-other-SQ tradespace analysis</td>
</tr>
</tbody>
</table>
To what extent do COCOMO II SU factors accurately assess to software maintainability?

**Hypothesis**

COCOMO II SU factors accurately assess to software maintenance effort.

**Null Hypothesis**

There is no significant relation among SU factors and effort spent on maintenance tasks.
Projects Selection

Selection Criteria

- Active online issue tracking system
- The latest stable release
- Has fully accessible source code
- Well-presented in the community
- Appropriate source code size and number of issues
Projects were selected from Apache community & SourceForge.

Java Projects:
- Docfetcher, DrJava, Aerosolve, Pinot, Auil, PMD

PHP Projects:
- Joomla, Composer, Codeception, Pagekit, PhpUnit
commit-impact: vulnerabilities (sonarqube_system_uni) in apache-phoenix
<table>
<thead>
<tr>
<th></th>
<th>Basic</th>
<th></th>
<th></th>
<th>Code Quality</th>
<th></th>
<th></th>
<th>Security</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>LC</td>
<td>FN</td>
<td>CS</td>
<td>CX</td>
<td>SM</td>
<td>PD</td>
<td>VL</td>
<td>SG</td>
<td>FG</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Avro</td>
<td>80</td>
<td>44</td>
<td>21</td>
<td>70</td>
<td>47</td>
<td>73</td>
<td>5.3</td>
<td>2.7</td>
<td>2.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Calcite</td>
<td>89</td>
<td>57</td>
<td>34</td>
<td>75</td>
<td>58</td>
<td>83</td>
<td>4.1</td>
<td>0.9</td>
<td>0.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-BCEL</td>
<td>54</td>
<td>16</td>
<td>5</td>
<td>31</td>
<td>45</td>
<td>53</td>
<td>4.6</td>
<td>3.0</td>
<td>4.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-Beanu.</td>
<td>47</td>
<td>17</td>
<td>7</td>
<td>27</td>
<td>28</td>
<td>40</td>
<td>0.0</td>
<td>0.7</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-Codec</td>
<td>42</td>
<td>20</td>
<td>6</td>
<td>23</td>
<td>26</td>
<td>36</td>
<td>0.8</td>
<td>0.5</td>
<td>0.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-Collect.</td>
<td>51</td>
<td>22</td>
<td>11</td>
<td>30</td>
<td>29</td>
<td>38</td>
<td>0.9</td>
<td>1.6</td>
<td>1.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-Comp.</td>
<td>64</td>
<td>32</td>
<td>10</td>
<td>49</td>
<td>39</td>
<td>54</td>
<td>3.5</td>
<td>2.3</td>
<td>1.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-Config.</td>
<td>65</td>
<td>34</td>
<td>9</td>
<td>37</td>
<td>28</td>
<td>43</td>
<td>0.0</td>
<td>0.9</td>
<td>0.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-CSV</td>
<td>39</td>
<td>16</td>
<td>9</td>
<td>25</td>
<td>20</td>
<td>32</td>
<td>0.7</td>
<td>1.2</td>
<td>0.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-DBCP</td>
<td>56</td>
<td>19</td>
<td>5</td>
<td>35</td>
<td>40</td>
<td>46</td>
<td>2.7</td>
<td>1.6</td>
<td>2.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-IO</td>
<td>55</td>
<td>29</td>
<td>8</td>
<td>38</td>
<td>35</td>
<td>45</td>
<td>1.4</td>
<td>1.8</td>
<td>1.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-JCS</td>
<td>79</td>
<td>47</td>
<td>17</td>
<td>57</td>
<td>48</td>
<td>66</td>
<td>8.3</td>
<td>3.0</td>
<td>3.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-Jexl</td>
<td>64</td>
<td>40</td>
<td>19</td>
<td>56</td>
<td>46</td>
<td>59</td>
<td>4.2</td>
<td>1.7</td>
<td>1.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-Net</td>
<td>59</td>
<td>21</td>
<td>6</td>
<td>39</td>
<td>39</td>
<td>51</td>
<td>4.5</td>
<td>1.4</td>
<td>0.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-Pool</td>
<td>44</td>
<td>18</td>
<td>8</td>
<td>27</td>
<td>27</td>
<td>36</td>
<td>3.6</td>
<td>0.7</td>
<td>0.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-SCXMS</td>
<td>74</td>
<td>32</td>
<td>14</td>
<td>52</td>
<td>49</td>
<td>63</td>
<td>4.2</td>
<td>0.0</td>
<td>0.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-Validat.</td>
<td>56</td>
<td>12</td>
<td>5</td>
<td>26</td>
<td>36</td>
<td>43</td>
<td>0.4</td>
<td>0.7</td>
<td>0.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>C-VFS</td>
<td>51</td>
<td>21</td>
<td>6</td>
<td>29</td>
<td>31</td>
<td>44</td>
<td>2.0</td>
<td>0.2</td>
<td>1.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CXF-Fediz</td>
<td>78</td>
<td>39</td>
<td>19</td>
<td>62</td>
<td>50</td>
<td>68</td>
<td>9.3</td>
<td>1.2</td>
<td>3.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Drill</td>
<td>88</td>
<td>48</td>
<td>29</td>
<td>74</td>
<td>58</td>
<td>80</td>
<td>8.3</td>
<td>1.5</td>
<td>3.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Flume</td>
<td>88</td>
<td>47</td>
<td>29</td>
<td>66</td>
<td>59</td>
<td>72</td>
<td>7.4</td>
<td>1.2</td>
<td>1.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Giraph</td>
<td>85</td>
<td>58</td>
<td>39</td>
<td>69</td>
<td>49</td>
<td>74</td>
<td>7.7</td>
<td>2.6</td>
<td>1.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hama</td>
<td>79</td>
<td>44</td>
<td>25</td>
<td>60</td>
<td>60</td>
<td>71</td>
<td>18.9</td>
<td>6.3</td>
<td>7.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Helix</td>
<td>87</td>
<td>47</td>
<td>24</td>
<td>70</td>
<td>67</td>
<td>78</td>
<td>9.0</td>
<td>1.4</td>
<td>2.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H-Client</td>
<td>73</td>
<td>34</td>
<td>14</td>
<td>54</td>
<td>40</td>
<td>56</td>
<td>2.9</td>
<td>2.0</td>
<td>1.6</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>H-Core</td>
<td>69</td>
<td>40</td>
<td>17</td>
<td>50</td>
<td>41</td>
<td>54</td>
<td>0.4</td>
<td>1.8</td>
<td>1.2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Mina</td>
<td>79</td>
<td>40</td>
<td>15</td>
<td>58</td>
<td>54</td>
<td>67</td>
<td>9.6</td>
<td>1.2</td>
<td>0.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>M-SSHID</td>
<td>85</td>
<td>52</td>
<td>30</td>
<td>75</td>
<td>58</td>
<td>75</td>
<td>17.0</td>
<td>2.6</td>
<td>3.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Nutch</td>
<td>81</td>
<td>33</td>
<td>17</td>
<td>62</td>
<td>59</td>
<td>74</td>
<td>10.1</td>
<td>3.2</td>
<td>4.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OpenNLP</td>
<td>68</td>
<td>39</td>
<td>22</td>
<td>54</td>
<td>56</td>
<td>61</td>
<td>6.8</td>
<td>4.6</td>
<td>4.1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P-MR</td>
<td>76</td>
<td>45</td>
<td>29</td>
<td>63</td>
<td>52</td>
<td>66</td>
<td>1.9</td>
<td>4.8</td>
<td>2.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phoenix</td>
<td>84</td>
<td>43</td>
<td>20</td>
<td>72</td>
<td>58</td>
<td>77</td>
<td>10.2</td>
<td>4.0</td>
<td>7.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Qpid-JMS</td>
<td>75</td>
<td>33</td>
<td>14</td>
<td>60</td>
<td>43</td>
<td>60</td>
<td>5.8</td>
<td>1.9</td>
<td>2.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Ranger</td>
<td>88</td>
<td>44</td>
<td>14</td>
<td>74</td>
<td>63</td>
<td>81</td>
<td>17.3</td>
<td>0.3</td>
<td>3.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Santuario</td>
<td>77</td>
<td>35</td>
<td>17</td>
<td>66</td>
<td>54</td>
<td>73</td>
<td>6.7</td>
<td>3.0</td>
<td>4.3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shiro</td>
<td>70</td>
<td>43</td>
<td>21</td>
<td>52</td>
<td>42</td>
<td>56</td>
<td>2.8</td>
<td>2.1</td>
<td>2.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tiles</td>
<td>82</td>
<td>54</td>
<td>28</td>
<td>67</td>
<td>56</td>
<td>68</td>
<td>7.1</td>
<td>0.6</td>
<td>2.9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zookeeper</td>
<td>83</td>
<td>39</td>
<td>15</td>
<td>71</td>
<td>53</td>
<td>69</td>
<td>8.0</td>
<td>1.8</td>
<td>4.0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>AVG</td>
<td>70</td>
<td>36</td>
<td>17</td>
<td>53</td>
<td>46</td>
<td>60</td>
<td>5.7</td>
<td>1.9</td>
<td>2.4</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>DEV</td>
<td>15</td>
<td>13</td>
<td>9</td>
<td>17</td>
<td>12</td>
<td>15</td>
<td>4.7</td>
<td>1.4</td>
<td>1.8</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Initial Quantitative Maintainability Assessment:
Evaluate SW Maintainability Index on Open Source Projects

- Evaluate MI across 97 open source projects
  - 3 programming languages: Java, PHP, Python
  - 5 domains: Web development framework, System administration, Test tools, Security/Encryption, Audio-Video
- Test MI invariance across languages, domains
- Evaluate completeness of MI vs. other sources
  - COCOMO II Software Understandability factors
    - Structuredness (cohesion, coupling)
    - Self-descriptiveness (documentation quality)
    - Application clarity (software reflects application content)
  - Other maintainability enablers (architecture, V&V support)
    - Repairability: Diagnosability, Accessibility, Testability, Tool support
    - Search for similar defects; root cause analysis
Web Development Framework has shown the highest medians and the highest maximum value.

Audio and Video has both the lowest maximum value and the lowest median value.

**PHP** may be a good option for projects that desires higher maintainability within Web Development Framework, Security/Cryptography and Audio and Video domain.

**Python** may be a good option for System Administrative Software.

**Java** may be a good option for Software Testing Tools.