In-Depth View of Software Maintenance Cost Estimation
Selected Topics Workshop Summary

28th International Forum on COCOMO® and Systems/Software Cost Modeling
October 22-24, 2013
University of Southern California, Los Angeles, CA
Contents

1 SWM-Work Breakdown Structure ........................................................................................................ 1
2 SWM Size ............................................................................................................................................... 2
3 Analogous Program Selection Criteria ................................................................................................. 2
4 Risk and Uncertainty Attributes ........................................................................................................ 3
5 Handout ................................................................................................................................................ 4
  5.1 Software Maintenance Work Breakdown Structure (v4.4a) ......................................................... 4
  5.2 Software Size .................................................................................................................................. 6
    5.2.1 Software Product Size .................................................................................................................. 6
    5.2.2 Software Change Size .................................................................................................................. 6
      5.2.2.1 Source of Change .................................................................................................................... 6
      5.2.2.2 Software Change Type .......................................................................................................... 6
      5.2.2.3 Software Change Effect ......................................................................................................... 7
      5.2.2.4 Software Change Priority ..................................................................................................... 7
  5.3 Software Application Super-Domains ............................................................................................... 8
  5.4 Software System Characterization .................................................................................................. 9
  5.5 Risk and Uncertainty ....................................................................................................................... 11
    5.5.1 Internal Software Product Attributes ....................................................................................... 11
    5.5.2 Program/Project Management Attributes ............................................................................... 11
    5.5.3 External Driver Attributes ....................................................................................................... 12
  5.6 Program/Project Management Attributes ..................................................................................... 13
  5.7 External Driver Attributes ............................................................................................................ 13
In-Depth View of Software Maintenance Cost Estimation
Selected Topics

This workshop covered four topics that impact SWM cost estimates. Each topic had an overview followed by a set of questions with discussion. The workshop had 21 attendees:

1. Jo Ann Lane, USC
2. Tom McGibbon, CSIAC
3. Gary Hafen, Lockheed Martin Aerospace
4. Jairus Hihn, JPL
5. Justine McNeill, Aerospace Corp.
6. Jim Cain, SPAWAR
7. Gary Palosaar, Aerospace Corp.
8. Ed Colbert, USC
9. Jim Alstad, USC
10. Michael Cleary, IBM & Cal State Dominguez Hills
11. Denton Tarbet, Galorath
12. Arlene Minkiewicz, Price Systems
13. Gan Wang, BAE Systems
14. Dan Houston, Aerospace Corp.
15. Ray Madachy, Naval Post Graduate School
16. Sue Koolmanojwong, USC
17. Linda Esker, Fraunhofer Inst.
18. Joe Provenzano, Aerospace Corp.
20. Alexey Tregubov, USC
21. Lori Saleski, BAE Systems
22. Brad Clark – USC & Moderator

1 SWM-Work Breakdown Structure

- **Handout** and review of the Army Software Maintenance Work Breakdown Structure
  1. Software Change Product
  2. System Project Management
  3. Software Licenses
  4. Certifications and Accreditations
  5. System Facilities
  6. Sustaining Engineering
  7. Field Support
  8. Support Infrastructure

- **Question:** Are there missing activities?
- **Discussion**
  - Not “product” oriented, i.e. what is being maintained?
    - If the software was spread across five components (e.g. communication, navigation, fire control, diagnostics, IFF), would the WBS as a whole be applied to each component?
  - Appears to apply to “Organic” labor, i.e., Army government personnel maintenance organizations
Software Maintenance Workshop Summary

2 SWM Size

- **Handout** description of Product and Change Size
  - Software Product size (SLOC): total code base, added, modified, reused, auto-generated, and deleted.
  - Software Change Size
    - Source of change
    - Software change type
    - Software change effect
    - Software change priority
- Question: What is the relevance of these different measures of workload to the cost of performing software maintenance?
- Discussion:
  - Missing the impact caused by making a change to one software configuration item as opposed to many configuration items, i.e.
    
    *There is a difference between making a 10 SLOC change to one configuration item than making a 1 SLOC change to 10 configuration items.*
  - Missing the impact of change on a software system constructed using object oriented technology. A change high-up in a class inheritance tree could have ripple effects for all derived classes compared to an instantiated class.
  - How experienced is the maintenance organization with the software being maintained as a percentage of the full software system, i.e., the organization has worked in 20% of the code base?
    - Experience in changing the design, source code and test procedures
    - Is the code well structured? Is the code self-descriptive? Is the code’s application aligned with the program purpose?
  - Software Change Type is missing Trusted/Security and Safety type of change. These changes can be very expensive.

—BREAK—

3 Analogous Program Selection Criteria

- **Handout** description of Super-Domains and Software System Attributes
- Questions:
  - What is the productivity rank of Domains within Super-Domains?
  - Are there major characterizations missing in the Software System attributes?
- Discussion:
  - Missing an important characterization addressing reliability (based on NASA classes / levels):
    - Class A: Human Rated systems. Ex is human rated systems. Would tend to map into Very High reliability
    - Class B: Mission Critical systems: Ex is robotic flight software. Would tend to map into Very High reliability

- Fire House Effect: some of the cost groups have to be funded whether you change software or not, i.e., the capability must exist much like a Fire House has to exist even if there are no fires.
- Class C: Mission Support Systems: Ex is ground data processing. Would tend to map into Nominal or Nominal+ reliability
  - COTS Product Incorporation characteristic is missing something on the existence of Vendor support.

4 Risk and Uncertainty Attributes

- Handout description of the different uncertainty attributes
  - Internal Software Product Attributes
    - Number Of External Interfaces
    - Execution Timing Constraints
    - COTS Product Incorporation
    - Critical Technology
  - Program/Project Factors
    - Management Personal Capability
    - Technical Personal Capability
    - Technical Process Capability
    - Facilities & Infrastructure Support
    - Local Maintenance/Funding Rhythm
  - External Drivers
    - Project & Program Management
    - External Stakeholders
    - Mandates
    - Policy-driven Maintenance/Funding Rhythm
  - Risk exposure is derived by asking what you should know about each attribute and what you do know about each attribute

- Questions:
  - Are there other attributes that should be considered?
  - What “should” you know at the different lifecycle phases?

- Discussion
  - Suggestion: Add another column to Internal Software Product Attributes for Assurance/Proof Complexity that addresses proving required reliability.
  - Comment: Delphi to determine table values are nice, but it would be good to validate them against real data.
  - Comment: The approach looks good. Perhaps it could also be applied to development costs.
5 Handout

5.1 Software Maintenance Work Breakdown Structure (v4.4a)

1.0 Software Change Product (System Specific)
Note: The software product includes builds, releases, individual changes, non-
developmental software updates and other configurable software products. The
products, as defined, are usually, but not always, delivered to the user base.

1.1 Change requirements
1.1.1 Individual change definition, characterization, and analysis
Note: This activity includes the technical and economic analysis of individual
software changes. Characterization attribute information may include:
• Source (derivation of change requirement, including IAVA)
• Funding/capability set assignment
• Technical characteristics/description
  - Size (multiple measures)
  - Complexity
  - Criticality
  - Reliability
  - Performance constraints (timing, etc.)
• Impact analysis
• Prioritization (single system or multiple systems)
• Other information

1.1.2 Product change allocation
Note: This activity includes the allocation of one or more individual changes into
a defined software product (e.g., change board prioritization and allocation
efforts, management of system change backlog, etc.).

1.2 Change development (Change driven code modifications)
1.2.1 Baseline analysis - change design
1.2.2 Code and unit test
1.2.3 Baseline modifications required to integrate COTS or upgraded non-
developmental software
1.2.4 Individual change and/or integration interface verification
1.2.5 Rework

1.3 Baseline integration and test
1.3.1 Interface testing - verification
1.3.2 System software testing
1.3.3 Quality Assurance - IV&V
2.0 System project and technical management (System Specific)
   2.1 Planning
   2.2 Execution management
   2.3 Configuration management
   2.4 Resource - team management
   2.5 Contracting
   2.6 Measurement - reporting

3.0 Software Licenses (System/Non-System Specific)
   3.1 Deployed systems
   3.2 Facility systems

4.0 Certification and Accreditation (System Specific)
   4.1 DITSCAP/DIACAP (IAVA cert happens here)
   4.2 Safety
   4.3 Networthiness

5.0 System Facilities (System/Non-System Specific)
   5.1 Software development assets/workstations
   5.2 System integration and test facilities
   5.3 Test equipment and tools

6.0 Sustaining Engineering (System Specific)
   6.1 Test support
   6.2 Software delivery
   6.3 User training (development and/or delivery)
   6.4 User support

7.0 Field Software Engineering (System Specific)
   7.1 On-site technical assistance
   7.2 Problem troubleshooting
   7.3 Software installation
   7.4 Operational assistance
   7.5 On-site training

8.0 Support Infrastructure (Non-system Specific)
   8.1 Operations
   8.2 Organization management
   8.3 Personnel management
   8.4 Financial management
   8.5 Information management
   8.6 Process management
   8.7 Change management
5.2 Software Size

5.2.1 Software Product Size

- Baseline Total Size (KSLOC)
- New / Added Size (KSLOC)
- Modified / Changed Size (KSLOC)
- Reused / No Change Size (KSLOC)
- Auto-Gen Product Size (KSLOC)
- Deleted Size (KSLOC)
- Comment Size (KCLOC)
- Final Total Size (KSLOC)
- Count Method

5.2.2 Software Change Size

5.2.2.1 Source of Change

Identifies the source of the requested software change. There are two primary sources: external and internal. For each Change Request, select one of the numbered items.

**External Drivers**

- Operational User Driven
  1. Functional modifications, i.e., *Bug Fixes*
  2. Functional additions
  3. Functional deletions
- Stakeholder Requirements
  4. Threat
  5. Mission doctrine
  6. System interoperability
  7. External testing / IV&V
  8. External audits
- Mandates
  9. Legal / Regulatory / Policy

**External Drivers (continued)**

- Technology
  10. Technology obsolescence
  11. Infrastructure changes

**Internal Drivers**

- Legacy Issues
  12. Technical debt
  13. Deferred functionality
  14. Bug Fixes
- Other (please specify)

5.2.2.2 Software Change Type

Identifies the nature of the change by technical category. For each Change Request, select one of the numbered items.

1. Computational (e.g. incorrect equation, truncation errors)
2. Logic (e.g. logic out of sequence, missing logic, incorrect condition test)
3. Input (Incorrect format, incorrect data source)
4. Data handling (e.g. data definition, initialization, boundaries)
5. Output (e.g. Incorrect output destination, incorrect format, garbled)
6. Interface (e.g. SW/HW, SW/User, SW/SW)
7. Operations (e.g. COTS/GOTS change, configuration control)
8. Performance (e.g. time / storage limits exceeded, inefficient design / code)
9. Specifications (e.g. requirements incorrect / inadequate, user docs inadequate)
10. Improvement (e.g. improve existing
output) function, improve interface)
11. Other (please specify)

5.2.2.3 Software Change Effect

Categorizes the “end item” effect related to the change. For each Change Request, select one of the numbered items.

1. Functionality
2. Usability
3. Security
4. Performance (Reliability, Availability)
5. Maintainability (Serviceability)
6. Other (please specify)

5.2.2.4 Software Change Priority

Provides a relative measure of the urgency for implementing the change. For each Change Request, select one of the numbered items.

1. Priority 1 - Critical
2. Priority 2 - Major
3. Priority 3 - Minor
4. Priority 4 - Unknown
5. Priority 5 - N/A
### 5.3 Software Application Super-Domains

<table>
<thead>
<tr>
<th>Super-Domain</th>
<th>Description</th>
</tr>
</thead>
</table>
| A. Real-Time | Most complex software type. These projects take the most time and effort for a given system size due to the lower language levels, high level of abstraction and increased complexity  
- Tightly coupled interfaces  
- Real-time scheduling requirements  
- Very high reliability requirements (life critical)  
- Generally severe memory and throughput constraints  
- Often executed on special-purpose hardware |
| B. Engineering | Medium complexity  
- Multiple interfaces with other systems  
- Constrained response time requirement  
- High reliability but not life critical  
- High degree of algorithmic or communication complexity  
Generally executed *on top of* COTS middleware / infrastructure products. For example:  
- Operating Systems  
- Simulation Platforms  
- Enterprise Services Management  
- Communication Services  
- Security Services  
- Content Discovery / Delivery  
- Audio & Video Transmission  
- Virtual Space Management  
- User Management. |
| C. Mission Support | Least complex software type.  
- Relatively less complex  
- Self-contained or few interfaces  
- Less stringent reliability requirement.  
Generally includes software in test equipment, training equipment, data loaders/extractors, and utility-type equipment. |
D. Automated Information Systems

Software that automates information processing often incorporates COTS products. Often written in more human-oriented languages and perform common mission or business functions such as:

- Intelligence Processing
- Situational Awareness
- Asset or Material Tracking
- Financial Transactions
- Storage/Retrieval Of Data

These applications allow the designated authority to exercise control over the accomplishment of the mission or business area. Humans manage a dynamic situation and the application responds to user-input to facilitate mission/business objectives.

### 5.4 Software System Characterization

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Description</th>
</tr>
</thead>
</table>
| 1. Number Of External Interfaces     | • Interoperability with other systems  
• Change in the number of system interfaces  
• Degree of external interfaces coupling, e.g., waiting problem  
• Integrated functionality (versus stand-alone, maybe implied by # interfaces)  
• Impact of eternal interface volatility  
• Implications:  
• What is indirectly affected by the number of interfaces |
| Rating: Few / Many                    |                                                                            |
| 2. Execution Timing Constraints       | • Event driven  
• Time driven (hard real-time)  
• Non-stop operation |
| Rating: Low / High                    |                                                                            |
| 3. COTS Product Incorporation         | • Number of COTS products  
• Are new products being incorporated? |
<p>| Rating: (Light / Heavy)               |                                                                            |</p>
<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4. Critical Technology</td>
<td>Overall rating: High / Low</td>
</tr>
<tr>
<td></td>
<td>A. Algorithmic complexity (High / Low)</td>
</tr>
<tr>
<td></td>
<td>• Maturity</td>
</tr>
<tr>
<td></td>
<td>• Human Machine Interface (HMI)</td>
</tr>
<tr>
<td></td>
<td>• Display complexity</td>
</tr>
<tr>
<td></td>
<td>• Failure impact</td>
</tr>
<tr>
<td></td>
<td>• Data fusion</td>
</tr>
<tr>
<td></td>
<td>• Security requirements including Information Assurance</td>
</tr>
<tr>
<td></td>
<td>• Pre-programed reactions to events</td>
</tr>
<tr>
<td></td>
<td>B. Communication Complexity (High / Low)</td>
</tr>
<tr>
<td></td>
<td>• Encryption</td>
</tr>
<tr>
<td></td>
<td>• Anti-jam</td>
</tr>
<tr>
<td></td>
<td>• Reliability</td>
</tr>
<tr>
<td></td>
<td>• Frequency hopping</td>
</tr>
<tr>
<td></td>
<td>C. Security Complexity (High / Low)</td>
</tr>
<tr>
<td></td>
<td>• Multi-level security in the same software product</td>
</tr>
<tr>
<td></td>
<td>• Updates to system parameters are done by different intelligence agencies</td>
</tr>
<tr>
<td></td>
<td>• Access group partitions functionality and data access, i.e. capabilities are turned on/off depending on the group.</td>
</tr>
<tr>
<td></td>
<td>D. Data Bandwidth Requirements (High / Low)</td>
</tr>
<tr>
<td></td>
<td>• Hard real-time requirement for data capture or transmission to moving vehicles</td>
</tr>
<tr>
<td></td>
<td>• Hard real-time requirements for capture of data from sensors or sending commands to actuators</td>
</tr>
<tr>
<td></td>
<td>• Data reduction &amp; analysis volume</td>
</tr>
<tr>
<td></td>
<td>• Storage constraints</td>
</tr>
</tbody>
</table>
5.5 Risk and Uncertainty

Objective is to improve the credibility of the cost estimate by:

- Identifying, characterizing and accounting for different cost performance factors (e.g., software product attributes, management factors, external program activities) and human estimation biases (e.g., anchoring, optimism bias, etc.) that may be sources of risk/uncertainty that can result in creating material impacts on a software sustainment and maintenance cost estimate.
- Making visible the “knowledge gap” (if any) between what should be known and what is known about the system being used to calculate a range of uncertainty associated the estimate.
- Fully documenting the key program issues and related performance factors that may influence the cost estimate and why.

5.5.1 Internal Software Product Attributes

<table>
<thead>
<tr>
<th></th>
<th>Number Of External Interfaces</th>
<th>Execution Timing Constraints</th>
<th>COTS Product Incorporation</th>
<th>Critical Technology</th>
</tr>
</thead>
<tbody>
<tr>
<td>What do we know?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>What should we know?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Individual Exposure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exposure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

5.5.2 Program/Project Management Attributes

<table>
<thead>
<tr>
<th></th>
<th>Management Personal Capability</th>
<th>Technical Personal Capability</th>
<th>Technical Process Capability</th>
<th>Facilities &amp; Infrastructure Support</th>
<th>Local Maintenance/Funding Rhythm</th>
</tr>
</thead>
<tbody>
<tr>
<td>What do we know?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>What should we know?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Individual Exposure</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exposure</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### 5.5.3 External Driver Attributes

<table>
<thead>
<tr>
<th></th>
<th>Project &amp; Program Management</th>
<th>External Stakeholders</th>
<th>Mandates</th>
<th>Policy-driven Maintenance/Funding Rhythm</th>
</tr>
</thead>
<tbody>
<tr>
<td>What do we know?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>What should we know?</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Individual Exposure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Exposure</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5.6 Program/Project Management Attributes

1. Management Personnel Capability
   • What percent of management is moderately to highly experienced in software maintenance?
   • What percent of management has worked on this type of system before?

2. Technical Personnel Capability
   • What percent of the personnel is moderately to highly experienced in software maintenance?
   • What percent of the personnel is moderately to highly experienced with this type of system?

3. Technical Processes Capability
   • What percent of the maintenance processes are useful/effective?
   • What percent of software trouble reports are traceable to process shortfalls?

4. Facilities & Infrastructure Support
   • What percent of the support tools is considered applicable/effective?
   • What percent of needed planned capital equipment (for example, for SILs, simulators, and emulators) is available?

5. Maintenance/Funding Rhythm
   • What percent of the maintenance work is discretionary as opposed to legally mandated?
   • What percent of needed funding was provided in the past fiscal year?
   • What percent of current maintenance work is attributed to backlog?
   • What percent of current maintenance work is attributed to technical debt?

5.7 External Driver Attributes

1. Project & Program Management
   • How experienced is the project/program management?
   • Is the project/program management team stable or changing?

2. External Stakeholders
   • How many external stakeholders are there?
   • Do external stakeholders provide funding, set requirements, or both?
   • Is there agreement or conflict among different stakeholders as to the system’s mission priorities?

3. Mandated Requirements
   • How stable are mandated policies and guidelines and do they conflict?
   • Are mandated policies, like security, fully funded?

4. Policy-driven Maintenance/Funding Rhythm
   • How long does the project/program funding stream look secure?
   • Are there planned and funded system upgrades?