Extending COCOMO II to Estimate the Cost of Developing Secure Software

Edward Colbert, Murali Gangadharan
Donald Reifer and Barry Boehm
{ecolbert, murali, boehm}@cse.usc.edu d.reifer@ieee.org

Ground System Architectures Workshops
GSAW 2003
March 4 - 6, 2003
Outline

- Why Extend COCOMO II for Security?
- The COCOMO II Family of Models
- New Security Cost Driver and its Factors
- COCOMO II Cost Driver Values
- Next Steps and Summary
Why Extend COCOMO II for Security

Getting money budgeted for security is easier this year than I can ever remember.

Did you say security? Here, take this!

How much do you need?!

Did someone say security?
Why Extend COCOMO II for Security

- Military projects have considered security in developing software since the early 1980s

- Until recently commercial projects often gave it little weight

- Threat to business-critical systems & private information has grown
  - Security can no longer be ignored

- Few cost models (including COCOMO II) include security factors
  - Based 1980s military perspective (Orange Book)
  - Developing secure systems has changed dramatically
Several approaches for addressing security

- Modernize existing models
  - Primarily by updating definitions of cost drivers
- Add new cost drivers to cost models
- Develop separate security model
Adding Security to Cost Models

- USC has taken intermediate approach
  - Add factor that addresses security from 3 viewpoints
    - Development
    - Operational
    - Physical
  - Include factors as appropriate to all COCOMO II family cost models
  - Address both commercial & military projects regardless of
    - Size
    - Domain
    - Level of maturity
Outline

- Why Extend COCOMO for Security?
- The COCOMO II Family of Models
- New Security Cost Driver and its Factors
- COCOMO II Cost Driver Values
- Next Steps and Summary
COCOMO II Refresher

Software size estimate

Software product, process, project and personnel cost drivers

Project scale factors: maturity, risk, flexibility, teamwork and precedentedness

Software organization’s project data

COCOMO II Model

Effort and duration estimates

Cost, schedule distribution by phase, activity, increment

COCOMO II recalibrated to organization’s data

Effort in Person Month

\[ PM_{estimated} = A \times (\text{Size})^{(SF)} \times \prod_{i} EM_{i} \]

SF: Scale Factors (5) EM: Effort Multipliers (17)
Outline

- Why Extend COCOMO for Security?
- The COCOMO II Family of Models
- New Security Cost Driver and its Factors
- COCOMO II Cost Driver Values
- Next Steps and Summary
COCOMO II Security Driver (SECU)

- Viewpoints
  - Physical Security
  - Operational Security
  - Development for Security

- Security strategies embraced
  - Ad hoc Defense (Low)
  - Passive Defense (Nominal)
  - Active Defense (High)
  - Layered Defense (Very High)
  - Defense in Depth (Extremely High)
Security Factors

- Development for Security
  - Effect of processes for development & validation when security a factor

- Operational Security
  - Effect of security policies, processes, tools and facilities that:
    - Permit identification of security events
    - Define subsequent actions to identify key elements
    - Report pertinent information to appropriate individual, group, or process

- Development Constraints
  - Constraints placed on development when protecting software facilities:
    - From outside perimeter to inside office space
    - Includes all of information system resources
Development for Security
Rating : Low & Nominal

- Low
  - No security requirements
  - No protection other than provided by execution environment

- Nominal

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Informal security requirements formulated for system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design</td>
<td>Analysis of security functions using</td>
</tr>
<tr>
<td></td>
<td>- Informal functional &amp; interface specification</td>
</tr>
<tr>
<td></td>
<td>- Descriptive high-level design</td>
</tr>
<tr>
<td></td>
<td>- Demonstration of corresponding pairs</td>
</tr>
<tr>
<td>Testing</td>
<td>Developer tests implementation of requirements</td>
</tr>
<tr>
<td></td>
<td>- Black box testing</td>
</tr>
<tr>
<td>Life-cycle controls</td>
<td>Simple Configuration Management (CM) with version</td>
</tr>
<tr>
<td></td>
<td>numbers</td>
</tr>
</tbody>
</table>
# Development for Security

**Rating: High**

- **Nominal +**

<table>
<thead>
<tr>
<th>Requirements</th>
<th>Fully defined external interfaces</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Informal security policy modeling</td>
</tr>
<tr>
<td>Design</td>
<td>Security enforcing high-level design</td>
</tr>
<tr>
<td></td>
<td>Informal low-level design description</td>
</tr>
<tr>
<td>Testing</td>
<td>Independent testing of all functional requirements</td>
</tr>
<tr>
<td></td>
<td>Inspection of COTS source code if available</td>
</tr>
<tr>
<td>Life-cycle controls</td>
<td>Detailed delivery &amp; installation procedures</td>
</tr>
<tr>
<td></td>
<td>Identification of security measures for life-cycle</td>
</tr>
</tbody>
</table>
# Development for Security

**Rating:** Extremely High

## Very High (+)

<table>
<thead>
<tr>
<th>Requirements</th>
<th></th>
</tr>
</thead>
</table>
|              | ✔ Fully defined external interfaces  
|              | ✔ Informal security policy modeling |

<table>
<thead>
<tr>
<th>Design</th>
<th></th>
</tr>
</thead>
</table>
|              | ✔ Semi-formal high level explanation  
|              | ✔ Structured implementation with reduction of complexity |
|              | ✔ Secure container for COTS and Open-source |

<table>
<thead>
<tr>
<th>Testing</th>
<th></th>
</tr>
</thead>
</table>
|              | ✔ Analysis of coverage of tests  
|              | ✔ Ordered functional testing with tests of low-level design |
|              | ✔ Covert channel analysis |

<table>
<thead>
<tr>
<th>Life-cycle controls</th>
<th></th>
</tr>
</thead>
</table>
|                     | ✔ Compete automation of CM  
|                     | – with coverage for developer tools |
|                     | ✔ Standardized life-cycle model  
|                     | – with compliance to implementation standards |
Operational Security
Rating: Low & Nominal

Low
- No organization-wide security policies
- Ad-hoc security practices
- Optional firewall & virus protection

Nominal

<table>
<thead>
<tr>
<th>Administration</th>
<th>Basic Security policies</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- inc.</td>
</tr>
<tr>
<td></td>
<td>• Password and Virus Protection policy</td>
</tr>
<tr>
<td></td>
<td>• Network access and system use policy</td>
</tr>
<tr>
<td></td>
<td>Guidelines for administrators &amp; users</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Protection</th>
<th>Reasonable practices for</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- Checksum Verification</td>
</tr>
<tr>
<td></td>
<td>- Software firewall(s)</td>
</tr>
<tr>
<td></td>
<td>- Operating system logging</td>
</tr>
</tbody>
</table>

| Authentication  | Simple password-based authentication schemes |
## Operational Security

**Rating: Very High**

- **High +**

### Administration

- Comprehensive Security policies
  - inc.
    - Business continuity plans
    - Disaster recovery plans
- Incident response teams handle security breaches

### Protection

- A layered defense strategy is implemented to protect the system with reasonable practices for
  - Proxy servers
  - Layered system monitoring with Intrusion Detection Systems

### Authentication

- Digital certificates & signatures used for
  - Authentication
  - Message integrity
  - Non-repudiation
Development Constraints
Rating Description

- **Nominal**
  - None

- **High**
  - All source materials are locked up when not in active use

- **Very High**
  - High +
    - Audited security markings in code

- **Extremely High**
  - Very High+
    - Multi-compartment developer communication constraints
Outline

- Why Extend COCOMO for Security?
- The COCOMO II Family of Models
- New Security Cost Driver and its Factors
- COCOMO II Cost Driver Values
- Next Steps and Summary
Draft Baseline Security Effort Distribution

- With wide standard deviations

<table>
<thead>
<tr>
<th>Phase</th>
<th>Security-Range</th>
<th>Security-Min</th>
<th>Base</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inception</td>
<td>6%</td>
<td>6%</td>
<td>6%</td>
<td>18%</td>
</tr>
<tr>
<td>Elaboration</td>
<td>24%</td>
<td>24%</td>
<td>24%</td>
<td>72%</td>
</tr>
<tr>
<td>Construction</td>
<td>48%</td>
<td>30%</td>
<td>76%</td>
<td>154%</td>
</tr>
<tr>
<td>Transition</td>
<td>6%</td>
<td>6%</td>
<td>12%</td>
<td>24%</td>
</tr>
</tbody>
</table>

With wide standard deviations
Outline

- Why Extend COCOMO for Security?
- The COCOMO II Family of Models
- New Security Cost Driver and its Factors
- COCOMO II Cost Driver Values
- Next Steps and Summary
Next Steps

- Reach consensus on cost drivers
  - FAA Workshop on security-May 03
  - Delphi run & calibration of factors in works

- Initiate efforts to statistically validate accuracy of model
  - Survey available 160+ COCOMO project data
  - Perform initial calibration

- Create enhanced COCOMO data collection forms
  - Gather security related efforts

- Compare actual project data to expert opinions
  - Calibrate the model by weighting
    - Actual data
    - Expert opinions using Bayesian statistical techniques
Summary

- Proposed extensions to COCOMO for development of Secure Systems
  - Based on Common Criteria
  - 1 Driver: SECU
  - 3 Factors:
    - Development for Security
    - Operational Security
    - Physical Security
  - Affects on other COCOMO II Drivers
    - RELY, CPLX, DOCU, SITE, TOOL
  - Affects on Size
  - Affects on project risk

- Hopefully stimulated your interest and motivated you to participate by sharing project data