(S3m) : Model to Evaluate and Improve the Quality of Software Maintenance Process

Alain April
Otto von Guericke University of Magdeburg, Germany
With the cooperation of
École de Technologie Supérieure de Montréal, Canada
Overview

- Research questions
- Part 1 - Related work in software maintenance
- Part 2 - Related work in software engineering maturity models
- Part 3 - Proposed process model
- Part 4 - Validation, conclusions and future work
Research questions

1- Is software maintenance a specific domain of software engineering? If so, what are its specific processes and activities?

2- Are the unique processes of software maintenance well reflected in the current international standards?

3- Is there an existing maturity model proposal that covers the entire set of software maintenance unique activities?

4- What would be the proposed architecture of a capability maturity model that could address the entire set of software maintenance unique activities?

5- Can the model be expressed formally and how can it be used in practice to support the improvement of software maintenance?
Part 1 - Related work in software maintenance
1 - Small maintenance?

1- Modification requests come in more or less randomly, and cannot be accounted for individually in the annual budget planning process;

2- Modification requests are reviewed and assigned priorities, often at the operational level - most do not require senior management involvement;

3- The maintenance workload is not managed using project management techniques, but rather queue management techniques;

4- The size and complexity of each small maintenance request are such that it can usually be handled by one or two maintenance resources; ISBSG and UKSMA \( \rightarrow 5 \text{ days or less} \)

5- Priorities can be shifted around at any time, and requests for corrections of application errors can take priority over other work in progress.
1 - Issues of maintenance

Top 10 issues by priority [Parik02 and Dekleva92]

1. Managing changing priorities
2. Inadequate testing techniques
3. Difficulty of measuring performance
4. Absent and/or incomplete software documentation
5. A large backlog of requests
6. Difficult to measure the maintenance team contribution to the organization
7. Low morale of maintenance personnel
8. Not many professionals in the domain
9. Little methodology, few standards, procedures and tools specific to maintenance
10. Source code in existing software is complex and unstructured
1 - New body of knowledge

Co-edited: SWEBOK chapter 5 -> Software Maintenance (www.swebok.org)

Fundamentals: Definitions, need, nature and costs
Key Issues: Technical, management and measurement
Maintenance Process: Processes and activities
Techniques for maintenance:
  Program comprehension and impact analysis
1 - Need for S3\textsuperscript{m}

- **CMM and CMM\textit{i} focus**
  - Software Development and Maintenance \textit{Projects}
  - Teams of developers

- **Software Maintenance Specific Processes (SWEBOK):**
  - Transition
  - Service Level Agreements
  - Acceptance/Rejection of Change and Corrective Requests
  - Planning Maintenance activities
  - Supporting operational software
Part 2 - Related work in software engineering maturity models
2- How to build a maturity model

1. Understand the knowledge area
2. Look in references to find processes, activities and best practices
3. Look for an Architecture to create domains and KPAs
4. Decide practices to be included in the model and their maturity level
5. Build or Refine the model Architecture
6. Pre and Post validate the model
7. Review the results with sponsors and experts
8. Improve model as necessary
2- Why the CMMi Architecture?

- Contains the essential elements of effective processes for software related activities
- Contains a framework that provides the ability to generate multiple models and associated training and assessment materials. These models may represent:
  - software and systems engineering
  - integrated product and process development
  - new disciplines
  - combinations of disciplines
- Provides guidance to use when developing processes
2- What current MM could help?

<table>
<thead>
<tr>
<th>Year</th>
<th>Software Engineering</th>
<th>CMMM proposals</th>
</tr>
</thead>
<tbody>
<tr>
<td>1991</td>
<td>Bootstrap</td>
<td></td>
</tr>
<tr>
<td>1992</td>
<td>Trillium</td>
<td></td>
</tr>
<tr>
<td>1993</td>
<td>CMM©</td>
<td></td>
</tr>
<tr>
<td>1994</td>
<td>Camélia, automated testing (Kra94)</td>
<td></td>
</tr>
<tr>
<td>1996</td>
<td>TMM (Bur96), Zit96, Dov96</td>
<td></td>
</tr>
<tr>
<td>1997</td>
<td>Som97</td>
<td></td>
</tr>
<tr>
<td>1998</td>
<td>Esi98, Top98, Baj98</td>
<td></td>
</tr>
<tr>
<td>1999</td>
<td>Wit99, Vet99, Sch99</td>
<td></td>
</tr>
<tr>
<td>2000</td>
<td>Cob00, Str00, Bev00, Lud00</td>
<td></td>
</tr>
<tr>
<td>2001</td>
<td>Kaj01d &amp; 01e, Ray01, Sch01, Luf01, Tob01, Sri01</td>
<td></td>
</tr>
<tr>
<td>2002</td>
<td>CMMi®, Nie02, Mul02, Vee02, Pom02, Raf02, Sch02, Ker02, Cra02</td>
<td></td>
</tr>
<tr>
<td>2003</td>
<td>Sche03, Wid03, Rea03, Dui03, Nas03, Usd03</td>
<td></td>
</tr>
</tbody>
</table>
2- Key References used

- ISO/IEC 15504 (Spice)
- ISO/IEC 12207, Information Technology Software Life Cycle Processes
- ISO/IEC 14764, Software Engineering, Software Maintenance
- IEEE 1219, Standard for Software Maintenance

**Framework**

- Zitouni/Abran Software Maintenance Model
- Camélia model
- CM³ Corrective Maintenance Maturity Model
- ITIL Service Delivery and Service Support
- IT Service CMM
- Cobit
- Malcolm Baldrige

**CMMI® Software Eng. v1.1 Process Area/Specific Practice**

**Standards**

**Best Practice guides and Maturity Models**
2- Chosen based on criteria

<table>
<thead>
<tr>
<th>Model</th>
<th>Criterion 1</th>
<th>Criterion 2</th>
<th>Criterion 3</th>
<th>Criterion 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>CMMi [Sei02]</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>ISO9001:2000 Interpretation [Iso00, Llo01]</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Zitouni [Zit96]</td>
<td>Yes</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>CM³ [Kaj01]</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Bootstrap [Boo91]</td>
<td>No</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>TeleSpice and R-Spice [Esi98b]</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Trillium/Camélia [Cam94, Tri96]</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Checklist Testing Maturity Model [Vee02]</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Testing Maturity Model [Bur96]</td>
<td>No</td>
<td>Yes</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Maturity Model (Automated Testing) [Kra94]</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>No</td>
</tr>
<tr>
<td>Outsourcing Management Maturity Model [Raf02]</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>IT Service Capability Maturity Model [Nie04]</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Business-IT Alignment Maturity Model [Luf01]</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>IT Management, Control and Audit Maturity [Cob00]</td>
<td>Yes</td>
<td>Yes</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Change Management Capability Model [Baj98]</td>
<td>No</td>
<td>No</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Publicly available, detailed, alive, all industrie
Part 3 - Proposed process model
Standards, are consensus-based documents that codify best practice. Consensus-based standards have seven essential attributes that aid in process engineering. They:

- represent the collected experience of others who have been down the same road,
- tell in detail what it means to perform a certain activity,
- can be attached to or referenced by contracts,
- help to assure that two parties have the same meaning for an engineering activity,
- increase professional discipline,
- protect the business and the buyer,
- improve the product.
3- CMMi structure in S3m

- Contains the essential elements of effective processes for software related activities
- Contains a framework that provides the ability to generate multiple models and associated training and assessment materials. These models may represent:
  - software and systems engineering
  - integrated product and process development
  - new disciplines
  - combinations of disciplines
- Provides guidance to use when developing processes
3- **S3m context** (Scope)

- **Customers and Users**
  - Request
  - Status

- **Help Desk**
  - Service Level Agreement
  - Maintenance services
  - Support

- **Software Development**

- **Software Maintenance**
  - Development projects
  - Initial Transition

- **Suppliers**

- **Computer Operations**
  - Problem
  - Resolution communications

- **Otto-von-Guericke Universität Magdeburg**
3- $S3^m$ Maintenance Processes
3- Proposal to ISO/JTC1 SC7

- This process model was presented to ISO/IEC 14764 to be used to update the current process model. Their answer was:

‘This comment is out of scope with the NWI which is to merge the content of the IEEE 1219 into ISO/IEC 14764. This comment will be placed in a "deferred comment database" for the future revision.’
3- Alignment to CMMi Arch.

- CMMi Process Domains
  - Process Management
  - Project Management
  - Engineering
  - Support

- S3 Process Domains
  - Process Management
  - Maintenance Request Management
  - Evolution Engineering
  - Support to Evolution Engineering
3 - S3\textsuperscript{m} - Resulting KPA's

<table>
<thead>
<tr>
<th>S3 \textsuperscript{m} Process Domains</th>
<th>Key Process Areas of Software Maintenance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process Management</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1- Maintenance Process Focus</td>
</tr>
<tr>
<td></td>
<td>2- Maintenance Process/Service definition</td>
</tr>
<tr>
<td></td>
<td>3- Maintenance Training</td>
</tr>
<tr>
<td></td>
<td>4- Maintenance Process Performance</td>
</tr>
<tr>
<td></td>
<td>5- Maintenance Innovation and deployment</td>
</tr>
<tr>
<td>Maintenance Request Management</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1- Request &amp; Event Management</td>
</tr>
<tr>
<td></td>
<td>2- Maintenance Planning</td>
</tr>
<tr>
<td></td>
<td>3- Monitoring &amp; Control of maintenance requests</td>
</tr>
<tr>
<td></td>
<td>4- SLA &amp; Supplier Management</td>
</tr>
<tr>
<td>Evolution Engineering</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1- Predelivery and Transition</td>
</tr>
<tr>
<td></td>
<td>2- Operational Support Services</td>
</tr>
<tr>
<td></td>
<td>3- Software Evolution &amp; Correction Services</td>
</tr>
<tr>
<td></td>
<td>4- Verification and Validation</td>
</tr>
<tr>
<td>Support to Evolution Engineering</td>
<td></td>
</tr>
<tr>
<td></td>
<td>1- Configuration and Version Management</td>
</tr>
<tr>
<td></td>
<td>2- Process and Product Quality Assurance</td>
</tr>
<tr>
<td></td>
<td>3- Measurement, Decision Analysis</td>
</tr>
<tr>
<td></td>
<td>4- Problem Resolution and Causal Analysis</td>
</tr>
<tr>
<td></td>
<td>5- Software Rejuvenation, Migration and Retirement</td>
</tr>
</tbody>
</table>
Part 4 – Validation (Pre-Release and Post-Release)
Pre-release validation

- Formalise some components of software maintenance maturity model [Wang 2000]
- Use the notation proposed by [Dumke 2005]
- Software Product Maintained
- Software Maintenance Process/Activities
- Software Maintenance Resources
Software Maintenance Ontology

Software Maintenance

Software Product
- Size
- Age
- Maturity
- Composition
- Application type
- Quality caract.

Maintenance Processes
- Type
- Input
- Outcome
- Performance
- Quality caract.

Maintenance Resources
- Type
- Description
- Requirements
- Performance
- Quality caract.
Software product ontology
Software product formalization

At first, we define the software product as a (software) system as:

\[ SP = (M_{SP}, R_{SP}) = (\text{artefacts}, R_{SP}) = (\{\text{programs, documentations}\}, R_{SP}) \]  

where the two sets are divided in the following elements

- \( \text{programs} \subseteq \{\text{sourceCode, objectCode, macro, plugIn}\} \)  
- \( \text{documentations} = \{\text{userManual, referenceManual, developmentDocumentation}\} \)

and \( R_{SP} \) describes the set of the relations over the \( SP \) elements.
Software product formalization

The given subsets could be described as follows:

developmentDocumentation = \{documentationElements\} \quad (4)

=\{productRequirements, productSpecification, productDesign, implementationDescription\}

documentationElements \subseteq \{model, chart, architecture, diagram, estimation, review, audit, verificationScript, testCase, testScript, pseudoCode, extensionDescription, qualityReport\} \quad (5)

productRequirements = systemRequirement \subseteq \{functionalRequirements, qualityRequirements, platformRequirements, processRequirements\} \quad (6)

We define software products formally using this approach.
Maintenance Process Ontology

- Maintenance
  - Investigation Activity
  - Modification
    - Correction
    - Improvement
      - Eliminating Problems
      - Preparing Problems
      - Changing Implementation
  - Retirement
  - Changing Requirements
Maint. Process formalization

\[ SM = (A_{SM}, R_{SM}) = \{\text{maintenanceActivity, maintenanceResources}\} \cup SP \]  

Therefore, we can define the software maintenance resources \( SR \) as follows:

\[ SR = (M_{SR}, R_{SR}) = \{\text{peoplewareResources, softwareResources, platformResources}\}, R_{SR} \]  

\[ \text{peoplewareResources} = \{\text{maintainer, analyst, developer, customer, user}\} \]

We define software maintenance resources formally using this approach.
Maint. Process formalization

\[ SM = (A_{SM}, R_{SM}) = (\{\text{maintenanceActivities}, \text{maintenanceResources}\} \cup SP_M, R_{SM}) \]

Where:

\textit{maintenanceActivities} = \{\text{investigationActivity, modification, retirement}\} \quad (9.11)

\textit{modification} = \{\text{corrective, improvement}\} \quad (9.12)

\textit{improvement} = \{\text{adaptive, perfective, preventive}\} \quad (9.13)
Maint. Process formalization

The definition of a corrective service can be expressed as the requirements identified on a specific Support Request that describes what has to be corrected = \textit{correctiveRequirements} with the use of the operational corrective workflow resulting in a software that is corrected:

\[ r_{SM} \in R_{SM} \cdot SP_M \times \text{correctiveRequirements} \]

\[ \times \text{maintenanceCorrectionWorkflow} \rightarrow SP_M^{(corrected)} \] 

(9.15)

We define software activities, resources and product formally using this approach.
Typical Post-release validation

- **ISO/IEC 15504 (SPICE) - 1994**
  - 35 Case studies (20: Europe, 14: Pacific Rim and 1: Canada);
  - Questionnaires (3), rating forms & report (3);
  - Demographic Analysis and questionnaire analysis.

- **IT Service CMM (Dr. Niessink) - 2000**
  - 2 case studies (a quick scan, a 3 day on-site assessment);
  - Questionnaires analysis, KPA discussions.

- **CM³ Corrective maint. MM (Dr. Kajko-Mattsson) - 2001**
  - 17 case studies (14 non-ABB, 3 ABB);
  - Checked if the CM³ proposed processes are present or absent ex: 14/17 document their problem management process;
4 - Post-Release Validation

Last update with conclusions of this thesis

Camélia Maturity Model

1995

Zitouni & Abran model

Maintenance Experts

Trial in Canadian telcomm.

ISO/IEC 12207

1999

S3 \textsuperscript{m} v1.0 model

ISO/IEC 14764

IEEE 1219

V&V1

V&V2

Swebok

2005

Update of the S3 \textsuperscript{m} v2.0 model

Update of the S3 \textsuperscript{m} v1.5 model

CobiT Malcolm-Baldrige

2002

ITIL

CM\textsuperscript{3} IT Service CMM

V&V2

Case studies in Middle east telecomm.
Thank You
4 - $S3^m$ Experimentation

- Consolidated Maturity Profile for Level 1 and 2 maturity

![Consolidated Maturity Profile Diagram]
4 - Middle-East Case Studies

- **S3m (April)**
  - 4 model V&V steps since 1999;
  - Experiment with 3 Case studies (Telecommunications Cie);
  - Four 3 days assessments with assessment plan;
  - Use the model, 1 context form & 1 observation/problem report form
  - Output:
    - Consolidated Maturity Profile for Level 1 and 2 maturity;
    - Identification of 3 company process improvement projects;
    - Model improvement list.

- **3 Improvement Projects- 2000-2 (Published Results)**
  1) Maintenance Process Measurement;
  2) Definition of a Maintenance Service Level Agreement (SLA);
  3) Software Product Measurement (source code measures).
## Model strengths and weaknesses

<table>
<thead>
<tr>
<th>Rank</th>
<th>Comments</th>
<th>+/-</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>There are important areas that the model does not address (ITIL, Cobit and Malcolm Baldrige)</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>The model accurately portrayed the process state of the organization</td>
<td>+</td>
</tr>
<tr>
<td>3</td>
<td>The model was useful for identifying what has to be improved</td>
<td>+</td>
</tr>
<tr>
<td>4</td>
<td>Model has too many practices to be used in a small maintenance organization</td>
<td>-</td>
</tr>
<tr>
<td>5</td>
<td>Some KPAs have too many practices in order to achieve the level</td>
<td>-</td>
</tr>
</tbody>
</table>
Conclusions of research questions

1- Software maintenance is a specific domain of software engineering and its specific processes/activities are published in SWEBOK.

2- Current Standards lack specific maintenance processes. We have made a proposal to ISO/IEC 14764 for a new process model.

3- No maturity model proposal covers the entire set of software maintenance unique activities as proposed by SWEBOK.

4- We have presented the proposed architecture of a capability maturity model that could address the entire set of software maintenance unique activities.

5- We have validated the model and shown how the model was used in practice to support the improvement of software maintenance.
Future Work

- Full model release in a French Book - February 2006
- Discussions on the English/German versions
- Evaluation tool built by Msc student
- Knowledge Based to support training
- Progress is posted on my WEB site at:
  
  Http://profs.logti.etsmtl.ca/aapril/English/autres/index.html
SM Assess tool
SMAssess tool
SMAssess tool

Legend

N  Not achieved
P  Partially achieved
L  Largely achieved
F  Fully achieved
Practice inexistant
Related Publication


3- Internal Technical Report 02-001, Montréal, ÉTS Software Engineering Laboratory, 10-11-2002.

4- Internal Technical Report 02-002, Montreal, ÉTS Software Engineering Laboratory, 30-12-2002.

5- 13th International Workshop on Software Measurement - IWSM Montréal (Canada), 2003.


8- Preprint, Faculty of Informatic, University of Magdeburg, November 2004.
Related Publication


10- 8th European Conference on Software Maintenance and Reengineering (CSMR2004). Tampere (Finland), Mar. 24-26, 2004


12- International Conference on Software Measurement (IWSM/MetriKon 2004), Königs Wusterhausen, Germany, 2004;


Related Publication


Accepted
