

Doktorandentag 2005

(SM^{mm}) :
Software Maintenance Maturity Model


Alain April

Reiner Dumke

Otto von Guericke University of Magdeburg, Germany

Alain Abran

École de Technologie Supérieure de Montréal, Canada


 Université du Québec
École de technologie supérieure



Otto-von-Guericke
Universität Magdeburg

Overview

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

 Université du Québec
École de technologie supérieure



Otto-von-Guericke
Universität Magdeburg

2

Research questions



- 1- If software maintenance is a specific domain of software engineering 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- How can such a model 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; ISBG and UKSMA → 5 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, reengineering, reverse engineering and impact analysis

1 - Need for SM^{mm}



❖ CMM and CMMi focus

- ♦ Software Development and Maintenance **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 standards to find processes, activities and best practices
- 3 Look to Framework and SWEBOK 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 Find a site and conduct case studies
- 7 Review the content with Experts
- 8 Improve model as necessary

2- How to validate a mat. model



- ❖ 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, 3ABB);
 - ♦ Checked if the CM³ proposed processes are present or absent ex: 14/17 document their problem management process;
 - ♦ Checked 10 control activities as well.

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?



Year Software Engineering CMM proposals

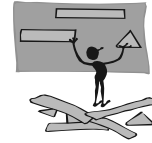
1991	Bootstap
1992	Trillium
1993	CMM®
1994	Camélia , automated testing (Kra94)
1996	TMM (Bur96), Zit96 , Dov96
1997	Som97
1998	Esi98, Top98, Baj98
1999	Wit99, Vet99, Sch99
2000	Cob00 , Str00, Bev00, Lud00
2001	Kaj01d & 01e , Ray01, Sch01, Luf01, Tob01, Sri01
2002	CMMi® , Nie02 , Mul02, Vee02, Pom02, Raf02, Sch02, Ker02, Cra02

2- Chosen based on criteria



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

Publicly available, detailed, alive, all industrie



Part 3 - Proposed software maintenance maturity model

3 - Standards used in SM^{mm}

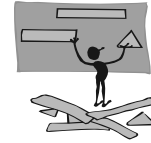


Standards, are consensus-based documents that codify best practice. Consensus-based standards have seven essential attributes that aid in process engineering. They:

- 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.

Source P.Croll: 14th Annual DoD Software Technology Conference - IEEE-Sponsored Track - 1 May 2002

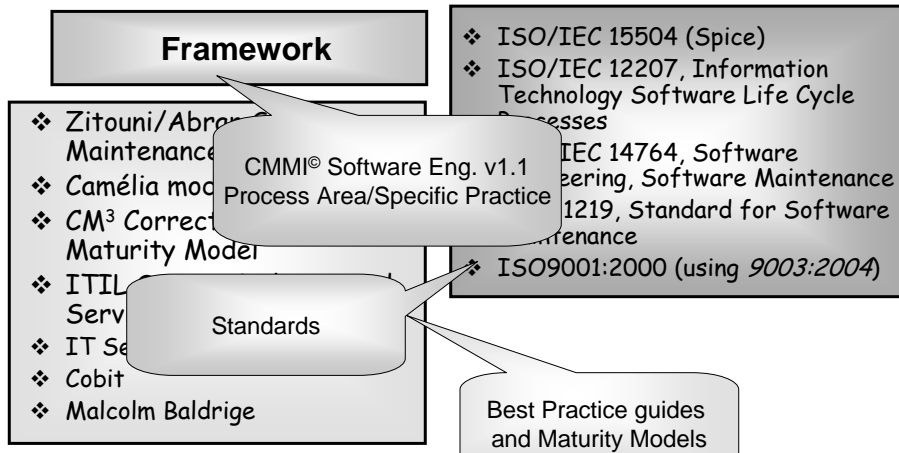
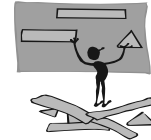
3- CMMi structure in SM^{mm}



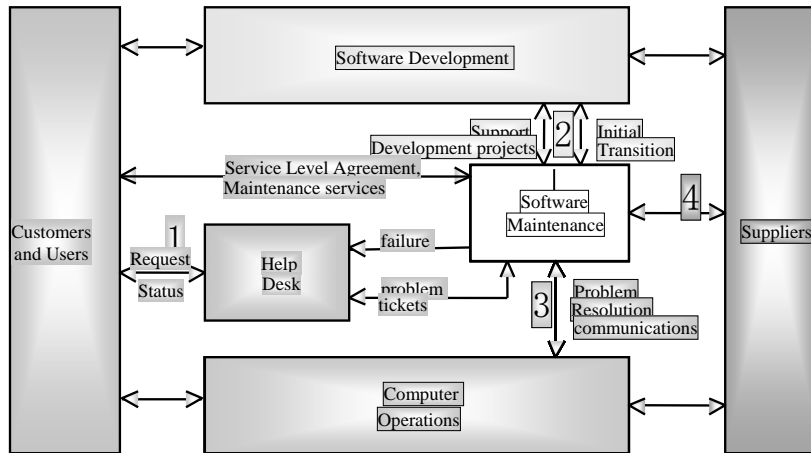
- ❖ Contains the essential elements of effective processes for software related activities
- ❖ Contains a framework that provides the ability to generate multiple models and associated training and assessment materials. These models may represent:
 - ◆ software and systems engineering
 - ◆ integrated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

Source P.Croll: 14th Annual DoD Software Technology Conference - IEEE-Sponsored Track -1 May 2002

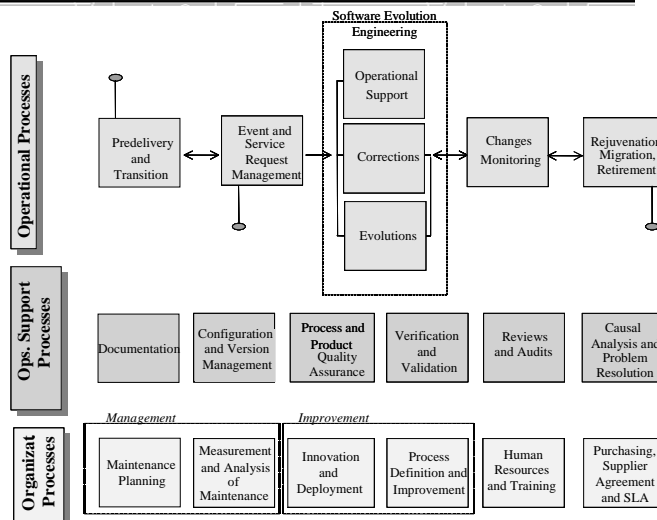
3- Sources to build SM^{mm}



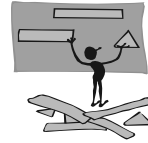
3- SM^{mm} context (Scope)



3- SM^{mm} proposed process model



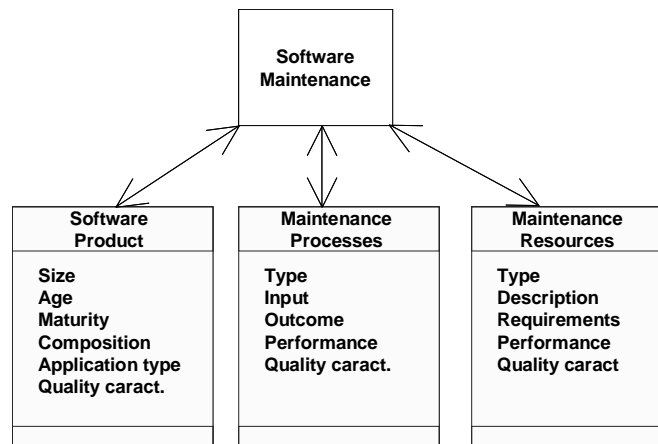
3- SM^{mm} proposed process model



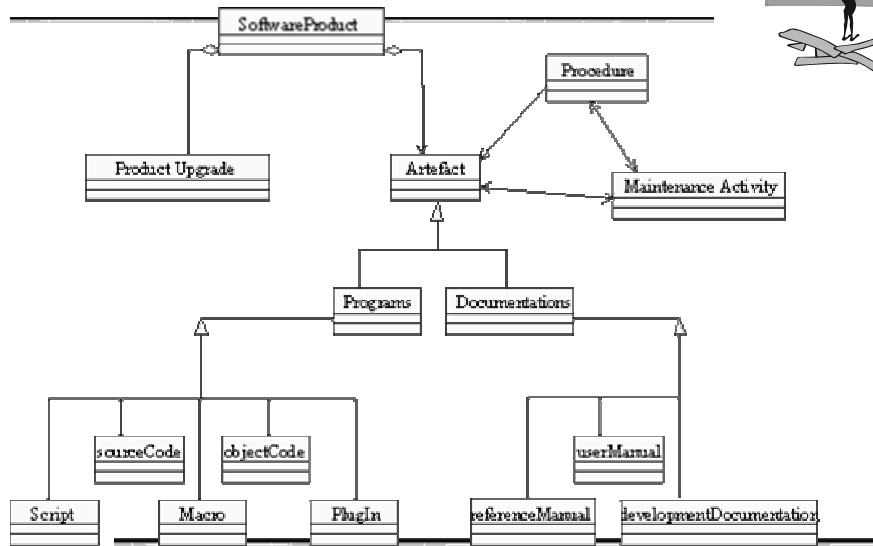
- ❖ The process model was presented to ISO/IEC 14764 to be used to update the current process model. Their answer was:

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. This revision will be started by a NWI proposal, subject to approval by SC7 and JTC 1.'

3- Influencing factors



Software product ontology



Software product formalization

At first, we define the software product as a (software) system as: $SP = (M_{SP}, R_{SP}) = (\text{artefacts}, R_{SP})$ (1)

$$= (\{\text{programs}, \text{documentations}\}, R_{SP})$$

where the two sets are divided in the following elements

$$\text{programs} \subseteq \{\text{sourceCode}, \text{objectCode}, \text{macro}, \text{plugin}\} \quad (2)$$

$$\text{documentations} = \quad (3)$$

$$\{\text{userManual}, \text{referenceManual}, \text{developmentDocumentation}\}$$

and R_{SP} describes the set of the relations over the SP elements.

Software product formalization



The given subsets could be described as follows:

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

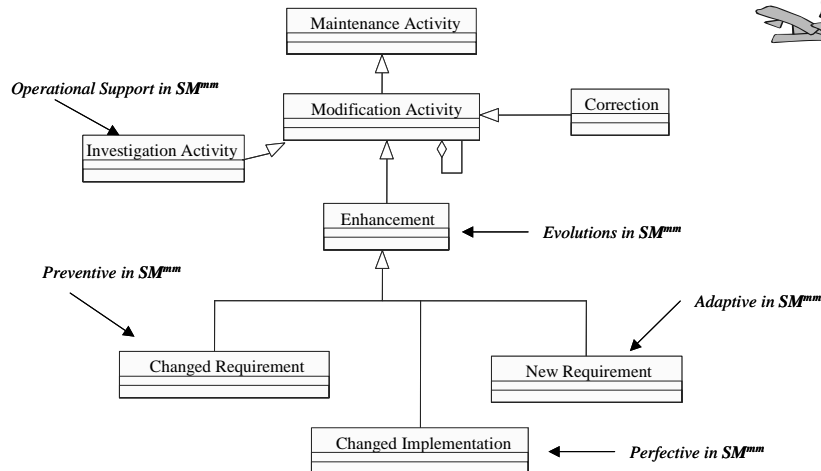
$= \{productRequirements, productSpecification, productDesign, implementationDescription\}$

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

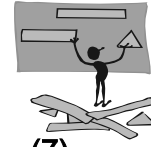
$productRequirements = systemRequirement \subseteq \{functionalRequirements, qualityRequirements, platformRequirements, processRequirements\}$ (6)

We define software products formally using this approach

Maintenance process ontology



Maint. process formalization



$$SM = (A_{SM}, R_{SM}) = (\{maintenanceActivity, maintenanceResources\} \cup SP) \quad (7)$$

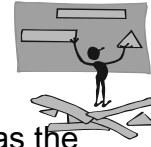
Where:

$$maintenanceActivity = \{modificationActivity, investigationActivity\} \quad (8)$$

$$modificationActivity = \{enhancement, correction\} \quad (9)$$

$$enhancement = \{adaptive, perfective, preventive\} \quad (10)$$

Maint. process formalization



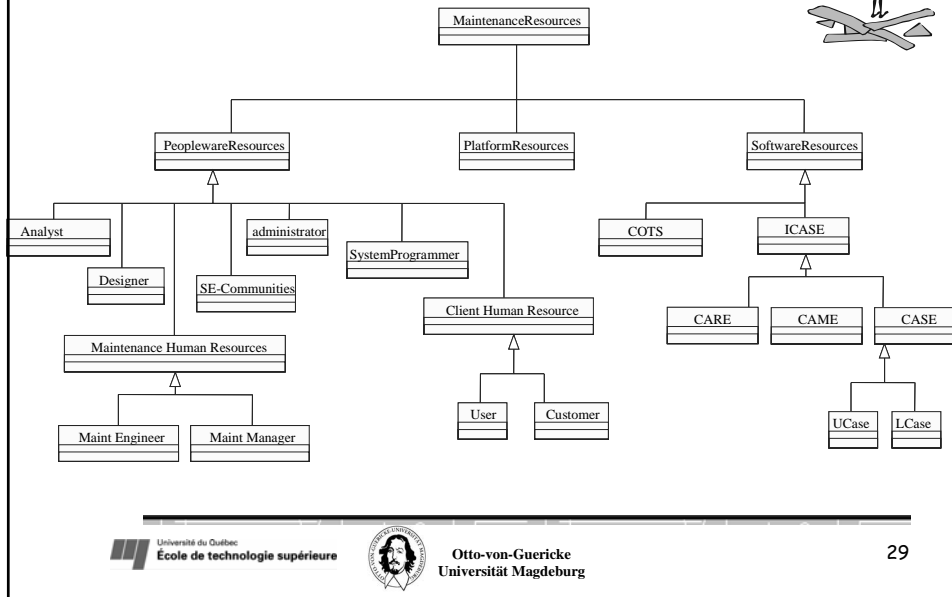
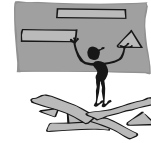
The definition of an investigation can be expressed as the requirements identified on a specific Support Request with the use of the operational support workflow resulting in a software that is investigated for this specific request:

$$a_{SM}^{(correction)} \in A_{SM}; SP \times SR_{supportRequirements} \quad (11)$$

$$\times operationalSupportWorkflow \rightarrow SP^{(investigated)}$$

We define software activities formally using this approach

Maintenance Resource Ontology



Model process formalization



$$SM = (A_{SM}, R_{SM}) = (\{maintenanceActivity, maintenanceResources\} \cup SP) \quad (7)$$

Therefore, we can define the software maintenance resources **SR** as follows:

$$SR = (M_{SR}, R_{SR}) = (\{peoplewareResources, softwareResources, platformResources\}, R_{SR}) \quad (12)$$

$$peoplewareResources = \{maintainer, analyst, developer, customer, user\} \quad (13)$$

We define software maintenance resources formally using this approach

Architecture alignment to CMMi



CMMi Process Domains	SM ^{mm} Process Domains
Process Management	Process Management
Project Management	Maintenance Request Management
Engineering	Evolution Engineering
Support	Support to Evolution Engineering

SM^{MM} - Resulting KPA's



SM ^{mm} Process Domains	Key Process Areas of Software Maintenance
Process Management	<ol style="list-style-type: none"> 1- Maintenance Process Focus 2- Maintenance Process/Service definition 3- Maintenance Training 4- Maintenance Process Performance 5- Maintenance Innovation and deployment
Maintenance Request Management	<ol style="list-style-type: none"> 1- Request & Event Management 2- Maintenance Planning 3- Monitoring & Control of maintenance requests 4- SLA & Supplier Management
Evolution Engineering	<ol style="list-style-type: none"> 1- Predelivery and Transition 2- Operational Support Services 3- Software Evolution & Correction Services 4- Verification and Validation
Support to Evolution Engineering	<ol style="list-style-type: none"> 1- Configuration and Version Management 2- Process and Product Quality Assurance 3- Measurement, Decision Analysis 4- Problem Resolution and Causal Analysis 5- Software Rejuvenation, Migration and Retirement

SM^{mm} - Overview

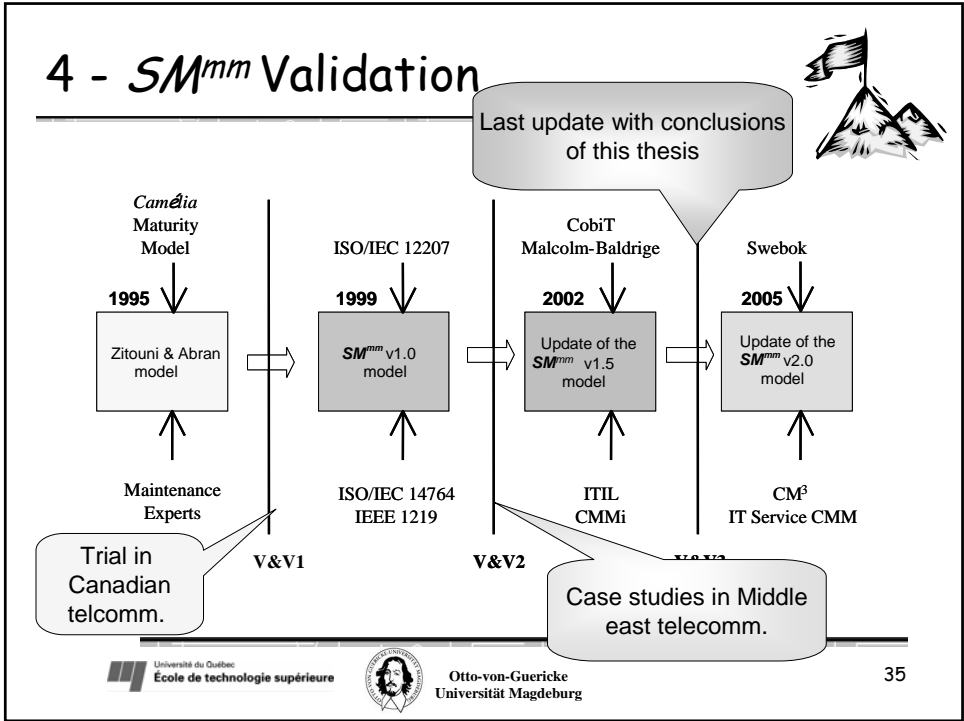


- ❖ Model in numbers
 - ◆ 4 Process Domains
 - ◆ 18 KPA's
 - ◆ 74 Roadmaps
 - ◆ 443 Practices with supporting text and references

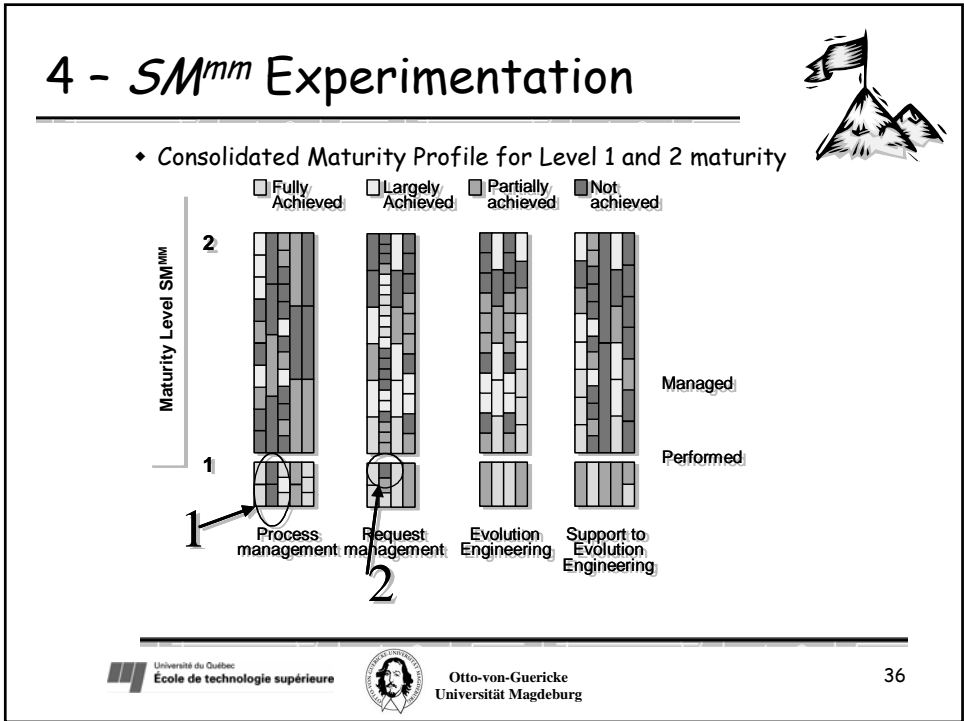


Part 4 - Case studies

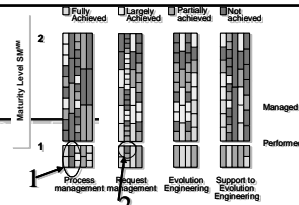
4 - SM^{mm} Validation



4 - SM^{mm} Experimentation



4 - SM^{mm} Validation



❖ SM^{mm} (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).

4 - SM^{mm} Evaluations



- ♦ Model strenght and weaknesses

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

Conclusions of research questions



- 1- Software maintenance is a specific domain of software engineering and its specific processes/activities are published in SWEBOK.
- 2- We are awaiting the ISO/IEC 14764 next NWI proposal on our process recommendations.
- 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 shown how the model was used in practice to support the improvement of software maintenance.



Future Work



- ❖ Full model release in a French Book - 2005
- ❖ Discussions on the English/German versions
- ❖ Evaluation tool built by Msc student
- ❖ Knowledge Based to support training 
- ❖ Are posted on my WEB site at:
<http://profs.logti.etsmtl.ca/aapril/English/Autres/index.html>





Thank
You



SMAssess tool

Choisir la langue à utiliser

1

Français
English
Deutsch

Suivant Quitter

Sélectionner les pratiques

3

- Domaines
 - Domaine1
 - KPA1
 - 1.1.0.1
 - 1.1.1.1
 - 1.1.1.2
 - 1.1.2.1
 - 1.1.2.2
 - 1.1.2.3
 - 1.1.2.4

Saisir les données de l'évaluation

2

Parrain

Relation du parrain à l'entreprise

Département à évaluer

Taille

Personnes

Objectif

Type d'évaluation

Quick Full

Date d'évaluation

(JJ.MM.AAAA)

Temps début de l'évaluation

(##:##M)

Suivant Précédent



SMAssess tool

Phase de réponse: choisir les réponses appropriées

DL_K1 | 01_K2 | 01_K3 | 01_K4 | 01_K5 | Analyse | 1

2

Niveau 0

1.1.01 Oui Non

Niveau 1

1.1.1.1 Oui Non

1.1.1.2 Oui Non

Niveau 2

1.1.2.1 N P L F

1.1.2.2 N P L F

1.1.2.3 N P L F

1.1.2.4 N P L F

1.1.2.5 N P L F

1.1.2.6 N P L F

1.1.2.7 N P L F

Niveau 3

1.1.3.1 N P L F

1.1.3.2 N P L F

1.1.3.3 N P L F

1.1.3.4 N P L F

1.1.3.5 N P L F

1.1.3.6 N P L F

1.1.3.7 N P L F

1.1.3.8 N P L F

1.1.3.9 N P L F

Niveau 4

1.1.4.1 N P L F

1.1.4.2 N P L F

1.1.4.3 N P L F

1.1.4.4 N P L F

1.1.4.5 N P L F

1.1.4.6 N P L F

1.1.4.7 N P L F

3 Pratiques: 1.1.01

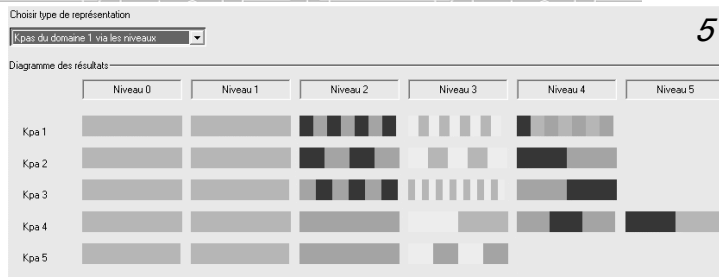
4 Commentaires de l'évaluateur

Descriptions

EST-CE QUE LA DÉCLARATION SUIVANTE DÉCRIT VOTRE SITUATION DE TRAVAIL... L'UNITÉ ORGANISATIONNELLE DE MAINTIENANCE DU LOGICIEL NE FAIT PAS D'ACTIVITÉ D'AMÉLIORATIONS STRUCTURELLES DES PROCESSUS MENANT À DES AMÉLIORATIONS DE PROCESSUS PÉRSISISTANTES ET CONTRÔLÉES. L'organisation n'a pas de processus de planification et d'implémentation de la qualité et une méthodologie de cycle de vie de la maintenance du logiciel. La direction des technologies de l'information ainsi que les dirigeants de la maintenance du logiciel ne reconnaissent pas qu'il y a...

OK Abbrechen Übernehmen Hilfe

SMAssess tool



Legend

- N Not achieved
- P Partially achieved
- L Largely achieved
- F Fully achieved
- Practice inexistant

Related Publication



- 1- *Software Measurement Conference (FESMA-AEMES)*, Madrid, Spain, October 18-20, 2000.
- 2- *Fourth European Software Measurement Conference (FESMA2001)*, Technologish Instituut vzw, Heidleberg, Germany, 2001.
- 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.
- 6- 11th International Workshop on Software Technology and Engineering Practice - STEP 2003, Amsterdam, Sept. 2003.
- 7- Maintenance Journal, February 2004.
- 8- Preprint, Faculty of Informatic, University of Magdeburg, November 2004.



Related Publication



- 9- (*IASTED 2004*) conference on Software Engineering, Innsbruck (Austria), Feb. 16-19, 2004.
- 10- *8th European Conference on Software Maintenance and Reengineering (CSMR2004)*. Tampere (Finland), Mar. 24-26, 2004
- 11- *Conference on Process Assessment and Improvement, Critical Software SA, (SPICE2004)*. Lisbon, Portugal, Apr. 27-99, 2004
- 12- International Conference on Software Measurement (IWSM/MetriKon 2004), Königs Wusterhausen, Germany, 2004;
- 13 - *Publication accepted in Journal of Software Maintenance and Evolution: Research and Practice* for early 2005 publication.

