Umsetzung der Vorgaben von CMMI mit Unterstützung durch Polarion

Dr. Ralf Kneuper
Beratung für Softwarequalitätsmanagement und Prozessverbesserung
Agenda

- Vorstellung des Verbesserungs- und Reifegradmodells CMMI
- Möglichkeiten und Grenzen der Werkzeugunterstützung
- Anforderungsmanagement - CMMI-Forderungen und Werkzeugunterstützung
- Test/Validierung der Anforderungen, Testmanagement – CMMI-Forderungen und Werkzeugunterstützung
- Projektplanung/-steuerung - CMMI-Forderungen und Werkzeugunterstützung
- Messung und Analyse - CMMI-Forderungen und Werkzeugunterstützung
- Konfigurationsmanagement - CMMI-Forderungen und Werkzeugunterstützung
Ralf Kneuper

Dipl.-Mathematiker, Univ. of Bonn
PhD Computing Science, Univ. of Manchester

1989-1995: Software AG
- Quality assurance, quality management, ISO 9000

1995-2005: Deutsche Bahn/TLC/DB Systems
- Senior consultant, project lead
- Quality management, internal CMM(I) consultant, development processes, project management

Since 2003: Independent consultant on CMMI

Former Speaker of GI SIG on software processes

SEI-Authorized CMM Lead Assessor, CMMI Lead Appraiser

Coordinator of the German CMM(I) Lead Appraiser and Instructor Board (CLIB)

Contact: ralf@kneuper.de
Underlying Premise of Process Improvement

“The quality of a product is largely determined by the quality of the process that is used to develop and maintain it.”

Based on TQM principles as taught by Shewhart, Juran, Deming and Humphrey.
Capability Maturity Model Integration for Development (CMMI-DEV)

Maturity model for the development of software and systems

- Used both for internal process improvement and as external proof of process maturity
- Defined improvement path

First published in 2002 as successor to CMM

- Author: Software Engineering Institute (SEI), Carnegie Mellon University in Pittsburgh, USA
- Current version v1.2 published in August 2006

Other „constellations“ of CMMI

- CMMI for Acquisition (CMMI-ACQ)
- CMMI for Services (CMMI-SVC)
The five maturity levels of CMMI

1. Initial
   - Unpredictable, little control
   - Project management

2. Managed
   - Can repeat earlier successes
   - Controlled process

3. Defined
   - Process documented, well understood
   - Product and process quality
   - Predictable process
   - Standardized consistent process

4. Quant. Managed
   - Process measured and controlled
   - Comprehensive development process

5. Optimizing
   - Process improvement
   - Managing change
   - Continuous process improvement

- Increased productivity and quality
- Reduced risk
<table>
<thead>
<tr>
<th>Process Mgmt.</th>
<th>Project Mgmt.</th>
<th>Engineering</th>
<th>Support</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>2</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Project Planning (PP)</td>
<td>Requirements Management (REQM)</td>
<td>Configuration Management (CM)</td>
</tr>
<tr>
<td></td>
<td>Project Monitoring and Control (PMC)</td>
<td></td>
<td>Process &amp; Product Quality Assurance</td>
</tr>
<tr>
<td></td>
<td>Supplier Agreement Management (SAM)</td>
<td></td>
<td>Measurement and Analysis (MA)</td>
</tr>
<tr>
<td><strong>3</strong></td>
<td>Organizational Process Focus (OPF)</td>
<td>Integrated Project Management (IPM)</td>
<td>Decision Analysis and Resolution (DAR)</td>
</tr>
<tr>
<td></td>
<td>Organizational Process Definition (OPD)</td>
<td>Risk Management (RSKM)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Organizational Training (OT)</td>
<td>Requirements Development (RD)</td>
<td></td>
</tr>
<tr>
<td><strong>4</strong></td>
<td>Organizational Process Performance (OPP)</td>
<td>Quantitative Project Management (QPM)</td>
<td></td>
</tr>
<tr>
<td><strong>5</strong></td>
<td>Organizational Innovation and Deployment (OID)</td>
<td></td>
<td>Causal Analysis and Resolution (CAR)</td>
</tr>
</tbody>
</table>
Tool Support for CMMI

- CMMI does not require any tools
  - It is difficult but not impossible to satisfy the CMMI requirements without any SW tools
- Tools can support the implementation and help to satisfy the CMMI requirements
- Tools cannot in themselves satisfy the CMMI requirements
  - If a tool is not used in the defined way (or perhaps not at all) it will not help with satisfying CMMI requirements
- To be truly useful, processes must be „institutionalized“ in order to ensure that they are used consistently
  - See generic goals and practices
Structure of CMMI V. 1.2
Staged Representation

- Maturity Levels
  - Process Area 1
  - Process Area 2
  - Process Area n

- Specific Goals
- Generic Goals

- Specific Practices
- Generic Practices
Generic Goal Level 2
Managed Process

Institutionalize a Managed Process (GG 2)

- Establish an Organizational Policy (GP 2.1)
- Plan the Process (GP 2.2)
- Provide Resources (GP 2.3)
- Assign Responsibility (GP 2.4)
- Train People (GP 2.5)
- Manage Configurations (GP 2.6)
- Identify and Involve Relevant Stakeholders (GP 2.7)
- Monitor and Control the Process (GP 2.8)
- Objectively Evaluate Adherence (GP 2.9)
- Review Status with Higher Level Management (GP 2.10)
Generic Goal Level 3
Defined Process

Institutionalize a Defined Process (GG 3)

- Establish a Defined Process (GP 3.1)
- Collect Improvement Information (GP 3.2)
Requirements Management

SG 1 Manage Requirements

- **SP 1.1** Obtain an Understanding of Requirements
- **SP 1.2** Obtain Commitment to Requirements
- **SP 1.3** Manage Requirements Changes
- **SP 1.4** Maintain Bidirectional Traceability of Requirements
- **SP 1.5** Identify Inconsistencies between Project Work and Requirements

Requirements documentation

Requirements traceability matrix
Why Traceability of Requirements?

Forward traceability:
- Ensure that all requirements are implemented in the work products building on them (design, code, test, etc)
- Helps to plan and track work
- Quickly identify impact of a change request

Backward traceability:
- Prevent „requirements creep“
- „Why do I create this product?“
- „What is affected if I leave out this test?“
Requirement Development

SG 1 Develop Customer Requirements
- SP 1.1 Elicit Needs
- SP 1.2 Develop the Customer Requirements

SG 2 Develop Product Requirements
- SP 2.1 Establish Product and Product-Component Requirements
- SP 2.2 Allocate Product-Component Requirements
- SP 2.3 Identify Interface Requirements

SG 3 Analyze and Validate Requirements
- SP 3.1 Establish Operational Concepts and Scenarios
- SP 3.2 Establish a Definition of Required Functionality
- SP 3.3 Analyze Requirements
- SP 3.4 Analyze Requirements to Achieve Balance
- SP 3.5 Validate Requirements

Customer Requirements

Product Requirements

Functional architecture (e.g. activity diagrams, use cases, etc.)

key requirements

Validated requirements
Supporting Requirements Management and Requirements Development Using the Polarion Toolset
Issues to Consider

- The Polarion toolset helps to manage the requirements and their traceability
- It cannot identify and understand the requirements for you
- It cannot know the relationships between requirements and project work. You have to identify and enter this information yourself so that the Polarion tools can then help you to manage it
- Maintaining traceability only becomes useful if this information is used, e.g. for identifying inconsistencies
Project Monitoring and Control

SG 1 Monitor Project Against Plan

- SP 1.1 Monitor Project Planning Parameters
- SP 1.2 Monitor Commitments
- SP 1.3 Monitor Project Risks
- SP 1.4 Monitor Data Management
- SP 1.5 Monitor Stakeholder Involvement
- SP 1.6 Conduct Progress Reviews
- SP 1.7 Conduct Milestone Reviews

SG 2 Manage Corrective Action to Closure

- SP 2.1 Analyze Issues
- SP 2.2 Take Corrective Action
- SP 2.3 Manage Corrective Action

Open issues list, corrective measures
Project Planning and Project Monitoring and Control in an Agile Environment

- Create a plan but adapt it when necessary
- Have a detailed short-term plan (e.g. covering one monthly Scrum sprint) plus a rough long-term plan
- Estimation is needed to plan a sprint / increment
- Plan and monitor in terms of user stories etc.

- Agile development usually monitors progress closely but with little defined measurements and little documentation
- CMMI requires more use of measurements for project tracking than usually used in agile development, and more documentation of the results

- Summary: agile development plus some add-ons can satisfy the CMMI requirements on PP and PMC
Supporting Project Planning and Monitoring Using the Polarion Toolset
Measurement and Analysis

SG 1 Align Measurement and Analysis Activities

- SP 1.1 Establish Measurement Objectives
- SP 1.2 Specify Measures
- SP 1.3 Specify Data Collection and Storage Procedures
- SP 1.4 Specify Analysis Procedures

Measurement Objectives

Analysis, interpretation of measurement data and reports

Metric Repository

Procedures & Tools (for collection and analysis)

Relevant stakeholder (data supplier or analyst, etc.)

SG 2 Provide Measurement Results

- SP 2.1 Collect Measurement Data
- SP 2.2 Analyze Measurement Data
- SP 2.3 Store Data and Results
- SP 2.4 Communicate Results
Measurement and Analysis

Project level

• Mainly used to keep project on track (→ Project Monitoring and Control)
• Typically measurements reported regularly in a status report, e.g. budget spent, earned value, etc
• Important to include measurements on ALL project goals, not just time and budget:
  – staff development
  – process improvement
  – customer satisfaction
  – ...

Organizational level

• Look at longer-term development of organization
• Typical example: Balanced Score Card
  – Financial view
  – Process view
  – Product view
  – Employee view
  – Customer view
Supporting Measurement and Analysis Using the Polarion Toolset
Issues to Consider

- By definition, a tool such as Polarion can strongly support the second specific goal for Measurement and Analysis, but provide very little support for the first specific goal.
- Remember to start from information needs, and then derive the relevant reports and measurements in Polarion, and not to start with the reports and measurements that are available but may not always provide the information you need.
Configuration Management

SG 1 Establish Baselines

SP 1.1 Identify Configuration Items

SP 1.2 Establish a Configuration Management System

SP 1.3 Create or Release Baselines

SG 2 Track and Control Changes

Configuration management system

Requirements database

Baselines (content & description)

Configuration items (CIs)

SG 3 Establish Integrity

SP 3.1 Establish Configuration Management Records

SP 3.2 Perform Configuration Audits

SP 2.1 Track Change Requests

SP 2.2 Control Configuration Items

Status (of CRs, CIs, baselines)

Audit results (non-compliances, actions)

CI = Configuration Item
CR = Change Request
A baseline

- is a consistent collection of work products
- forms the basis for further activities

This implies that a baseline

- is planned
- is reviewed for consistency, correctness and completeness
- is put under change control and must be kept consistent
  - therefore files that are not maintained should not be part of the baseline
- is more than a backup of the current project directory at a certain point in time

Typically, a baseline is created at major milestones such as the end of a development phase
Change Management can mean very different things

Manage changes at different levels:

• Change of requirements
  – in CMMI covered by REQM, SP 1.3
  – typically leads to changes of multiple work products
• Change of work products
  – in CMMI covered by CM, SG 2
• Change of own organization
  – training, rollout, etc.
  – in CMMI covered by generic goals and practices, OPF, OT
• Change of customer organization
  – training, rollout, etc.
  – not covered by CMMI but e.g. by ITIL
Supporting Configuration Management Using the Polarion Toolset
Further reading

