Why Isn’t Someone Coding Yet (WISCY)?

Avoiding Ineffective Requirements

Charlene Gross, Sr Member Technical Staff
Software Engineering Institute

Presented at the SEPG, May 2004, in Orlando, Florida
Agenda

- Requirements and Their Impacts
- Basic Requirements Definitions
- Comparison to Capability Maturity Model Integration® (CMMI®) Designations
- Requirements Development and Management
Didn’t We Solve the Requirements Problem?¹

- Sample of approximately 428 CMM-Based Assessments for Internal Process Improvement (CBA-IPI)

- Analyzed data from 1997 through August 2001

- Of the assessments conducted, only 33 percent fully satisfied the Requirements Management KPA

[Crosstalk, April 2002]
Didn’t We Solve the Requirements Problem?²

“[Disciplines for performance-based contracting to be successful] start with requirements definitions, and that takes a skill set. . . . . It’s a very difficult process to get a good set of requirements. There are not a ton of folks who are really good at that and you have to apply that very early in the process. That is the first discipline.”

--Ed Meagher, Acting CIO, Veteran’s Administration

Requirements and Their Impact

As Requirements Go, So Goes the Project
What is a Requirement?

- **Standard Definition**
  - Something that the product must do or a quality that the product must have

- **More Ways to Characterize**
  - Something you discover BEFORE YOU START TO BUILD YOUR PRODUCT
    --Robertson and Robertson (1999)
  - Agreement reached between the customer and the developers on what the system will do
The greatest control on software quality can be exercised during requirements phases.
[Stevens, 1999]

“Quality is conformance to requirements”
[Philip Crosby, 2000]

"Quality is conformance to requirements. Everything else is bull...."
[Forsha, 1992]
Requirements: A Project Foundation²

○ Planning Foundation
  - Clear and concise communication to all the team members
  - Alive and active throughout the lifecycle
  - Solution must reflect requirements

○ ROI Foundation
  - BASIS FOR EFFORT ESTIMATES and thus cost and profit
Size of the Problem

- 40 – 60% of errors in systems have been traced back to the requirements and analysis phase
- 70 – 85% of total revisions can be attributed to requirements errors

[Leffingwell, 1997]
68 times the effort is required to correct a post-production defect traced to an erroneous requirement than to correct the requirement in the requirements phase.

[Boehm, 1981]
Requirements Development
Traditional Requirements Categories

- Business
- User
- Functional
- Non-functional
Requirement Type - Business

Meaning:
- What the organization hopes to achieve
- The business benefits that the product will offer

Eliciting the Business Requirements:
- How will this project (product) improve the business or organization?
- What will you be able to do that you cannot do now?
Requirement Type - User

Meaning
- What the user requires to complete tasks
- Business rules, data representation requirements, logical models, and acceptance criteria that user will employ

Eliciting the User Requirement
- Tasks that need to be accomplished?
- Required business rules?
- Deciding if the new system/product is working?
## Requirement Type - Functional

### Meaning
- What software system should do
- What it does to have effect on outside domain

### Documenting the Functional Requirement

<table>
<thead>
<tr>
<th>Functional Requirement FR#</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority: (Select <strong>High</strong> if must have, <strong>Medium</strong> if Important but not Critical, <strong>Low</strong> if Nice to Have)</td>
</tr>
<tr>
<td>Description:</td>
</tr>
<tr>
<td>Related User Requirements UR#</td>
</tr>
<tr>
<td>Input information:</td>
</tr>
<tr>
<td>Output information:</td>
</tr>
</tbody>
</table>
# Requirement Type – Non-Functional

<table>
<thead>
<tr>
<th>Meaning</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standards</td>
<td>Enterprise Standards</td>
</tr>
<tr>
<td>Regulations</td>
<td>Government Regulations</td>
</tr>
<tr>
<td>Constraints</td>
<td>Platform</td>
</tr>
<tr>
<td>Interfaces</td>
<td>Legacy Interfaces</td>
</tr>
<tr>
<td>Quality attributes that affect how the</td>
<td>Usability</td>
</tr>
<tr>
<td>system must perform</td>
<td>Performance</td>
</tr>
<tr>
<td></td>
<td>Scalability</td>
</tr>
<tr>
<td></td>
<td>Security</td>
</tr>
<tr>
<td></td>
<td>Flexibility</td>
</tr>
<tr>
<td></td>
<td>Portability</td>
</tr>
</tbody>
</table>
Successful Requirements Development

- Place high emphasis on requirements -
  - About 15% of the project life should be spent on requirements development activities before any final deliverable is built. [Rubin, 1999]

- Use a variety of methods for obtaining requirements
  - Unstructured interviews – no particular format
  - Structured interviews – specific questions and format
  - Observation – view and record user actions
  - Brainstorming – facilitated or non-facilitated group elicitation

- Devise a consistent method for describing requirements
Elements of a Good Requirement

- Necessary
- Verifiable
- Feasible
- Clear and concise
- Complete
- Consistent
- Traceable
- No implementation bias

[Kar and Bailey, 1996]
Structure of a Requirement

- Keep sentences and paragraphs short
- Use active voice
- Use complete sentences with proper grammar, spelling and punctuation
- Use consistent wording
- Reduce ambiguity by avoiding vague and subjective terms
- Avoid comparative words and ambiguous language; quantify statement
Comparison to Capability Maturity Model Integration® (CMMI®) Designations
Model Overview

- **Capability Maturity Model (CMM®)**
  - Philosophy that quality processes enable quality products
  - Essential elements of effective processes for one or more bodies of knowledge
  - First CMM released in 1991 and targeted Software Engineering (SW-CMM)
  - Other discipline-specific CMMs created, e.g:
    - Systems Engineering
    - Integrated Product and Process Development
    - Supplier Sourcing
    - Others . . .
Model Overview²

Issues with multiple models

- Hampered ability to focus improvements where multiple disciplines present
- More costly in terms of training, appraisals, and improvement activities when applied within an organization

Solution

- An integration of three source models
- Addresses multiple disciplines
- Integrates training, appraisal support, and improvement activities
Model Overview³

○ Capability Maturity Model Integration® (CMMI®)
  ● Cohesive set of integrated models for organizations already using other CMMs, as well as by those new to the CMM concept
  ● More information at http://www.sei.cmu.edu/cmmi/
CMMI® and Requirements

- **Two principal process areas (PA)**
  - Maturity Level 2 – Requirements Management
  - Maturity Level 3 - Requirements Development

Purpose of Requirements Development

PA - produce and analyze:
  - Customer requirements
  - Product requirements
  - Product-component requirements
  - Derived requirements
CMMI and Requirements

Customer requirements

- An understanding of what will satisfy stakeholders
- Transformed stakeholder needs, expectations, constraints, and interfaces
- May be stated in technical or non-technical terms
- May also provide specific design requirements
CMMI and Requirements³

- Product requirements – a work product delivered to the customer
  - More detailed and precise sets of requirements
  - Expressed in technical terms or parameters
    - Functionality, including actions, sequence, inputs, and outputs
    - Qualities it must possess
    - Constraints that the system and its development must satisfy

[CMMI, Software Engineering Institute, 2003]
CMMI and Requirements

Product-component requirements – lower level components of the product

- Example - a car engine and a piston are product components of a car (the product)
- Allocated from product requirements
- Complete specification, including fit, form, function, performance, and any other requirement
- Sufficiently technical for use in the design of the product component
CMMI and Requirements\textsuperscript{5}

- Derived requirements – discovered and/or implied
  - Not explicitly stated but inferred from:
    - Customer requirements
    - Contextual requirements (e.g., applicable standards, laws, policies, common practices, and management decisions)
    - Contractual commitments such as data rights for delivered commercial off-the-shelf (COTS), and non-developmental items (NDIs); terms and condition, delivery dates, and milestones with exit criteria
CMMI and Requirements

Derived requirements (cont.)

- Factors arise as part of:
  - Selected architecture
  - Design decisions
  - Developer’s unique business considerations

- May also address the cost and performance of other life-cycle phases and other non-technical requirements
  - Training requirements
  - Site requirements
  - Deployment schedules
Relating CMMI Requirements Categories

Customer Requirements

Product Requirements

Product-Component Requirements

Derived Requirements

Discover & Allocate

Refine

Allocate
Relating Two Schemas - 1

**CMMI® Terms**
- Customer Requirements
- Product Requirements
- Product-Component Requirements
- Derived Requirements

**Traditional Terms**
- Business Requirements
- User Requirements
- Functional Requirements
- Non-Functional Requirements
Relating Two Schemas - 2

Customer Requirements

Product Requirements

Product-Component Requirements

Derived Requirements

Business Requirements

User Requirements

Functional Requirements

Non-Functional Requirements

Traditional Terms

CMMI® Terms

© 2004 by Carnegie Mellon University
Relating Two Schemas - 3

Customer Requirements

Product Requirements

CMMI® Terms

Product-Component Requirements

Derived Requirements

Business Requirements

User Requirements

Functional Requirements

Non-Functional Requirements

Traditional Terms

© 2004 by Carnegie Mellon University
Relating Two Schemas - 4

- Customer Requirements
- Product Requirements
- Product-Component Requirements
- Derived Requirements

Traditional Terms

- Business Requirements
- User Requirements
- Functional Requirements
- Non-Functional Requirements

CMMI® Terms
Implementing Requirements Management
Requirements Management Processes¹

- Change Control
- Version Control
- Requirements Tracing
- Requirements Status
- Requirements Measures
Requirements Management Processes

- Change Control Processes – controlling and authorizing changes
  - Documentation and baseline of requirements
  - Submission and documentation of changes
  - Impact analysis and negotiation with stakeholders
  - Change Control Board Infrastructure
  - Update and recording of disposition of change request
Requirements Management Processes\(^3\)

- Version Control Processes – ensuring correct version availability
  - Configuration management of requirements repository
  - Version maintenance and history throughout iterations
  - Designated read, write, delete and update permissions
  - Check In-Check out capability
  - Labeling and annotation schemas
Requirements Management Processes

- Requirements Tracing Processes – forward and backward requirements audit trail
  - Bidirectional linking to system elements
  - Capture of allocation rationale, accountability, and test/validation
  - Identification of inconsistencies
  - Capabilities to view/trace links
  - Verification of requirements
  - History of requirements changes

[Kean, 1998]
Requirements Management Processes\textsuperscript{5}

- Requirements Status Processes – status of activity on requirements
  - Categories for status, e.g., proposed, approved, implemented, verified, deleted, and/or rejected
  - Methods of tracking
  - Escalation standards
Requirements Management
Processes\(^6\)

- Requirements Measures – metrics for requirements activities and status
  - Requirements change requests – status, number, age
  - Number of requirements in a particular status category
  - Time spent on traceability and other requirements activities

[Weigers, 2001]
Requirements Management Tools

- **Database-centric**
  - Store all requirements, attributes, and traceability information in database
  - Examples are Caliber-RM, DOORS/ERS, RTM Workshop

- **Document-centric**
  - Treats word processing document as primary requirements container
  - May provide link to database or allow user to identify text as requirement
  - Examples are Requireit and RequisitePro
Managing Customer Expectations

A Bill of Rights and a Bill of Responsibilities
Origin and Importance

- Developed by Karl E. Weigers for his book, Software Requirements

- Delineates what customer should expect from project team

- Clarifies what customer needs to commit to providing to project team
Customer Bill of Rights

- Expect analysts to speak your language.
- Expect analysts to learn about your business and your objectives.
- Expect analysts to structure the information you present during requirements capture into a written software requirement specification.
- Have developers explain all work products created from the requirements process.
- Expect developers to treat you with respect and to maintain a collaborative and professional attitude throughout your interactions.
- Have developers provide you with ideas and alternatives both for your requirements and for implementation of the product.
- Describe characteristics of the product that will make it easy and enjoyable to use.
- Be presented with opportunities to adjust your requirements to permit reuse of existing software components.
- Be given good-faith estimates of cost, impacts, and trade-offs when you request a change in the requirements.
- Receive a system that meets your functional and quality needs, to the extent that those needs have been communicated to the developers and agreed upon.
Customer Bill of Responsibilities

- Educate analysts about your business and define business jargon.
- Spend the time it takes to provide requirements, clarify them, and iteratively flesh them out.
- Be specific and precise when providing input about the system’s requirements.
- Make timely decisions about requirements when requested to do so.
- Respect a developer’s assessment of the cost and feasibility of requirements.
- Set priorities for individual requirements, system features, or use cases.
- Review requirements documents and prototypes.
- Communicate changes to the project requirements as soon as you know about them.
- Follow the development organization’s defined process for requesting requirements changes.
- Respect the processes the developers use for requirements engineering.
Bibliography

Contact Information

Charlene C. Gross
Sr. Member Technical Staff

Software Engineering Institute
4301 Wilson Boulevard
Arlington, VA 22203
Phone Number - 703-908-8205
Email – cgross@sei.cmu.edu