

Constructive Quality Modeling for Defect Density Prediction: COQUALMO

Sunita Chulani

IBM Research, Center for Software Engineering

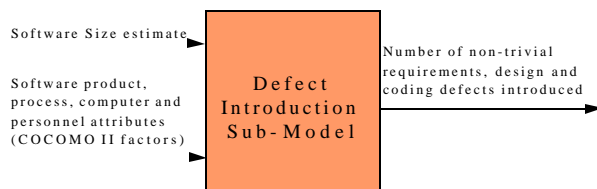
1-408-927-1767; sunita@us.ibm.com

The aim of this paper is to present COQUALMO, a quality prediction model. COQUALMO predicts the defect density of the software under development where defects conceptually flow into a holding tank through various defect introduction pipes and are removed through various defect removal pipes. COQUALMO consists of 2 sub-models, namely the 'Defect Introduction (DI)' and the 'Defect Removal (DR)' models. The DI model is formulated using product, process, computer and personnel attributes (based on COCOMO II, USC-CSE, 1999) and predicts the number of requirements, design and coding defects that are introduced during various activities of the development life cycle. The DR model captures the effects of 3 relatively orthogonal profiles of defect removal techniques, namely Automated Analysis, People Reviews, Execution Testing and Tools, and predicts the number of requirements, design and coding defects that are eliminated. The residual number of defects is the difference between the number of defects introduced and the number of defects removed. As discussed below, the model has been validated against published studies on defect densities and gives comparable results.

1. The Defect Introduction (DI) Model

Defects can be introduced in several activities of the software development life cycle and are classified based on their origin. The three types of defect artifacts of interest for this model are Requirements Defects, Design Defects and Coding Defects. The Defect Introduction Model has been summarized below.

Figure 1: The Defect Introduction Model



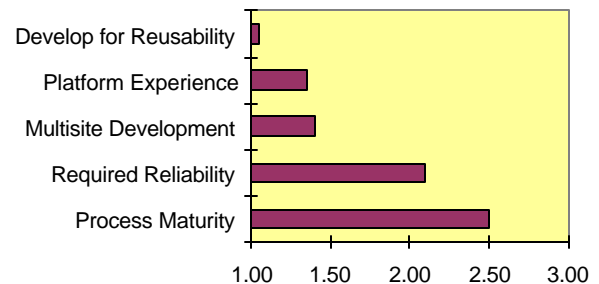
Non-trivial defects include **Critical** (causes a system crash or unrecoverable data loss or jeopardizes personnel), **High** (causes impairment of critical system functions and no workaround solution exists),

Medium (causes impairment of critical system function, though a workaround solution does exist).

Defect Introduction Range

The Defect Introduction Range for a particular attribute (product, process, computer or personnel) is the ratio between the highest DI rating and the lowest DI rating. Figure 2, for example, provides a graphical view of the impact of some of the DI drivers on the introduction of Coding defects. As seen in the figure, Process Maturity has a higher impact than Required Reliability, Multisite Development etc. The DI ranges for requirements and design defects will be presented at the conference with validation on how the author reached these conclusions.

Figure 2: Example Defect Introduction Ranges for Coding Defects



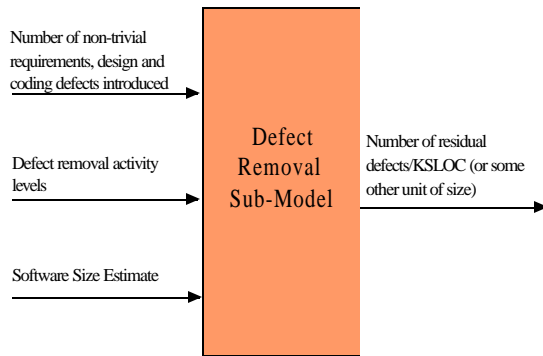
2. The Defect Removal (DR) Model

The Defect Introduction Model described in section 1 estimates the number of Requirements, Design and Coding defects introduced in the software product under development. The aim of Defect Removal Model is to estimate the number of defects removed by several defect removal activities. The defect removal activities are classified into three relatively orthogonal profiles with each profile having 6 levels of increasing defect removal efficiency, namely 'Very Low', 'Low', 'Nominal', 'High', 'Very High' and 'Extra High' with 'Very Low' being the least effective and 'Extra High' being the most effective in defect removal. Table 1 gives an example of the three profiles for three example rating levels, namely Very Low, Nominal and Extra High.

Table 1: Defect Removal Profiles

Rating	Automated Analysis	Peer Reviews	Execution Testing and Tools
Very Low	Simple compiler syntax checking.	No peer review.	No testing.
Nom-inal	Some compiler extensions for static module and inter-module level code analysis, syntax, type-checking. Basic requirements and design consistency, traceability checking.	Well-defined sequence of preparation, review, minimal follow-up. Informal review roles and procedures.	Basic unit test, integration test, system test process. Basic test data management, problem tracking support. Test criteria based on checklists.
Extra High	Formalized* specification and verification. Advanced distributed processing and temporal analysis, model checking, symbolic execution. *Consistency-checkable pre-conditions and post-conditions, but not mathematical theorems.	Formal review roles and procedures for fixes, change control. Extensive review checklists, root cause analysis. Continuous review process improvement. User/Customer involvement, Statistical Process Control.	Highly advanced tools for test oracles, distributed monitoring and analysis, assertion checking. Integration of automated analysis and test tools. Model-based test process management.

Figure 3: The Defect Introduction Model



3. Conclusions and Other Remarks

This paper describes COQUALMO, and its two sub-models, the Defect Introduction and the Defect Removal models. Initial results using the COQUALMO framework based on expert-judgment and early data-collection activities is presented. Validation of COQUALMO's output against published studies written by well-known figures in the Software Engineering community on observed defect densities gives very good matches. For example, when the author did a study using Jones data [Jones, 1998] with the SEI-published CMM-distribution of 807 organizations [SEI, 1999], she got an average defect density of 12 defects/kSLOC. This maps quite well into the residual defect densities estimated by COQUALMO; which says that a nominal project will have a residual defect density of approximately 14 defects/kSLOC.

4. References

- USC-CSE, 1999** - "COCOMO II Model Definition Manual", Computer Science Department, University of Southern California, Center for Software Engineering, Los Angeles, CA 90089-0781, 1999.
- SEI, 1999** - "Process Maturity Profile of the Software Community: 1998 Year End Update," Carnegie Mellon University, Software Engineering Institute's Technical Report, Mar 1999.
- Jones, 1998** - "The Impact of Poor Quality and Canceled Projects on the Software Labor Shortage," Capers Jones, Technical Report, Software Productivity Research, Inc.(an Artemis company), 1998.