A Model of Predelivery Problem Management

Mira Kajko-Mattsson, Eirini Tsotra

Abstract—Often, one connotes problem management with a postdelivery process for resolving problems within corrective maintenance. Very seldom, however, one relates it to the testing process within development, evolution and maintenance. In this paper, we propose a model of predelivery problem management. Using the model, we study the industrial status within eight companies situated in Greece. Our results show that all the organizations studied conduct a predelivery problem management process within system testing. However, only three out of eight companies perform it within central integration testing.

***Index Terms*—testing, change control, release, defect.**

I. Introduction

Often, one connotes problem management with a postdelivery corrective maintenance phase, during which one attends to problems as reported by the customers. Very seldom however, one relates it to the predelivery phase, during which problem management acts as a steering engine of the overall testing process.

Predelivery problem management plays an important role. It functions as a communication channel between engineers and testers. It also controls the testing process and provides an important feedback for decision making by various roles, such as project managers, quality managers, release managers, testers, and the like. Despite this, it has been little explored. To the knowledge of the authors of this paper, there are no process models whatsoever defined for this important activity. Software organizations do not have any standard model to follow in order to control their testing process via a predelivery problem management process.

In this paper, we suggest a predelivery problem management process model to be performed during the testing process. Our goal is to suggest an optimal model of how to control the testing process using problem management. Using the model, we then study the industrial status within eight companies situated in Greece.

The remainder of this paper is as follows. Section II describes our research method. Section III places problem management process within the development and evolution maintenance processes. Sections IV and V describe the predelivery problem management process model and its status within the organizations studied. Finally, Section VI makes conclusions and suggestions for future work.

Manuscript received December 30, 2007.

Mira Kajko-Mattsson is with the Department of Computer and Systems Sciences, Stockholm University/Royal Institute of Technology, Forum 100, SE-16440. Kista, Sweden. (Phone: +46-8-162000; fax: +46-8-7039025; e-mail: jaana@dsv.su.se).

Eirini Tsotra is with the Department of Computer and Systems Sciences, Stockholm University/Royal Institute of Technology, Forum 100, SE-16440. Kista, Sweden. (E-mail: mira@dsv.su.se).

II. Research Method

In this section, our research method is described. Section II.A presents the organizations studied. Section II.B describes the research steps taken during the study, and finally, Section II.C describes the scope and validity of our work.

A. Organizations

The companies involved in our research were experts in the fields of assurance, IT services, food industry, travel industry, car manufacturing and finance solutions. They were:

1. *Generali*: Greek branch of multinational assurance company [5],
2. *Toyota-Europe*: European branch of well-known car manufacturing company [13],
3. *Amadeus*: French branch of a leading provider of IT solutions in the travel industry [1],
4. *PWC*: Greek branch of assurance and advisory services multinational company having many well-known multinational companies as clients [11],
5. *Cronos:* Belgian company providing various services in the IT world [3],
6. *Cogmed Systems AB* Swedish software-based company [2],
7. *Unilever S.A.:* Greek member of the multinational group *ELAIS-Unilever S.A.* an expert in the food industry [4],
8. *Unilever Hellas S.A.:* Greek branch of the multinational group *Unilever S.A.* dealing with consumer products [14].

Regarding *Cronos*, it cooperates on some projects with *Toyota-Europe*. When receiving the responses from these companies, *Cronos* was working on an on-going project at *Toyota-Europe* and had responded on behalf of *Toyota-Europe's* IT department. For the credibility of our research results, we treat these two companies as two separate cases.

B. Method Phases

Our study consisted of the following steps; (1) *Literature Study*, (2) *Model Creation*, and (3) *Model Evaluation*.

Within the *Literature Study* phase, we searched for various printed materials dealing with problem management within testing. To our surprise, we found almost nothing describing

Table 1. Our questionnaire

Unit and Unit Integration Testing	**System Testing**
• Do you conduct unit and unit integration testing?	• Do you conduct system testing?
• Do you prepare test cases?	• Do you prepare tests cases for system testing?
• If you get a problem within testing, do you record it?	• At exactly what phase do you prepare test cases?
• If you record the problem, what do you use it for?	• If you have a problem, do you formally report it?
• Do you encounter any difficulties by *not* recording the information about the problem?	• If you formally report a problem, what do you do next?
• Do you believe that making notes will help you? If yes, why?	• Do you encounter any difficulties by *not* formally reporting the problem?
Central Integration Testing	• Do you communicate with the responsible developer in order to help him with reproducing the problem?
• Do you conduct central integration testing?	• Do you describe the problem to the responsible developer?
• Do you prepare central integration test cases?	• Do you agree that the application-software should follow the flow that is given in our predelivery problem management process model?
• If you have a problem, do you formally report it?	• If not, what is wrong with it?
• Do you encounter any difficulties by *not* formally reporting the problem?	• Do you think Formal Problem Management Report can help to improve the system testing process? If yes, in what way?
• Do you communicate with the responsible engineer in order to help him reproduce the problem? In what way?	
• Do you describe the problem to the responsible developer?	
• Do you believe that formal problem management process will help the testing process? If yes, in what way?	

this important process. The only material dealing with problem management were various articles on postdelivery problem management within corrective testing. The majority of them however, were written by the lead author of this paper [7-9]. Some information on problem management within testing was, however, slightly mentioned in one testing book [10].

With this state of art, we had to rely on our own experience of problem management within testing as elicited within two ABB organizations while creating *CM³: Problem Management*, a postdelivery process model to be used within corrective maintenance. Using this experience, we outlined a problem management process model and placed it on major testing process phases.

In order to evaluate the proposed model of problem management, we created an open-ended structured questionnaire. The questionnaire is presented in Table 1. Via email, we then sent the description of our model and the questionnaire to the companies studied. They provided us with their responses to the questions via email as well.

C. Scope and Sampling

Problem management within testing and problem management within corrective maintenance are different processes. To distinguish between them, we use the term *predelivery problem management* to refer to the problem management within testing and the term *postdelivery problem management* to refer to the back-end problem management process within corrective maintenance [7].

The predelivery problem management covers all the four standard testing phases: developers' testing, integration testing, system testing and acceptance testing. In this paper however, we limit our scope to only the first three testing phases. We exclude acceptance testing phase due to the difficulties of contacting the customers within the organizations studied. Also, in our testing process, we assume that the integration testing is conducted by an independent role, called *System Integrator*.

The choice of the organizations studied was made according to the convenience sampling method [12]. This means that we evaluated our model within the organizations that agreed to be studied. The small sample size and the convenience sampling method should not allow us to generalize our results. More studies need to be made to explore the domain of predelivery problem management.

III. Placing Problem Management within the Development and Evolution Processes

Although the problem management within testing and problem management within corrective maintenance are different processes, they are tightly related to each other. Figure 1 illustrates two main milestones within the development and evolution processes that are relevant from the problem management perspective. These milestones designate the point in time when these problem management processes start and end.

Before *Milestone 1 (M1* in Figure 1*)*, the system is not subject to formal change control. The functionality under development or change (evolution) is not complete and not fully tested. At *M1*, a component has reached sufficient functionality and stability to start the formal change control.

Between the milestones *M1* and *M2*, the system is subject to formal change control. At this phase, one conducts various levels of testing. Problems revealed during testing are then reported to the developers/maintainers via the predelivery problem management process dedicated to the testing phase. The predelivery problem management is a simple process,

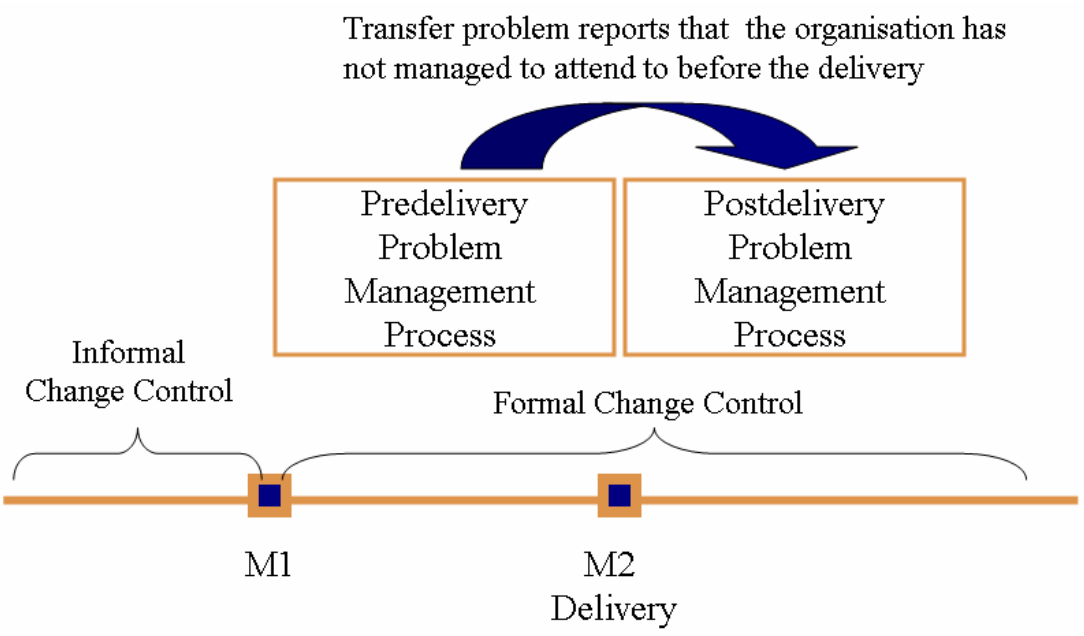

Figure 1. Placing the two problem management processes within the development and maintenance phases

during which testers report on problems in the defective components and developers attend to these components and supply testers with new corrected ones.

At milestone *M2*, the system gets delivered to the customer. However, not always all the problems discovered during the testing get resolved. The management of some of them will have to be postponed to the post-delivery problem management phase. In order not to loose track of these unresolved problems, they are transferred from the predelivery problem management process dedicated to testing to postdelivery problem management process dedicated to the postdelivery corrective maintenance phase.

IV. Predelivery Problem Management Process Model

Testing is an iterative process. As illustrated in Figure 2, predelivery problem management process is an integral part of an overall testing process. It starts at the developers' level and ends at the acceptance testing level.

The first phase is developers' testing. Here, the engineers should test their code before sending it for system integration. They should conduct both unit and unit integration tests [9]. Although these two tests result in the same process activities, they vary somewhat.

Unit testing encompasses testing of individual methods, irrespective of whether they are constituents of a class or not. It ensures that a specific method has successfully undergone a test. Unit integration testing, on the other hand, tests several units together, that have been developed by the same developer. It covers one or several units/classes that have been developed or changed by the responsible developer. The stress is put on the interfaces among the classes and functionality or sub-functionality as developed, evolved or maintained by the developer.

The predelivery problem management does not explicitly exist in this phase. However, in our model, we strongly recommend that developers make notes on various problems that they have encountered when testing and integrating their units. These notes help the developers to efficiently manage and control their problems and to learn new lessons.

As soon as engineers have written their code and tested it, they should send it for central integration testing. The successfully integrated and tested parts are then sent for system and acceptance tests.

Depending on the complexity of the system, the integration and system tests may be conducted on different system levels. For instance, components may be integrated into a system which, in turn, may constitute a subsystem to another system and so forth. Hence, one may need to conduct integration and system tests on different system levels.

All testing processes depicted in Figure 2 are iterative. These iterations may be conducted on several levels. From the developers' perspective, this means that developers iteratively test and modify their components before sending them for integration. This is the first iteration level.

The second iteration level is as follows. The software components, sent for central integration, may have resulted in problems. These problems are reported by the integrators to the developers via the predelivery problem management process. The engineers then attend to the reported problems and send the corrected code to the integrators. The reporting activity and the correction of code is part of the predelivery problem management process model.

The same procedure applies to system and acceptance testing. As soon as system or acceptance testers encounter problems, they should report them to the developers responsible for the problematic code. The developers should correct the code, and, the corrections should then be sent to the integrators. If successfully integrated, the system is then sent for system and acceptance testing.

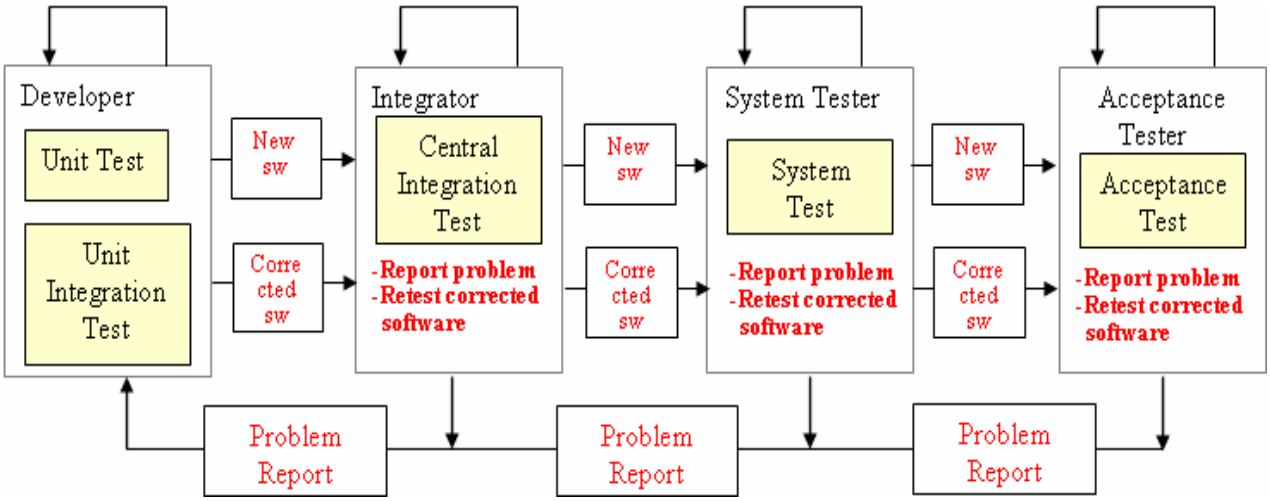

Figure 2. Predelivery problem management process within testing

From the predelivery problem management process perspective, the integrators and system testers should write formal problem reports and send them to the responsible engineers. These problem reports provide an important feedback for (1) the efficient management and control of all the problems encountered during testing, (2) efficient communication among all the roles involved within testing, (3) evaluation of the system quality, (4) decision making on whether and when to release, and (5) estimation of the remaining cost and time of testing.

V. EVALUATION RESULTS

In this chapter, we present and analyze the results as collected via the questionnaires. The results provide inside information of the state of predelivery problem management practice within the eight companies studied. When evaluating the status, we follow the order of the questions as stated in our questionnaire in Table 1. The evaluation results involving the three basic phases of testing (*Developers' Testing, Central Integration Testing and System Testing)*, are presented in Sections IV.A-IV.C, respectively.

A. Developers' Testing

Within six out of the eight companies, engineers conduct both unit and unit integration testing. The remaining two companies did not provide us with any information concerning this testing phase.

Regarding the problem management process, not all engineers make notes about the problems that they have encountered when testing their own components. Only five out of eight companies record a problem when it gets encountered at this testing level. Two of the three remaining companies only record major and more important problems.

The engineers mainly make notes in order not to forget the problem. They use it for planning their next-coming work and for tracking the coding and testing activity.

Five out of eight companies have had difficulties by not making notes about the problems encountered during the developers' testing process. One company has not provided us with any information about it. The remaining two companies state that they do not have any difficulties regarding this practice. The major difficulty that the developers encounter when not making notes concerns the planning of their own testing process and making decisions on when to stop testing.

Six companies believe that making notes at this testing level will help them (1) have a clearer understanding of their own testing process, (2) solve problems in a more efficient way, (3) retrieve past cases and apply their solutions, (4) observe problem/defect/error patterns and use them to improve future development, (5) possess a reference base for finding out what has been tested, where and how, and (6) identify bottlenecks within testing.

The respondents claimed that the above-listed benefits may only be gained if these notes are well organized and documented.

B. Central Integration testing

Six out of eight companies conduct central integration testing and prepare test cases for this testing phase. The remaining two companies have chosen not to reveal any information concerning this activity.

Regarding the problem reporting activity, only three organizations formally report on problems encountered during integration testing to the responsible developer. To facilitate the developers' debugging process, the integrators within these organizations record the testing context and action sequence in which the problem was revealed. Another three companies do not formally report on integration testing problems at all. Finally, the remaining two companies have not answered this question.

The companies that report on integration problems claim that formal reporting within integration testing is pivotal. Lack of it may lead to difficulties such as confusion due to misunderstanding of the integration problems and unrealistic feedback for the scheduling of the remaining testing activities.

In addition to formal problem reporting, the integrators within six organizations interact with the developers in order to solve the integration problems. The integrators do the following:

- Provide the developer with data that caused the integration problem. They do it usually via emails;
- Provide the developer with relevant examples, either orally or in writing;
- Explain the problem to the developer by walking through the integration testing and reproducing the problem, or by recording the context and sequence of testing actions during which the problem was detected.

Some problems cannot be easily explained. In this case, the integrators provide as detailed information as possible on the sequence of actions leading to the problem occurrence.

All eight companies, even the ones that have not implemented the formal problem reporting within integration testing phase, believe that formal problem management process, as proposed in this paper, is of great importance. According to them, it leads to the following benefits:

- Reduced misunderstanding of a problem;
- Better clarification of the problem;
- More efficient communication on problems to all relevant parties;
- Easier tracing of solutions to problems;
- More efficient problem resolution process;
- Assurance that all resolution actions have been done;
- Increased testing efficiency in the long run.

These benefits however must be weighed against the cost of the time consuming problem management procedures.

C. System testing

All eight companies conduct system testing in their organizations. They all create system test cases. The point in time when the test cases are created are the following:

- Early phase in the development/evolution cycle;
- Later phases of the requirements specification phase, when the usage of the application is well understood;
- After completion of central integration testing;
- Early stages of system testing.

All the organizations studied formally report on problems encountered during system testing. The reported problems may be assigned a criticality value by the system testers. What happens after the problem got reported by the system testers varies depending on the testing status and the character of the encountered problems. System testers may continue with the testing of other parts of the system while awaiting the corrections. When delivered, these corrected components (corrections) are retested to make sure that their problems got resolved.

The corrections delivered by the responsible developers are retested at the system level within all the organizations studied. However, only in one organization, the corrections are also retested at the integration testing level.

Some companies encounter some difficulties in cases when system testing problems do not get formally reported to the responsible developers. Lack of the reporting procedure may lead to confusion, discovery of ripple-effect problems in other parts of the software system, and delay of the whole testing process.

In all eight companies, the integrators communicate with the responsible developers in order to help reproducing the problems reported during the system testing phase. If the problems cannot be reproduced, the integrator explains the sequence of actions before running into the problem as detailed as possible.

When being asked about the opinion on the outline of our predelivery problem management process model and the flow of problem reports and corrections, six out of eight companies agreed upon it. The seventh company pointed out that the testing time and cost may be substantially multiplied with our model. The eighth company responded that for simple problems that can be resolved right away, the flow might be redundant. Still however, all the companies studied believe that formal predelivery problem management process can help improve the testing process in the following way:

- It leads to a fast solution;
- It helps all the relevant parties to continuously review the testing status;
- It helps reduce misunderstandings related to the reported problems;
- It assures that all required actions have been performed;
- It serves as a formal documentation for future use and referencing;
- It prevents similar problems from occurring;
- It increases testing efficiency in the long run.

As a general conclusion, the companies are positive to our proposed model as an approach to manage the overall testing process. Although it implies a cost in time and energy, it is however worth the effort.

VI. FINAL REMARKS

In this paper we have presented a predelivery problem management process model. Our model can be used as a guideline for how to efficiently drive the overall testing process and how to manage communication on problems among the various stakeholders involved in the testing process.

We have evaluated the process model within eight companies situated in Greece. Their main business domains are food industry, travel industry, car manufacturing, finance solutions and assurance. They are not pure software developing organizations.

The results achieved during the model evaluation show that the practice of performing predelivery problem management varies depending on the testing phase. While a predelivery problem management is used as a main driving vehicle for managing system tests, it is not as well exploited within the integration testing phase. Also, not all resolved problems that have been reported by system testers get retested at the integration testing level.

The results presented herein do not allow us to make generalizations about the predelivery problem management status. Due to the fact that the organizations chosen for this study do not have a software production as a main business driver, we will have to repeat a similar study in the context of

pure software producing companies. Still however, results presented in this paper provide an important indication that a predelivery problem management process is important as a steering engine of the overall testing process.

REFERENCES

[1] Amadeus, Internet site address, http:// www.amadeus.com/amadeus/amadeus.html , accessed on 2006/11/15.

[2] Cogmed Systems AB, Internet site address, http:// www.cogmed.com/cogmed/ , accessed on 2007/02/10.

[3] Cronos, Internet site address, http:// www.cronos.be/index.html, accessed on 2007/02/05.

[4] Elais-Unilever S.A., Internet site address, http:// www.elais.gr/unilever/unilever.jsp , accessed on 2007/01/28.

[5] Generali assurance, Internet site address, http:// www.generali.gr/index.php?lang=en&topmenuid=1&sidemenuid=1, accessed on 2006/06/06.

[7] Kajko-Mattsson, M, Problem Management Maturity within Corrective Maintenance, Journal of Software Evolution and Maintenance: Research and Practice, John Wiley & Sons, May/June, Volume 14, Issue 3, May/June, 2002, pp. 197-227.

[8] Kajko-Mattsson M., Evaluating CM^3: Problem Management, in Lecture Notes in Computer Science, Conference on Software Advanced Information Systems Engineering, Springer-Verlag, Volume 2348, 2002, pp. 436-451.

[9] Kajko Kajko-Mattsson M., Björnsson, T., Outlining Developers' Testing Process Model, In Proceedings, 33rd Euromicro Conference on Software Engineering and Advanced Applications, IEEE, Computer Society Press: Los Alamitos, CA, 2007.

[10] Kaner C, Falk J, Nguyen H Q, Testing Computer Software, John Wiley & Sons, Inc. 1999.

[11] PriseWaterhouseCoopers PWC, Internet site address, http:// www.pwc.com/extweb/home.nsf/docid/d6f13ee15d69057780256fe000437808 , accessed on 2007/01/30.

[12] Robson., C., Real World Research. Blackwell Publishing, 2002.

[13] Toyota-Europe, Internet site address, http:// www.toyota.eu/ , accessed on 2006/11/11.

[14] Unilever Hellas S.A., Internet site address, http:// www.unilever.gr/ , accessed on 2007/02/20.