Tailoring the Process for Automotive Software Requirements Engineering

Beatrice M. Hwong, Xiping Song

Siemens Corporate Research Inc.
755 College Road East
Princeton, NJ 08540

{ Beatrice.Hwong , Xiping.Song, } @siemens.com

ABSTRACT

When an automotive organization goes from being part of the OEM organization to being a supplier organization, several Requirements Engineering issues must be faced. The first is how the organization adapts to a new parent organization and its software development processes. Another is recognizing the need for more formal artifacts to better define the responsibility boundary between customer and supplier. One other change is adapting to working in a distributed fashion when software development is split among multiple locations, while manufacturing entails yet another location. We recently piloted an automotive software development group through this change and this paper will report on the lessons we learned.

Categories and Subject Descriptors

General Terms

Requirements Engineering, Process Tailoring.

Keywords

1. MOTIVATION

Software is increasingly providing the key differentiator in automotive products. The complexity resulting from embedding more software into automotive applications requires more effective yet managed development process and more systematic and formal requirements engineering.

We have had recent experience working with an automotive organization that moved from being part of the OEM organization to being a supplier to the OEM. Their primary product was instrument clusters for a major auto-maker with configuration and variation done by software. Since the software was an integration of the source code of the supplier with that of the customer, a cultural change was required to move to a process that defined a boundary between customer and supplier.

The group we were working with is a small software development team (5-6 staff) that needs to use a corporate-level software development process [3] from their new parent company. The business goal of using this process is to help them to be certified with certain CMMI level as required by their customer. Introducing Requirement Engineering into their practice is a major goal as well. This paper discusses our lessons learned from this project.

2. PROCESS CHANGE

The first step of the project is to tailor the corporate-level process that mainly aims at a large software development, to fit such a small project. Although the process was well supported by an internal consulting group with a web site rich with documentation and process tailoring guidelines, the group was new to software process and requirements engineering, and thus need further guidance for how to apply the process to their specific needs (e.g., to achieve CMMI[2]). In addition they also needed extra resources to help with document creation, reviews and change management as required by the new software process.

Since it was a small group, the first challenge was defining who would undertake the various roles defined by the process, roles previously done only informally (systems analyst, test manager, configuration manager, quality manager, etc.). In addition to doing software design and coding, each person had to take on these additional responsibilities. We took on the roles of configuration manager and quality manager. The group quickly decided that the project lead should take the requirements engineer role.

The next challenge was to tailor 300 process activities to fulfill the mandatory requirements of the overall corporate process without unduly burdening the small group while they were developing the software. We first iterated between a corporate recommended process tailoring list and a project document list to fulfill the mandatory steps of the umbrella process. In the Requirements Engineering area, there were a total of sixteen mandatory activities to consider.

For example, in the Software Requirements Analysis process area, the group rejected three out of ten mandatory activities with reason. For example, “Identify Software Safety Requirements’ was rejected since no hazards were identified. Since the group incorporated the release schedule into the project schedule, they decided there was no need to create a separate Software Release and Integration Plan.

The group at first resisted adopting two of the six mandatory activities in the Requirements Management area, related to requirement tracing despite our advice that CMMI level 2 would require it. They cited having no formal customer requirements specification and a lack of a requirements management tool. However, when they began to consider integration and system

International Automotive Requirements Engineering Workshop (AuRE'06)
0-7695-2710-8/06 $20.00 © 2006 IEEE

testing, later in the development lifecycle, they realized the value of documenting the traces between the requirements and testing documents. At one point, they thought that the Software Requirements Specification (SRS) could simply be an excel spreadsheet. The group later changed their minds, as explained below in the section Lessons Learned.

3. DISTRIBUTED DEVELOPMENT

Requirements and module specifications that specify the functionalities of each module play an important role for the distributed development within this project.

In addition to their own code, the group had to integrate with code from other sources. One source was the platform code for the microcontroller which came from the parent company in Germany. Another source was OEM supplied code, auto generated by an application level model. One more source was networking support code for the automotive communication bus. A new release of any of these code sources required re-synchronization.

At first all the members of the group were co-located in Alabama, but as it became apparent that more help was needed for design and testing, a designer from Michigan joined the team part-time and then a another person who would do the testing in the manufacturing location in Mexico joined. As supplemental help to the group, we were located in New Jersey and in a different time zone. We facilitated many document reviews among those distributed locations through teleconferences. About 10 module specifications were reviewed and helped clarify the key functionalities to be provided by each module. The clarification was very useful for the distributed team to establish common understanding.

Since manufacturing would take place in Mexico the group needed to define system testing with a well defined specification. A small, co-located group might not ordinarily need such formal documents, but with geography, time zones, and functionality imposing separations, these artifacts (e.g., SRS) were better than relying on non documented, private communications.

4. LESSONS LEARNED

What we have learned from this pilot process echoes what has been reported earlier [4] by an OEM organization. The current challenge is more in requirements management than in the requirements development. Automotive organizations, like many other complex engineering domains, are facing the evolving challenges of shifting from electro-mechanical to purely electronic components, where software becomes the differentiator.

Process, with appropriate tailoring, can indeed be fitted to a small group without undue burden. The resulting process from the tailoring is reusable for subsequent small group projects. Some tasks may be assigned to external consultants – e.g. configuration management, quality management, preparation and moderation of reviews. Requirements and testing were assigned to the core group since they had closest contact with the customer and manufacturing.

The checklists, templates, and example documents provided by the umbrella process were useful, but even more useful were existing specifications or reports that could be tweaked to be project specific. Since automotive designs are based on model years, some specifications were easier to create by modifying an existing one (e.g. base a model year ’08 design on the model year ’07 design). If a product line approach were used for developing the products, then the templates would be more effective.

A Software Quality report alerted the group to their deficiency in requirements, since the requirements were not fully implemented and the completion of the SRS was delayed. Since it was an engineering group, their initial focus was on module functionality, fulfilled by writing module specifications and implementing the code. But as the time to do first code integration with the OEM approached, their need to complete the Software System Test Specification (SSTS) made them recognize the value of requirements. Rather than using an excel spreadsheet for a SRS, a full requirements specification was created along with a traceability matrix so that the SSTS could be linked to the requirements and vice-versa. The group increasingly realized that documenting the traces between the requirements and testing as necessary. This was especially important at release/integration time when the code had to integrate with OEM code and they had the documentation to show what was implemented and how it fulfilled requirements with the test results.

Overall, we believe, based upon this experience, that the product line approach would help this automotive software development organization to reuse the software documents and code more effectively.

5. REFERENCES

[1] X. Song, G. Matos, B. Hwong, A. Rudorfer, and C. Nelson, S-RaP: A Concurrent Prototyping Process for Refining Workflow-Oriented Requirements. In *13th IEEE International Requirements Engineering Conference*, (2005), pp. 416-420.

[2] M. Chrissis, M. Konrad, and S. Shrum, “CMMI Guidelines for Process Integration and Product Improvement”, Addison-Wesley, 2003.

[3] SMK, Siemens internal documents and websites.

[4] M. Weber and J. Weisbrod, “Requirements Engineering in Automotive Development: Experiences and Challenges“, IEEE Software, Vol. 20, No. 1, January-February, 2003, pp. 16-24.

0-7695-2710-8/06 $20.00 © 2006 IEEE