Requirements for Flexible Software Development Processes

O. Jaufman, A. Dold, T. Haeberlein, C. Schlumpberger, M. Stupperich

Abstract — At present, software development in the automotive industry is characterized by frequent changes caused by new innovations, fast-growing system complexity, growing software portion in cars, changing business relationships. This dynamical environment demands for flexible software processes. In order to improve a software development process with respect to flexibility, it is necessary to characterize what kind of flexibility is required. Therefore, we defined a set of requirements for desired processes based on our process analysis in DaimlerChrys-ler's engineering departments and analysis of related contributions proposed in the literature. Based on this requirement the current processes can be analyzed to identify its improvement potential. The application of the requirements is illustrated in the context of a case study.

Index Terms — Process flexibility, requirements, c ase study, software development

1 INTRODUCTION

Nowadays, the automotive industry is confronted with a highly dynamic environment which is chara cterized by frequent changes due to innovations (e.g., new switch strategy for a gear), business rel ationships (e.g., new international collaboration), and new experiences (e.g., a project team follows a prescriptive process and recognizes that the process is not really efficient to perform module testing). One of the reasons for this trend is that software has come to play a more and more important role in the automotive industry and will be the major source of innovation in the future. In order to be able to quickly react to the changes in the development environment, flexible software develop ment processes are needed.

By a *software development process* we understand a set of partial causal and constrained activities performed by project team to produce software. The process constraints define milestones to deliver product artifacts to the customer, maturity level of the artifacts, and activities have to be performed to insure the quality of end product and satisfy the customer. The abstraction level of activities' specification depends on project team's needs.

By a *flexible* process we understand a process that allows the organization to react quickly and effectively to predictable and non-predictable changes in business und development environment.

A *quick reaction* is characterized by the ability to quickly identify

- whether a process adaptation (e.g., process refinement, remove of some constraints etc.) is needed
- what kind of process adaptation is required
- what the extent of the required process adaptation is
- what the strategy to perform the process adaptation is, and, accordingly, if a process adaptation is needed

For an *effective reaction* the benefit of performing the rea ction activities (1)-(4) must be higher than the effort required to perform the activities. The problem is that classical methods proposed for embedded software development (e.g., V-Model [3], RUP [1]) are not flexible enough for a frequently changing environment. Automotive companies usually follow plan driven methods for software development (e.g., for control devices). The consequence of the application of inflexible processes is that project teams are not able to quickly react to changes, therefore, project teams are pressed for time often resulting in fire-fighting and overtime. Therefore, there is a demand for more flexible processes to better support our engineering departments. To address this point, agile methods [1] have been considered. Unfortunately, agile methods do not clearly define how to deal with process constraints. In our environment, the process constraints are milestones to which artifacts are to be delivered, maturity of artifacts at given milestones, changes to be considered since a given milestones, management and communication structure etc. These aspects are important for long (e.g., 36 months) large (e.g., hundreds of people coming from several organizations) projects, with constant budget and constant ground functionality as it is the case in automotive industry. These aspects are important in order to better plan and control the project. The planning and control is necessary in order to be able in time to develop required functionality in given budget constraints.

Agile methods do not address the process constraints, because they usually aim on small project teams and assume the project team members being very experienced people who are able to very well plan and organize the irwork. In the large projects, in which very safety criticall software should be developed, there is no person which can in deep understand the work performed by all team members in order to be able to communicate with right persons

- *O. Jaufman is with the DaimlerChrysler AG, Ulm, HPC U800, E-mail: Olga.Jaufman@DaimlerChrysler.com.*
- *A. Dold is with the DaimlerChrysler AG, Ulm, HPC U800, E-mail: Axel.Dold @DaimlerChrysler.com.*
- *T. Haeberlein is with the DaimlerChrysler AG, Ulm, HPC U800, E-mail: Tobias. Haeberlein @DaimlerChrysler.com.*
- *C. Schlumpberger is with the DaimlerChrysler AG, Ulm, HPC U800, E-mail: Claudia. Schlumpberger @DaimlerChrysler.com.*
- *M. Stupperich is with the DaimlerChrysler AG, Ulm, HPC U800, E-mail: Michael.Stupperich@DaimlerChrysler.com.*

in write time, to deliever the right products of the right maturity to the right time. Therefore the *research question* arising is fundamental: What should a flexible software development process look like? In order to answer this question we investigate the requirements for process flexibility in this paper. This is done as following. Chapter 2 outlines the derivation of the requirements for process flexibility within the domain "automotive software". Related work used as input for the requirements definition is discussed at relevant places throughout the paper. In Chapter 3, the application of the requirements to a specific process is illustrated by means of a case study. Finally, Chapter 4 summarizes the most important contributions of the paper and addresses future work.

2 Requirements for Processes

Our main objective is to provide a specific set of requirements for needed process flexibility. In order to derive the requirements, four steps illustrated in Figure 1 are performed.

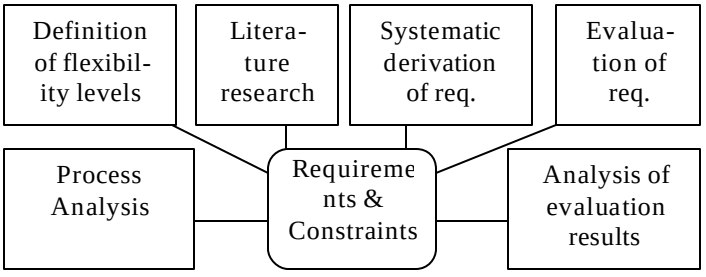

Fig. 1. Steps performed

Process Analysis software processes at DaimlerChrysler are analyzed in order to better understand why existing software development processes are considered as inflexible by the software development team. In addition, interviews with project teams were performed. We identified that project teams desire to be able to modify the prescriptive software development process during the project in order to be able to perform their work more effectively and efficiently. The reason for this desire is that it is not possible to decide what is the most effective way of performing the development and management activities at the beginning of the project. In addition, the project teams in our engineering departments desire to have their process more flexible at the start of the project, in order to adequately react to many changes coming at the start of the project. First, in the course of project, their software development process should be more stringent, because, empirically, there are lesser changes here. By a flexible process, the project team understands a process

- that provides a clear overview about the activities to be performed, if a change event (e.g., it is identified that developers are not able in time to remedy their faults) occurs.
- allows the project team to perform their work in the usual way as far as possible.
- fault tolerant (i.e., process is iterative and it is possible from each step to go back by using existing recovery mechanisms).
- corresponds to the terminology and the way of thinking of team members.

Definition of flexibility level: Based on results of the process monitoring, levels of process flexibility are defined and the flexibility level relevant for our specific software development process is identified. Figure 2 shows four levels of flexibility. Waterfall processes belong to the process class having "no flexibility" level (see curve 1). Here it is exactly known what should be done and no (or very few) change drivers (such as new gained experience, users' faults) are available. An example for such processes is an administrative process. The iterative processes, where the activities to be performed and the sequence of activities are exactly known belong to the process class having "adaptability" level. The terminology of such process activities and role concept are adaptable to a project team. An example for such kind of processes is a process for development of well known products.
Processes where for parts of the process it is not really known, what are the activities to be performed and what is the optimal order of the activities belong to the process class having "partial emergence" level of flexibility(see curve 3). Furthermore, the software development process usually becomes more stringent in the course of the project. This class of flexibility corresponds to the needs of DaimlerChrysler's engineering departments, because about 80% of the process aspects are predefined by the norms and standards for software development (e.g., SPICE [4], quality gates etc) and only about 20% of process aspects could be defined by the project team.

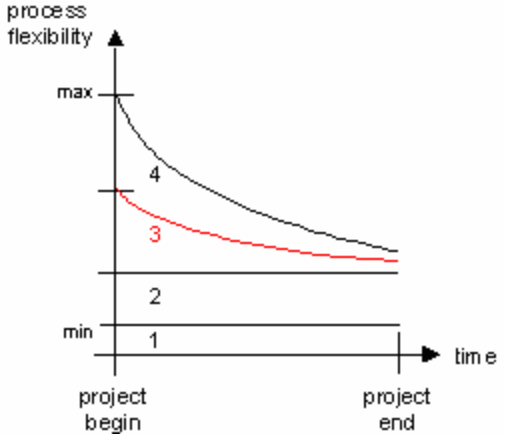

Fig. 2. Flexibility levels

Processes, were it is not really known at all, how the process should be performed belong to the fourth class "total emergence" (see curve 4). An example for processes having fourth flexibility level is the processes applied for development of innovation at research centers.

Literature Research: after we have determined what kind of flexibility we need in our company, the flexibility and

agility contributions provided in the literature are analyzed. Because it is widely assumed that agile methods (e.g., XP [1]) provide more flexibility than plan driven methods (e.g., V-Model [3], RUP [6]). The XP, the V-Model, and the RUP are analyzed with respect to flexibility. We found out that XP provides flexibility by focusing on time schedule and clear definition of roles and of responsibilities. The competence to decide about activities to be performed and artifacts to be delivered is subject to the project team members. In contrast to the XP, the V-Model prescribes activities to be performed and artifacts to be delivered. The V-model allows tailoring the activities to be performed and the artifacts to be delivered to a given context. Furthermore, the applicability of flexibility related aspects (e.g., change drivers, types of changes etc.) found in the literature [2], [7] to the context "software development processes" are investigated by means of a case study. These concepts are taken as an input for the systematical derivation of requirements related to process flexibility.

Systematic derivation of requirements and constraints for process flexibility are systematically identified following the GQM method [5] based on the knowledge gained in the previous steps. It is distinguished between requirements for process definition, process application, and domain constraints. The domain constraints are considered, in order to provide as far as possible realistic requirements on the processes in domain "automotive software development".

Requirements for process definition

Req. 1 - Priorisation
the process should distinguish between core activities (that must be performed), special activities (that should be performed), and nice to have activities (that can be performed). This is needed to be able to quickly react on time pressure.

Req. 2 - Hierarchy
the process should be modeled in a hierarchical way in order to be transparent.

Req. 3 - Traceability
the process should be traceable (vertical and horizontal), in order to be able to quickly understand what kind of process adaptation is needed, if a change happens. Vertical traceability means that the relationships between different abstraction levels of the process are clear. Horizontal traceability means that the relationships between different views on the process are clear (e.g., between a project manager view and a view of a tester)

Req. 4 - Views and Roles
the process should provide different views on the process (e.g., view for project manager, view for tester etc.) and clear responsibility concept, in order to be more transparent and understandable for a project team.

Req. 5 - Modularity
the process should be modular in order to support an easy and quick process adaptation. The process modules should have as much as possible cohesion. The number of dependencies between process modules should be as low as possible.

Req. 6 - Parameterization
the predictable process changes should be taken as parameter (e.g., number of verification activities varies depending on safety criticality of product component).

Req. 7 - Iterations
the process should be iterative in order to be able to quickly react to the change drivers (e.g., developers' faults).

Req. 8 - Tailor ability
the process should be tailored on the application domain, organization structure, terminology and the way of thinking of the project team in order to have acceptance by the project team.

Requirements for process application

Req. 1 - Learning
the process should support systematical learning from past experience and knowledge, in order to better (1) estimate risks, benefits, and costs of a process adaptation and (2) integrate gained knowledge and experience in the process.

Req. 2 - Communication
the process should support a good (i.e., open and honest) communication in the project team in order to quickly identify a need for a process modification and to correctly perform the needed modifications.

Req. 3 - Modeling Language
the process should be modeled by a modeling language that supports quick and easy adaptation.

Req. 4 - Tool Support
the process should be supported by a tool that allows to quickly adapt the process.

Req. 5 - Revision
the operational process should be revised each two weeks and if needed adapted to a new situation.

Req. 6 - Process Constraints
manager should define only "must" constraints on the software development process and let as much as possible developers' freedom to perform their operational work in desired way.

Domain constraints

Con. 1- Quality Gates
quality gates (i.e., milestones) to which product artefacts should be developed (e.g., by quality gate A lab tested software should be provided). The quality gates have to be considered, because quality gate method is used for automotive software development at DaimlerChrysler.

Con. 2 – SPICE Compatibility
activities to be performed must be compatible with SPICE standard [I98] (e.g., reviews, tests). The compatibility with SPICE is desired, because it is the actual standard in our context.

Evaluation of the requirements and constraints

In order to make sure that the theses meet the needs of project teams, interviews with 5 project managers, 2 quality manager and 6 developers from different engineering departments, and workshops with 16 experts in the domain "process design" are performed. This is done following the process shown in Figure 8.

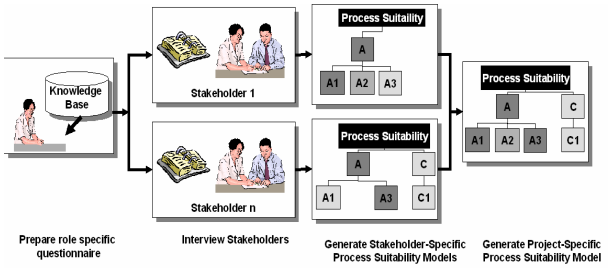

Fig. 3. Evaluation Process

The evaluation process consists of four main steps: (1) Prepare role specific questionnaires, (2) Interview stakeholders, (3) Generate stakeholder specific process models, (4) Generate project specific stakeholders process model.

The first step is to identify those stakeholders that are considered in the project specific software development processes. Potential stakeholders are project managers, quality assurance managers, developers, etc.

The second step is to interview individual representatives from the selected stakeholder groups. The objective of these interviews is to identify those theses for processes that are important from the viewpoint of the stakeholder group. Therefore, structured interviews are held, in which each stakeholder rates the importance of the requirements and constraints defined in the previous section. To support these interviews, the interviewer derives a questionnaire based on knowledge of the knowledge base. For each thesis its name and definition is given, as well as a 4-point-importance scale. An excerpt from such a questionnaire is shown in Table 1.

Table 1: Exerpt from the questionnaire

Thesis 1 – priories process activities	
Definition	the process should distinguish between core activities (that must be performed), special activities (that should be performed), and nice to have activities (that can be performed) with respect to a considered release product. This is needed to be able to quickly react on time pressure.
Evaluation	☐ Very important ☐ Important ☐ Somewhat important ☐ Not important at all

In order to detect whether the set derived from the knowledge base is complete, the interviewees are asked about missing theses that are to be considered.

Additionally, stakeholder-specific process models are generated. As multiple stakeholders' groups are interviewed, the individual viewpoints are integrated. The aggregated answers are visualized in a colored quality tree: the colors indicate whether requirements or constraints are rated as *very important*, *important*, or *somewhat important*. Theses that were rated as not important at all are not included.

Finally, we performed Steps 3 and 4 of the Evaluation Process: Based on the interview results we generated the stakeholder-specific and final process model. The final model is derived by aggregation of role specific constraints based on the arithmetic method. This is done as following: to each scale value a number is assigned: very important=3, important=2, somewhat important=1, and not important at all=0. Then for each requirement and constraint, the evaluation values given by stakeholders are added and divided by number of the values.

Fig. 4 and Fig. 5 show the stakeholder-specific models for the Project Manager and the Developer, respectively. Figure 12 shows the final model. (Due to space constraints these figures show only excerpts of the models.)

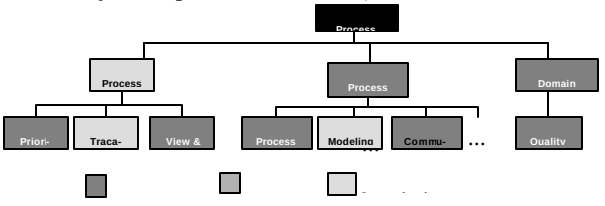

Fig. 4. Developer role-specific Process Suitability Model (Example)

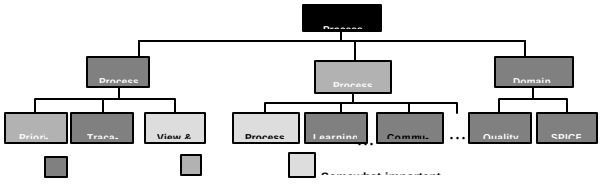

Fig. 5 Manager role-specific Process Suitability Model (Example)

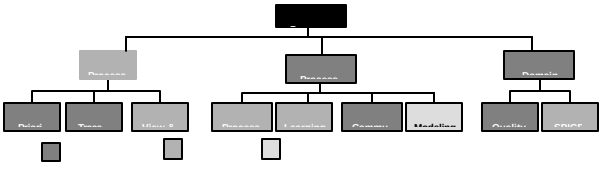

Figure 1: Project-specific Process Suitability Model (Example)

These models show for example that the developers' process theses related to process definition (e.g., provide different role and views, prioritize of activities) play a larger role than for the manager. Also it can be observed that the developer would prefer different process models than the project manager. These visualizations help to make different views explicit and allow a focused discussion on what kind of processes is desired.

3 A Case Study

In order to evaluate our requirements and constraints, we performed a case study in the context of Software Issue Tracking process applied at DaimlerChrysler's engineering departments. The issue tracking process consists of activities presented in Figure 3. The first activity in the process is to "capture" an issue. Here customer, supplier, or another authorized member of project team generates a request form for his issue and fills out the form. An issue can be a fault, a requirement's change, or a new requirement.

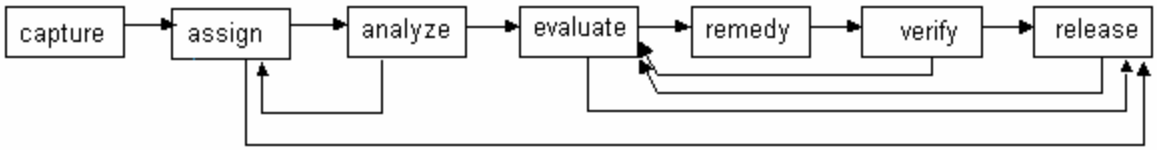

Fig. 1. Software issue tracking process

TABLE 2: EVALUATION RESULTS

	Requirements for process definition								Requirements for process application						Constrains	
	Req. 1	Req. 2	Req. 3	Req. 4	Req. 5	Req. 6	Req. 7	Req. 8	Req 1	Req. 2	Req. 3	Req. 4	Req.5	Req. 6	Con. 1	Con. 2
Importance of requirements	1,2	2	2,8	3	1	0.8	2,4	3	2,2	1,4	1,8	3	1,4	2,6	3	3
Process evaluation	-	-	o	+	+	o	+	+	-	o	o	+	+	+	+	+

After the form is filled out, a responsible system expert decides whether the issue is a software, electronic, actuator, or gear issue, and assigns the issue to personal responsible for the issue. If the issue is a software issue, a responsible expert analyzes the risk and the effort to remedy the issue. Based on this analysis, the change control board decides whether the issue should be remedied. If the decision is positive, a responsible developer remedies the issue. This implementation is verified by the system expert. If the implementation is correct, change management decides when the issue is to integrate in the system and close the issue.

The evaluation of issue tracking process is performed in two phases. In the first phase, the importance of the requirements and constraints is evaluated by users of issue tracking process based on the scale: very important (3), important (2), somewhat important (1), and not important at all (0). The aggregation of the importance values is performed by the method "arithmetic middle value". In the second phase, researchers evaluated the issue tracking process with respect to the requirements and constraints based on the scale (+: fully fulfilled, o: partly fulfilled, -: not fulfilled). Each evaluation is qualitatively argued as following: for example first requirement is not fulfilled, because issue tracking process does not distinguish between core activities (that must be performed), specific activities (e.g., that should be performed in order to be conform to third SPICE level), and nice to have activities. The results of this evaluation are visualized in Table 1 and are explained as following.

Table 2 shows the requirements for process definition in the second row of the first top line, the requirements for process application in the third row of the first top line, and finally, the domain constraints in the fourth top line. In the third line, the importance of the requirements and constraints for issue tracking process from process users' point of view is presented. In the fourth line, the evaluation' results of the issue tracking process with respect to requirements and constraints are presented.

Based on the evaluation results, the process improvement potential needed from process users' point of view can be identi fied. For example, hierarchical process modeling and explicit learning from past experience seem to be important and not fulfilled by the process.

4 SUMMARY AND FUTURE WORK

Present software development is characterized by frequent changes caused by new innovations, fast-growing system complexity, growing software portion in cars, changing business relationships. So, in order to develop software effectively, more flexible processes are needed. Ignoring process flexibility can make software developers not follow the prescriptive process, because it does not reflect the most effective and efficient way of performing and supporting the development.

In order to improve the process flexibility, explicit requirements for the process flexibility are needed. In this paper, we have presented and discussed a set of process flexibility requirements together with constraints which are specific for the automotive industry. This set is not to be considered as "a golden set of requirements" but rather as knowledge that should be adaptable on the specific context. In order to make such an adaptation easier, a case study is performed.

In a similar manner, existing software development processes can be analyzed, in order to identify the realization ideas on how a flexible process should look like. Based on the ideas, improvements for the software development processes performed at DaimlerChrysler's engineering departments will be proposed.

REFERENCES

[1] Boehm, B. and Turner, R., Balancing Agility and Discipline: A Guide for the Perplexed, Addison Wesley, 2004.
[2] Bylesjo, H. Modeling Output Flexibility in process industry, Performance Measurement Conference, 2000.
[3] Droschel, W., Wiemers, H. Das V-Modell 97, Der Standard für die Entwicklung von IT-Systemen mit Anleitung für den Praxiseinsatz, Oldenbourg, 2000.
[4] ISO/IEC TR 15504-2:1998(E), ISO Standard. Information technology — Software process assessment — Part 2: A reference model for processes and process capability, ISO/IEC, 1998.
[5] Rombach, R. Practical benefits of goal-oriented measurement. In N. Fenton and B. Littlewood, editors, Software Reliability and Metrics. Elsevier Applied Science, London, 1991.
[6] Versteegen, G. Projektmanagement mit dem Rational Unified Process, Springer Verlag, Berlin, 2000.
[7] Wadhwa, S. Flexagility and Agility For Enterprise Synchronization: Knowledge and Innovation Management Towards Flexagility, Studies in Informatics and Control, Vol. 12, No. 2 June 2003.