Proceedings of DET2007
4th International Conference on Digital Enterprise Technology
Bath, United Kingdom
19-21 September 2007

TRANSFERRING SELECTED SOFTWARE ENGINEERING PRINCIPLES TO THE COMMERCIAL VEHICLE DOMAIN

Jan C. Aurich
University of Kaiserslautern
aurich@cck.uni-kl.de

Nico Wolf
University of Kaiserslautern
wolf@cck.uni-kl.de

Jürgen Münch
Fraunhofer IESE
juergen.muench@iese.fhg.de

Andreas Grzegorski
University of Kaiserslautern
grzegorski@cck.uni-kl.de

Axel Wickenkamp
Fraunhofer IESE
axel.wickenkamp@iese.fhg.de

ABSTRACT

Quality criteria such as high reliability or proper diagnosability play an increasing role during the entire lifecycle of automotive and commercial vehicles. Therefore, setting up and adjusting appropriate defect detection and prevention mechanisms is a key element to assure such quality attributes. One challenge is to systematically select, combine, and balance defect detection and prevention mechanisms in order to guarantee a certain quality level under various constraints such as cost limitations. This article presents an approach for transferring selected software engineering principles to the commercial vehicle domain in the context of a continuous quality improvement paradigm. Additionally, the article gives recommendations on how to apply these software engineering principles in practice. Finally, lessons learned about the application of the software engineering principles in the commercial vehicle domain and related work are sketched.

KEYWORDS

Quality Assurance (QA), Quality Improvement Paradigm (QIP), Defect Detection, Defect Flow Model, Orthogonal Defect Classification

1. INTRODUCTION

Present challenges in commercial vehicle development and production are driven by an increase in the distribution of development and production processes (concurrent with a decrease in the vertical range of manufacturing), increasing interconnectivity of EE systems, and the need for multidisciplinary collaboration (e.g., electric/electronics, mechanics, software). Leading companies in the commercial vehicle industry are currently struggling for market shares in all relevant markets around the globe. This situation results in an ongoing industry consolidation as well as in multinational development and production networks. The latter finally leads to processes and process chains in product development, production planning, and production across plant and country borders. The ability to control such complex systems has become a decisive factor in competition. The same applies to the increasing integration of supply companies into the product development process. A continually rising number of interfaces between plants, companies, countries, or cultural regions entail the risk of upcoming information asymmetries or even losses. As a result this situation could lead to “stand-alone-solutions” if requirements regarding the entire system are not defined early and communicated to the suppliers. Furthermore, this could affect the diagnosability (i.e., the capability to be diagnosed) of the entire system as well as that of its components. Due to distributed development processes and the resulting differences in work progress, reliability verifications of depending components might not be possible. The stress of competition enforces rapid development. This is due to a much shorter time to market for the launch of innovative products.

The commercial vehicle industry is notably struck by recent trends towards stricter legislation aimed at improved environmental conservation. In order to meet the topical goals of exhaust and noise emission standards, completely new engine generations have to be developed. Besides allocating the tremendous development costs to high production numbers, cost pressure requires a rethinking of how much money should be spent on development and production activities and how much effort should be spent on quality assurance activities and, more specifically, on which of them.

Quality aspects (such as reliability and diagnosability) have to be carefully considered throughout the whole lifecycle, and proper quality assurance strategies have to be applied in order to cope with these challenges. One interesting question is if it is possible to learn from other domains about how they cope with these challenges.

As in many engineering disciplines, defect prevention and detection across the whole lifecycle are key quality assurance activities in the commercial vehicle domain. However, mature concepts for deploying a lifecycle-oriented quality assurance strategy in the context of distributed, multi-discipline development are widely missing. Questions that should be addressed by such a strategy include "how much" of each quality assurance activity should be performed at what point in the process or what the current defect baseline is.

A domain where this problem has been addressed for a long time is software engineering. Due to the observation that finding defects early in the development process avoids high rework costs, many principles, methods, and tools have already been developed to support defect prevention and detection as well as to balance quality assurance activities with respect to cost, resulting quality, and schedule. It could be promising to analyze if and how principles and methods of the software engineering domain are transferable into the commercial vehicle development domain.

Taking into account the situation described above, conventional strategies to ensure reliability and diagnosability show deficits. Thus, a holistic view on quality aspects in the area of commercial vehicle development is becoming more and more important.

This article is structured as follows: Section 2 sketches quality assurance approaches in both domains, the commercial vehicle domain and the software engineering domain, and describes examples from the software engineering domain (i.e., defect flow models, defect classification techniques) in more detail. Section 3 presents an approach for transferring selected software engineering principles to the commercial vehicle domain in the context of a continuous quality improvement paradigm. Section 4 gives recommendations on how to apply these software engineering principles in practice. Finally, lessons learned about the application of software engineering principles in the commercial vehicle domain and related work are sketched.

2. QUALITY PLANNING

Quality planning intends to assure both quality and reliability. Before identifying, classifying, and assessing quality features, quality planners therefore have to derive these features from every single product requirement in as much detail as possible. (Grothe, 2004)

2.1 QUALITY ASSURANCE IN MECHANICS

Quality planning relies on various quality assurance techniques covering a variety of different product lifecycle (PLC) phases. Quality assurance techniques usually focus on product-related quality issues with regard to the product development process chain, the production process chain, as well as the utilization. (Hering, 2003) Well-known and established techniques are Quality Gates, Quality Function Deployment (QFD), and the Failure Mode and Effects Analysis (FMEA) (Pfeifer, 2002). Furthermore, Statistical Experiment Methods (SEM) or supplier assessments focus in particular on process planning activities ➔ Figure-1.

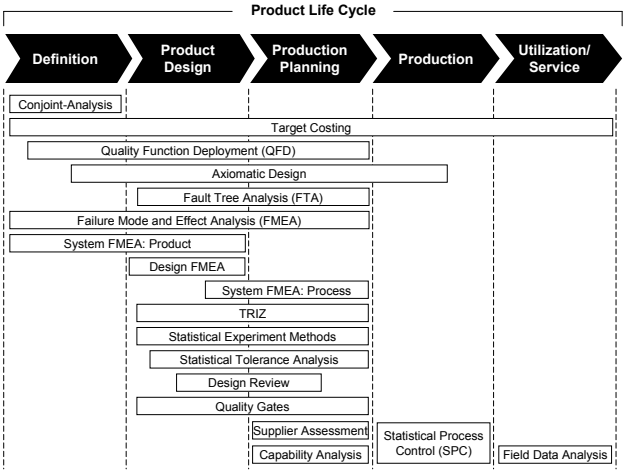

Figure 1 - Quality Assurance Techniques within the PLC

2.1.1 Preventive quality assurance measures

Traditional quality assurance techniques are regularly applied as preventive measures. However, their usage is usually limited to the early phases of the PLC (Danke, 2000). Additionally, hardly any of the existing techniques considers product quality throughout the entire PLC.

The target costing approach can be seen as a technique that intends to cover the entire PLC. The approach is used for analyzing product and process design and involves estimating target costs and designing the product correspondingly to meet these costs. Usually, the cost objective results from the targeted market price. However, besides the given cost objective target costing is not related to any other quality feature such as reliability (Clifton, 2004).

As can be seen in Figure-1, quality assurance (QA) has traditionally been dominated by preventive measures addressing the early lifecycle phases. Within later phases of the PLC, reactive measures gain increased significance as they aim for quality improvement of existing products and processes. For that reason, deficits in quality are examined and analyzed. Subsequently, opportunities for improvement actions can be derived and appropriate remedial actions can be initiated to increase product quality.

2.1.2 Reactive quality assurance measures

In comparison to preventive quality assurance approaches, reactive approaches consider the underlying PLC while analyzing defects and revealing defect causes. Thus, searching for process-specific shortcomings is important for uncovering technical causes for deficits. However, such analyses are limited to specific PLC phases (Schäfer, 2003). These sections were usually predetermined in earlier stages of the corresponding quality assurance measure.

To evaluate and improve product quality, the product features have to be linked to the PLC phases that either define features or have a significant influence. An appropriate approach that continuously covers the PLC is not available yet. Especially for the development of mechatronic products, such a concept considering process-based deficits within the context of the underlying PLC would provide a vital benefit.

2.2 QUALITY ASSURANCE IN SOFTWARE ENGINEERING

According to IEEE Std. 610, quality assurance is defined as (1) a planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product confirms to established technical requirements, and (2) a set of activities designed to evaluate the process by which products are developed or manufactured. In today's software development, the ever increasing complexity and size of software artifacts results in a high demand for systematic quality assurance. Generally, constructive quality assurance and analytical quality assurance can be distinguished.

Constructive QA includes organizational and technical activities that are likely to guarantee that high quality software artifacts are produced (in the sense of correct by construction). Examples for constructive QA elements are principles (e.g., "divide and conquer"), methods (e.g., coding guidelines, model-driven development), formalisms (e.g., special textual or graphical notations like UML), and tools (e.g., code generators).

Analytical QA activities aim at detecting and correcting quality defects. The most common analytical QA activities in software development are verification (e.g., software inspections) and validation (e.g., software testing). Software inspections are engineering practices for the detection and correction of defects in software artifacts, and for preventing their escape into the field. Software inspections were pioneered by IBM in the 1970s (Fagan, 1976). A software test as defined in (IEEE Std 610) is "the process of operating a system or component under specified conditions, observing or recording the results and making an evaluation of some aspects of the system or component".

Both software inspections and software testing aim at the detection of defects but are applied to different types of artifacts. Software inspections can be applied in early phases of a development project, to informal artifacts (like requirements or problem descriptions), while testing first becomes applicable first when executable components are available. While inspections focus on the correctness of the development artifacts, testing focuses on the reliability of the executables.

2.3 DEFECT FLOW MODELS

Defect measurement and thus defect classification plays an important role in most software measurement programs. This importance stems from the fact that defects are an essential impact factor on software failures, and that quality improvement in terms of reliability improvement is an essential dimension of many process improvement programs.

Defect flow models provide a means for characterizing and analyzing defect baselines throughout the lifecycle. Determining the number of defects that are detected during the different defect detection activities in the entire lifecycle provides a so-called defect profile. This defect profile forms a baseline for quality assurance processes. It is expected that changes to these processes can be observed as changes to this baseline. For example, introducing requirements and design inspections will cause more defects to be detected early in the life cycle (Freimut et al., 2000).

Further analysis can be done by considering the origin or injection phase of a defect. Here, the analysis question is, in which phase or activity the defect was introduced into the system. With this information, it is possible to determine the number of defects that were introduced in a given phase or activity, and it is possible to determine the effectiveness of quality assurance techniques by relating the number of defects found to the number of defects not found (e.g., as defect removal effectiveness or yield). By determining the distance between defect injection and defect detection, reasoning about improving defect detection activities in between can be done.

Classifying the origin of a defect is one example of a defect classification. A defect classification scheme contains one or more defect classification attributes that capture various aspects of a defect.

To establish a measurement instrument for the quality assurance process with the intention of establishing quantitative management of the inspection and testing activities, it is necessary to capture the number of defects per defect detection activity and the origin of the respective defects (i.e., the activity in which it was injected). This information can be synthesized as a defect flow model as shown in ➔ Figure-2.

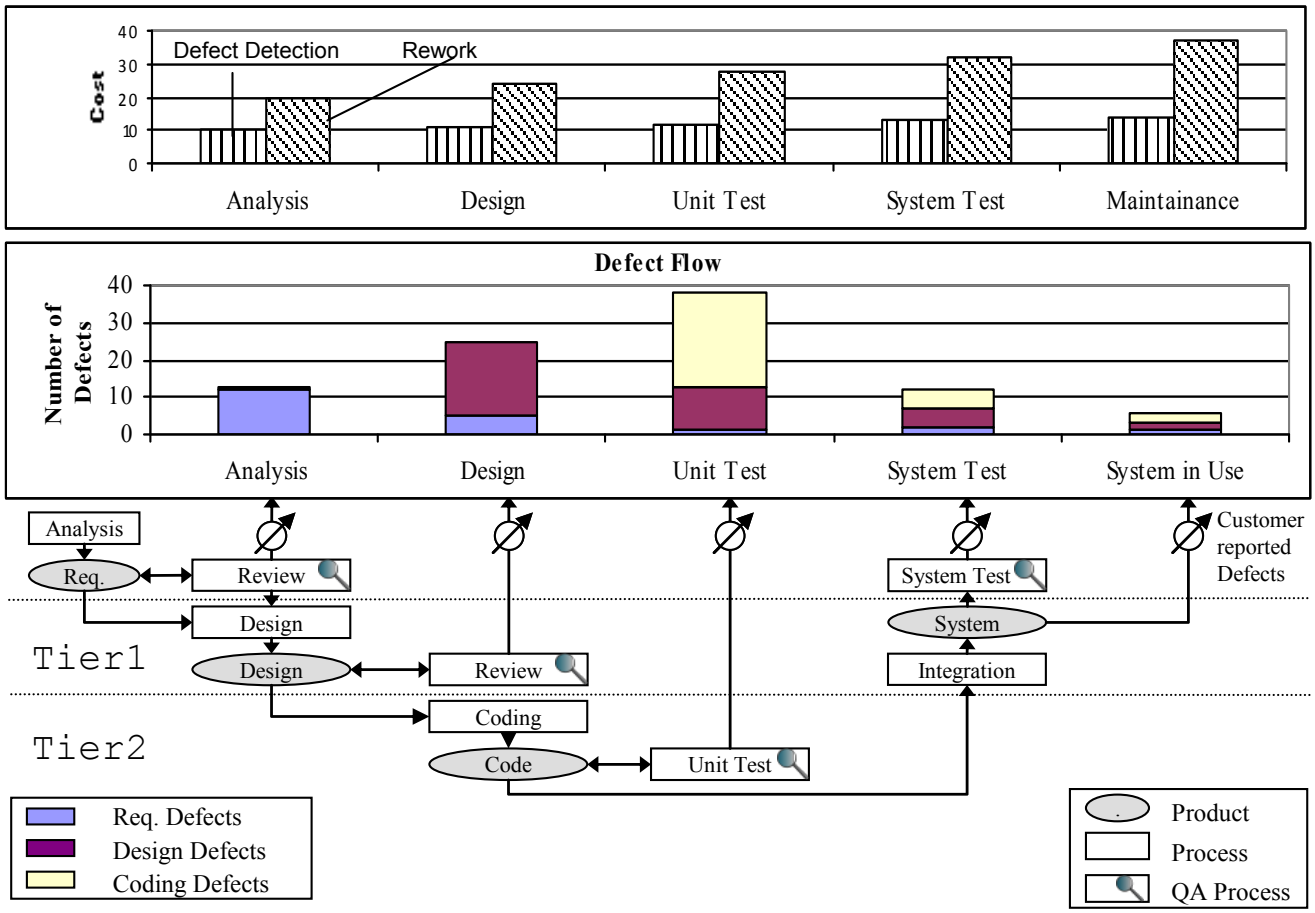

Figure 2 - Defect Flow Model

In the lower area of Figure-2, the main activities of a simplified software development process are shown along with typical products developed within the process. For reasons of simplicity, a sequential software development model in which development is seen as a flow through the phases *analysis*, *design*, *coding*, *integration*, and *test* is used. In each phase, a major development artifact is produced and verified by a review activity or validated by a testing activity. The vertical lines in the process exemplarily indicate organizational borders. Software development is typically done in a distributed style with several organizations involved. Optimizing quality strategies requires considering all these organizations and their local quality assurance strategies in a systematic way.

Typically, in each defect detection step, defects are found and fixed. In the review of the requirements document, only defects that are injected in the analysis activity can be found, since this is the only preceding development activity (assuming that the underlying problem is correctly defined). In the design review step, most of the defects found will probably have been injected in the design activity, but typically, some analysis defects will be found, too. These defects slipped

through the requirements review and cause additional rework in the design phase. Generally, the later defects injected in early phases are found the greater the amount of rework will be in later stages to fix them.

The middle area of Figure 2 shows a chart representation of the defect flow. On the positive y-axis, the defect flow model provides information on how many defects of which type are introduced into the development cycle during each development step. The colors indicate the origin of the defects. Thus, a defect flow model provides useful information on the lifecycle of a defect and represents a first measurement baseline. To fulfil the objective of creating an integrated quality strategy, it is necessary to enhance the defect flow model with defect information on a more detailed level. Attributes like *defect type* allow for valuable analysis in order to optimize a quality strategy, as these attributes characterize a defect at a more detailed level of granularity. Based on such detailed information, it is possible to determine, which quality assurance techniques can address which defect types.

The upper area of Figure-2 shows a cost distribution of defect detection and rework cost. Integrating cost aspects into the analysis can further support the reasoning about appropriate quality assurance strategies.

2.4 DEFECT CLASSIFICATION SCHEMES

Defect classification schemes are used by researchers to characterize the defect detection ability of quality assurance techniques (such as reading techniques (Porter et al, 1996)), or to describe defects found in particular kinds of documents such as use cases.

The defect classification schemes referenced most often from an industrial point of view are the HP Scheme (Grady, 1992), which aims at deriving process improvement proposals, and the ODC scheme from IBM (Chillarege et al, 1992), which also aims at controlling the progress of a development project.

The HP Scheme is shown in ➔ Figure-3. Once a defect is detected, the developer determines the activity in which the defect was introduced and assigns an appropriate attribute value of the attribute *origin*. Next, the attribute *type* describes the defect of a particular origin in more detail. Finally, the attribute *mode* describes why the defect was a defect.

Orthogonal Defect Classification has been developed by IBM (Chillarege, 1994). In ODC, defects are viewed from two perspectives. In the so-called *Opener Section* (executed when a defect is detected), a defect is classified along the attributes *activity* (in which process step was the defect detected), *trigger* (how the defect was detected), and *impact* (what would a customer have noticed if the defect had escaped into the field). In the *Closing Section* (executed when a defect has been fixed), the defect is classified along the attributes *target* (which high-level entity has been fixed), *source* (who developed the target), *age* (what is the history of the target), and *defect type* (what exactly had to be fixed). Additionally, the defect is assigned a *defect qualifier* (missing, incorrect, or extraneous).

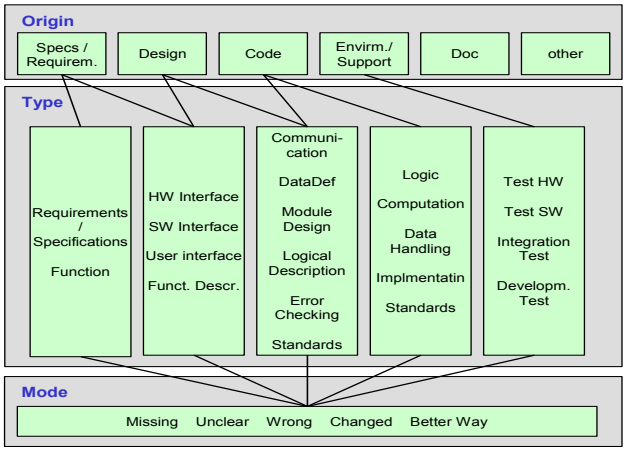

Figure 3 - HP Scheme (Grady, 1992)

3. APPLYING DEFECT FLOW MODELS TO COMMERCIAL VEHICLE LIFECYCLES

In the following section, we present an initial concept on how to transfer software engineering principles from the software engineering domain to the commercial vehicle domain. The focus here is on transferring principles from the defect flow model. The findings we present stem from observations of industrial commercial vehicle development and production processes as well as from a series of interviews with practitioners in the field. The interviews were performed in the context of a research project.

We illustrate the implementation of a defect flow model in the commercial vehicle domain using a continuous improvement approach well known in the software engineering domain. The Quality Improvement Paradigm (Basili et al., 1994) is a systematic approach to continuous software quality improvement ➔ Figure-4. The approach is similar to the Plan/Do/Check/Act (Deming, 1986) cycle used in industry for the implementation of quality management plans. As such, QIP is a feedback loop process (for projects and organizations) and consists of the following six steps:

Step 1: *Characterize and understand* the environment based upon available models, data,

intuition, etc. Relevant characteristics of the organization and of the projects to be measured have to be identified in order to characterize the context of an improvement program. This characterization might be used in other improvement scenarios to decide, if and under which restrictions the gathered knowledge can be reused. Typical questions concern:

- the kind of product that is being developed
- the process that is being used, and
- the main problems encountered during the projects.

This characterization is mainly qualitative in nature, even though previously existing data may be reused. In our specific situation, we characterize context and the lifecycle process with respect to defects (or their capability to be diagnosed). For the definition of *diagnosability* we refer to IEEE Std. 1522-2004: The degree to which faults within a system can be confidently and efficiently identified. Confidently means frequently and unambiguously detecting and isolating faults when they occur. Efficiently means optimizing the resources required to achieve fault isolation.

By considering the functional organization of product development departments of typical companies in the commercial vehicle industry, it becomes apparent how complex the entire system must be. Product development departments are subdivided analogous to the related subsystems: chassis, cabin, powertrain, technical systems, etc. At lower hierarchical levels, specialized technical knowledge increases, and the understanding of the entire system decreases due to the system's complexity. Therefore, the definition and management of organizational interfaces have become one of the most important challenges in the automotive and especially in the commercial vehicle industry.

At the same time, research and development projects as well as the related processes (development, test, construction, etc.) have to be effectively and efficiently integrated into the company's organization. With regard to concepts like "design for manufacturing", other departments have to be involved in the processes, too (i.e., product application center, production, after-sales service, spare part management or supplier management). Nowadays, this is realized by implementing functionally with diverse teams.

The most significant challenges regarding the diagnosability of commercial vehicles and their components are as follows:

- A simultaneous and distributed development process for complex systems or components requires effective coordination mechanisms and tools. Thus, the concentration on the relevant preceding goals within the development process has to be assured.
- In addition, controversial requirements that derive from being present in globally distributed markets lead to contradiction in objectives that need to be carefully considered. This especially applies to the field of after-sales service, where strategies and tools for diagnosis need to be developed or kept in stock.
- Finally, the high number of vehicle and component variants has to be taken into consideration during the product development process. Components are applied to different vehicles with underlying multiple usage and load scenarios.

Due to the product generation's long lifecycle, not every usage scenario is well known at the beginning of the development and test phase. Therefore, the subsequent appearance of new variants involves the danger of insufficient testing or a general lack of suitability with regard to constructive solutions.

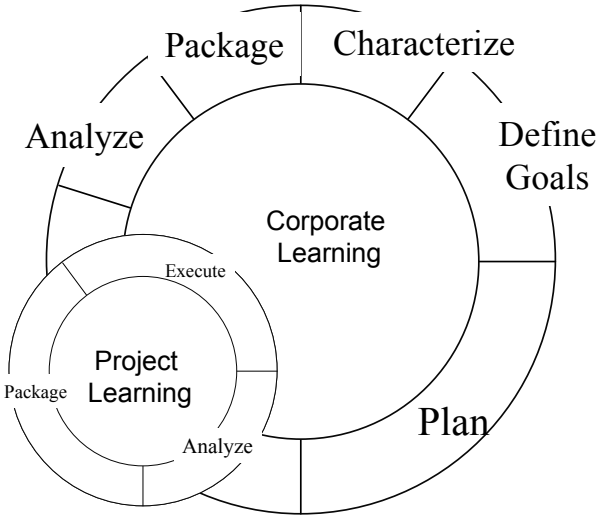

Figure 4 - Quality Improvement Paradigm

Step 2: *Define Goals*. In step two, measurement goals are identified and measurement plans are developed accordingly. The baseline established in the previous step is used to define reasonable and quantifiable goals. For each measurement goal, it is explicitly stated which attributes have to be measured and who is responsible for collecting measurement data. The definition of measures as well as their underlying motivation is documented in the measurement plan.

In our context, we formulate the following goals: (1) Reduce defects detected in field, and (2) improve the diagnosability of selected subsystems and components. These goals are refined into measurement goals. An essential objective of

diagnosis in the field of commercial vehicles is the detection of defects or wear with the highest possible level of accuracy and within the shortest time. Moreover, the smallest technical unit that needs to be changed is to be identified unequivocally in order to prevent waste in the overhaul or repair process. As a result, the required time for the identification of a defect or the diagnosis' accuracy can be taken as measures for the product's diagnosability.

Step 3: *Choose Processes, Methods, Techniques and Tools*. In this step, the data collection procedures for all measures identified during the second step have to be defined, i.e., how and when the data has to be collected and who will collect it. For the implementation of a defect flow model in the commercial vehicle domain, a set of quality gates has to be identified that typically detect defects in products or product components.

Information that might lead to improved diagnosability in the field of commercial vehicles can be identified at every stage of the product development and production process. Therefore, typical core processes like development, production, and service need to be analysed carefully. In addition, the analysis of support processes such as supplier and quality management or production planning can deliver valuable results as well.

In our case, all departments have developed strategies and quality assurance techniques to document defects and ensure the identification of root causes. Complex organizations usually provide a broad range of adequate techniques. Commonly, these techniques display different degrees of complexity and effectiveness themselves. Unfortunately, most of these techniques show deficits by the time they have to deal with inconsistent or incomplete data. On top of that, information that might be relevant for diagnostic purposes frequently exists only as implicit knowledge. Structured and formal documentation is still an exception.

Accordingly, different challenging fields regarding diagnosability can be identified. Apart from the technical challenges related to the product and its components, process organization also obtains an important role.

A formal improvement of the product development process needs to be based on a demand-oriented classification of defects that considers diagnosability. With regard to the diverse requirements of certain defect classes, adequate concepts and strategies can be implemented in order to solve topical problems as well as to avoid identical or similar issues in the future.

Step 4: *Process Execution*. The improvement program gets executed in this step. Measurement data is collected, analyzed, and interpreted. In a complex QIP scenario, the execution step might result in a nested QIP circle ➔ Figure-4, where the outer circle represents the corporate learning loop and the inner circle represents a project learning loop. Such inner loops might be implemented at a supplier's site that delivers components or sub-components.

As part of the research project, interviews were performed in cooperation with a leading manufacturer of commercial vehicles. Thus, a number of representatives from different departments were involved. Their practical experiences regarding the detection of defects as well as the related root cause analysis were the most important objectives. The existence of strategies for defect prevention was analyzed, as were the existing systems, methods and tools, for reliability and diagnosability improvement.

Step 5: *Analyze Results*. In this step, the collected data is analyzed and interpreted and lessons learned are identified.

Findings as well as results identified throughout the research project can be divided into two basic categories. On the one hand, possible solutions and concepts have been examined focusing on improved reliability and diagnosability of a sample mechatronic product. On the other hand, concepts and solutions in the fields of process design and improvement have been generated. These issues have led to a superior and more sustainable treatment of identified and documented defects.

The latter concepts facilitate tracing defects from the moment of detection to the point where defects could have been avoided and to the points where the defect's root cause is located. Moreover, the concept allows prioritization with regard to the defect's impact on the entire system. This procedure is comparable to the well-known Pareto Analysis.

Complementary to the findings of the practical analysis, the concept allows a holistic view on processes and process chain within the considered corporation. This point of view needs to be broadened during the next steps of the research project. Due to the minimal vertical integration of manufacturing and development within the automotive industry, it is critical to apply the concept to the entire supply chain.

Finally, a holistic view on product reliability, diagnosability, and related issues is supported by the proposed approach. The integration of all involved parties is one main focus of the concept.

Step 6: *Package and store experience*. This step includes the packaging of experience for future use. In our case, this includes the packaging of defect

profiles together with context information and the description of lessons learned.

4. LESSONS LEARNED AND RECOMMENDATIONS

Having performed the described research project the following issues could be identified and documented as "lessons learned":

- Due to a large organization's complexity, the analysis needs often to be limited to selected sections or departments.
- Using a certain aspect or component supports a focused analysis regarding the entire process chain.
- With regard to their requirements, the entire group of stakeholders has to be considered carefully (e.g., customers and their requirements regarding the product, or the production department and their requirements regarding the ease of assembly).
- In order to be able to cooperate effectively and efficiently, interdisciplinary teams have to be defined. Thus, specific topics have to be assigned to these teams.
- The adequate consideration of stakeholders' requirements can be supported by implementing transparent processes.
- For dealing with controversial requirements concerning the entire system or its components (e.g., ease of assembly vs. ease of maintenance), transparent decision processes have to be established.
- Defects that have been identified throughout the product development process or even during the phase of utilization have to be tracked and documented carefully. At the same time, a sustainable elimination of the defect's root cause has to be ensured.
- Based on defect identification and documentation activities have to be defined that effectively prevent from identical defects and their impacts in upcoming product or process generations.

6. SUMMARY AND CONCLUSIONS

This article sketched the initial application of software engineering principles to the commercial vehicle domain. The approach is based on observations from industrial processes and interviews with practitioners. The focus is on defect detection, defect prevention, and diagnosis aspects. The main findings are: (1) Software engineering principles in the area of defect flow models help to systematically analyze defect flows in the commercial vehicle domain and provide a basis for deriving quality assurance strategies in this domain. (2) Many challenges with respect to applying defect flow models (such as establishing appropriate measurement competence, missing transparency between organizations) are also relevant in the commercial vehicle domain. (3) The complexity of the context in the commercial vehicle domain (such as the interaction between suppliers and OEMs during development and production) is very high and should be carefully understood so that defect stream models can be applied.

Directions for future research include the use of more refined defect classification schemes and cost aspects for analyzing defect streams as well as the selection and tailoring of appropriate defect detection techniques (such as perspective-based reading) based on the defect baselines.

REFERENCES

V. R. Basili, G. Caldiera and D.H. Rombach, "The Experience Factory", John Wiley and Sons: Encyclopedia of Software Engineering (ed. Marciniak) 1994

V. R. Basili, C. Caldiera. H. D. Rombach, "Goal Question Metric Paradigm", Encyclopedia of Software Engineering, Vol. 1, 1994, pp. 528-532

L. C. Briand, C. M. Differding and H. D. Rombach, "Practical Guidelines for Measurement-Based Process Improvement", Software Process - Improvement and Practice, Vol. 2 No. 4, 1996

R. Chillarege and I. Bhandari and J. Chaar, and M. Halliday and D. Moebus and B. Ray and M. Wong, "Orthogonal defect classification – A concept for in-process measurements", IEEE Transactions on Software Engineering, vol. 18, pp. 943-956, 1992

M. B. Clifton, H. M. B Bird and R. E. Albano, "Target Costing: Market Driven Product Design", Marcel Dekker, New York, Basel, 2004

S. M. Danke, "QM-Konzept zur präventiven Qualitätssicherung für die Montageplanung in der Automobilindustrie", Weißensee, Berlin, 2000

E. Deming, "Out of the Crisis", MIT Center for Advanced Engineering Study, MIT Press, Cambridge, MA, 1986

M. Fagan, "Design and Code Inspections to Reduce Errors in Program Development." IBM Systems Journal 15, 3, 1976, pp 182-211.

B. Freimut, C. Denger and M. Ketterer, "An Industrial Case Study of Implementing and Validating Defect Classification for Process Improvement and Quality Management", Software Metrics, 2005. 11th IEEE International Symposium

K. Frühauf, J. Ludewig, H. Sandmayr, „Software-Prüfung", Vdf Hochschulverlag, 2004.

K. H. Grote, J. Feldhusen, “Dubbel – Taschenbuch für den Maschinenbau”, 21st Edition, Springer, Berlin, 2004

E. Hering, et. al.: Qualitätsmanagement für Ingenieure, 5th Edition, Springer, Berlin, 2003

IEEE Std 610 1990, “Standard Glossary of Software Engineering Terminology, New York, USA: Institute of Electrical and Electronic Engineers, 1990

IEEE Std 1522-2004, “IEEE Trial-Use Standard for Testability and Diagnosability Characteristics and Metrics”

R. Grady, “Practical Software Metrics for Project Management and Process Improvement”, Prentice Hall, 1992

T. Pfeifer, “Quality Management. Strategies, Methods, Techniques”, Hanser, München, Wien, 2002

A. Porter, L. G. Votta, V. R. Basili, Comparing Detection Methods for Software Requirements Inspections: A Replicated Experiment, IEEE Transactions on Software Engineering, 1996, pp 563-575.

L. Schäfer, “Analyse und Gestaltung fertigungstechnischer Prozessketten”, PhD thesis, University of Kaiserslautern, Kaiserslautern, 2003