Suitable Development Processes

Dr. Erich Meier, method park Software AG

Abstract

If you ask computer users for their assessment of software, most are dissatisfied. Software is seen as inconvenient, slow and plagued with errors. The aim of this article is to bring together tried-and-tested measures for counteracting this phenomenon. In so doing, both process standards and reference models such as CMMI® and SPICE™ will be analyzed, as will agile methods.

Suitable development processes have a considerable influence on improving software quality. Norms, standards and reference models may be used as building blocks of tried-and-tested procedures for designing such processes. What is absolutely essential, however, is the acceptance of the development processes among all those involved. The strategy for making the processes known must, therefore, pay attention both to the automation of process elements, as well as to adequate levels of method training and to optimum process implementation. Process portals play a key role in this. The benefit to the user must always be at the forefront of considerations.

Introduction

If you ask computer users who do not come from the IT industry about the reputation of software and IT, the answer is often "bad" or "very bad". The situation is the same if you ask users of cell phones or even normal car drivers about their experience with the respective technologies. Typical comments include: “complicated operation”, “too slow”, “too prone to errors”. Why is that so? Software Engineering has been developing technologies for usability engineering, for requirements analyses and for designing reliable software for decades now. Are they all unfit for use? Reference models such as CMMI or SPICE have been deployed for a long time for improving development processes. Why is it that these efforts remain almost fruitless?

The aim of this article is to bring together tried-and-tested measures for counteracting this phenomenon.

In our experience, we have found that the actual reasons for the inadequacy of software lie not so much in over-complicated technology or in the insufficient qualifications of developers and engineers as in the lack of or incorrect use of development processes, for example, or of technologies for requirements management, test management, usability engineering or project management. If these technologies are used, the pendulum tends to swing in an undesired direction, as applying them to the full often generates a multitude of additional activities and documentation steps. These, in turn, are seen by the developer as superfluous extra work, and they are thus carried out in a correspondingly unmotivated way and without due care.

In short: one key lies in the definition of suitable development processes and in the acceptance and implementation of these processes within the organization.

Norms, Standards and Reference Models

Are process standards helpful? Standards are a dime a dozen. In the area of safety alone there are a multitude of standards (e.g. IEC 61508, EN 50128, IEC 60601). There are standards for usability engineering (ISO 9241), for testing processes (TPI) and for project management (ISO 10006 / ISO 10007). Fig.1 provides an impression of which standards are currently influential on development processes in the automotive sector.

When asked for the essence of the safety standard, IEC 61508, safety expert, Günther Glöe, from the TÜV Nord (= technical inspection agency for Northern Germany) provides a simple answer: “The IEC 61508 demands engineering, and forbids just patching things together.” In other words, it ensures controlled and methodical procedures that are carried out according to plan instead of development at any cost. This statement may be applied to almost all process standards and all software engineering technologies.

However, the IEC and ISO standards are limited to abstract requirements, particularly in the area of processes, and seldom provide concrete specifications on implementing them. This is understandable considering the lengthy time it takes to create a standard. Under such circumstances, the standard is unable to keep up with the ever-changing state-of-the-art at any one time.

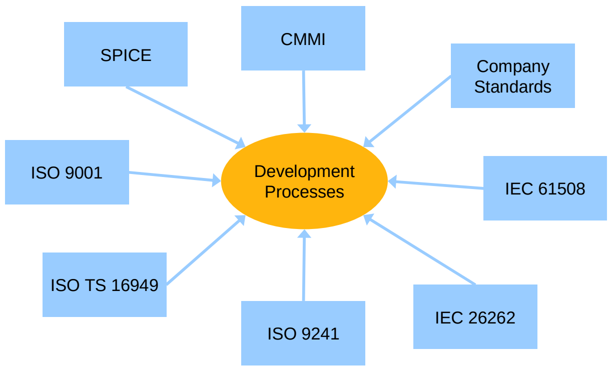

Fig. 1: Relevant process standards in the automotive branch

This is also valid, in principle, for process reference models such as CMMI und SPICE, which contain a multitude of specifications that development processes need to fulfill. They thus represent a collection of “Best Practices” on the subject of engineering. Are they then a solution to the problem? The answer is “yes” insofar as the processes are more complete after deploying the reference models than they were before, as they then demonstrate a certain minimum standard. In spite of this, the reference models – like all norms and standards - nevertheless remain on a relatively abstract level and hardly provide any concrete guidelines on implementation.

Particularly problematic, however, are inexperienced assessors and auditors who work with methods that are lacking in practical relevance, who merely stick to formal specifications, and who thereby demand overly bureaucratic and unsuitable processes. It is a good sign that the certification of iNTACS-SPICE assessors will, in future, ensure that the candidates are able to demonstrate comprehensive practical experience in the development field, and not purely in the area of quality assurance alone.

However, it must also be noted that, as experience has shown, good processes may well ensure better products, but good processes alone are by far no guarantee for perfect products.

Agile Methods

If one turns instead to the agile methods with their focus on lightweight procedures, it becomes clear that they are developer-oriented, and thus they automatically enjoy a higher level of acceptance within the development team. Agile methods (e.g. XP or SCRUM) have influenced the design of processes in two main ways in the past, namely through test-driven development and iterative-incremental procedures. The principle of covering all system functionalities by means of automated tests contributes significantly to reducing the proneness to errors of software products, while the emphasis on an iterative-incremental procedure represents an important step towards creating a software product which fulfils the requirements and expectations of its users.

Being too lightweight when it comes to method selection can lead to problems, however. At the latest these become discernible if software systems need to be in operation for a long period of time, i.e. for decades. It is no easy task to make changes to a system whose only documentation consists of the program code and whose development team has, in the meantime, scattered to the four winds.

It must be pointed out, however, that this is not an inherent problem of the agile methods

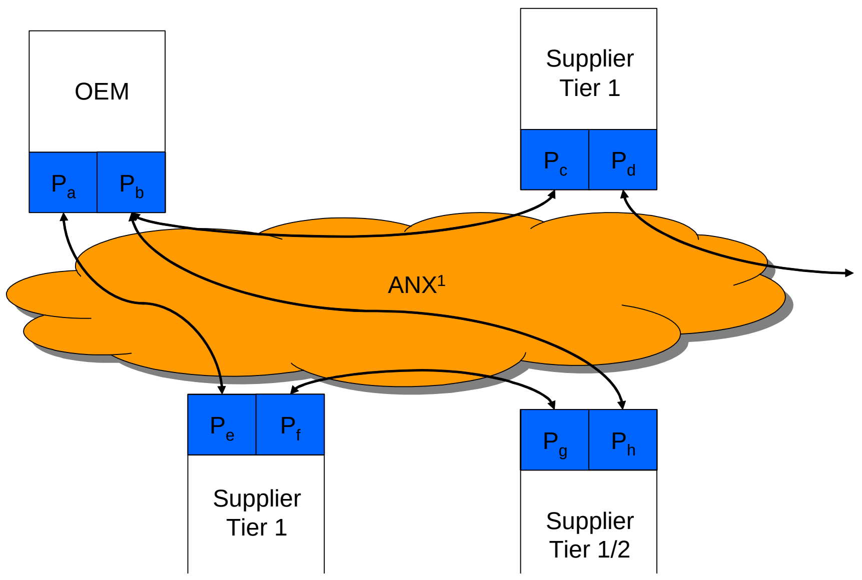

Fig. 2: Connected processes for cross-company development processes

themselves. Rather, the blame lies with those users who falsely believe that using agile methods means that no additional measures (e.g. methodical requirements analyses, design, system documentation, traceability) are needed at all.

Much more of a problem is successfully implementing agile methods in development projects which are distributed across several supplier companies and in which not only software but complete systems are developed. This is the case in branches such as the automotive industry in which the actual development of complete systems, such as electronic braking assistants, is carried out by the supplier companies, with the automotive manufacturers increasingly assuming the role of pure system integrators. The system suppliers, in turn, draw upon specialist companies for a large proportion of the hardware and software they require. Standard architectures such as AutoSAR, which aim to define combinable and interchangeable components, will probably serve to reinforce this trend.

In such constellations it is almost impossible to establish agile principles such as the on-site customer, who is available to the development team for information on system requirements over a long time period. Rather, a sophisticated requirements management system needs to be established which assists in meeting the flexibility required on the manufacturer side, and which also helps to create the stability needed by the supplier. Possible approaches to solving this challenge are currently being piloted in the automotive industry with the aid of the standardized interchange format, RIF (Requirement Interchange Format).

General approaches towards managing such multi-tier development projects that are not merely limited to requirements management are currently being discussed under buzzwords such as “Connected Processes” or “Process Standardization”. The connected process approach involves companies carrying out an initial standardization of their processes using reference models such as SPICE. As a second step, they make the interfaces available to their suppliers and couple the process results to each other. Fig. 2: Connected processes for cross-company development processes illustrates this principle, with the electronic market places in the area of procurement and logistics serving as a model.

Suitable Development Processes

The great challenge is now to learn both from standards and reference models, as well as from the agile world. In other words, it is necessary to bring together the principle of process orientation from the reference models with the right techniques and procedures from the agile methods. In an ideal world, users would be part of the process without noticing it.

Which steps could we undertake to achieve this – admittedly ambitious – goal?

Firstly, here are two approaches which proved unfavorable in practice. On the one hand, nothing is to be gained by defining development processes and then exerting pressure, with those at the top decreeing that these processes be deployed by everyone. All this does is to damage the process acceptance enormously. Developers are generally creative people, and they will find ways and means of working around a process whose use has been decreed.

Secondly, processes should not simply be put together by consultants alone, as this is not the road to process acceptance, either. Instead, what one will generally hear is statements such as “That is all very well and good, but we do things completely different here”. It is thus absolutely essential to take the time to work on a suitable development process in an iterative way together with those involved and accompanied by experienced consultants. “Suitable” in this context means that every process component must serve a worthy goal, or to put it like Einstein: “…as simple as possible, but no simpler.” It is worthwhile recording this goal, for example by means of a diagram of the process step according to SPICE, CMMI or an IEC standard. In this way, it may be drawn upon again if necessary. To determine the level of detail for the description, an inexperienced user should serve as the gauge. After reading the process description, this user should be able to roughly find his/her way around and to successfully carry out initial steps.

Furthermore, a suitable development process must also contain certain degrees of freedom that enable users to adapt the process to their specific project situation, i.e. to carry out process tailoring. These degrees of freedom should not be limited to merely being able to omit one or two process steps, but must allow the user to really have an influence on the process.

An experienced test manager once said of the process definition: “First be creative and then bureaucratic, not vice versa.” In other words, define a good, suitable process and then stick to it stringently. With many development processes, however, one has the impression that bureaucratic process monsters have been created which are then taken apart, manipulated or worked around by the users with astonishing levels of creativity.

Acceptance and Implementation of Development Processes

Once a suitable development process has been worked out, the question arises as to which principles should be used to implement it, i.e. how can it be introduced into the organization?

The key to this is demonstrating the value of the system by allowing people to use it. If the process user recognizes that added value is generated by deploying the process (e.g. in the form of time savings or better results), the most important hurdle to acceptance has already been overcome. In order to convince the user of the benefits of the process, process management tools are usually deployed. In selecting such a tool, the following three principles need to be taken into account:

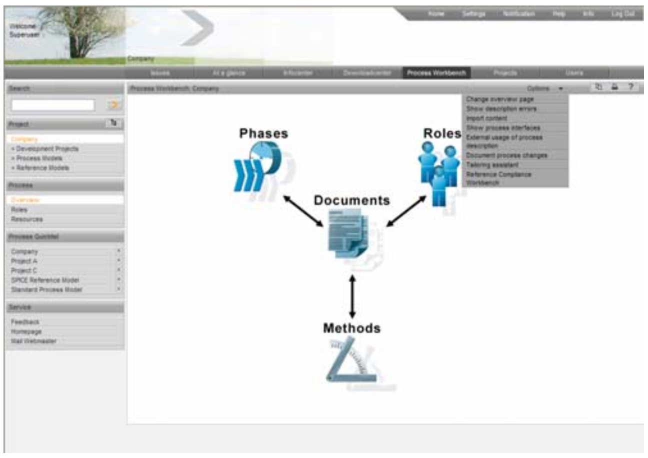

Fig. 3: "project kit" process portal

- Simplicity: access to the development processes needs to be quick and simple
- Adaptability: the process structure must be adaptable to the customer requirements
- Integration: the process design must be carried out in such a way that it integrates with the existing IT environment and the existing tools

According to our analyses, the most important purpose of such tools is to gain rapid access to the work products (e.g. project plan, test plan, requirements). The user is mostly seeking as comprehensive an answer as possible to the questions: “What do I need to produce, and when, how and why do I need to do this?” and: “Who or what can assist me?” Process portals, as illustrated in Fig. 3: "project kit" process portal, can offer valuable services in this context.

By the way, tools which whisper to the user, “Deploy me and you will have CMMI Level 5” very rarely keep their promise!

In order to ensure that users are not overly burdened with additional work on top of their actual development activities, parts of the development process need to run automatically as far as possible. Examples include:

- Reviewing and approving documents
- Checking for consistency and process conformity (e.g. with QA-C or Checkstyle) – and termination of the process in cases where non-conformities are determined. From experience, warnings are usually ignored!
- No or only minimal manual efforts for metrics logging

However, it is not advisable to allow the development processes to run completely as workflow, as they contain too many degrees of freedom for that. Complete automation would either rob the process of all flexibility, or it would make the control of the process extremely complicated. All trials in this direction that are known to us have failed.

In those areas where automation reaches its limits (e.g. constructing the architecture, creating the design, compiling the test concept) optimum understanding of the process among those involved in it must be achieved instead. Targeted process and process method training (e.g. architecture and design, requirements management) are indispensable.

Furthermore, optimum process provision is also essential. The most important requirements for a good process representation are:

- Intuitive representation of the entire process and of sub-processes (e.g. through role-based views)

- No endless wallpaper-sized sheets of workflow, but short, concise activity lists
- Templates, checklists and example documents must be available and ready to hand.
- The process representation shows the tailored project process, and not just any standard process, as a .pdf file

Parallel coaching measures to accompany the introduction of new processes are also indispensable. The project leader must always highlight to the team the reasons for introducing the development processes. Correctly designed checklists reduce hassle for the process user, particularly in critical situations, as there is no danger of forgetting an important step. However, it should not take half an hour to locate the right checklist only to be unsure upon finding it as to whether it is the most up-to-date version, or not. A process management system must ensure rapid and reliable access to all process components.

Process improvement also involves regularly questioning the meaningfulness of process specifications. The correct use of a CIP (Continual Improvement Process) not only involves adding activities, but also merging steps, as well as consistently deleting steps or even entire process areas that are no longer relevant.

Summary

Suitable development processes have a considerable influence on improving software quality. Norms, standards and reference models may be used as building blocks of tried-and-tested procedures for designing such processes. What is absolutely essential, however, is the acceptance of the development processes among all those involved. The strategy for making the processes known must, therefore, pay attention both to the automation of process elements, as well as to adequate levels of method training and to optimum process implementation. Process portals play a key role in this. The benefit to the user must always be at the forefront of considerations.

However, the prerequisite for successful process deployment is that those involved in the process are also granted adequate time for their work. The proficiency and the will to deliver high-quality work are almost always given.