# Suitability of Software Engineering Models for the Production of Usable Software

Karsten Nebe[1] and Dirk Zimmermann[2]

[1] University of Paderborn, C-LAB, Fürstenallee 11, 33098 Paderborn, Germany
[2] T-Mobile Deutschland GmbH, Landgrabenweg 151,
53227 Bonn, Germany
Karsten.Nebe@c-lab.de, Dirk.Zimmermann@t-mobile.de

**Abstract.** Software Engineering (SE) and Usability Engineering (UE) both provide a wide range of elaborated process models to create software solutions. Today, many companies have understood that a systematic and structured approach to usability is as important as the process of software development itself. However, theory and practice is still scarce how to incorporate UE methods into development processes. With respect to the quality of software solutions, usability needs to be an integral aspect of software development and therefore the integration of these two processes is a logical and needed step. One challenge is to identify integration points between the two disciplines that allow a close collaboration, with acceptable additional organizational and operational efforts. This paper addresses the questions of where these integration points between SE and UE exist, what kind of fundamental UE activities have to be integrated in existing SE processes, and how this integration can be accomplished.

**Keywords:** Software Engineering, Usability Engineering, Standards, Models, Processes, Integration.

## 1 Introduction

Software engineering is a discipline that adopts various engineering approaches to address all phases of software production, from the early stages of system specification up to the maintenance phase after the release of the system ([14],[17]). Software engineering tries to provide a systematic and planable approach for software development. To achieve this, it provides comprehensive, systematic and manageable procedures, in terms of software engineering process models (SE Models).

SE Models usually define detailed activities, the sequence in which these activities have to be performed and the resulting deliverables. The goal in using SE Models is a controlled, solid and repeatable process in which the project achievement do not depend on individual efforts of particular people or fortunate circumstances [5]. Hence, SE Models partially map to process properties and process elements, adding concrete procedures.

Existing SE Models vary with regards to specific properties (such as type and number of iterations, level of detail in the description or definition of procedures or activities, etc.) and each model has specific advantages and disadvantages, concerning

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-3-540-92698-6_37

J. Gulliksen et al. (Eds.): EIS 2007, LNCS 4940, pp. 123–139, 2008.

predictability, risk management, coverage of complexity, generation of fast deliverables and outcomes, etc.

Examples of such SE Models are the Linear Sequential Model (also called Classic Life Cycle Model or Waterfall Model) [15], Evolutionary Software Development [12], the Spiral Model by Boehm [1], or the V-Model [9].

### 1.1 Linear Sequential Model

The Linear Sequential Model divides the process of software development into several successive phases: *System Requirements, Software Requirements, Analysis, Program Design, Coding, Testing* and *Operations.* On the transition from one phase to the other it is assumed that the previous phase has been completed. Iterations between neighboring phases are planned to react on problems or errors which are based on the results of the previous phase. The Linear Sequential Model is document-driven. Thus, the results of each phase are documents that serve as milestones to track the development progress.

### 1.2 Evolutionary Development

In the Evolutionary Development the phases *Software Specification*, *Development* and *Validation* are closely integrated. Evolutionary Development is especially well suited for software projects where the requirements cannot be defined beforehand or in which the requirements are likely to change during the development process. The procedure is always a sequence of iterative development-cycles which results in an improved version of a product on the end of each sequence. There is no explicit maintenance phase at the end of the lifecycle. Necessary changes after the product delivery are solved in further iterations. Within Evolutionary Development the end users and the customers are closely involved in the development process. The goal of Evolutionary Development is "to avoid a single-pass sequential, document-driven, gated-step approach" [10].

### 1.3 Spiral Model

The Spiral Model is a refinement of the Linear Sequential Model in which the single phases are spirally run through. This cycle in the spiral is repeated four times, for *System Definition, Software Requirements, Conception (Architecture Design)* and *Realisation (Detail Conception, Coding, Test, Integration and Installation)*. The nature of the model is risk-driven. At the end of each cycle the current project progress is being analyzed and the risk of project failure is evaluated. Depending on the evaluation outcome the project goals are (re)defined and resources are (re)allocated or – in the worst case - the development is being discontinued if necessary for the subsequent phases. Unlike the Linear Sequential Model, risks are identified throughout the process which leads to a more control- and planable process. The failure of a project can be significantly minimized.

### 1.4 V-Model

The V-Model represents the development process in a symmetric model in which the validation is performed inversely to the system compilation, starting from module up

to the acceptance test [13]. The V-Model is based upon the Linear Sequential Model but emphasis is laid on the assurance of quality (e.g. connections between basic concepts and the resulting products). Inspections take place at multiple test phases testing different levels of detail of the solution and not only at the end of development as other models propose. Compared to the Linear Sequential Model or the Spiral Model, the V-Model is more precise in its description of procedures and measures.

### 1.5 Standards in Software Engineering

Software engineering standards define a framework for SE Models on a higher abstraction level. They define rules and guidelines as well as properties of process elements as recommendations for the development of software. Thereby, standards support consistency, compatibility and exchangeability, and cover the improvement of quality and communication.

The ISO/IEC 12207 provides such a general process framework for the development and management of software. "The framework covers the life cycle of software from the conceptualization of ideas through retirement and consists of processes for acquiring and supplying software products and services." [7]. It defines processes, activities and tasks and provides descriptions about how to perform these items on an abstract level.

In order to fulfill the superordinate conditions of software engineering standards (and the associated claim of ensuring quality) the SE Models should comply with these conditions. In general, standards as well as SE Models can not be directly applied. They are adapted and/or tailored according to the corresponding organizational conditions. The resulting instantiation of a SE Model, fitted to the organizational aspects, is called software development process, which can then be used and put to practice. Thus, the resulting Operational Process is an instance of the underlying SE Model and the implementation of activities within the organization.

This creates a hierarchy of different levels of abstractions for software engineering: Standards that define the overarching framework, process models that describe systematic and traceable approaches and the operational level in which the models are tailored to fit the specifics of an organization (Figure 1).

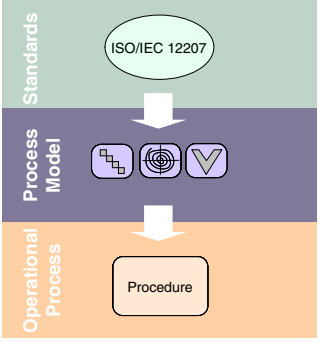


**Fig. 1.** Hierarchy of standards, process models and operational processes in software engineering

### 1.6 Usability Engineering

Usability Engineering is a discipline that is concerned with the question of how to design software that is easy to use (usable). Usability engineering is "an approach to the development of software and systems which involves user participation from the outset and guarantees the efficacy of the product through the use of a usability specification and metrics." [4]

Usability engineering provides a wide range of methods and systematic approaches for the support of development. These approaches are called Usability Engineering Models (UE Models) or Usability Lifecycles, such as the Goal-Directed-Design [2], the Usability Engineering Lifecycle [11] or the User-Centered Design-Process Model of IBM [6]. All of them have much in common since they describe an idealized approach that ensures the development of usable software, but they differ their specifics, in the applied methods and the general description of the procedure (e.g. phases, dependencies, goals, responsibilities, etc.) [18]. UE Models usually define activities and their resulting deliverables as well as the order in which specific tasks or activities have to be performed. The goal of UE Models is to provide tools and methods for the implementation of the user's needs and to guarantee the efficiency, effectiveness and users' satisfaction of the solution.

Thus, usability engineering and software engineering address different needs in the development of software. Software engineering aims at systematic, controllable and manageable approaches to software development, whereas usability engineering focuses on the realization of usable and user-friendly solutions.

The consequence is that there are different views between the two disciplines during system development, which sometimes can be competing, e.g. SE focuses on system requirements and the implementation of system concepts and designs, whereas UE focuses on the implementation of user requirements and interaction concepts and designs. However, both views need to be considered in particular.

### 1.7 Standards in Usability Engineering

Usability Engineering provides standards similar to the way Software Engineering does. They also serve as a framework to ensure consistency, compatibility, exchangeability, and quality which is in line with the idea of software engineering standards. However, usability engineering standards lay the focus on the users and the construction of usable solutions. Examples for such standards are the DIN EN ISO 13407 [3] and the ISO/PAS 18152 [8].

The DIN EN ISO 13407 introduces a process framework for the human-centered design of interactive systems. Its' overarching aim is to support the definition and management of human-centered design activities, which share the following characteristics:

1) the active involvement of users and a clear understanding of user and task requirements (Context of use)
2) an appropriate allocation of function between users and technology (User Requirements)
3) the iteration of design solutions (Produce Design Solutions)
4) multi-disciplinary design (Evaluation of Use)

These characteristics are reflected by the activities (named in brackets), which define the process framework of the human centered design process, and have to be performed iteratively.

The ISO/PAS 18152 is partly based on the DIN EN ISO 13407, and describes a reference model to measure the maturity of an organization in performing processes that make usable, healthy and safe systems. It describes processes and activities that address human-system issues and the outcomes of these processes. It provides details on the tasks and artifacts associated with the outcomes of each process and activity.

There is a sub-process called *Human-centered design* which describes the activities that are commonly associated with a User Centered Design Process. These activities are *Context of use*, *User requirements*, *Produce design solutions* and *Evaluation of use*, which are in line with the DIN EN ISO 13407. However, by being more specific in terms of defining lists of activities (so called Base Practices), that describe how the purpose of each activity is achieved (e.g. what needs to be done to gather the user requirements in the right way). The ISO/PAS 18152 enhances the DIN EN ISO 13407 in terms of the level of detail and contains more precise guidelines.

In order to ensure the claims of the overarching standards, UE Models need to adhere to the demands of the corresponding framework. Thus, a connection between the standards and the UE Models exists which is similar to the one the authors described for software engineering. There is a hierarchy of standards and subsequent process models, too.

Additionally there are similarities on the level of operational processes. The selected UE Model needs to be adjusted to the organizational guidelines. Therefore, a similar hierarchy of the different abstraction levels exists for software engineering and for usability engineering (Figure 2). Standards define the overarching framework, models describe systematic and traceable approaches and on the operational level these models are adjusted and put into practice.

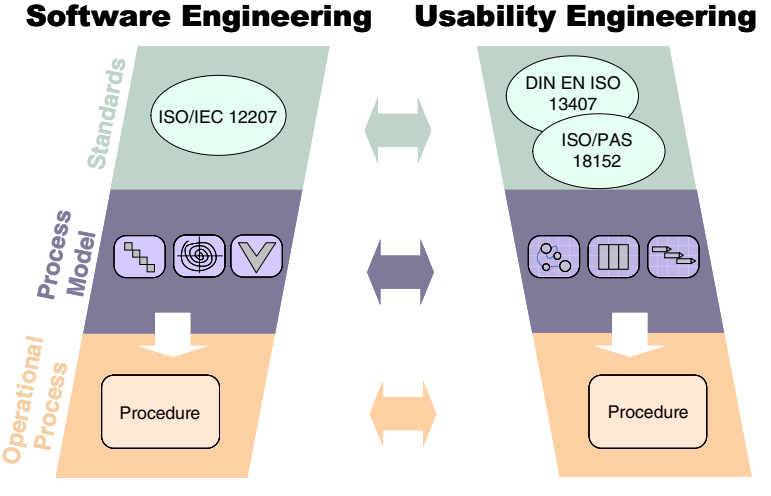


**Fig. 2.** Similar hierarchies in the two disciplines software engineering and usability engineering: standards, process models and operational processes

## 2 Motivation

For development organizations SE Models are an instrument to plan and systematically structure the activities and tasks to be performed during software creation.

Software development organizations aim to fulfill specific goals when they plan to develop a software solution. Such goals could be the rapid development of a new software solution, to become the leader in the application area or to develop a very stable and reliable solution e.g. because to enhance the organization's prestige – and of course, to generate revenue with it. Depending on their goals an organization will chose one (or the combination of multiple ones) SE Model for the implementation that will in their estimate fit best. However, these goals are connected with criteria which can manifest themselves differently. These could be organization-specific characteristics, such as the planability of the process or project, quality of the process, size/volume of the project, organizational structures, types of qualification, etc. These could also be product-specific characteristics, like security and reliability, verification and validation, innovation, etc.

Thus depending on the goals of an organization the decision of selecting an appropriate SE Model for the implementation is influenced by the underlying criteria. As an example, the Linear Sequential Model with its' predefined results at the end of each phase and its sequential flow of work certainly provides a good basis for a criterion such as planability. On the other hand, the Evolutionary Development might not be a good choice if the main focus of the solution is put on error-robustness because the continuous assembling of the solution is known to cause problems in structure and the maintenance of software code.

As usability engineering put the focus on the user and usability of products, which is an important aspect of quality, usability is important for the development process. Usability could take up both either product-specific characteristics (such as the efficiency, effectiveness and satisfaction of using a product) or organizational-specific characteristics (like legal restinctions or company guidelines such as producing usable products to distinguish on the market). Thus, usability is also an important – even crucial – criterion for organizations to choose a well-suited SE Model.

However, one problem remains – usability engineering activities are not an inherent part of software engineering, respectively of SE Models. Indeed, many different models for software engineering and usability engineering exist but there is a lack of systematic and structured integration [16]. They often coexist as two separate processes in an organization and therefore need to be managed separately and in addition need to be synchronized. However, as usability is an important quality aspect it needs to be an integral part of software engineering and of SE Models. It seems reasonable to extend the more extensive proceeding with the missing parts, which in this case means to add usability engineering activities to the software engineering process models, to integrate these two disciplines.

Beside the need for integration it is, however, important to consider both views, the systematic, controllable and manageable approaches of SE and the realization of usable and user-friendly solutions of UE, respectively. It should not be tried to cover one view with the other. The goal is to guarantee an efficient coexistence but to retain the specific goals and approaches of each discipline.

According to the hierarchy of standards, process models and operational processes an integration of the disciplines has to be performed on each level. This means that for the level of standards needs to be proven that aspects of software engineering and usability engineering can coexist and can be integrated. On the level of process models it has to be ensured that usability engineering aspects can be incorporated with SE Models. And on the operational level activities a close collaboration needs to be achieved, resulting in acceptable additional organizational and operational efforts.

## 3 Proceedings

In order to identify the integration points between software engineering and usability engineering, the authors examined the three different levels, based on the hierarchies of standards, process models and operational processes (Figure 2):

1. On the abstract overarching level of Standards in software engineering and usability engineering, serving as a framework to ensure consistency, compatibility, exchangeability, and quality within and beyond the organizational borders and to cover the improvement of quality and communication.
2. On the level of Process Models for software engineering and usability engineering, to provide a procedural model and more refined approach that can serve as a framework for an organization, providing specific advantages and disadvantages, like predictability, risk management, coverage of complexity, generation of fast deliverables and outcomes, etc.
3. On the Operational Process level which reflects the execution of activities and the processing of information within the organization. It is an instance of the underlying model and the implementation of activities and information processing within the organization.

The goal of analysis on the level of standards is to identify similarities in the description of standards between SE and UE. They could be found in definitions of activities, tasks, goals, procedures or deliverables. With the focus on activities the authors will create a framework of activities, representing SE and UE likewise. Such a framework can be used to set limits for the following analysis, on the level of process models.

Based on the framework different SE Models are being analyzed in terms of how they already support the implementation of activities from a usability point of view. Criteria are being defined to measure the significance of UE activities within the SE Models. Based on the results and identified gaps recommendations for the enhancements of SE Models are being derived. These enable the implementation of activities on the level of models to ensure the development of user friendly solutions.

On the operational level the analysis is used to examine whether the recommendation meet the requirements of the practice. Measures regarding a specific SE Model in practice are being derived, evaluated and analyzed. As a result statements about the efficiency of the measures in making a contribution to the user-centeredness of the operational process could be made.

In this paper the authors will show the proceedings and first results of the analysis on the level of standards and of the level of process models. The derivation of recommendations, the refinement of the analysis methods and the analysis on the operational level are currently in progress and will be published by future work.

### 3.1 Analysis of Standards

To figure out whether software engineering and usability engineering have similarities on the level of standards, the standards' detailed descriptions of processes, activities and tasks, output artifacts, etc. have been analyzed and compared. For this the software engineering standard ISO/IEC 12207 was chosen to be compared with the usability engineering standard DIN EN ISO 13407.

The ISO/IEC 12207 defines the process of software development as a set of 11 activities: *Requirements Elicitation, System Requirements Analysis, Software Requirements Analysis, System Architecture Design, Software Design, Software Construction, Software Integration, Software Testing, System Integration, System Testing* and *Software Installation*. It also defines specific development tasks and details on the generated output to provide guidance for the implementation of the process.

The DIN EN ISO 13407 defines four activities of human-centered design that should take place during system development. These activities are the *Context of use, User Requirements, Produce Design Solutions und Evaluation of Use*. The DIN EN ISO 13407 also describes in detail the kind of output to be generated and how to achieve it.

On a high level, when examining the descriptions of each activity, by relating tasks and outputs with each other, similarities were found in terms of the characteristics, objectives and proceedings of activities. Based on these similarities single activities were consolidated as groups of activities (so called, Common Activities). These common activities are part of both disciplines software engineering and usability engineering on the high level of standards. An example of such a common activity is the Requirement Analysis. From a software engineering point of view (represented by the ISO/IEC 12207) the underlying activity is the *Requirement Elicitation*. From the usability engineering standpoint, specifically the DIN EN ISO 13407, the underlying activities are the Context of Use and User Requirements, which are grouped together. Another example is the Software Specification, which is represented by the two software engineering activities *System Requirements Analysis* and *Software Requirements Analysis*, as well as by Produce Design Solutions from a usability engineering perspective.

The result is a compilation of five common activities: Requirement Analysis, Software Specification, Software Design and Implementation, Software Validation, Evaluation that represent the process of development from both, a software engineering and a usability engineering point of view (Table 1).

These initial similarities between the two disciplines lead to the assumption of existing integration points on this overarching level of standards. Based on this, the authors used these five common activities as a general framework for the next level in the hierarchy, the level of process models.

However, the identification of these similar activities does not mean that one activity is performed in equal measure in SE and UE practice. They have same goals on the abstract level of standards but they differ in the execution at least on the operational level. Thus, Requirement Analysis in SE focuses mainly on system based requirements whereas UE requirements describe the users' needs and workflows. The activity of gathering requirements is equal but the view on the results is different. Another example is the Evaluation. SE evaluation aims at correctness and correctness of code whereas UE focuses on the completeness of users' workflows and the fulfillment of users' needs.

**Table 1.** Comparison of software engineering and usability engineering activities on the level of standards and the identified similarities (Common Activities)

| **ISO/IEC 12207**<br>**Sub- Process: Development** | **Common Activities** | **DIN EN ISO 13407** |
|---|---|---|
| Requirements Elicitation | **Requirement Analysis** | Context of Use<br>User Requirements |
| System Requirements Analysis<br>Software Requirements Analysis | **Software Specification** | Produce Design Solutions |
| System Architecture Design<br>Software Design<br>Software Construction<br>Software Integration | **Software Design and Implementation** | n/a |
| Software Testing<br>System Integration | **Software Validation** | Evaluation of Use |
| System Testing<br>Software Installation | **Evaluation** | Evaluation of Use |

Consequently it is important to consider these different facets of SE and UE likewise. And as usability has become an important quality aspect in software engineering, the identified common activities have not only to be incorporated in SE Models from a software engineering point of view, but also from the usability engineering point of view. Some SE models might already adhere to this but obviously not all of them. To identify whether usability engineering aspects of the common activities are already implemented in SE Models (or not), the authors performed a gap-analysis with selected SE Models. The overall goal of this was to identify integration points on the level of process models.

Therefore, the authors first needed a deep understanding about the selected SE Models and second, needed an accurate specification of the requirements that put demands on the SE Models from the usability engineering perspective, on which the SE Models then could be evaluated.

### 3.2 Analyzed SE Models

For the analysis of SE Models four commonly used models were selected: the Linear Sequential Model, the Evolutionary Development, the Spiral Model and the V-Model. They were examined and classified, in particular regards to their structural characteristics (e.g. classification of activities, proceedings, etc.), their specifics (e.g. abilities, disabilities, etc.) and their individual strengths and weaknesses.

The descriptions of the SE Models in literature served as the basis of the analysis. Improvements or extensions based on expert knowledge or practical experiences were not taken into account to retain the generality of statements. A sample of the results is represented in the following table (Table 2).

The gap-analysis surfaced particular characteristics of the considered models. Based on the identified strengths and weaknesses first indicators were derived that are in the authors eyes crucial for the model selection on the operational level. For example, the Evolutionary Development could be a good choice if the organization wants

to get results fast because of its ability to produce solution design successively and its ability to deal with unspecific requirements. A disadvantage of Evolutionary design is however, that due to the continuous changes and adjustments, the software quality and structure can suffer. However for the development of safety-relevant products,

**Table 2.** Strength/Weaknesses-Profiles of software engineering models

| | Basic properties | Specifics | Strength | Weakness |
|---|---|---|---|---|
| **Linear Sequential Model** | - division of the development process into sequent phases<br>- completeness of previous phase requirement for the next phase<br>- successive development<br>- iterations between contiguous phases<br>- deliverables define project's improvement | - document-driven<br>- phase-oriented | - controllable management<br>- controlling the complexity by using encapsulation | - lack of assistance with imprecise or incorrect product definitions<br>- problems with supplementary error identification and experiences from development |
| **Evolutionary Development** | - intertwined specification, development and evaluation phases<br>- no distinct phases<br>- successive requirement processing (and elicitation, if applicable)<br>- sequence of development cycles<br>- version increment at the end of every cycle<br>- no explicit but implicit maintenance phase<br>- high customer and user involvement | - successive solution design<br>- ability to deal with unspecific requirements<br>- avoids single-pass sequential, document-driven, gated-step approaches | - compilation of "quick solutions"<br>- ability to react to changing requirements<br>- small changes lead to measurable improvements<br>- user-oriented<br>- early identification of problems and shortcomings | - problems in software quality and structure caused by continuous changes and adoptions<br>- maintainability<br>- maintenance and quality of the documentation<br>- difficulties in measuring the project progress<br>- precondition is a flexible system |
| **Spiral Model** | - enhancement of the phase model<br>- phases are distributed in a spiral-shaped form<br>- development within four cycles<br>- evaluation, decision making, goal definition & planning of resources at end of each cycle | - successive solution design<br>- risk-driven | - risk management<br>- simultaneous control of budget and deliverables<br>- independent planning and budgeting of the single spiral cycles<br>- flexible, pure risk oriented but controlled response to current status | - high effort on management and planning |
| **V-Model** | - based on the Linear Sequential Model<br>- enhancement regarding quality assurance<br>- symmetric process<br>- evaluation reverse to system development<br>- evaluation on different levels of detail | - continuous evaluation<br>- quality assurance | - measures for continuous evaluation of the quality assurance<br>- verification and evaluation on all levels of detail | - initial planning efforts<br>- basically for large projects |

which need to adhere to a detailed specification, the Linear Sequential Model could be a good choice because of its stepwise and disciplined process. For developing new, complex and expensive software solutions the Spiral Model could be the method of choice because of its risk-oriented and successive development approach.

The results of the SE Model analysis are Strength/Weakness-Profiles that guide the selection of a specific SE Model based on organization specific criteria. Afterwards, with the detailed knowledge about the selected SE Models the maturity of these models in creating usable products was examined.

### 3.3 Gap-Analysis of SE Models

To assess the ability of SE Models to create usable products, requirements need to be defined first that contain the usability engineering demands and that can be used for evaluation later on.

As mentioned above the DIN EN ISO 13407 defines a process framework with the four activities Context of Use, User Requirements, Produce Design Solutions und Evaluation of Use. The reference model of the ISO/PAS 18152 represents an extension to parts of the DIN EN ISO 13407. Particularly the module *Human-centered design* of the ISO/PAS 18152 defines base practices for the four activities of the framework. These base practices describe in detail how the purpose of each activity is achieved. Thus, it is an extension on the operational process level. Since the ISO/PAS 18152 is aimed for processes assessments, its base practices describe the optimal steps. Therefore they can be used as usability engineering requirements that need to be applied by the SE Models to ensure to create usable products. According to this, there is an amount of requirements where each activity can be evaluated against. The following Table (Table 3) shows the base practices of the activity User Requirements.

**Table 3.** Base practices of the module HS.3.2 *User Requirements* given in the ISO/PAS 18152

| **HS.3.2 User Requirements** | |
|---|---|
| BP1 | Set and agree the expected behaviour and performance of the system with respect to the user. |
| BP2 | Develop an explicit statement of the user requirements for the system. |
| BP3 | Analyse the user requirements. |
| BP4 | Generate and agree on measurable criteria for the system in its intended context of use. |
| BP5 | Present these requirements to project stakeholders for use in the development and operation of the system. |

Based on these requirements (base practices) the authors evaluated the selected SE Models. The comparison was based on the description of the SE Models. For each requirement the authors determined whether the model complied to it or not. The results for each model and the regarding requirements are displayed in Table 4. The quantity of fulfilled requirements for each activity of the framework informs about the level of compliance of the SE Model satisfying the usability engineering requirements. According to the results statements about the ability of SE Models to create usable products were made. Table 5 shows the condensed result of the gap-analysis.

**Table 4.** Results of the gap-analysis: Coverage of the base practices for the Linear Sequential Model (LSM), Evolutionary Development (ED), Spiral Model (SM) and V-Model (VM)

| Modul | Activity | LSM | ED | SM | VM |
|---|---|---|---|---|---|
| **HS 3.1** | **Context of use** | | | | |
| 1 | Define the scope of the context of use for the system. | - | - | + | + |
| 2 | Analyse the tasks and worksystem. | - | - | - | + |
| 3 | Describe the characteristics of the users. | - | - | - | + |
| 4 | Describe the cultural environment/organizational/management regime. | - | - | - | + |
| 5 | Describe the characteristics of any equipment external to the system and the working environment. | - | - | - | + |
| 6 | Describe the location, workplace equipment and ambient conditions. | - | - | - | + |
| 7 | Analyse the implications of the context of use. | - | - | - | + |
| 8 | Present these issues to project stakeholders for use in the development or operation of the system. | - | + | - | - |
| **HS 3.2** | **User Requirements** | | | | |
| 1 | Set and agree the expected behaviour and performance of the system with respect to the user. | - | - | + | + |
| 2 | Develop an explicit statement of the user requirements for the system. | - | + | + | + |
| 3 | Analyse the user requirements. | - | + | + | + |
| 4 | Generate and agree on measurable criteria for the system in its intended context of use. | - | - | + | + |
| 5 | Present these requirements to project stakeholders for use in the development and operation of the system. | - | - | - | - |
| **HS 3.3** | **Produce design solutions** | | | | |
| 1 | Distribute functions between the human, machine and organizational elements of the system best able to fulfil each function. | - | - | - | - |
| 2 | Develop a practical model of the user's work from the requirements, context of use, allocation of function and design constraints for the system. | - | - | - | - |
| 3 | Produce designs for the user-related elements of the system that take account of the user requirements, context of use and HF data. | - | - | - | - |
| 4 | Produce a description of how the system will be used. | - | + | + | + |
| 5 | Revise design and safety features using feedback from evaluations. | - | + | + | + |
| **HS 3.4** | **Evaluation of use** | | | | |
| 1 | Plan the evaluation. | - | + | + | + |
| 2 | Identify and analyse the conditions under which a system is to be tested or otherwise evaluated. | - | - | + | + |
| 3 | Check that the system is fit for evaluation. | + | + | + | + |
| 4 | Carry out and analyse the evaluation according to the evaluation plan. | + | + | + | + |
| 5 | Understand and act on the results of the evaluation. | + | + | + | + |

The compilation of findings shows, that for none of the SE Models all Base Practices of ISO/PAS 18152 can be seen as fulfilled. However, there is also a large variability in the coverage rate between the SE Models. For example, the V-Model shows

a very good coverage for all modules except for smaller fulfillment of HS 3.3 Produce Design Solution criteria, whereas the Linear Sequential Model only fulfills a few of the HS 3.4 Evaluation of use criteria and none of the other modules.

Evolutionary Design and the Spiral Model share a similar pattern of findings, where they show only little coverage for Context of Use, medium to good coverage of User Requirements, limited coverage for Produce Design Solution and good support for Evaluation of Use activities.

**Table 5.** Results of the gap-analysis, showing the level of sufficiency of SE Models covering the requirements of usability engineering

| | **Context of Use** | **User Requirements** | **Produce Design Solutions** | **Evaluation of Use** | ***Across Activities*** |
|---|---|---|---|---|---|
| **Linear Sequential Model** | **0 %** | **0 %** | **0 %** | **60 %** | ***13 %*** |
| **Evolutionary Development** | **13 %** | **40 %** | **40 %** | **80 %** | ***39 %*** |
| **Spiral Model** | **13 %** | **80%** | **40 %** | **100 %** | ***52 %*** |
| **V-Modell** | **88 %** | **80 %** | **40 %** | **100 %** | ***78 %*** |
| ***Across Models*** | ***28 %*** | ***50 %*** | ***30 %*** | ***85 %*** | |

By looking at the summary of results (Table 5) and comparing the percentage of fulfilled requirements for each SE Model, it shows that the V-Model performs better than the other models and can be regarded as basically being able to produce usable products. With a percentage of 78% it is far ahead of the remaining three SE Models. In the comparison, the Linear Sequential Model cuts short by only 13%, followed by Evolutionary Development (39%) and the Spiral Model (52%).

If one takes both the average values of fulfilled requirements and the specific base practices for each usability engineering activity into account, it shows that the emphasis for all SE Models is laid on evaluation (Evaluation of Use), especially comparing the remaining activities. The lowest overall coverage could be found in the Context of Use and Produce Design Solution, indicating that three of the four SE models don't consider the relevant contextual factors of system usage sufficiently, and also don't include (user focused) concept and prototype work to an extent that can be deemed appropriate from a UCD perspective.

### 3.4 Interpretation and Results

Based on the relatively small compliance values for the Context of Use (28%), User Requirements (50%) and Produce Design Solutions (30%) activities across all SE

models, the authors see this as an indicator that there is only a loose integration between usability engineering and software engineering. There are less overlaps between the disciplines regarding these activities and therefore it is necessary to provide suitable interfaces to create a foundation for the integration.

The results of the gap-analysis can be used to extend the Strength/Weakness-Profiles in a way that these can be supplemented by statements about the ability of the SE Models to produce usable products. Thus, the quality criterion usability becomes an additional aspect of the profiles and for the selection of appropriate SE Models.

The presented approach does not only highlight weaknesses of SE Models regarding the usability engineering requirements and corresponding activities, it also pinpoints the potential for integration between software engineering and usability engineering:

- Where requirements are not considered as fulfilled, recommendations could be derived, which would contribute to an accomplishment.
- The underlying base practices and their detailed descriptions provide appropriate indices what needs to be considered in detail on the level of process models.

To give some examples, first high-level recommendations e.g. for the Linear Sequential Model could be made as followed: Besides phases likes *System Requirements* and *Software Requirements* there needs to be a separate phase for gathering user requirements and analysis of the context of use. As the model is document driven and completed documents are post-conditions for the next phase it has to be ensured that usability results are part of this documentation. The evaluation is a downstream phase that is performed after completing of the solution (or at least of a complex part of the solution). User centered validation aspects should take place already as early as possible, e.g. during the *Preliminary Design.* For the Spiral Model user validations should be introduced as an explicit step at the end of each cycle in order to avoid the risk of developing a non-usable solution.

According to the given approach and the results it shows that any SE Model can be similarly analyzed and mapped with the requirements of usability engineering to be then adapted or extended according to the recommendations based on the gap-analysis results in order to ensure the creation of usable products. This can be used a foundation for implementing the operational process level and will guarantee the interplay of software engineering and usability engineering in practice.

## 4 Summary and Outlook

The approach presented in this paper was used to identify integration points between software engineering and usability engineering on three different levels of abstractions. The authors showed that standards define an overarching framework for both disciplines. Process models describe systematic and planable approaches for the implementation and the operational process in which the process models are tailored to fit the specifics of an organization.

On the first level of standards the authors analyzed, compared and contrasted the software engineering standard ISO/IEC 12207 with the usability engineering standard

DIN EN ISO 13407 and identified common activities as part of both disciplines. They define the overarching framework for the next level of process models.

Based on this, the authors analyzed different software engineering process models. These models were classified, in particular regarding their structural characteristics (e.g. classification of activities, proceedings, etc.), their specifics (e.g. abilities, disabilities, etc.) and their individual strengths and weaknesses. As a result, the authors derived Strength/Weaknesses-Profiles for each model that helps organizations to select the appropriate process model for the implementation.

In order to identify the maturity of these software engineering process models' ability to create usable products, the authors synthesized demands of usability engineering and performed an assessment of the models. The results provide an overview about the degree of compliance of the models with usability engineering demands. It turned out that there is a relatively small compliance to the usability engineering activities across all software engineering models. This is an indicator that there only little integration between usability engineering and software engineering exists. There are less overlaps between the disciplines regarding these activities and therefore it is necessary to provide suitable interfaces to create a foundation for the integration.

But, the presented approach does not only highlight weaknesses of software engineering process models, it additionally identifies opportunities for the integration between software engineering and usability engineering. These can be used a foundation to implement the operational process level and will help to guarantee the interplay of software engineering and usability engineering in practice, which is part of the authors' future work.

However, the analysis results and regarding statements about the software engineering models are currently only based on their documented knowledge in literature. The authors are aware of the fact that there are several adoptions of the fundamental/basic models in theory and practice. Hence, in future research the authors will include more software engineering process models, even agile development models, to provide more guidance in selecting the most suitable model and to give more precise and appropriate criteria for selection.

The demands of usability engineering used in this paper are based on the base practices of the ISO/PAS 18152, which was a valid basis for a first analysis of the selected software engineering models. It is expected that there is a need for a different procedure in analyzing agile models because they are not as document and phase driven as classical software engineering models and the ISO/PAS 18152 are. The authors will apply the given approach to evaluate whether agile process models are better able to suit the demands of usability engineering than formalized approaches compared in this paper.

Regarding the current procedure the authors discovered that more detailed/adequate criteria for the assessment are necessary by which objective and reliable statements about process models and their ability to create usable software could be made. Therefore the authors plan to conduct expert interviews as a follow-up task to elicit appropriate criteria for the evaluation of SE models. Based on these criteria the authors will perform another gap-analysis of selected software engineering models (including agile approaches). The authors expect to derive specific recommendations to enrich the SE Models by adding or adapting usability engineering activities, phases, artifacts, etc. By doing this, the development of usable software on the level of proc-

ess models will be guaranteed. Furthermore, hypothesizes about the process improvements are expected to be made for each recommendation which then can be evaluated on the Operational Process level. Therefore, case studies will be identified based on which the recommendations could be transferred in concrete measures. These measures will then be evaluated by field-testing to verify their efficiency of user-centeredness of software engineering activities. This will help to derive concrete measures that result in better integration of software engineering and usability engineering in practice and hopefully more usable products.

## References

1. Boehm, B.: A Spiral Model of Software Development and Enhancement. IEEE Computer 21, 61–72 (1988)
2. Cooper, A., Reimann, R.: About Face 2.0. Wiley, Indianapolis (2003)
3. DIN EN ISO 13407. Human-centered design processes for interactive systems. CEN - European Committee for Standardization, Brussels (1999)
4. Faulkner, X.: Usability Engineering, pp. 10–12. PALGARVE, New York (2000)
5. Glinz, M.: Eine geführte Tour durch die Landschaft der Software-Prozesse und –Prozessverbesserung. Informatik – Informatique, 7–15 (6/1999)
6. IBM: Ease of Use Model (November 2004), http://www-3.ibm.com/ibm/easy/eou_ext.nsf/publish/1996
7. ISO/IEC 12207. Information technology - Software life cycle processes. Amendment 1, 2002-05-01. ISO copyright office, Switzerland (2002)
8. ISO/PAS 18152. Ergonomics of human-system interaction — Specification for the process assessment of human-system issues. First Edition 2003-10-01. ISO copyright office, Switzerland (2003)
9. KBST: V-Modell 97 (May 2006), http://www.kbst.bund.de
10. Larman, C., Basili, V.R.: Iterative and Incremental Development: A Brief History. Computer 36(6), 47–56 (2003)
11. Mayhew, D.J.: The Usablility Engineering Lifecycle. Morgan Kaufmann, San Francisco (1999)
12. McCracken, D.D., Jackson, M.A.: Life-Cycle Concept Considered Harm-ful. ACM Software Engineering Notes, 29–32 (April 1982)
13. Pagel, B., Six, H.: Software Engineering: Die Phasen der Softwareentwicklung, 1st edn., vol. 1. Addison-Wesley Publishing Company, Bonn (1994)
14. Patel, D., Wang, Y. (eds.): Annals of Software Engineering. In: Editors' introduction: Comparative software engineering: Review and perspectives, vol. 10, pp. 1–10. Springer, Netherlands (2000)
15. Royce, W.W.: Managing the Delopment of Large Software Systems. Proceedings IEEE, 328–338 (1970)
16. Seffah, A. (ed.): Human-Centered Software Engineering – Integrating Usability in the Development Process, pp. 3–14. Springer, Dordrecht (2005)
17. Sommerville, I.: Software Engineering, 7th edn. Pearson Education Limited, Essex (2004)
18. Woletz, N.: Evaluation eines User-Centred Design-Prozessassessments - Empirische Untersuchung der Qualität und Gebrauchstauglichkeit im praktischen Einsatz. Doctoral Thesis. University of Paderborn, Paderborn, Germany (April 2006)

## Questions

***Jan Gulliksen:***
*Question: One thing I miss is one of the key benefits of the ISO standards, namely the key values provided by the principles. Your analysis of process steps fails to address these four values. The software engineering process covers much more.*

Answer: The goal is to be more specific in describing the assessment criteria and therefore it does not address the principles. We plan to develop more specific criteria through interviews and use these criteria to assess the process models including software engineering models in more detail. Then we will go back to the companies to see how they fit.

***Ann Blandford:***
*Question: Work is analytical looking at how things should be – what confidence do you have about how these things can work in practice? Methods are always subverted and changed in practice anyway.*

Answer: Documentation is not representative enough – plan to do more specific work with experts in the field. SE experts could answer, for example, whether the criteria for usability engineering fit into an underlying model. We will then map these to criteria in order apply them in practice.