20

Integrating Software Engineering and Usability Engineering

Karsten Nebe[1], Dirk Zimmermann and Volker Paelke[2]
[1]*University of Paderborn (C-LAB),* [2]*Leibniz University of Hannover (IGK)*
Germany

1. Introduction

The usability of products gains in importance not only for the users of a system but also for manufacturing organizations. According to Jokela, the advantages for users are far-reaching and include increased productivity, improved quality of work, and increased user satisfaction. Manufacturers also profit significantly through a reduction of support and training costs (Jokela, 2001). The quality of products ranks among the most important aspects for manufacturers in competitive markets and the software industry is no exception to this. One of the central quality attributes for interactive systems is their usability (Bevan, 1999) and the main standardization organizations (IEEE 98, ISO 91) have addressed this parameter for a long time (Granollers, 2002). In recent years more and more software manufacturer consider the usability of their products as a strategic goal due to market pressures. Consequently, an increasing number of software manufacturer are pursuing the goal of integrating usability practices into their software engineering processes (Juristo et al., 2001). While usability is already regarded as an essential part of software quality (Juristo et al., 2001) the practical implementation often turns out to be a challenge. One key difficulty is usually the integration of methods, activities and artefacts from usability engineering into the existing structures of an organization, which typically already embody an established process model for product development and implementation. Uncoordinated usability activities that arise frequently in practice have only a small influence on the usability of a product. In practice the activities, processes and models applied during development are usually those proposed by software engineering (Granollers et al., 2002). A systematic and sustainable approach for the integration of usability activities into existing processes is required.

In order to align the activities from both software engineering (SE) and usability engineering (UE) it is necessary to identify appropriate interfaces by which the activities and artefacts can be integrated smoothly into a coherent development process. The central goal of such integration is to combine the quality benefits of usability engineering with the systematic and planable proceedings of software engineering. This chapter provides a starting point for such integration. We begin with a discussion of the similarities and differences between both disciplines and review the connection between models, standards and the operational processes. We then introduce integration strategies at three levels of abstractions: the level of

standards in SE and UE, the level of process models, as well as the level of operational processes.
On the most abstract level of standards we have analyzed and compared the software engineering standard ISO/IEC 12207 with the usability engineering standard DIN EN ISO 13407 and identified 'common activities'. These define the overarching framework for the next level of process models.
Based on this, we have analyzed different SE process models at the next level of abstraction. In order to quantify the ability of SE process models to create usable products, a criteria catalogue with demands from usability engineering was defined and used to assess common SE models. The results provide an overview about the degree of compliance of existing SE models with UE demands. This overview not only highlights weaknesses of SE process models with regards to usability engineering, but also serves to identify opportunities for improved integration between SE and UE activities.
These opportunities can form the foundation for the most concrete implementation of an integrated development approach at the operational process level. We present recommendations based on the results of the analysis.

2. Software Engineering

Software engineering (SE) 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 (Patel & Wang, 2000; Sommerville, 2004). SE tries to provide a systematic and planable approach for software development. To achieve this, it provides comprehensive, systematic and manageable procedures, in terms of SE 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 results do not depend on individual efforts of particular people or fortunate circumstances (Glinz, 1999). 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 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) (Royce, 1970), Evolutionary Software Development (McCracken, & Jackson, 1982), the Spiral Model by Boehm (Boehm, 1988), or the V-Model (KBST, 1997).
SE 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.' (ISO/IEC, 2002). It defines processes, activities and tasks and provides descriptions about how to perform these items on an abstract level.

In order to fulfil the conditions of SE standards (and the associated claim of ensuring quality) SE models should comply with these conditions. In general, standards as well as SE models cannot be directly applied. They have to be adapted and/or tailored to the specific 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 SE: 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).

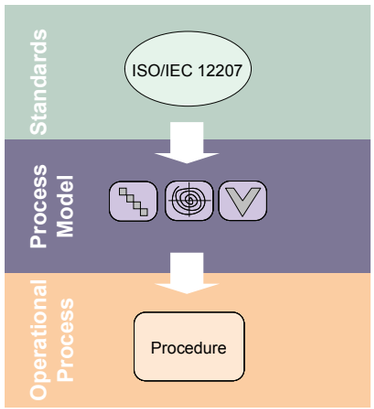

Figure 1. Hierarchy of standards, process models and operational processes in SE

3. Usability Engineering

Usability Engineering (UE) is a discipline that is concerned with the question of how to design software that is easy to use (usable). UE 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.' (Faulkner, 2002a)
UE 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. Examples include Goal-Directed-Design (Cooper & Reimann, 2003), the UE Lifecycle (Mayhew, 1999) or the User-Centred Design-Process Model of IBM (IBM, 1996). All of them have much in common since they describe an idealized approach that ensures the development of usable software, but they differ in their specifics, in the applied methods and the general description of the procedure (e.g. phases, dependencies, goals, responsibilities, etc.) (Woletz, 2006). 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, UE and SE address different needs in the development of software. SE aims at systematic, controllable and manageable approaches to software development, whereas UE 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 lead to conflicts, e.g. when SE focuses on system requirements and the implementation of system concepts and designs, whereas UE focuses on the implementation of user requirements, interaction concepts and designs. For successful designs both views need to be considered and careful trade-offs are required.

UE provides standards similar to the way SE does. These are also intended to serve as frameworks to ensure consistency, compatibility, exchangeability, and quality, which is in line with the idea of SE standards. However, UE standards focus on the users and the construction of usable solutions. Examples for such standards are the DIN EN ISO 13407 (1999) and the ISO/PAS 18152 (2003).

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

- The active involvement of users and a clear understanding of user and task requirements ('context of use')
- An appropriate allocation of function between users and technology ('user requirements')
- The iteration of design solutions ('produce design solutions')
- Multi-disciplinary design ('evaluation of use')

These characteristics are reflected by the activities (named in brackets), which define the process framework of the human-centred 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 artefacts associated with the outcomes of each process and activity.

There is a sub-process called human-centred design that describes the activities that are commonly associated with a user centred 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. They are, however, more specific in terms of defining lists of activities (so called base practices) that describe how the purpose of each activity can be achieved (e.g. what needs to be done to gather the user requirements in the right way). Thus, 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 identified for SE above. Similar to SE, there is also a hierarchy of standards and subsequent process models.

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 SE and for UE (Figure 2). Standards define the overarching framework, models describe systematic and traceable approaches and on the operational level, the SE models are adjusted and put into practice.

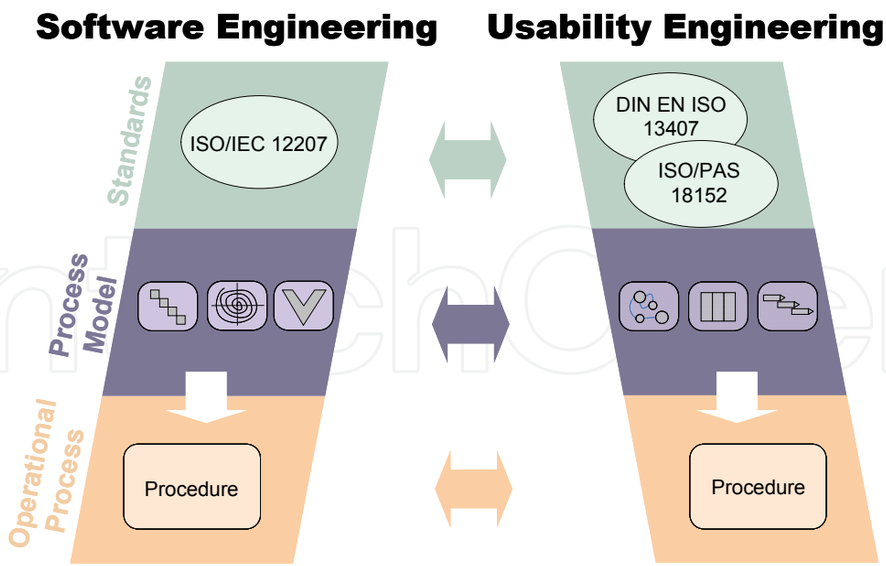

Figure 2. Similar hierarchies in the two disciplines SE and UE: standards, process models and operational processes

4. State of the Art: Integration of Standards, Models and Operational Processes

In general, standards and models are seldom applied directly, neither in SE nor in UE. Standards merely define a framework to ensure compatibility and consistency and to set quality standards. For practical usage, models are being adapted and tailored to organizational conditions and constraints, such as existing processes, organizational or project goals, legal policies, etc. Thus models are refined by the selection and definition of activities, tasks, methods, roles, deliverables, etc. as well as the responsibilities and relationships in between. The derived instantiation of a model, fitted to the organizational aspects, is called software development process (for SE models) or usability lifecycle (for UE models). The resulting operational processes can be viewed as instances of the underlying model. This applies to both SE and UE.

In order to achieve sufficient alignment between the two disciplines, all three levels of abstraction must be considered to ensure that the integration points and suggestions for collaboration meet the objectives of both sides and the intentions behind a standard, model or operational implementation is not lost.

The integration of SE and UE leads to structural, organisational and implementation challenges, caused by 'differences relating to historical evolution, training, professional orientation, and technical focus, as well as in methods, tools, models, and techniques' (Constantine et al., 2003).

As discussed in sections 2 and 3, SE and UE have evolved with different objectives. SE's main goal is the design of software, covering the process of construction, architecture, reliable functioning and design for utility (Sutcliffe, 2005). Historically, UE has been considered in contrast to this system-driven philosophy of SE (Norman & Draper, 1986) and focuses on social, cognitive and interactive phenomena (Sutcliffe, 2005). This conceptual difference between the two disciplines highlights the challenges for integration in practice.

A lack of mutual understanding between practitioners of both disciplines is common. As reported by Jerome and Kazmann (2005) there is substantial incomprehension of the other field between software engineers and usability specialists. Apparently, scientific insights on design processes have had little impact on the exchange and communication so far. Practitioners from both disciplines commonly hold widely disparate views of their respective roles in development processes. This leads to a lack of communication and collaboration. Often cooperation is deferred to late stages of the development lifecycle where it is usually too late to address fundamental usability problems.

Hakiel (1997) identifies additional areas where understanding is lacking. A central problem is the perception of many project managers that the activities and results of UE are irreproducible and result from unstructured activities. This perception is in part caused by the missing interrelation/links between activities of SE and UE and indicates the need for the education of project management about UE goals and practices (Faulkner & Culwin, 2000b; Venturi & Troost, 2004). This is a prerequisite to achieve adequate executive and managerial support and to consider UE in the project planning from the outset (staffing, organisational structure, scheduling of activities) (Radle & Young 2001).

Many of the mentioned problems are caused by the act of changing established processes by adding and adapting UE activities. Particularly there, where formal SE process models are applied UE activities are rarely considered as a priority. Granollers et al. (2002) summarize this in their study: „the development models used by the software industry in the production of solutions are still those proposed by SE. SE is the driving force and the UE needs to adapt to this in order to survive'. While so called UE models, which address the complete development from the usability perspective, have been proposed (e.g. Mayhew, 1999, Cooper & Reimann 2001), a complete replacement of SE engineering processes by these is often infeasible. Rather, it seems necessary to gather knowledge about the fundamental concepts, the methodologies and techniques of UE in a form that is accessible to all stakeholders in a development process and to incorporate this knowledge and the corresponding activities into existing software development processes (Aikio, 2006). Early on Rideout (1989) have identified the use of effective interdisciplinary design teams, the creation and dissemination of company and industry standards and guidelines, and the extension of existing processes to encompass UE concerns as keys to the acceptance and success of UE. Most existing approaches have tried to achieve unification at specific levels of abstraction. While this can lead to successful results, it fails to provide a continuous strategy for integration that considers the structural, organizational and operational aspects on equal footing and is adaptable to a variety of established and evolving SE practices. This has been illustrated by the advent of so called agile development approaches that have become popular in recent years, resulting in the need for newly adapted strategies for UE integration. In the following sections we introduce a systematic approach that covers the three levels of abstraction from standards over process models to operational processes to address this challenge in a systematic way.

5. Integration on three Levels of Abstraction

In order to identify integration points between the two disciplines examination and analysis has to be performed on each level of the abstraction hierarchy: On the most abstract level of standards it has to be shown that the central aspects of SE and UE can coexist and can be integrated. On the level of process models it has to be analyzed how UE aspects can be

incorporated into SE models. And on the operational level concrete recommendations for activities have to be formulated to enable effective collaboration with reasonable organizational and operational efforts. In previous work the authors have addressed individual aspects of this integration approach (Nebe & Zimmermann, 2007a; Nebe & Zimmermann, 2007b; Nebe et al., 2007c), which are now composed into a coherent system.

5.1 Common Framework on the Level of Standards

To figure out whether SE and UE have similarities on the level of standards, the standards' detailed descriptions of processes, activities and tasks, output artefacts, etc. have been analyzed and compared. For this the SE standard ISO/IEC 12207 was chosen for comparison with the UE 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-centred design that should take place during system development. These activities are labelled 'context of use', 'user requirements', 'produce design solutions' und 'evaluation of use'. 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 can be identified 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 SE and UE on the highest level of standards. An example of such a common activity is the Requirement Analysis. From a SE point of view (represented by the ISO/IEC 12207) the underlying activity is the Requirement Elicitation. From the UE 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 SE activities System Requirements Analysis and Software Requirements Analysis, as well as by 'produce design solutions' from a UE 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 SE and a UE 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 similar ways in SE and UE practice. They may have similar goals on the abstract level of standards but typically differ significantly 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 identical but the view on the results is different.

Another example is the Evaluation. SE evaluation aims at functional correctness and correctness of code whereas UE focuses on the completeness of users' workflows and the fulfilment of users' needs.

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

Table 1. Comparison of SE and UE activities on the level of standards and the identified similarities (Common Activities)

Consequently, it is important to consider these different facets of SE and UE likewise. And as usability becomes an important quality aspect in SE, the 'common activities' not only have to be incorporated in SE models from a SE point of view, but also from the UE point of view. Some SE models already adhere to this, but obviously not all of them. To identify whether UE aspects of the 'common activities' are already implemented in SE models, the authors performed a gap-analysis with selected SE models. The overall goal of this study was to identify integration points on the level of process models.
Therefore a deep understanding about the selected SE models and an accurate specification of the requirements that specify demands from an UE perspective was required.

5.2 Ability of SE models to Create Usable Products

In order to assess the ability of SE models to create usable products, criteria are needed to define the degree of UE coverage in SE models. These criteria must contain the UE demands and should be used for evaluation later on.
To obtain detailed knowledge about UE activities, methods, deliverables and their regarding quality aspects, the authors analyzed the DIN EN ISO 13407 and the ISO/PAS 18152.
As mentioned above the DIN EN ISO 13407 defines a process framework with the four activities 'context of use', 'user requirements', 'produce design solutions' and '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-centred 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 at process assessments, its base practices describe the established steps. Therefore they can be used as UE requirements that

need to be applied by the SE models to ensure to create usable products. For the following analysis they form the basic requirements against which each activity is evaluated. As an example table 2 shows the base practices of the activity 'user requirements'.

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.

Table 2. Base practices of the module HS.3.2 User Requirements given in the ISO/PAS 18152.

Based on these requirements (base practices) the authors evaluated the selected SE models. The comparison was based on the description of the SE models in the standard documents and official documentation. For each requirement the authors determined whether the model complied with it or not. An overview of the results for each model is shown in table 3. The quantity of fulfilled requirements for each activity of the framework provides some indication of the level of compliance of the SE model with the UE requirements. According to the results, statements about the ability of SE models to create usable products were made. Table 4 shows the condensed result of the gap-analysis.

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 fulfilment of 'produce design solutions' HS 3.3 criteria, whereas the Linear Sequential Model only fulfils a few of the 'evaluation of use' (HS 3.4) criteria and none of the other modules.

Evolutionary Design and the Spiral Model share a similar pattern, 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.

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.	+	+	+	+

Table 3. 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)

	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 %	

Table 4. Results of the gap-analysis, showing the level of sufficiency of SE models covering the requirements of UE

The summary of results (Table 4) and a comparison of the percentage of fulfilled requirements for each SE model, 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 falls short at 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 UE activity into account, this analysis 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.

The relatively small compliance values for the 'context of use' (28%), 'user requirements' (50%) and 'produce design solutions' (30%) activities across all SE models, can be interpreted as an indicator that there is only a loose integration between UE and SE. There are few overlaps between the disciplines regarding these activities and therefore it is necessary to provide suitable interfaces to create a foundation for integration.

This approach does not only highlight weaknesses of SE models regarding the UE requirements and corresponding activities, it also pinpoints the potential for integration between SE and UE: Where requirements are currently considered as not fulfilled, recommendations for better integration can be derived.

The underlying base practices and the corresponding detailed descriptions provide indicatory on what needs to be considered on the level of process models.

As an example, initial high-level recommendations e.g. for the Linear Sequential Model could be as followed: In addition to 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 approach can be applied in a similar way to any SE model, to establish the current level of integration and to identify areas for improved integration. This can be used a foundation for implementing the operational process level and will improve the interplay of SE and UE in practice.
The results confirmed the expectations of the authors, indicating deficiencies in the level of integration between both disciplines on the level of the overarching process models. Thus, there is a clear need to compile more specific and detailed criteria for the assessment of the SE models. The analysis also showed that the base practices currently leave too much leeway for interpretations. In addition, it turned out that a dichotomous assessment scale (in terms of 'not fulfilled' or 'fulfilled') is not sufficient. A finer rating is necessary to evaluate process models adequately. Thus, the documentation analysis of the SE models produced first insights but it turned out that the documentation is not comprehensive enough to ensure the validity of recommendations derived from this analysis alone.

6. Detailed Criteria of Usability: Quality aspects in UE

In order to gather more specific and detailed criteria for the assessment of SE models and the derivation of recommendations a further analysis had to be performed.
The authors believe that there is such thing as a 'common understanding' in terms of what experts think of when they talk about UE and this is certainly represented by the definition of the human-centred-design process in the DIN EN ISO 13407. Although the definitions of base practices defined in the ISO/PAS 18152 are not considered as invalid they leave leeway for interpretation as shown in section 5.
However, while this 'common understanding' seems true on a very abstract level, strong differences in how to implement these in practice can be expected. The key question therefore is not only what should be done, but rather how it can be assured that everything needed is being performed (or guaranteed) in order to gain a certain quality of a result, an activity or the process itself. In addition, the completeness and correctness of the base practices and human-centred design activities as defined in the ISO/PAS 18152 itself needs to be verified.
For a more detailed analysis, based on current real-world work practices, the authors performed semi-structured interviews and questionnaires with six experts in the field of UE. These experts are well grounded in theoretical terms, i.e. standards and process models, as well as in usability practice. As a result, overarching process- and quality characteristics were derived that led to statements about the relevance, the application and need of usability activities, methods and artefacts to be implemented in SE.
The following results highlight initial insights, especially with regards to the quality characteristics/aspects of UE activities.
A substantial part of the interviews referred explicitly to quality characteristics/aspects of the four human-centred design activities of the DIN EN ISO 13407: 'context of use', 'user requirements', 'produce design solutions' and 'evaluation of use'. The goal was to identify what constitutes the quality of a certain activity from the experts' point of view and what kind of success and quality criteria exist that are relevant on a process level and subsequently for the implementation in practice.

In summary, it can be said that the quality of the four activities essentially depends on the production and subsequent treatment of the result generated by each activity. From the quality perspective it is less important how something is accomplished, but rather to guarantee the quality of the results. In order to answer the question what constitutes this quality, the analyzed statements of the experts regarding each activity are discussed in the following paragraphs. The core characteristic (essence) of each activity is described, followed by requirements regarding the generation and treatment of content, a summary of (measurable) quality criteria and success characteristics, as well as a list of operational measures that can be used for the implementation in practice.

6.1 Context of Use

There is a common agreement of all experts involved in the study in that the fundamental goal of the activity is the formation of a deep understanding of the users, their goals, needs and the actual work context. This then forms the base for the derivation of requirements as well as for validating user requirements of the solution. The focus of the context analysis should therefore be on the original tasks and workflows, independent of concrete solutions and/or any supporting systems.

Apart from a documentation or rather communication of the analyzed knowledge it is crucial to anchor the activity within the overall process model. The context analysis provides a base for the entire process of development, in particular for deriving requirements, which provide the link to the next process step/sub-process. It generally applies that an output is only as valuable as it serves as input for a following sub-process. Thus, the context analysis must start at an early stage, preferably before any comprehensive specification is being created. This could be in a pre-study phase of the project, where neither technical platform nor details about the implementation have been determined. The quality of the context analysis and its results also depends on organisational conditions. A sufficient time schedule and the allocation of adequate resources is required in order to be able to accomplish the context analysis in an appropriate way. The support by the management and the organization is crucial in this case.

As mentioned before, the results are the most significant factor for the quality. In their comments, most experts focused on the documentation of the results, but it turned out that this is not a necessity. The important aspect is not the documentation itself. Rather, the results must exist in an adequate form so that all people involved can access or read it, and that it is comprehensible for anyone involved. With regard to the description (respectively the communication) of context information two major points have been identified that need to be considered and distinguished: First, the context information must be formulated in an appropriate way so that even people who were not involved in the process of analysis can comprehensibly understand its content. And second, the information must be unambiguous, consistent and complete. That is, only reasonable and context-relevant data is gathered and completeness (pertaining to the systems that are essential for the accomplishment of the tasks) is assured.

One major quality aspect of a successful context analysis is the ability to identify so called ‘implied needs’ based on the context information. Implied needs are ‘those needs, which are often hidden or implied in circumstances in which requirements engineers must sharpen their understanding of the customer's preferred behavior’ (ProContext, 2008). Hence, the context information must contain all details needed to derive the user requirements.

All experts agreed on the fact that the quality of the results strongly depends on the experience and qualification of the analysts. The abilities to focus on the substantial and to focus on facts (rather that interpretation) are crucial for this activity. Additionally, the experiences of the users are important as well. Their representativeness, ability to express themselves and the validity of their statements are a prerequisite for good results.

A measureable quality criterion is the amount of predictable user requirements derived based on the context information. One example: If five comparable skilled experts are asked to derive all user requirements based on the given context information independently, and if all experts finally compile a similar set of requirements in terms of quantity and meaning, then the context information could be rated as high quality. This implies that the entire user requirements could be derived based on the context information.

A success criterion for the analysis and its context information is the ability to identify coherent patterns, e.g. similar workflows, similar habits, the usage of similar tools, etc. Accordingly, the identification of non-similar workflows might imply the need for further analysis. Another success factor is the feedback obtained from presenting the results not only to the users of the system but also to the customers. Good feedback from the users implies a good solution and in combination with good customer feedback it indicates a good balance between user and business goals.

The success of a context analysis activity itself can often only be estimated/evaluated afterwards, then when the users have not found any major gaps or critical errors in the concept of the solution. Summative evaluation is a suitable method to measure this.

An additional, but difficult to quantify, quality criterion is the acceptance and the utility of the results (context information) for the process and for the organization. If the context information enables the derivation of user requirements and if it is useful to create concepts and designs out of it than it has been obviously good.

A central characteristic of good context information is that all questions that appear during the design process can be answered from it.

An important measure for ensuring the quality is the training of the analysts in applying the methodologies and methods. The qualification of the analysts determines the result's quality. Qualification means not only to know (theoretically learned), but rather to perform (practically experienced). The context analysis is the corner stone for the user-centred development and it is crucial for the success of a solution. In order to ensure its quality, it requires the integration of the activities and regarding results (deliverables) into the overall process, the supply of sufficient and qualified resources as well as appropriate time for the execution of an entire analysis within the project plan. Therefore, the support by the management and the organization is necessary.

6.2 User Requirements

The main goal of the activity 'user requirements' is to work out a deep understanding about the organisational and technical requirements, the users' workflows and the users' needs and goals of the future system. This results in a valid basis for system specifications. From the UE perspective it is crucial that this is not purely technically driven perception (as often in SE processes) but extents to a utilization- and situational-view.

The majority of the experts described the core of this activity as the specification and documentation of requirements in coordination with development, users and customers. However, the result does not always have to be defined in the way of fine-granular

requirements and extensive specifications. More important is to transfer this knowledge successfully into the development process. Similar to the activity of context analysis, both the experience and skills of the analysts are essential for the success and for the quality of the results, as well as the users' representativeness, ability to express themselves and the validity of their statements.

According to statements of the experts, a set of quality criteria can be defined, both at the execution of the activity, and at the result. User requirements should have a certain formulation quality (legibility, comprehensibility, consistency, etc.) and should be formulated system-neutrally. Requirements should be based on demands (prerequisite for something that should be accomplished), which guaranteed the validity. This can be important for the argumentation when trade-offs in the development process are required. User requirements must be adequate and precisely formulated on level of the tasks. A negative example would be: „The system must be usable' - this requirement is not embodied at a task level, it is just an abstract goal. Good user requirements are not interpretable per se. The solution itself derived from the requirements is interpretable of course, but the requirements are not.

In particular the consideration of user goals and requirements and the trade-off with business goals is seen as an important success criterion. Further success criteria are the comprehensibility and utility of the results (requirements) in the further process.

In order to achieve this quality the experts recommended several measures, such as an iterative procedure during the collection and derivation of requirements. The involvement of all stakeholders (users, customer, management, development, etc.) at all stages of the process is inevitable. In particular the presentation of user requirements to the management is considered as an important means for arising awareness. Dedicated and qualified roles are also crucial for the success of the entire process (e.g. no developer should write the specification – this should be done by someone qualified and skilled).

6.3 Produce Design Solution

The design activity Produce Solution covers the creative process of transferring user requirements and knowledge about the user domain and the business perspective into design concepts for new solutions. For this, different ideas have to be produced, investigated and critiqued. As most design activities, this is an open-ended problem solving process without a unique solution, especially regarding the user interface. The main goal is to provide a functional system with a user interface that allows satisfactory interaction and efficient use (all information, no errors, no unnecessary steps, etc.), as described in the seven ISO dialogue design principles (DIN EN ISO 9241-Part 110, 2006).

Essential for a good design solution is that it handles all requirements collected in the specification (validity) but also (increasingly) that the user feels safe and confident in the use of the designed system and feels satisfied with the results. The form in which the design is created, communicated and documented is not the decisive factor. Representations can range from sketches over formal specifications to working prototypes. What is central is that the representation provides information at a level of detail that is deemed useful for implementation and can be successfully communicated.

As in most design disciplines it is considered good practice to consider design alternatives that are critiqued by experts and evaluated with users to guide the design process. Process support for such exploratory activities is considered useful and important. If questions from

the users - but also from the development team - arise during these reviews it is often a strong indication that further analysis and review are required. The quality of the proposed solutions depends critically on the experience and knowledge of the persons involved in the design work. The experts consider a multidisciplinary qualification, but with a clear account of the assigned roles and competencies (developer, analyst, designer, etc.) as the key to the production of successful high quality design.

With regards to design, criteria for success and measurable quality criteria can be difficult to define. What can, however, be checked and measured are for example, the implementation of user requirements through the design, the design accordance with the principles of dialogue design (DIN EN ISO 9241-110, 2006) and information presentation standards (DIN EN ISO 9241-Part 12, 1998). Comparative studies of alternative designs and performance measurements can provide further information on design quality to guide further refinement or to identify problematic design choices.

Appropriate measures for quality assurance are to strengthen the links with the related process activities of 'user requirements' and the 'evaluation of use'. An established best practice is the use of iterative approaches in which alternative design proposals are examined and refined to evolve a suitable solution. To ensure a wide creative potential and a correspondingly large solution space the integration of a variety of roles and experts from different backgrounds into this activity is advisable. It is, however, vital that the leadership of this activity is assigned to a qualified user interface designer, whose role is explicitly communicated and who has the final decision power in this activity. This must be communicated to the complete team and adequately supported by the management.

6.4 Evaluation of Use

The central aim of the 'evaluation of use' activity is to collect feedback on the practical use of a system design in order to refine the design and the system. Problems in use are to be identified and subsequently corrected. The key to a successful integration of evaluation into a development process is to consider it as a continuous ongoing activity throughout the process. Of course, the methodology and applied techniques have to be adapted to the maturity of the design representations/implementation at different stages of development. Evaluation is most useful in the larger process context, as many evaluation methods identify problems, but provide no solution for the problems found. It is therefore important to not only identify problems but to feed this information back into the design process to resolve the identified issues. This should be reflected in the process. It is critical that the results of this activity are used in the ongoing process. Therefore the information must be available in a form that is understandable to the stakeholders in the development process.

The quality of the activity 'evaluation of use' is primarily defined by the results. In this activity what was previously designed (based on the preceding analysis) and implemented is now evaluated. It is crucial that mayor usability problems are identified at this stage, especially issues that are seen as disruptive from the user's perspective. Second categories of problems cover issues that are valid but do not necessarily disturb users, which are less critical. The activity can be considered completed, once no new significant problems are found. The commitment of all stakeholders, the expertise of usability experts and the ability of stakeholders to accept criticism and to act constructively on it are of central importance for the success of the evaluation activity. Measurable quality criteria can be derived from the selection and use of established evaluation methods (Freymann, M. 2007).

Decisive measures for quality assurance include the qualification of key stakeholders in evaluation techniques and process, the explication of the evaluation as an essential activity in the process and the allocation of sufficient time for multiple iterations of design-evaluation cycles in the process plan.

7. Summary & Outlook

Today, the usability of a software product is no longer only a crucial quality criterion for the users but also for the organizations. It can be seen as unique selling point in the competitive market. However, the various differences in each organisation's structure, its process model, development process and the organizational conditions makes it difficult to transfer the knowledge about usability methodologies, methods and techniques into practice. There are many different approaches to integrate usability engineering and software engineering but each focuses on different measures. Some are very specific, e.g. to an organization or to a development process, while others are very abstract, e.g. to process models. However, each of them has its authority and each leads to the regarding results in detail on a specific level of abstraction. However, there are fewer approaches that combine approaches on all levels of abstractions. The presented approach identifies 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 ('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 a set of 'common activities' as part of both disciplines. These activities define the overarching framework for the next level, the 'level of process models'.

In order to identify the maturity of software engineering process models' ability to create usable products, the authors used a two-step approach to synthesize the demands of usability engineering and performed an assessment of selected software engineering models.

To obtain detailed knowledge about usability engineering activities, methods, deliverables and their regarding quality aspects, the authors analyzed the two usability engineering standards DIN EN ISO 13407 and the ISO/PAS 18152. The ISO/PAS 18152 defines detailed base practices that specify the tasks for creating usable products. These base practices have been used as a foundation to derive requirements that represent the 'common activities'' usability engineering perspective. The quantity of fulfilled requirements for each activity of the framework informs about the level of compliance of the software engineering model satisfying the base practices and therewith the usability perspective of activities.

The results of the assessment provide an overview about the degree of compliance of the selected models with usability engineering demands. It turned out that there is a relatively small compliance to the usability engineering activities across all selected software engineering models. This is an indicator that 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.

The analysis of software engineering models also showed that more detailed and 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 performed a second analysis and conducted expert interviews and questionnaires to elicit appropriate criteria for the evaluation of software engineering models. A substantial part of the questions referred explicitly to quality characteristics/aspects of the four human-centred design activities of the DIN EN ISO 13407: 'context of use', 'user requirements', 'produce design solutions' and 'evaluation of use'. The goal was to identify, what constitutes the quality of a certain activity from the experts' point of view and what kind of success and quality criteria exist that are relevant on a process level and subsequently for the implementation in practice.

The results highlight first insights especially regarding quality aspects of usability engineering activities. Even if these could not be generalized, the results reflect the experts' fundamental tendencies and opinions.

Beyond this, it could have been proved that there is such thing as 'common mind' in terms of what experts think of when they talk about user centred design and this is certainly represented by the definition of the human-centred-design process in the DIN EN ISO 13407. However, even if the experts' opinions sometimes differ on how to implement these activities (e.g. in using methodologies, methods, tools, etc.), they follow the same goal: to ensure a specific quality of these activities. Thus, there is no generic answer of how an activity has to be performed in detail – the question is how to assure that everything needed is being performed in order to gain a certain quality of a result, an activity or the process itself. This article shows first results on this important question.

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 as 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.

The gathered knowledge about the quality aspects of usability engineering can serve as a basis for a successful integration with software engineering and leads to high quality results and activities.

In the future, the authors expect to derive specific recommendations to enrich software engineering models by adding or adapting usability engineering activities, phases, artefacts, etc. By doing this, the development of usable software on the level of process models will be guaranteed. Furthermore, the authors will present further results based on the questionnaires and will work out a guideline/checklist of usability engineering demands that account for the integration of usability engineering into software engineering models.

8. Acknowledgement

We would like to express our sincere appreciation to all those who have contributed, directly or indirectly, to this work in form of technical or other support. In particular we would like to thank Markus Düchting for his assistance. A special thank to Lennart Grötzbach always helping us to find the right words. We would like to thank Thomas Geis and Jan Gulliksen for the fruitful and intensive discussions and the helpful input. We also like to thank the remaining interview partners for their help and efforts.

9. References

Boehm, B. (1988). A Spiral Model of Software Development and Enhancement. *IEEE Computer*, Vol. 21, pp. 61-72

Constantine, L., Biddle, R. and Noble, J. (2003). Usage-centered design and software engineering. Models for integration, In: *IFIP Working Group 2.7/13.4*, ICSE 2003 Workshop on Bridging the Gap Between Software Engineering and Human-Computer Interaction, Portland, Oregon

Cooper, A. & Reimann, R. (2003). *About Face 2.0.*, Wiley, Indianapolis, IN

DIN EN ISO 13407 (1999). Human-centered design processes for interactive systems, CEN - European Committee for Standardization, Brussels

DIN EN ISO 9241-12 (1998) Ergonomic requirements for office work with visual display terminals (VDTs) - Part 12: Presentation of information, ISO Copyright Office, Geneva, Switzerland

DIN EN ISO 9241-110 (2006), Ergonomics of human-system interaction - Part 110: Dialogue principles, ISO Copyright Office, Geneva, Switzerland

Faulkner, X. (2000a). *Usability Engineering*, pp. 10-12, PALGARVE, New York, USA

Faulkner, X. & Culwin, F. (2000b). Enter the Usability Engineer: Integrating HCI and Software Engineering. *Proceedings of ITicSE 2000 7/00,* ACM Press, Helsinki, Finnland.

Freymann, M. (2007). Klassifikation nutzerzentrierter Evaluationsmethoden im User Centered Design Prozess, *Diploma Thesis*, University of Paderborn, Germany

Glinz, M. (1999). Eine geführte Tour durch die Landschaft der Software-Prozesse und -Prozessverbesserung, *Informatik – Informatique*, Vol. 6/1999, pp. 7-15

Granollers, T., Lorès, J. & Perdrix, F. (2002). Usability Engineering Process Model. Integration with Software Engineering, *Proceedings of HCI International 2003*, Crete, Greece, June 22-27-2003, Lawrence Erlbaum Associates, New Jersey, USA

Hakiel, S. (1997). Delivering ease of use, In: *Computing & Control Engineering Journal*, Vol. 8, Issue 2, 04/97, p. 81-87

IBM (2004). Ease of Use Model. Retrieved from http://www-3.ibm.com/ibm/easy /eou_ext.nsf/ publish/1996, (11/2004)

ISO/IEC 12207 (2002). Information technology - Software life cycle processes, Amendment 1, 2002-05-01, ISO copyright office, Switzerland

ISO/PAS 18152 (2003). Ergonomics of human-system interaction — Specification for the process assessment of human-system issues, First Edition, 2003-10-01, ISO copyright office, Switzerland

Jerome, B. & Kazman R. (2005). Surveying the Solitudes. An Investigation into the relationships between Human Computer Interaction and Software Engineering in Practice, *In: Human-Centered Software Engineering – Integrating Usability in the Software Development Lifecycle*, Ahmed Seffah, Jan Gulliksen and Michel C. Desmarais, 59-70, Springer Netherlands, 978-1-4020-4027-6, Dordrecht, Netherlands

Jokela, T. (2001). An Assessment Approach for User-Centred Design Processes. In: *Proceedings of EuroSPI 2001*, Limerick Institute of Technology Press, Limerick

Juristo, N., Windl, H., Constantaine, L. (2001). Special Issue on Usability Engineering in Software Development, In: *IEEE Software*, Vol. 18, no. 1

KBST (2006). V-Modell 97. Retrieved from: http://www.kbst.bund.de , 05/2006

Mayhew, D. J. (1999). *The Usablility Engineering Lifecycle*, Morgan Kaufmann, San Francisco

McCracken, D.D., Jackson M.A. (1982). Life-Cycle Concept Considered Harm-ful. *ACM Software Engineering Notes,* 4/1982, pp. 29-32

Nebe, K. & Zimmermann, D. (2007a). Suitability of Software Engineering Models for the Production of Usable Software, *Proceedings of the Engineering Interactive Systems 2007*, Lecture Notes In Computer Science (LNCS), Salamanca, Spain (in print)

Nebe, K. & Zimmermann, D. (2007b). Aspects of Integrating User Centered Design to Software Engineering Processes, *Human-Computer Interaction. Interaction Design and Usability,* Vol. 4550/2007, pp. 194-203, Beijing, P.R. China

Nebe, K., Düchting, M., Zimmermann, D. & Paelke, V. (2007c). Qualitätsaspekte bei der Integration von User Centred Design Aktivitäten in Softwareentwicklungsprozesse, *Proceedings of Mensch & Computer 2008*, Lübeck, Germany (in print)

Norman, D.A. & Draper, S.W. (1986). Eds. User Centered System Design. Laurence Erlbaum

Patel, D., Wang, Y (eds.) (2000). Comparative software engineering. Review and perspectives, In: *Annals of Software Engineering*, Vol. 10, pp. 1-10, Springer, Netherlands

ProContext (2008). retrieved from: http://www.procontext.com/en/methods/context-scenario/implied-needs.html, 07/2008

Radle, K. & Young, S. (2001). Partnering Usability with Development. How Three Organizations Succeeded, In: *IEEE Software*, January/February 2001.

Rideout, T., Uyeda, K. & Williams, E. (1989). Evolving The Software Usability Engineering Process at Hewlett-Packard, *Proceedings of Systems, Man and Cybernetics 1989*, Vol. 1, pp. 229-234

Royce, Winston W. (1970). Managing the Development of Large Software Systems: Concepts and Techniques. In: *Technical Papers of Western Electronic Show and Convention (WesCon),* pp. 328-338, August 25-28, Los Angeles, USA

Sommerville, I. (2004). *Software Engineering*, 7th ed, Pearson Education Limited, Essex, GB

Sutcliffe, A. (2005). Convergence or competition between software engineering and human computer interaction, In: *Human-Centered Software Engineering – Integrating Usability in the Software Development Lifecycle*, Ahmed Seffah, Jan Gulliksen and Michel C. Desmarais, 71-84, Springer Netherlands, 978-1-4020-4027-6, Dordrecht, Netherlands

Woletz, N. (2006). Evaluation eines User-Centred Design-Prozessassessments - Empirische Untersuchung der Qualität und Gebrauchstauglichkeit im praktischen Einsatz. *Doctoral Thesis*, 4/2006, University of Paderborn, Paderborn, Germany

Venturi, G. & Troost, J. (2004). Survey on the UCD integration in the industry. *Proceedings of NordiCHI '04*, pp. 449-452, Tampere, Finland, October 23-27, 2004, ACM Press, New York, USA

Advances in Human Computer Interaction

Edited by Shane Pinder

ISBN 978-953-7619-15-2
Hard cover, 600 pages
Publisher InTech
Published online 01, October, 2008
Published in print edition October, 2008

In these 34 chapters, we survey the broad disciplines that loosely inhabit the study and practice of human-computer interaction. Our authors are passionate advocates of innovative applications, novel approaches, and modern advances in this exciting and developing field. It is our wish that the reader consider not only what our authors have written and the experimentation they have described, but also the examples they have set.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Karsten Nebe, Dirk Zimmermann and Volker Paelke (2008). Integrating Software Engineering and Usability Engineering, Advances in Human Computer Interaction, Shane Pinder (Ed.), ISBN: 978-953-7619-15-2, InTech, Available from: http://www.intechopen.com/books/advances_in_human_computer_interaction/integrating_software_engineering_and_usability_engineering

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen