Fraunhofer Institut
Experimentelles
Software Engineering

UseLine: Process Description and Case-Study

Authors:
Isabel John
Kirstin Kohler
Martin Schmettow
Daniel Kerkow

Gefördert von der Stiftung Rheinland Pfalz für Innovation
Förderkennzeichen:
15202-38 62 61 / 621

IESE-Report No. 074.04/E
Version 1.0
June, 2004

A publication by Fraunhofer IESE

Fraunhofer IESE is an institute of the Fraunhofer Gesellschaft.
The institute transfers innovative software development techniques, methods and tools into industrial practice, assists companies in building software competencies customized to their needs, and helps them to establish a competitive market position.

Fraunhofer IESE is directed by
Prof. Dr. Dieter Rombach
Sauerwiesen 6
67661 Kaiserslautern

Abstract

In this report we describe the results of the UseLine Project, a project that was funded by the „Stiftung Rheinland Pfalz für Innovation“ under grant number 0621.
In this project, the interplay between the two Software Engineering disciplines Usability Engineering on the one hand and Product Line Engineering on the other hand is investigated. Although both disciplines seem to contain contradictory elements – e.g. Usability Engineering focuses on the individual user whereas Product Line engineering concentrates on reuse and commonality between products an integrated approach that focuses on both aspects has advantages. Especially for small and medium enterprises who traditionally have a strong customer focus and do often not have the time and money to introduce both – Product Line engineering and Usability engineering – an integrated approach can help to achieve a large market advantage – a usable Product Line.

In this report we describe the useline approach for planning and requirements engineering of usable Product Lines. The approach explicitly considers the business goals of an organisation, helps to identify user tasks and product features and domains and improves the Usability of central and reusable tasks. For the latter a new Usability inspection method was developed which should suit well for small development teams and projects and persons which are unskilled in Usability methods.

The approach was applied in a small company in Kaiserslautern. We also describe our experiences with applying the approach there and give examples on the artefacts produced.

Keywords: Usability; Product Line Engineering; UseLine; Scoping; Feature; Task

Table of Contents

1 Goal of the Use-Line Project

Usability Engineering and Product Line Engineering are two upcoming topics in software engineering. They both aim at improving software and software development. Usability Engineering wants to improve he ease of use of software products by integrating Usability principles into development. Product Line Engineering wants to improve reuse and reUsability of software by explicitly planning and developing a platform for a set of software products.

As Usability focuses on the user, on the individual usage of the software and Product Line Engineering focuses on reuse and commonality of software they are, at first sight, contrary approaches. Usability Engineering focuses on the user, its tasks and its interaction with the system [Lau03], Product Line Engineering focuses on the systems or products and their commonalities and variabilities [CN02].

But an integrated view on Usability and Product Line engineering has major advantages: Both approaches focus on business goals and often entail major changes in software structure and development structure. By jointly looking at Usability aspects and commonalities and variabilities of existing and planned products improvements can fertilize each other and can be put into practice more cost effective.

Especially small and medium sized companies (SMEs) often have deficiencies in Usability of their systems that they want to improve and, on the other hand have the need to build and release products more systematically with their limited manpower. Those SMEs do not have time and money to invest in both the introduction of Usability and Product Line engineering principles into their software.

In this report we present the UseLine Approach for an integrated Product Line and Usability analysis. The approach is based on business goals and focuses on planning aspects and the early development phases. With the approach, weaknesses in the products (on Usability and Product Line side) are identified and improvement suggestions for the development of future products and current products are given. With the approach we integrate the task-oriented mode of working of Usability approaches and the feature- and component oriented mode of Product Line engineering approaches. And therefore provide one step towards bridging the gap between Usability and software engineering [KLB03].

2 Context

In this section we describe the larger context of the approach – the two disciplines Usability Engineering and Product Line Engineering that we want to integrate.

2.1 Usability Engineering

The field of human-computer interaction (HCI) is concerned with the joined performance of tasks by humans and computational machines [CL99], [CL99], [HB96]. From the computer science perspective one essential contribution of HCI is the design, evaluation and implementation of interactive computing systems for human use. The process that guides these tasks by specifying, measuring and improving the Usability of a product is commonly called Usability engineering. Example for Usability engineering approaches are the User Centered Design Process (UCD for short) [Ma99] and the Contextual Design Process [BH98].

Usability refers to the extend, to which a product can be used by specific users in a specific usage context to achieve specific goals in an effective, efficient and satisfying way [ISO01].

Usability in a simplified understanding is a non-functional property that characterizes the functionality of a product. In this sense missing functionality is as much of a thread to Usability as the so-called featuritis (which refers to a large amount of unneeded functions)[Nor88]. This simplified view leads many novices to the false assumption that Usability is an inherent characteristic of a product – independently from the situational context of the person using it [MB03]. A more elaborated understanding of Usability is the concept of "quality in use". Quality in use as it is defined in ISO 9126 [ISO98] addresses more than a product characteristic. It refers to the perception of the overall quality of a product in use and can only be expressed accurately, if all quality aspects such as Usability, efficiency, safety and security are taken into account [KDP04]. The perception of quality aspects of a system in use is also known as the users experience and differs from individual to individual [KKD03].

The discipline of Usability engineering develops methods to meet these challenges. Even though methods for all phases of the software engineering lifecycle have been introduced in the past, the highest emphasis has been on evaluative activities. Fewer methods are known in constructive phases such as requirements analysis or design.

Activities of Usability engineering have the challenges in mind that raise with the construction of solutions that meet usage-context-specific requirements. They involve the future users of the system as much as resource limitations, that every software project has, allow. The integration of specific users as well as the concept of "quality in use" leads to certain decisions during software development. These decisions take place on four abstraction levels: task level, domain level, interaction level and system level (see Figure 1). For a detailed discussion of the design decisions compare [PK03a] and [PK03b]:

- Task level
  Decisions on this level determine the user roles and the tasks of these roles to be supported by the system. This meets the challenge of supporting specific users in specific contexts.
- Domain level
  The user tasks consist of several activities. On domain level, decisions about activities in the as-is process and those in the to-be process are made, as well as decisions about which of those activities will be the systems responsibilities (features). Further, decisions about typical data that is required in the domain are made.
- Interaction level
  System responsibilities are realized by system functions.
  The decision about the system functions determines the border between user and system. It has to be decided how the user can use the system function to achieve the system responsibilities. This determines the interaction between user and system. Decisions about input and output data have to be made as well as decisions about the grouping of data and system functions in different workspaces.
- System level
  a) Code related decisions
  Architectural decisions about the code realizing the system have to be made. System functions are modularised into different components. Decisions have to be made regarding the internal system actions that realize the system functions. The internal system data refines the interaction data to the granularity of the system actions.
  b) Interface related decisions
  It has to be decided how the user can navigate between different screens during the execution of system-functions. This determines the navigation functions. In addition support functions that facilitate the system-functions have to be defined. These functions realize parts of system-functions that are visible to the user in form of GUI elements. For each interaction the detailed control of the user has to be decided. This determines the dialog. It consists of a sequence of support and navigation functions executions.
  For each navigation- and support-function the input data provided by the user as well as the output data provided by the system has to be defined.

The screen-structure groups navigation- and support-functions as well as UI-data into different screens.

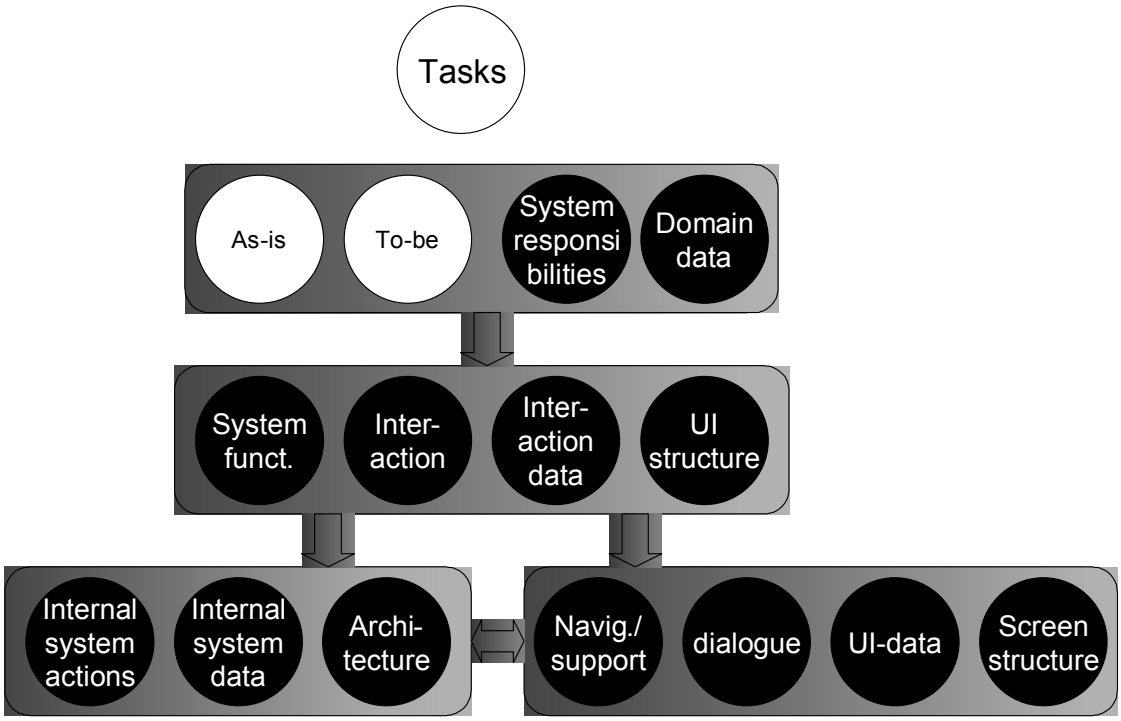


Figure 1 The four levels of design decisions in Usability engineering (see [JK04])

The maximization of user and usage specific issues during these design decisions seems to be in conflicts with Product Line Principles. In the following sections we will describe properties of Product Line engineering and discuss these conflicts.

2.2 Product Line Engineering

Product line engineering can be described as a technology providing methods to plan, control, and improve a reuse infrastructure for developing a family of similar products instead of developing single products separately. This reuse infrastructure manages commonality and controls the variability of the different products. Examples for Product Line approaches are PuLSE [BFK+99], Fast [WL99] and the SEI Product Line Practice Initiative [CN01].

The goal of Product Line engineering is to achieve planned domain-specific reuse by building a family of applications. Distinct from single system software development there are two life cycles, domain engineering and application en-

gineering. In domain engineering the reusable asset base is built and in application engineering this asset base is used to build up the planned products (cf. Figure 2

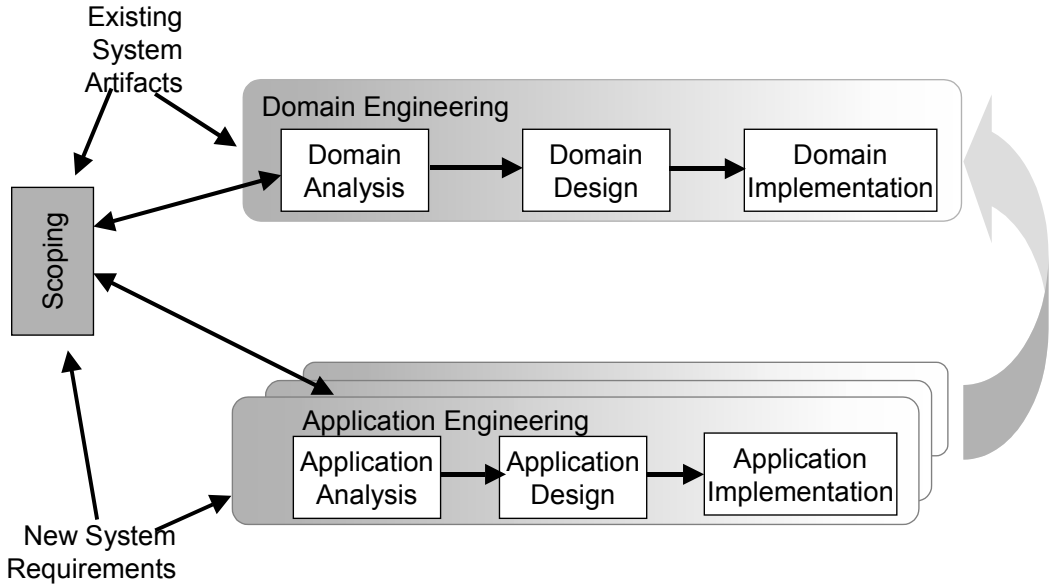


Figure 2 Product line engineering: A two lifecycle approach

One benefit of the Product Line approach is the idea that all members of the Product Line are based on a single set of assets. Thereby, the maintenance effort is reduced to this single asset base. Hence, existing products must always follow the evolution of the asset base. This is done by instantiating the changed domain model while reusing the existing resolution of the domain decision model.

Product line engineering stands on a middle ground between specific assets (single system development, one time use) and general assets (general use libraries or components, applicable anywhere; generator based approaches). The idea is to define a Product Line, which captures the intended scope of reuse. The products included in a Product Line are determined to have sufficiently common characteristics to make it more efficient to study the commonalities and variabilities once for all products of the Product Line and to build reusable assets, than to study and build all the products separately. Product line engineering approaches define how to leverage the commonalities and create reusable assets that increase the efficiency of developing the products [BMW99].

Domain analysis or domain modelling, is requirements engineering for Product Lines. Domain analysis methods provide processes for eliciting and structuring the requirements of a domain, or Product Line. The results are captured in a

domain model. A domain model must capture both the common characteristics of the products and their variations. The domain model is the basis for creating other reusable assets like a domain specific language or a component-based architecture. For a domain analysis method to be applicable it must be appropriate and it must provide enough guidance so that it can be carried out. As in other areas of software development, the context for each domain analysis application varies, and methods that are appropriate in one context will not be in others. This fact is especially important for domain analysis because of the compound effects of inappropriate models over multiple products and over the whole lifecycle. Therefore, a generally applicable domain analysis method should be customisable to the context of the application.

The properties which are specific to requirements engineering for Product Lines, are:

- Commonality and Variability
  When doing domain analysis the properties of several products have to be modelled at once. As the planned products that are analysed during domain analysis differ in their features [KCH09] and in their functional and non-functional requirements, the commonalities and variabilities between those products have to be captured and adequately modelled.
- Instantiation Support
  As several products are modelled in one domain model it must be clear, which part of the model or which requirement belongs to which product. In order to have an application specific view on the product the instantiation of the generic and variable model for several products has to be supported.
- Decision Modelling
  To get this instantiation support, the decisions that have be made have also to be captured in a separate model. This model collects and abstracts the information on which requirement is instantiated in which product and supports the instantiation.
- Traceability
  providing traceability from the requirements to the product and from the requirements to architecture, implementation and tests is very important in Product Line engineering. As a Product Line spans over several products and several releases of the products it has to be ensured that those two dimensions of traceability (traceability through products and through lifecycles) are provided
- Evolution
  Product Lines are a means to cope with evolution. With Product Lines evolution in space (the space of the planned products) is controlled.

2.3 Integration of Usability and Product lines

A substantial strength of the German industry lies in the professional engineering of highly sophisticated products [BHK+01]. Individual production instead of mass production is a characteristic of SMEs (Small and medium enterprises and reflects itself also in software-developing companies. These SMEs often develop customized software solutions for certain application domains, i.e. no standard software to a predominant portion. The different desires of the customers are ideally fulfilled by a carefully planned variant building on the basis of a common platform. This variant formation concerns the architecture of the software altogether but for the customer particularly materializes itself in the use surface (variant feature, different processes etc).

The Goal of the UseLine Project is it to integrate Usability and Product Line Aspects both into an integrated approach for software development

Usability and Product Lines engineering are complementary approaches at first sight. Usability focuses on the user, on the concrete tasks and activities of the user with the system and on the individual, concrete system. Product Line engineering tries to find commonalities and systematize differences to synchronize the planned systems and to make them common as far as possible.

Both approaches thus can profit from each other: The collection of requirements and analysis of systems from the Usability viewpoint gives valuable assistance for functionality which can be systematized in the Product Line analysis. On the other hand the systematic collection of variability within Product Line analysis can help to uncover differences with the use and the Usability of the existing and planned systems and so to work out the different types of usage and the different involvement of tasks and activities in the different products more clearly .

In order to let both approaches benefit from each other and to let Product Line engineering leverage Usability aspects and otherwise we have to address the following requirements on an integrated approach:

- Product Line Engineering does an explicit scoping [Sch03]. Product line scoping is the problem of determining what is inside and what is outside of a Product Line. With the help of scoping, the features (user visible characteristics of a system), the domains (areas of common functionalities, common features or concepts within a system) are determined, and the commonalities and variabilities between the features and domains of the different products are determined. This feature oriented and static view on the systems has to be more user centred and dynamic in order to cover Usability aspects during scoping. Especially the link between the features and the user tasks has to be made explicit.

- Usability Engineering often works with tasks, scenarios and user stories [Ma99], [CL99], [Co99], Product Line engineering with features and domains. By an approach that integrates those two views, a mapping between the different types of artefacts hast to be provided.

- Product line engineering as well as Usability engineering (especially customer centred design processes) focus on business goals. So, synergies can be used here. Each discipline considers different aspects that affect their viewpoint. So, an integrated approach should supports a broader and more comprehensive view on business goals to help determine the most valuable priorities in a given the business situation.

- As we only describe a planning approach, not the actual architecting and implementation, the approach has to provide explicit interfaces to a deeper requirements engineering, to architecture development and to the actual implementation of Product Line and Usability aspects. This can e.g. be done by prioritising activities, by giving recommendations for notations and activities and by suggesting architecture, design and implementation pattern for certain parts of the Product Line. A

- Pattern are a good means to unify the Usability-driven activities and so to support the built up of commonalities. By building a pattern based approach, recommendations for Usability pattern should be given. By giving recommendations for pattern, the commonalities for the Product Line can be established.

So the main challenge that we want to tackle with our approach is it to integrate Product Line engineering and Usability engineering in a way hat both approaches benefit from each other. As SMEs often cannot invest in both improving (or starting) Product Line engineering and Usability we need an approach that integrates the advantages of both approaches and reduces the effort of introducing Usability and Product Line engineering into a development organisation.

2.4 Context of our integrated Process

The approach that we describe here is only applicable in certain contexts. It is not valid for all possible contexts but it can be customized for other approaches too.

For introducing and applying Product Line engineering we identified the following areas for customisation [SW00] that also partially hold for Usability engineering. The customisation factors capture many different aspects of the situation in which a process will be applied. During the definition of these factors for PuLSE the customisation factors were classified into six categories. These in-

clude organizational aspects, characteristics of the domain, and project characteristics. The list of the different categories of customisation decisions is given in Table 1. The purpose of these categories is to provide a structuring of the set of customisation factors and thus to structure and simplify elicitation of these factors.

Domain Characteristics	factors which relate to the domain itself and are independent of implementation aspects (e.g., domain complexity which represents how difficult to understand a domain is)
Information Sources	factors that relate to the availability of information about the domain
Implementation Characteristics	factors that influence implementations in the domain
Integratable Software Artefacts	factors that relate to the actual artefacts that can be integrated in a new Product Line
Project context	factors relating to the specific project that is planned
Enterprise context	factors relating to the enterprise in general

Table 1 Areas of customization factors

To support resolving the customisation decisions, the customisation requires that, for each customisation decision, the information that may influence the resolution is defined. This information can be at two levels. The first level includes information that can be directly collected from an organization — for example, the skills of personnel and existing development practices. Information at this level is called a customisation factor. The second level includes information that itself is influenced by other information and therefore has to be resolved; for example, the level of domain understanding in an organization is a complex factor that is influenced by other factors. Information at this level is called an intermediate decision.

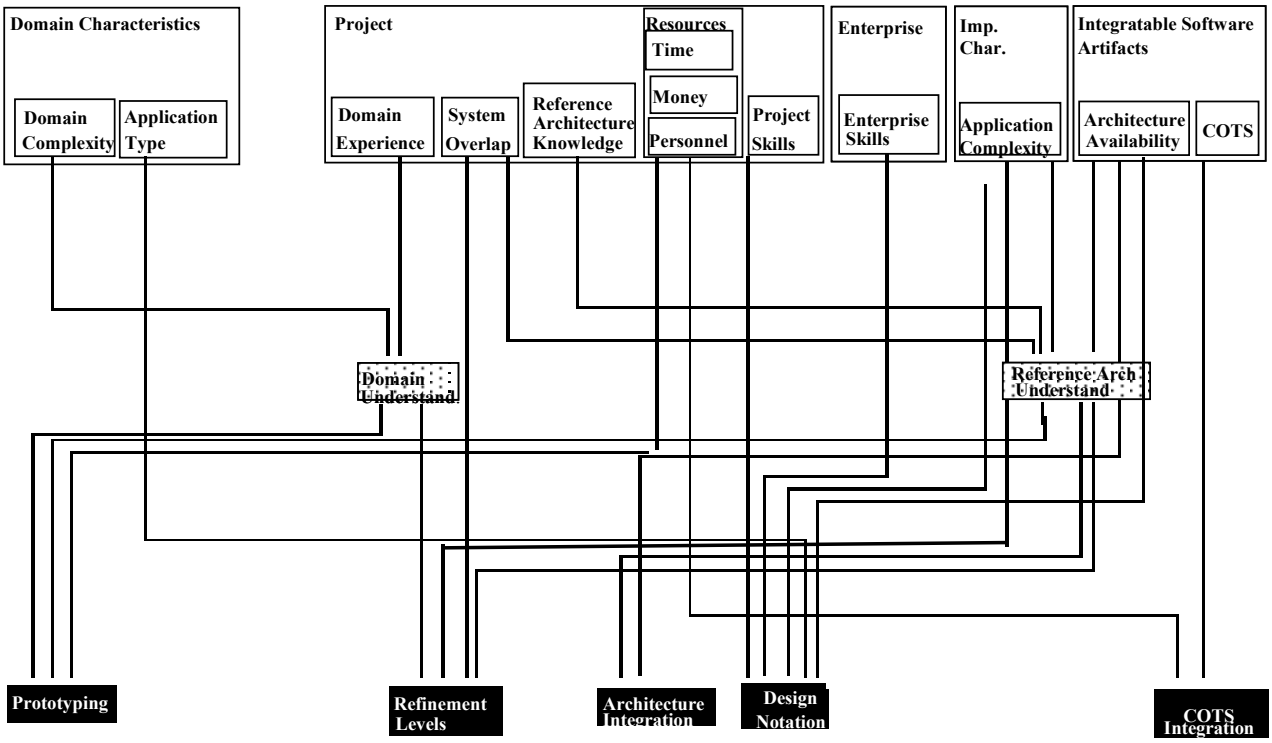


Figure 3 Product Line Customization Decisions

How the categories and intermediate decisions influence each other and how they influence the actual decision can be seen in Figure 3. When applying these customisation factors to the UseLine Approach we come to the following restrictions:

Customisation Factors

- Integratable SW Artefacts
  As a Usability evaluation should be part of the analysis, there should be existing systems in the domain. As the goal of the approach is it to identify reusable assets and to improve the Usability of these assets, it is essential that some reusable assets exist. It is not mandatory that the existing systems will be reusable and will be reused as a whole but there should be some existing functionality. So, the approach is not applicable if a Product Line is planned from scratch, without integrating existing artefacts.

- Domain Characteristics
  As Usability of a system is manifested in the user interface, the domain that is investigated should be an externally visible domain. If no user interface is planned for the Product Line (e.g. if the planned Product Line

is a Product Line of different protocols that is customized through a parameter file, or if the Product Line effort is concentrated on an internal database subdomain of the systems that is not visible to the user of the systems) it is not reasonable to integrate Usability aspects into the Product Line effort.
On the other hand, the domain should have a certain amount of systematizable variability in order to make a Product Line effort useful. There can be situations when a Usability improvement in a domain makes sense but the Usability aspects are so diverse (or so unknown from the beginning) between the different products that it doesn't seem possible to systematize them or to find commonalities and variabilities.

- Information Sources
  Concerning the information sources there main restriction is that there should be at least one running system to be used as an information source in order to make a Usability analysis with the system. All other information sources (experts, books, documentation, code, contracts etc.) can be used in different extents.

- Enterprise Context
  There is no restriction on the enterprise context. Although the approach is especially valuable for SMEs that want to integrate Product Line engineering and Usability engineering and benefit from the integration of both, also larger companies can apply the approach if they want to have an integrated effort.

- Project Context
  It is important for both Usability and Product Line engineering to get expert input for the analysis. Especially for finding business goals and for planning new features it is important that experts from former similar projects are available to give input. There are no further restrictions on the size or duration of the project or on the maturity of the staff.

With these customisation factors we can also derive restrictions on the intermediate decisions. Those intermediate decisions can be partially resolved with the above given restrictions on the factors but they can only be totally resolved in a concrete project context:

Intermediate Decisions

- Level of Detail
  There are no restrictions on the level of detail of the artefacts that should be produced.

- Coverage
  The Product Line and Usability efforts do not have to cover all subdomains/areas/components of the systems. A priorization of subdomains should be given to have an order for the following activities. Furthermore it is important to cover at least some of the areas that are influenced or influence the user interface in order to have the Usability aspects included. As with the approach we want to focus on those areas where an improvement pays best, not the whole Product Line should be covered with the activities but only those areas that have a high priority concerning the business goals and the Usability problems. Therefore no full Usability evaluation is performed but only an evaluation of those tasks that are the most important ones in terms of commonality and variability and fulfilment of the business goals.

- Information Types
  There is no hard restriction on the information types to be produced. As Usability often is seen as task-centred, Product Line engineering is feature centred, we recommend to produce at least task (+ descriptions) and features but this is not a must.

- Paradigm
  There are no restriction on the software development paradigm. The integrated Product Line engineering and Usability activities can be implemented with an object oriented development paradigm, with generators, within an existing framework etc.

Based on those factors we made for our approach the following customisation decisions:

Customisation Decisions

- Information Gathering
  The information gathering is done in a mix of gathering from experts and from documents. So, the approach can be customized based on the available of experts and of documentation that is valuable for the approach.

- Modelling Process
  In order to integrate Usability and Product Line aspects, the modelling process produces Product Line artefacts (feature and domain descriptions) and Usability artefacts (task and scenario description) and integrates them in a separate step. We support incremental introduction of Usability and Product Line aspects (domain per domain integration, maybe also applicable in product per product integration).

- Workproduct Types
  As UseLine has a strong focus of business goals and on the integration of Usability and Product Line aspects in (multi-) project planning, the following workproducts should (at least) be produced:
  - A prioritisation of business goals
  - A product feature matrix that describes which features can be found in which product
  - A domain description for each subdomain of the systems in order to find out about the Usability aspects of the subdomain
  - A description of the typical users of the system (e.g. as a persona description) for each product
  - A description of the tasks that can be performed with the systems e.g. In form of a business process and scenarios.
  - A list of Usability weaknesses based on Usability evaluations of existing systems.
  - Recommendations for development. These recommendations can e.g. be given in the form of Usability pattern, in the form of prioritisation lists etc.

- Workproduct Representation
  There are no restrictions on the workproduct representation. As the Usability and Product Line aspects are both equally important, a representation should be chosen that is understandable by the users (at least partially or for some of the artefacts), by the domain experts and at least partially by the developers of the systems.

3 Description of the Process

In this section we describe the integrated Usability and Product Line analysis process that we developed. We fist describe the integration aspects of Product Lines and Usability and where we had to change and extend the respective processes in order to integrate them. We then give an overview of the integrated process and describe each process step in detail.

3.1 Integration Aspects

Product Line Engineering and Usability follow similar processes in process steps but produce and consume different artefacts. The differences and similarities can be seen in Figure 4.

Both processes start with a business goal analysis, the outcome of the Product Line business goal analysis are the reuse goals and the product strategy, the outcome of the Usability business goal analysis are the Usability goals. The actual analysis and how the business goals are elicited overlaps. Additionally, most of the business goals analyzed in Product Line engineering also play an important role for Usability (e.g. time to market, decreased support cost, customer satisfaction, market extension or opening new markets). But there are also some possible business goals for Product Line engineering that do not play a role for Usability engineering (e.g. increased maintainability, increased reuse) and vice versa (e.g. increased task efficiency).

The next step in Product Line planning as it is done in PuLSE [BMW99] and Pulse-Eco [Sch03] is a feature analysis. The features (user visible functional and non functional properties of a system or Product Line) and domains (areas of functionality of a system or Product Line) are elicited and captured. The features are located in the domains and the commonalities and variabilities between the different products that exist and that are planned are captured. The artifact that is produced here is a product-feature –map [Sch00], this artifact will be explained in detail in section 3.3.3. With this information about the features and domains a domain prioritization is done. The domains are prioritized in a kind of product assessment and a prioritization given on factors like alignment with the business goals, systematic variability, existing implementation etc is given. With this list of domain, features, existing artifacts etc. the domain engineering can start with the highest priority domains, an architecture can be built for the Product Line and the platform supporting commonalities and systematic variabilities can be built.

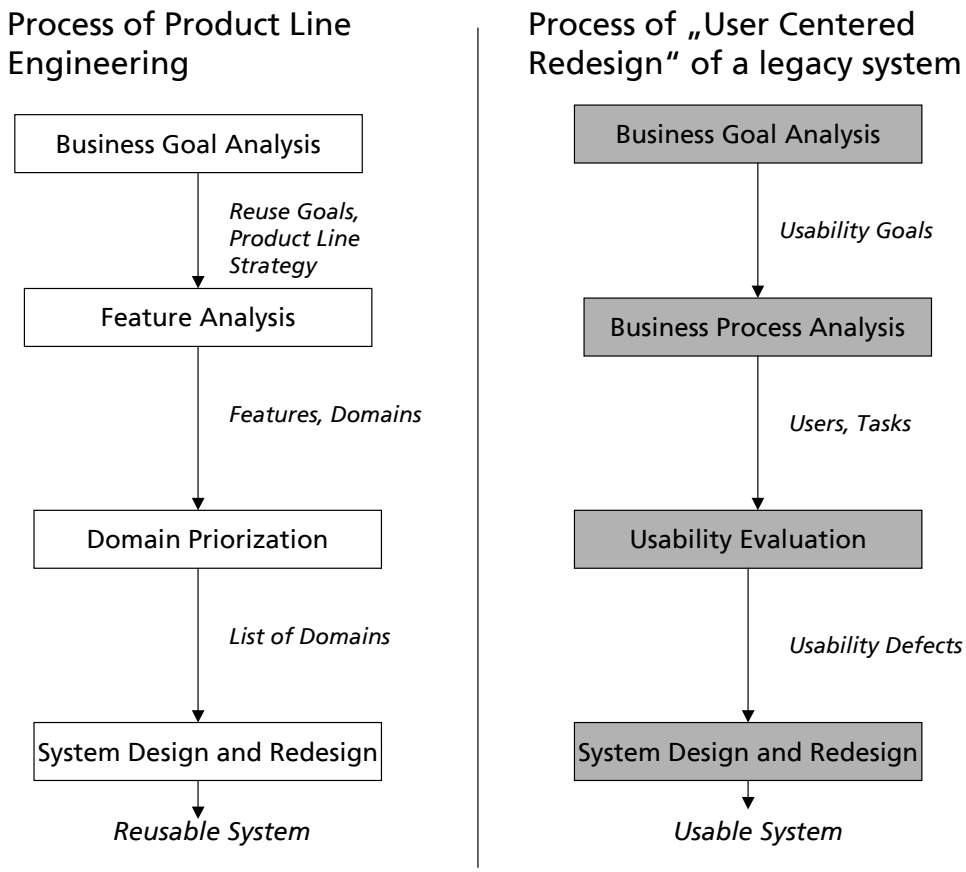


Figure 4 Product Line and Usability Analysis Process

In Usability Engineering, the business goals are elicited similar to Product Line engineering but then the focus is more on the processes and Tasks (c.f. [PK03a] and [PK03b]) that the user performs with the system. The business process with its involved roles and activities is captured and fine-grained tasks are found. Un the Usability Evaluation those tasks are processed with the system and Usability defects are found. Those defects are accompanied with improvement suggestions for making the system more usable that go into redesign of the old and design of the new systems.

3.2 Process Overview

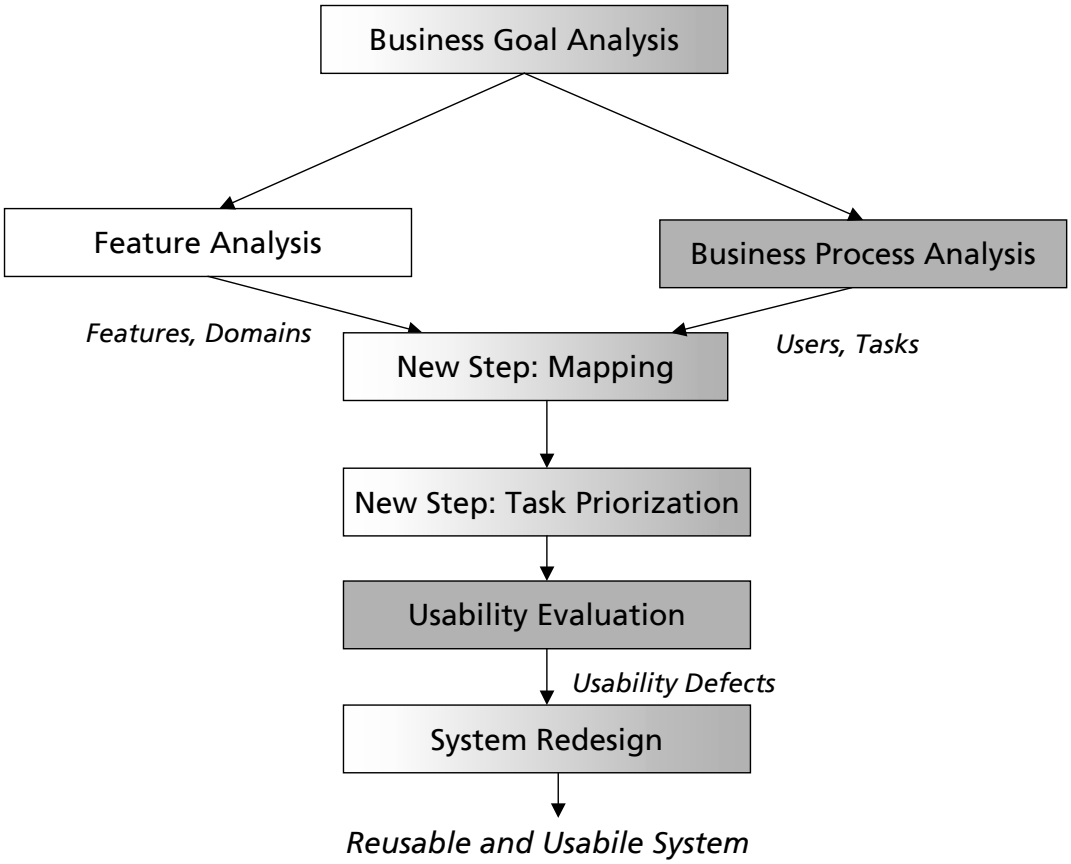


Figure 5 Overview of the UseLine Process

We have integrated those to similar but different process into a joint Usability and Product line planning process, the UseLine process. You can see an overview of the UseLine Process in Figure 5. First, an analysis of the business goals of the company is done in order to find out the overall goal of improving Usability and commonality and variability of the systems. Then the business process that the existing and planned systems within the Product Line cover is captured and the variabilities between the tasks and sub processes within the business process are captured as far as possible (they can be changed and refined later on). The next step is an extraction and analysis of the features as it is described in [Sch03]. The outcome of this step is a product-feature matrix that also includes the domains and subdomains of the systems and a description of the subdomains. This step also includes an analysis of the data that the system consumes and produces. In the Mapping step the features, data and tasks are mapped against each other. In the next step, the domains, features and tasks are prioritised according to the business goals. Then the most critical tasks are

evaluated in the Usability evaluation step by doing a Usability evaluation with an existing system. There, the variabilities of the existing and planned features can be explicitly taken into account as there is a mapping of the tasks to the variable features. Based on the Usability evaluation and the business goals recommendations are given in the form of Usability pattern as they can be best mapped to the Product Line architecture. In the last step, consolidation, the recommendations of all analyses are harmonized and an overall recommendation for improving Usability and Product Line engineering is given.

The actual implementation of the recommendation (e.g. implementation of user interface pattern in the form of reusable widgets) is not part of process. These aspects are covered in a similar but in focus different project, the Research Lab Product Lines and Usability. So the process focuses in the planning and on parts of the requirements engineering phases in the software development lifecycle.

In the following we describe the sub processes of our approach. We describe each sub process in a different section and we describe each sub process with the following attributes:

– Goal: The general goal of the process step
– Method: The method or methodical support that is applied or given in order to perform the process. This can on one hand be a general description of the method but also be a pointer on templates, checklists etc.
– Input Artefacts: The input that is needed to perform the step, including the templates, checklists etc that are needed.
– Output artefacts: The artefacts that are built while performing the step and that are useful as intermediate product for the general process or as general result

3.3 Description of Activities

3.3.1 Business Goal Analysis

Goal

The Goal of this step is to identify the business goals of the company that drive the improvement in the area of Usability and product-line engineering. A deeper understanding of the business goals is beneficial for several reasons:

- It leads to better understanding of the sub characteristics of Usability, the so called Usability goals by deriving a first draft of a quality model [KKD03].

- It helps focusing the Product Line aspects on the important factors and main parts of the system from the business perspective.
- It helps judging the importance of the improvement steps and therefore the willingness of the company to invest in process changes. By doing so it is possible to scale the changes in concert with the business context.
- Additionally, the context of the project (e.g. number of developers duration of the project etc) is determined which can help in customizing the process on the context of the organization as described in section 2.4.

Method

The business goal analysis is conducted by a guided interview of the managers of the company, which takes around 2 hours. This interview is guided by a set of questions (for an example see section4).

The results of the interview are analyzed and transferred to the description of the business goals, Usability goals and a first version of the Usability quality model. This set of documents is reviewed by the interview partners in order to correct or complete the derived documents.

Input Artifacts

Interview questions

Output Artifacts

Description of business goals, First draft of the Usability quality model, Usability goals

3.3.2 Business-Process Analysis

Goal

The goal of this step is to identify the typical user roles of the product variants, their tasks and the variation between the user roles and task for the different variants of the product family. As a result of this step the understanding of the business process, in which the software products are used has to be developed.

Method

There are various methods to conduct this business process analysis. The cheapest is to talk to the managers or the marketing of the company, which usually have a very solid understanding of the business of their clients. In that case the business process can be derived based on a tow to four hours interview guided by a given set of questions. In addition to interviews user observations. can complete the analysis [DM96]. While user observations provide first

hand information about users and their tasks (and therefore we strongly recommend them), at the same time they are more expensive and need more experienced people to perform them.

The results of the business process analysis are condensed in one to tow page graphical overview of the business process, showing the tasks of the various user roles and the information that is transferred between these activities. The business process gives a complete overview over the process including the steps, that are currently not supported by the system. In addition to the business process overview a description of the various user roles as Personas [Co99], [Ni03].and a set of tasks summarizes the outcome of the business process analysis.

The user roles and business process are elicited for the variants of the product. The differences between the variations are condensed in the graphical overview as well as in a table listing the tasks of the business process.

The business process should either be documented in ARIS or UML-Activity diagrams. The differences in the business process for the variations of the product should be represented in one graphical overview [SBK+00]

Input Artefacts

Guidelines for Interview including questions

Output Artefakts

Personas, Business Process Overview, Table with Task of the different product variations. Examples for these artefacts are listed as part of the case study in section 4.2.3.

3.3.3 Feature Analysis

Goal

Schmid identifies three levels of scope [Sch00]:

- The product portfolio defines the various products that are part of the Product Line, and their requirements.
- The domain scope bounds the domain(s) that are relevant for reuse and what should be part of them.
- The asset scope identifies which assets should be part of the reuse infrastructure.

The goal of this step is to find domains (concept areas of the system that contain matching functionality; domains can be related to components but do not have to) and subdomains of the system and to identify the common and varying features (user visible capabilities) of the Product Line. To perform this analysis techniques from PuLSE-ECO [Sch03] are used.

Method

Domain structuring, domain descriptions and feature analysis (see [Sch03] and [Sch00] for further descriptions- The Feature Analysis is generally done with guided interviews , existing user documentation can be used as a source as described in [JD03].

Input Artifacts

Business Process, Business Goals, User documentation of existing systems (if available). Templates for Product-Feature Matrix, Structure Chart and domain descriptions (can be found in [Sch01])

Output Artifacts

Product Feature Matrix, Structure Chart, Domain descriptions, Data Table

3.3.4 Mapping of Features, Tasks and Data

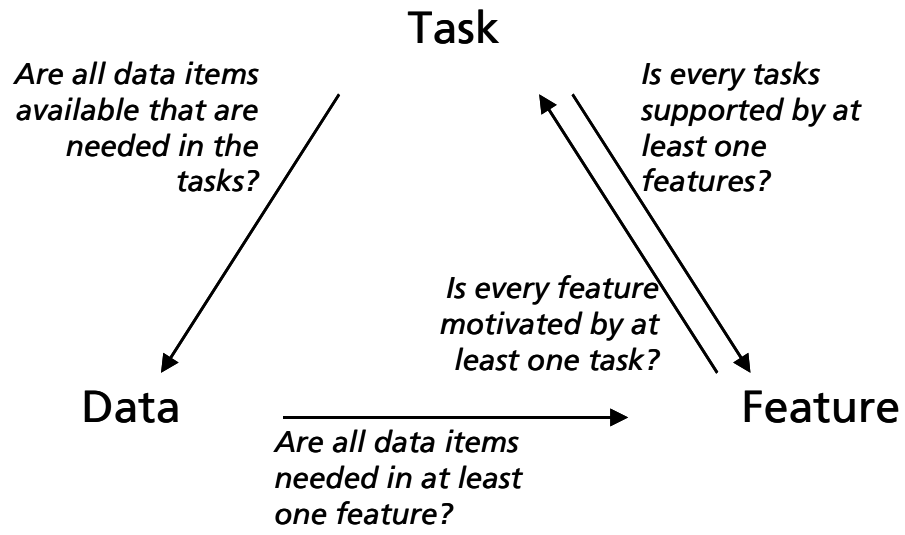


Figure 6: Mapping of tasks, features and data

Goal

The goal of this step is to ensure the quality of the developed model in respect to completeness, consistency and minimality. In addition this step aligns the results of the business process analysis and the feature analysis.

Method

A simple method to conduct this step is to create tables that align the features identified in step Feature Analysis to the tasks identified in step quality model, Usability goals

Business-Process Analysis as well as aligning the data to the tasks. As Figure 6 shows the mapping of the features to tasks ensures two things. First, it verifies that every feature is motivated by at least one tasks, which means we do not provide features that are unnecessary for the user (minimality). Second, it ensures that all tasks are supported by features, which means the system provides the desired support for the user (completeness). Additional one can make the same kind of mapping between data and tasks or data and features to ensure the completeness and minimality of the data model.

But mapping the tasks and features is not only an activity to ensure the quality, it also combines the results of the business process analysis that represents a Usability engineering oriented method and the feature analysis that is Product Line oriented. It joins the user-oriented view and the software/architectural oriented view to a coherent picture on the requirements.

As part of this step we also have to resolve the identified inconsistencies or gaps. This has usually be done by the experts, responsible for the software.

Input Artefacts

- Task list from the step quality model, Usability goals
- Business-Process Analysis
- Feature list and data model form the Feature Analysis .

Output Artifacts

- Table with mapping of tasks and features: Which features are "used" within which task?
- Table with mapping of tasks to data or features to data: Where is which data element involved?
- List of gaps or faults identified during this step

3.3.5 Task prioritisation

Goal

Especially in SMEs there is not enough time to analyze all tasks in depth. So there has to be a priorization of the tasks to give an order of processing and to give a focus Based on the information found so far the Tasks are prioritized. The tasks are prioritized with the help of the artifacts produced (e.g. the table with the mapping of the tasks and features). The tasks are prioritized based on the following factors:

- Commonality: Is there enough commonality between the features that are used within the task between different products. If there is no commonality this means that this task is done differently in different products and there is no reuse potential to be expected from improving the Usability there.
- Systematic Variability: On the other hand, as a Product Line also consists of systematic variabilities one important factor for a highly prioritized task is that some features differ (systematically) within the task. If all features of the task are common features, it can be expected that the existing implementation will be reused one to one, so there is no need to touch the code and so a Usability improvement in this task would not be efficient
- System Coverage: The business process also consists of tasks that are performed without the system or just with minor support of the system. Those tasks that are not performed with the system are of low importance for a Usability analysis as there is no large improvement potential for this task without changing the external business process.
- Importance for the business goals: Some tasks are more, some tasks are less important for the business goals.

The first three factors can be counted from the existing artifacts (e.g. from the product feature matrix), the last factor is based on subjective judgment and can be elicited in interviews.

Method

Similar to the PuLSE-ECO: Benefit and Risk Assessment [Sch03] interviews can be performed. The existing artifacts are analyzed and the numbers for the priorization factors can be computed from the task- and feature lists

Input Artifacts

Business Goals, , Tasks, Product Feature Matrix, , Task-Feature Mapping, Expert Opinion

Output Artifacts

Commented List of prioritized tasks of the system

3.3.6 Usability Evaluation

Goal

The goal of this step is to identify the weaknesses of the product (and its variations) in respect to Usability under consideration of the given Usability goals and the results of the domain priorization.

Especially in SMEs there are not enough resources to conduct participatory studies like Usability tests. Therefore there is a strong indication towards an analytical approach (expert evaluations).

Another aim is that the evaluation method not just exhibits design defects but proposes better design solutions.

Method

In our view analytical evaluation methods generally can be decomposed into two aspects: the Usability knowledge component and the walkthrough component. The Usability knowledge component contains rules, heuristics and best practice examples from the body of Usability experience. The walkthrough component describes how the examiner makes his/her way through the application (e.g. by task scenarios) and defines the decision points where he/she is to apply the Usability knowledge. Additionally useful is a report component which consists of a structured presentation of the evaluation results.

The most prominent analytical evaluation methods are probably the Heuristic Evaluation [NI94] and the Cognitive Walkthrough [WRL94]. Under perspective of the mentioned components these both have different strength and weaknesses. The Heuristic Evaluation uses a set of well established Usability heuristics and so has a medium rich Usability knowledge component - medium rich because the Usability knowledge is presented in a very condensed and generalized form. This requires the examiner to draw conclusions, which may by vague and subjective and require sound knowledge of user expectations and behaviour at the evaluator. The walkthrough component of Heuristic Evaluation is only ill defined. The Cognitive Walkthrough on the other hand has a very precise walkthrough component based on a detailed hierarchical task decomposition and it is strong in the Usability knowledge component, which is well founded on psychological theory of learning. But it is also restricted to judgements on ease of learning which limits its applicability.. Both methods focus on identifying usability problems but lack in providing useful hints on better design solutions.

Having especially the latter in mind we came across the upcoming field of Usability patterns (e.g. [Cas97]). In Usability patterns Usability knowledge is presented in a structured way, matching problems and specific contexts to proven design solutions. Patterns are often arranged in pattern collections or languages

[MJ98] which provide easier access by classifying patterns and defining relations between them.

While pattern collections are primarily meant to be useful during the design phase of an application it is also seems obvious to use them as a source of Usability knowledge for Usability inspections. Instead of using somewhat abstract heuristics and guidelines to identify Usability defects, using patterns one could simply judge if the design of an application element conforms to a well established standard solution. If it does not, a better design solution can be directly given from the pattern description. Therefore a collection of Usability patterns serves as the Usability knowledge component of our inspection method.

To complete our method we needed a well defined walkthrough component. It is obvious to use a scenario based walkthrough where the examiner makes his way through the application by doing a user task under typical circumstances. The scenarios can be obtained from the prioritized task list (see 3.3.5) but further interviews with domain experts and consulting the user manual may be required. In order to provide clear defined decision points during this walkthrough, we developed the following enhancement:

We collected a set of typical but somewhat abstract activities users are in while doing a task with a (nearly) arbitrary application. At first it is important to note that these activities are user centered (as opposed to system centered) in the sense of "What am I doing at this moment?". Examples for these activities are: data entry, seeking orientation, monitoring and error recovery. Then we classified all patterns in our collections according to these "activity modes". The examiner is now instructed to monitor his/her sequence of activities while doing the scenario. With a little practice in what one could call metacognitive monitoring he/she will be able to identify a set of patterns matching his/her current situation and judging if the canonical solution suggested by these patterns was applied by the designer.

The reporting of the Usability issues is supported by a database tool. During the inspection the deviations from patterns are recorded stating the task, the application element including a screenshot, the description of the current solution and references to the patterns which should be applied. From this record a inspection report is automatically generated.

As a result we established a method for Usability evaluation which has

- a rich Usability knowledge component, which can easily be customized and enhanced
- a well defined walkthrough component with clear decision points
- a structured and detailed report component with adequate tool support

We consider our method to be easy to learn and apply even for Usability non-experts. The judgement of possible Usability issues are based on confounded Usability knowledge and give detailed design suggestions to the interface designers. We suppose that the method can be easily integrated into the development process at different stages like design, early prototyping and testing.

Last but not least the pattern approach strongly promotes the reuse of prior established and well understood solutions. This is in line with the ideas of Software Engineering in general and especially Product Line Engineering.

Input Artifact

Prioritized task list

Scenarios

Output Artifact

Report with identified Usability issues and design recommandations

3.4 Results

The process that we described in the proceeding section helps small and medium enterprises in analysing usability and product line aspects in parallel. It gives business goal aligned recommendations for improvement, based on the description of the business processes, tasks, features and data of the system.

The general results of the process are:

- An explicit list of the business goals of the company for the product line
- A variant business process covering all aspects of user tasks, system tasks and user interaction with the system
- Tasks and features of the existing, planned and envisioned products including a prioritisation of the tasks. This prioritisation can be used for aligning forthcoming activities to the tasks where activities really pay (in terms of systematizable variability, number of features that are covered and usability importance)
- Recommendations for usability improvements and design recommendations. Those recommendations can be used to improve the design and usability of the product line within the highly prioritised tasks

So generally the results of applying the UseLine Process are Requirements that are focused on those parts of the system that are critical from usability as well as product line point of view.

4 Case Study: Sieda

4.1 Context

The company SIEDA – Systemhaus für intelligente EDV Anwendungen GmbH was created 1993 in Kaiserslautern and has a staff of 14 employees at the moment. The SIEDA GmbH is a specialist for the development of intelligent software for the solution of complex optimization problems in particular short-time manpower planning. Their main system "ORBIS Dienstplan" is a roster planning system with application emphasis in the public service. Contrary to most products available at the market it is specialized in planning by multi-layer models (e.g. with services around the clock). For intelligent planning constraint solver methods are used, Apart from the actual planning also the performed services are entered through the system and the data for the wage and salary statement can be computed with consideration of tariff regulation.

In 2002, first Product Line concepts had been introduced and a Product Line has started including a product providing web-based access to the roster planning system [JM02]. The current plans of Sieda at the beginning of the UseLine Project was it to built up a so called "light product" that includes a reduced functionality and is sold for a reduced price for persons that do roster planning in a smaller and more manageable environment like nursing homes with one single station. Here Usability and Product Line engineering are both important issues as the system shall be easily usable for an end user that does not use the planning functionality so often and the new system should be derived from the existing Product Line systematically but with low effort.

This section contains sanitized versions of the artifacts produced during the case study due to confidentiality reasons. Parts of the real artifacts can be found in the Appendix.

The general goal that we had in this case study was it to show that y integrating Usability analysis and Product Line Analysis we find more information in less time than with two single analyses. Furthermore, as this was the first and rather experimental application of the use line process we wanted, in the tradition of action research [RB01] just apply the process, see how it works an gain experience through closely watching the application and correcting the process if necessary. So there are no explicit hypotheses for this case study, we just want to show the applicability of the UseLine Process and give examples for how to perform the process steps and for the artefacts that are produced.

By integrating Usability and Product Line engineering, we are able to handle contradictory requirements (reuse and Usability) in an integrated approach.

4.2 Performing the Case study

4.2.1 Overview

The Case Study was performed in the Sieda Context between June? 2003 and January 2004. Two Persons from IESE and two persons from Sieda were involved. The Goal of Sieda in this Case Study was it to derive a third product from their existing Product Line of rostering software systems, a so-called light product that should serve a low-end market. While deriving the product the overall Usability of the system should be evaluated and improved.

So the Usability goals of the company were:

- Overall improvement of the systems in the Product Line
- Derivation of a usable light product from the Product Line

The Product Line goals of the company were:

- Controlled derivation of the light product from the existing Product Line while reusing as much functionality and components as possible
- Improving the composition of the light product (selection of the right features for the envisioned market

4.2.2 Business Goal Analysis

The results of the business goal analysis are the following:

The primary business goal for the new product is to address a new segment of the market. It should be more consumer-oriented and it should be a door-opener for the old product. In terms of non-functional requirements for the software this means, it needs less functionally, but has to persuade the end user in a very short time, that it is a good product. This means it has to be cheap and it has to have a "easy-to-use" look and feel. It can only be cheep , if it needs less implementation effort (which can be achieved by reusing as much of the existing code as possible) and by having minimal training and support costs. Both minimal training and support cost require an improved Usability for the new product.

Figure 7 shows the described dependency between the business goal and the non-functional requirements. It also includes first measurable non-functional goal which could be derived and which should following design decision.

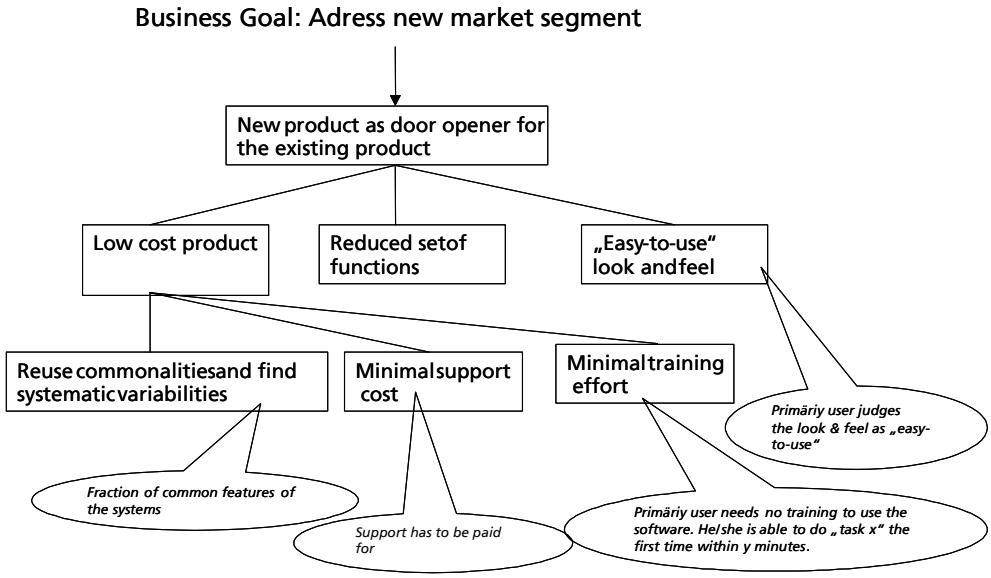


Figure 7. Business goals and non-functional requirements

4.2.3 Business-Process Analyse

The business process analysis showed, that four different user groups work with the system: Pflegepersonal (short: Pp), "Stationsleitung (short: Sl), and Personalabteilung (short: Pa). The "Stationsleitung" as the planning person is the main target group for both variations of the product. For the light product of the version it is the only user role that has to be supported. A description of this primary persona is shown in the appendix. The focus of both product variants one the same user group already indicates at this stage, that it is very likely to reuse many of the Usability improvements for both product variants.

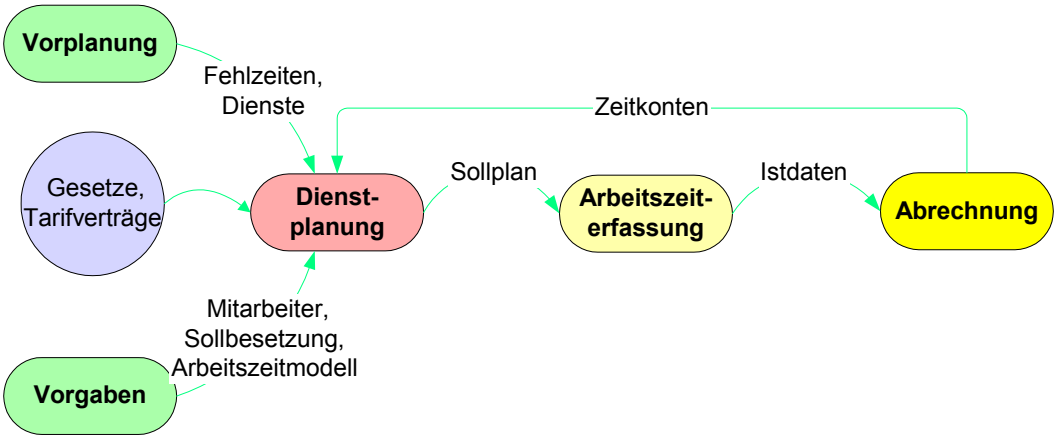


Figure 8 Overview of the Sieda Business Process

A task list summarizes the tasks that are supported by the different variants of the system (see Table 2).

The business process in Figure 8) shows sanitized of the process supported by the Sieda products. The real process is more complex and more concrete and can be found in the appendix.

Tasks	Orbis	Orbis-Light
Pa: Personaldaten einpflegen	X	X
Pl: Dienste & Schichtfolgen definieren	X	X
Pl: Dienstplan genehmigen	X	
Pl: Dienstplan kontrollieren	X	
Pl: Statistiken für Kontrolling erstellen	X	
Pp: Dienstplan zur Kenntnis nehmen	X	
Pp: Terminwünsche/Urlaubszeiten angeben	X	X
Sl: Dienstplan publizieren	X	X
Sl: Dienstplanung vornehmen	X	X
Sl: Ist-Daten eingeben	X	X
Sl: Monat abschließen	X	X
Sl: Statistiken für Controlling erstellen	X	X
Sl: Umplanen	X	X
Sl: Urlaub genehmigen	X	
Pa: Bezügenachweise erstellen		
Pa: Bezüge berechnen und auszahlen		
Pp: Bezüge prüfen		
Pp: Dienst ausführen		
Pp: Geleistete Ist-Zeit melden		
Pp: Ungeplante Abwesenheit melden		

Table 2 Task list showing the tasks supported by the variants of the system. An "X" marks tasks that are supported.

4.2.4 Feature Analysis

In the feature analysis, the common and variable Domains and Subdomains as well as the features of the Orbis and the Orbis Light system were identified. The Features and Domains of the Orbis System and the special product Orbis-Rettungsdienst were found by doing a CaVE-Analysis (see [JD03]) on the User Documentation of the Orbis Systems. The features, domains and the data were marked in the documentation (see 9) and afterwards brought in form of a matrix. The results of this analysis and the CaVE-Process are described in [JD04a] and [JD04b]. The result were domain descriptions (for an example see Figure 10, a domain structure chart (can be found in the appendix) and a preliminary product feature matrix (for an excerpt of the product feature matrix see Table 3, the complete matrix can be found in the appendix. About 130 Features, 7 Domains and12 Subdomains were identified. The matrix and the domain descriptions were then validated and corrected by the domain experts and the features for Orbis light were added by the experts.

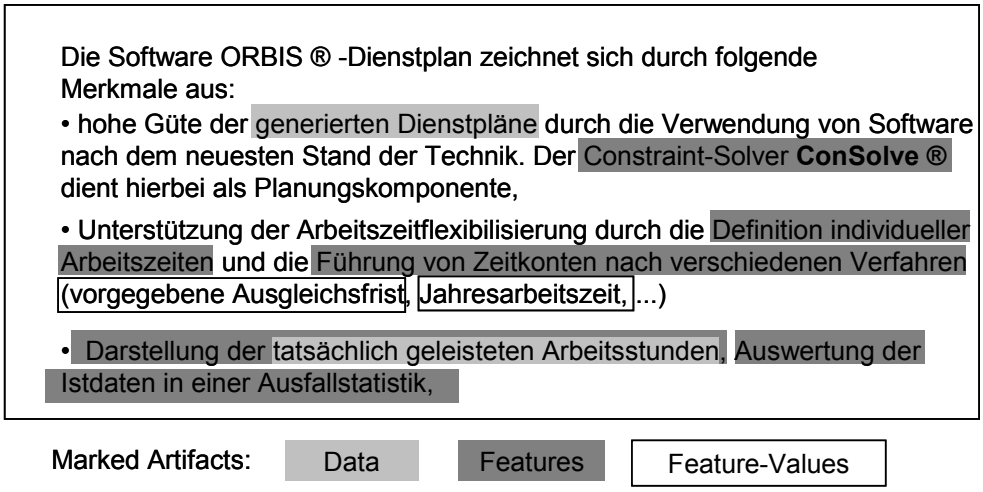


Figure 9 Analysis of the Orbis Documentation

Domäne: Systemverwaltung
Informationsquelle: Orbis Handbuch
Funktion: Die Systemverwaltung schafft die Voraussetzung für die unterschiedlichen Benutzer, mit ORBIS®-Dienstplan zu arbeiten. Die Benutzer müssen zusammen mit ihren Rechten und ihrer Sicht auf die Organisation in das System eingegeben werden
Domänengrenzen:
In:
Out:
Higher Level Domänen:
Lower Level Domänen: Datenbank
Domänen auf gleichem Level: -
Subdomänen: Allgemeine Verwaltung, Konten, Konfiguration, Personaldaten, Benutzergruppen
Übergeordnete Domäne: -
Funktionen: ....Konten verwalten, Konfiguration Einstellen, Personaldaten eingeben, Benutzergruppen anlegen...
Daten: ..... Alle Systemverwaltungsdaten incl. Konten und Personaldaten

Figure 10 An example domain description (in german)

Domain	Subdomain	Feature	Values	Product1	Product2	Product3
System-verwaltung	allgemein			X	X	
		Unterstützung verschiedener Diensttypen	Frühdienst, Tagesdienst, Spätdienst, Nachtdienst....	Alle	Alle	N
		Bereitschafts-dienste		X	X	
	Konfigura-tion			X	X	X
		Tarifgruppen-bezug		X	N	X
		Freie Fehlzei-ten		X	X	N
		….				

Table 3 An excerpt of the product feature matrix

4.2.5 Mapping

In the mapping step, a quality check was done by correlating the features to the tasks. The features in the product feature matrix were mapped to the tasks by combining the two matrices in Table 2 and Table 3. Some missing features and a few features that were on the wrong place were identified by this analysis. The combined matrix is then used for prioritising the tasks.

4.2.6 Task Prioritisation

After the check, the tasks were prioritized as described in section 3.3.5 with the factors commonality, systematic variability, system coverage and importance for the business goals. As a result the following five tasks were selected for the Usability evaluation:

- „Dienste und Schichtfolgen definieren
- „Terminwünsche angeben“
- „Dienstplanung vornehmen“
- „Ist-Daten eingeben“
- “Umplanen”

4.2.7 Usability Evaluation

Preparation

The Usability Evaluation followed the described evaluation method based on Usability Patterns. At first the examiner interviewed a member of the support staff to

- gain basic domain knowledge
- create more detailed scenarios from the prioritized tasks
- check if the sample data allows to run the task with the test installation

The results of the interview were recorded in a standardized form, which states prerequisites, typical users, data, dialogues etc. of the scenario.

Another result from the interview was that one of the tasks (“Umplanen”) is identical to another one (“Ist-Daten eingeben”) in terms of required data and dialogues, so only four scenarios were required for the evaluation.

For one of the tasks (3: “Dienstplanung vornehmen”) it was not possible to conduct the examination because (for yet unknown reasons) the corresponding function was deactivated in the test installation. Instead the dialogues were reviewed using the screenshots and descriptions in the online help; all identified deviations were comparable or at least transferable from the previous results in the directly examined dialogues.

The examination of a dialogue was conducted following the steps:

1. Identify the current User Activity from the flow of the scenario
2. Get a list of patterns from the pattern repository
3. From the list select patterns which possibly match the current dialogue or situation
4. Compare each pattern to the current dialogue
5. Describe deviations of the dialogue from the proposed patterns
6. Outline a design solution based on the proposed patterns

Results

During the evaluation 18 examinations following the above steps were conducted. 9 examinations belonged to the first scenario, 5 to the second and 4 to the fourth scenario. The first scenario required more steps because it was actually composed of two activities (Dienste definieren, Schichtfolgen anlegen).

We used 6 of the user activities in our repository to select the patterns. The frequencies of use are shown in Table 4. The user activity "Orientieren" was most often used. The other activities are typical for data oriented applications.

User activity	**Frequency of use**
Orientieren	8
Funktion suchen	2
Daten suchen	3
Daten eingeben	4
Bedienung lernen	1
Fehler beheben	1
Daten lessen	1

Table 4 Frequency of User activities used for pattern selection

Altogether 32 distinct patterns were used during the evaluation, ranging from 2 to 10 Patterns per step. A total of 95 examination points (one pattern tested against one dialogue) were conducted.

Table 5 shows the 8 (first quartile) most often tested patterns during the examination. These 8 patterns will be used further on to outline the results of the usability evaluation.

Pattern	Times tested
Central Working Surface	8
Overview besides Detail	7
Demonstration	4
Narrative	5
Repeated Framework	4
Control Panel	8
Hierarchical Set	3
Step-by-Step Instructions	4

Table 5 patterns in the examination

One could roughly categorize these 8 patterns into the following three categories:

General layout or interaction pattern:

- Central Working Surface
- Overview besides Detail
- Hierarchical Set

Patterns concerned with ease of learning:

- Demonstration
- Narrative
- Repeated Framework

Patterns describing more specific design solutions:

- Control Panel
- Step-by-Step Instructions

In the following we present a concrete example of identifying a usability problem and outlining a recommendation using the Usability Pattern Inspection Method

We often examined patterns which describe certain dialogue elements for entering data and choosing actions more detailed:

In many dialogues the entry of date and time data is required. The consistent solution for this data entry in the application is the *Structured Text Entry*, which forces the user to give the data in the required (system-defined) syntax. This helps avoiding wrong data entry but might be annoying to the user. There exist two alternative patterns for this problem: *Forgiving Text Entry* states that the data entry's format should not be restricted but the system should infer and normalize the time or date. For example the user entry "4.5." becomes 04.05.2004 if the year is clear from the context (which should be the case in many dialogues). As another alternative the *Unambiguous Format* pattern could be applied which would result in a calendar widget to enter date items.

In this case we recommend to provide more intelligent data entry fields, e.g. free date and time entry or unambiguous widgets.

5 Conclusions

In the UseLine project, the interplay between the two Software Engineering disciplines Usability Engineering on the one hand and Product Line Engineering on the other hand is investigated. Although both disciplines seem to contain contradictory elements – e.g. Usability Engineering focuses on the individual user whereas Product Line engineering concentrates on reuse and commonality between products an integrated approach that focuses on both aspects has advantages. Especially for small and medium enterprises who traditionally have a strong customer focus and do often not have the time and money to introduce both – Product Line engineering and Usability engineering – an integrated approach can help to achieve a large market advantage – a usable Product Line.

We described here the useline approach for planning and requirements engineering of usable Product Lines. The outcomes of the UseLine approach are:

- An explicit list of the business goals of the company for the product line
- A variant business process covering all aspects of user tasks, system tasks and user interaction with the system
- Tasks and features of the existing, planned and envisioned products including a prioritisation of the tasks. This prioritisation can be used for aligning forthcoming activities to the tasks where activities really pay (in terms of systematizable variability, number of features that are covered and usability importance)
- Recommendations for usability improvements and design recommendations. Those recommendations can be used to improve the design and usability of the product line within the highly prioritised tasks

So generally the results of applying the UseLine Process are Requirements that are focused on those parts of the system that are critical from usability as well as product line point of view.
The approach was exemplarily applied in a small and medium enterprise, the Sieda GmbH. All steps of the approach could be efficiently performed and led to useful results for the company. The design recommendations that were given as the last step of the approach are currently being implemented, first results will be in public when the new product will be released next year.

References

[BFK+99] J. Bayer, O. Flege, P. Knauber, R. Laqua, D. Muthig, K. Schmid, T. Widen, J.-M. DeBaud. PuLSE: A Methodology to Develop Software Product Lines. In Proc. of the Symposium on Software ReUsability (SSR'99), pp. 122-131, 1999.

[BHK+01] M. Broy; S. Hartkopf; K. Kohler; D. Rombach. Germany: Combining Software and Application Competencies. IEEE Software (18), 2001.

[BMW99] J. Bayer, D. Muthig, and T. Widen. Customizable Domain Analysis. In Proceedings of the First International Symposium on Generative and Component-Based Software Engineering (GCSE '99), Erfurt, Germany, September 1999

[BH98] Beyer, H., Holtzblatt, K., *Contextual Design: Defining Customer Centered Systems*, Morgan Kaufmann Publishers,1998

[Cas97] Casaday, G., Notes on a Pattern Language for Interactive Usability. In. CHI Electronic Publications: Late Breaking/ Short Talks, 1997.

[CN01] P. C. Clements and Linda Northrop. Software Product Lines: Practices and Patterns. SEI Series in Software Engineering. Addison-Wesley, August 2001

[CL99] Constantine, L., Lockwood, L., *Software For Use*, Addison Wesley, 1999

[Co99] Cooper, A., The inmates are running the asylum, Chapter 9: Designing for Pleasure, SAMS Publishing, 1999

[DM96] Dray, S. & Mrazek, D. "A day in the life:" Studying context across cultures. In J. Nielsen & E. del Galdo, Eds. *International User Interfaces*, John Wiley & Sons, 1996.

[HB96] Hewett, T., Baecker, R., Card, S., Carey, T., Gasen, J., Mantei, M., Perlman, G., Strong, G., and Verplank, W., ACM SIGCHI Curricula for Human-Computer Interaction, 1996, htttp://www.acm.org/sigchi

[ISO01] DIN EN ISO 9126-1 Software Engineering - Product Quality - Part 1: Quality Model, 2001

[ISO98] DIN EN ISO 9241-11. Ergonomische Anforderungen für Bürotätigkeiten mit Bildschirmgeräten – Teil 11: Anforderungen an die Gebrauchstauglichkeit; Leitsätze (ISO 9241-11:1998). Berlin, 1998.

[JD03b] I. John, J. Dörr Extracting Product Line Model Elements from User Documentation. IESE Report 112.03/E, Fraunhofer IESE, 2003

[JD04a] I. John, J.Dörr. Requirements Engineering aus existierenden Dokumenten. In Produktlinienentwicklung in Deutschland. dpunkt, 2004, to appear

[JD04b] I. John, J. Dörr, K. Schmid. User Documentation Based Product Line Modeling. IESE Report 004.04/E, Fraunhofer IESE, 2004

[JK04] I. John, D. Kerkow, K. Schmid. Usable Product Lines: Issues and requirements. ICSE 2004 Workshop on Bridging the Gaps Between Software Engineering and Human-Computer Interaction; Edinbourgh, Scotland, May 2004.

[JM02] I. John, D. Muthig, E. Tolzmann, P. Sody Efficient and Systematic Software Evolution Through Domain Analysis. Proceedings of RE'02, Essen, Germany,September 2002.

[KCH90] K. Kang, S. Cohen, J. Hess, W. Novak, S. Peterson. Feature-Oriented Domain Analysis (FODA) Feasibility Study. CMU/SEI-90-TR-21, Software Engineering Institute, 1990.

[KDP04] D. Kerkow, J. Dörr, B. Paech, T. Olsson & T. Koenig. Elicitation and Documentation of Non-functional Requirements for Sociotechnical Systems. In: J. L. Maté & A. Silva (Ed.), Requirements Engineering for Sociotechnical Systems. Idea Group, Inc., to appear 2004.

[KKD03] D. Kerkow, K. Kohler & J. Dörr. Usability and Other Quality Aspects Dervied from Use Cases, forUse 2003.

[KLB03] Kohler, K., Leidermann, F., Birk, A., Der Weg zu einer stärkeren Verzahnung von Usability Engineering und Software Engineering Proceedings of the 1st annual GC-UPA Track Stuttgart, September 2003

[Lau03] S. Lauesen, "Task Descriptions as Functional Requirements", IEEE Software, March/April 03

[MB03] J. Machate, M. Burmester (Hrsg.). User Interface Tuning: Benutzungsschnittstellen menschlich gestalten. - Frankfurt am Main: Software und Support Verl., 2003

[Ma99] Mayhew, D.J., The Usability Engineering Lifecycle, Morgan Kaufmann Publishers, 1999

[MJ98] Mahemoff, M. J., Johnston, L. J., Pattern Languages for Usability: An Investigation of Alternative Approaches, In: Tanaka, J. (Ed.), Asia-Pacific Conference on Human-Computer Interaction 98 Proceedings, 1998.

[MK04] M. Medina, K. Kohler A Brief introduction to Usability Patterns. IESE Report, Fraunhofer IESE, 2004, to appear.

[Ni94] Nielsen, J., Heuristic Evaluation. In: [NM94], 25-62.

[NM94] Nielsen, J., and Mack, R. L. (Eds.), Usability Inspection Methods, John Wiley & Sons, 1994.

[Ni03] Nielsen, J., Constructing the User. HCII2003, Crete, Lawrence Erlbaum, 2003 (http://www.cbs.dk/staff/lene/Artikler.htm)

[Nor98] D. A. Norman. The Psychology of Everyday Things, New York, 1988.

[PK03a] B. Paech, K. Kohler. Usability Engineering integrated with Requirements Engineering. ICSE 2003 Workshop on Bridging the Gaps Between Software Engineering and Human-Computer Interaction; Portland, USA, May 2003.

[PK03b] B. Paech, K. Kohler Task-driven Requirements in object-oriented Development In J. Leite, J. Doorn, (eds.) Perspectives on Software Requirements, Kluwer Academic Publishers, 2003.

[RB01] P. Reason, H. Bradbury (eds.). Handbook of Action Research. Sage Publications, London, 2001

[Sch03] K Schmid. Planning Software Reuse – A Disciplined Scoping Approach for Software Product Lines. PhD Theses in Experimental Software Engineering, Fraunhofer IRB, 2003

[Sch00] K. Schmid. Product Line Mapping Report. IESE-Report 028.00/E,Fraunhofer IESE 2000.

[SBK+00] K. Schmid, U. Becker-Kornstaedt, P. Knauber, and F. Bernauer. Introducing a software modeling concept in a medium-sized company. In Proceedings of the 22nd International Conference on Software Engineering (ICSE 2000), Limerick, Ireland, 2000.

[SV02] K. Schmid and M. Verlage. The Economic Impact of Product Line Adoption and Evolution. IEEE Software, 19(4):50--57, JulyAugust 2002

[SW00] K.Schmid and T.Widen. Customizing the PuLSE Approach to the Demands of an Organization. 7th European Wokshop on Software Process Technology (EWSPT2000), LNCS 1780, Febuary 2000.

[WL99] D. Weiss, C. Lai. Software Product Line Engineering. Addison-Wesley, 1999.

6 Appendix

This appendix contains the artifacts that were produced for Sieda during the UseLine Project. Some of the artifacts produced during the projects are confidential and therefore appear only in the confidential version o the report.
As the Project was done in Germany, most of the Artifacts and descriptions are in German.

6.1 Business Goal Overview

Die folgende Tabelle zeigt die priorisierten Business Goals der Sieda Software Orbis und Orbis Light

	Orbis	**Orbis Light**
Verringerung der Time to Market		unwichtig
Reduzierung des Entwicklungsaufwands		eher unwichtig
Reduzierung des Wartungsaufwands		Sehr wichtig
Erhöhung der Qualität (welcher Qualitätsaspekt)		wichtig
Erhöhung der Kundenzufriedenheit	wichtig	wichtig
Reduzierung der Supportkosten		Seht wichtig
Reduzierung der Trainingskosten		Sehr wichtig
Erhöhung der Task Effizienz	wichtig	Sehr wichtig
Risikominimierung (in der Entwicklung)		Unwichtig
Reduktion der Expertenlast		Unwichtig

Bemerkungen zu den Business Goals:

- Für die Qualität des Produktes ist maßgeblich wie fehlerhaft das Produkt ist. Dabei werden Fehler als unerwartetes Systemverhalten definiert. Erwartetes Systemverhalten ist in der Systemdokumentation dokumentiert. Fehler bedeuten Abweichungen von diesem erwarteten Systemverhalten.
- Time to Market ist nicht so wichtig, wie Software Qualität. „Sonst ist die Gefahr groß, dass der Kunde aussteigt.

- Gewinnung neuer Kunden (auch für Orbis) ist ein primäres Ziel von Orbis Light
- Erhöhung der Qualität und Reduktion der Supportkosten beeinflussen sich gegenseitig. Die Reduktion von Fehlern ist also sehr wichtig.
- Wichtig sind zusätzlich die Kosten die bei der Entwicklung entstehen

6.2 High Level Target Group and Task Analysis

Folgende Zielgruppen gibt es:

A Pflegedienstleitung (Kernanwender):

- Im Krankenhaus: Stationsleitung (Oberschwester) meist langjährige Pflegeerfahrung
- Im Altenpflegebereicht: Ausbildungsstand eher geringer (Pflegeausbildung). In der Altenpflege wird Pflegedokumentation auch mit dem Computer gemacht (hier wird manchmal jeder Handgriff dokumentier), ist sehr unterschiedlich von Altersheim zu Altersheim
- Im Rettungsdienst: Rettungsdienstleiter (=Ausbildung zum Rettungsassistent), leitet die Schicht, teilt die Sanitäter ein
- Zukünftig evtl. Polizei (Ausbildungsstand hier unbekannt)
- Bei der Feuerwehr ist ähnlich zum Rettungsdienst
- In Behindertenwerkstätten

-> Alles öffentlicher Dienst.

Für das "Light Produkt" erwarten man eine breitere Nutzergruppe: Dienstplanung auch im privaten Bereich. Schwimmhallenbetreiber, Wachpersonal (eher schlecht, weil wir nicht mit wechselnden Standorten umgehen können) – machbar bei Industriebetrieb oder Werkstatt mit Schichtbetrieb. Die primäre Zielgruppe für Orbis Light ist die Stationsleitung im Krankenhaus die zuhause Dienstpläne erstellen will oder die Gruppenleitung oder Pflegedienstleitung im Altenheim. Die Kunden kennen oft nur das Orbis System als einzige Software, u.U. noch Standardsoftware wie Word oder Software zur Pflegedokumentation. Die Kunden sind also oft keine versierten Computernutzer (können aber versiert sein)

Aufgaben des Pflegedienstleitung:

- Abrechnung als wichtigste Aufgabe: Ermittlung der Zeiten für die Bruttolohnberechnung (je nach Zulagen, Wechselschichtzulagen)
- Erstellen der Pläne
- Erfassen die Ist-Daten (Bewegungsdaten; Korrektur der Pläne, Arbeitszeiterfassung)
- Urlaubsplanung erstellen.

– Controlling (Ausfallstatistik, …, Optimierung der Ausgaben, möglichst wenig Zuschläge – wie sieht der billigste Dienstplan aus). Controlling wird zunehmend wichtig (bisher herrscht hier noch viel Verwirrung, bzgl. dessen, was die Kunden hier wissen wollen)
– System an den Haustarifvertrag anpassen

Die Dienstpläne werden ausgedruckt und an die Wand gehängt (Datenschutztechnisch sehr fragwürdig) , Alternativ (Web-Portal für die Mitarbeiter), Spezifikation von Wünschen (z.B. „meine Oma wird 80“) – heute über das Web-Portal

A’ Andere Dienstplanersteller:

– Ärzte verplanen sich selbst, heute meist ohne Software – für die Zukunft aber wünschenswert
– Einzelhandel (ist kein Marktsegment, Konkurrenz (Invision) – hier müssten noch andere Aspekte abgedeckt werden, Orbis ist hierfür nicht flexible genug)

B. Schichtangestellte (Krankenschwestern, Sanitäter, ...)

Die Leute die verplant werden, verwenden das System nur über das Web-Portal. Außerdem bekommen sie Berichte ausgedruckt, die mit dem System generiert werden.

C. Administrator –

– Sehr unterschiedliche Ausbildung (z.B. umgeschulte Psychologen)

Aufgaben:
Einrichtung der Software, Zeitwirtschaftsfunktionen (Stechuhr) ergänzen

D. Personalabteilung

– Sorgt dafür das Personaldaten gepflegt werden
– Sorgt dafür, dass die Abrechungsdaten in ein Zahlungssystem übertragen werden (löst die Zahlung aus)
– Korrigieren von Konten (z.B. wenn Zeitkonten ausgezahlt werden

E. Betriebsrat um Einsicht zu gewinnen:

– schaut sich die Dienstpläne an, weil der Mitspracherecht
– überwacht ob z.B. Überstundenregelungen eingehalten werden

F. Oberarzt/Chefarzt/Klinischer Direktor

Nutzt die Software zu Controlling Zwecken (Prüfen, ob der Dienstplan schon erstellt werden)

Klinische Direktor – Auch Controlling;, vergleicht verschiedenen Abteilungen und Berufsgruppen (Ausfallzeiten von Pflegepersonal im Unterschied zu Ärzten)

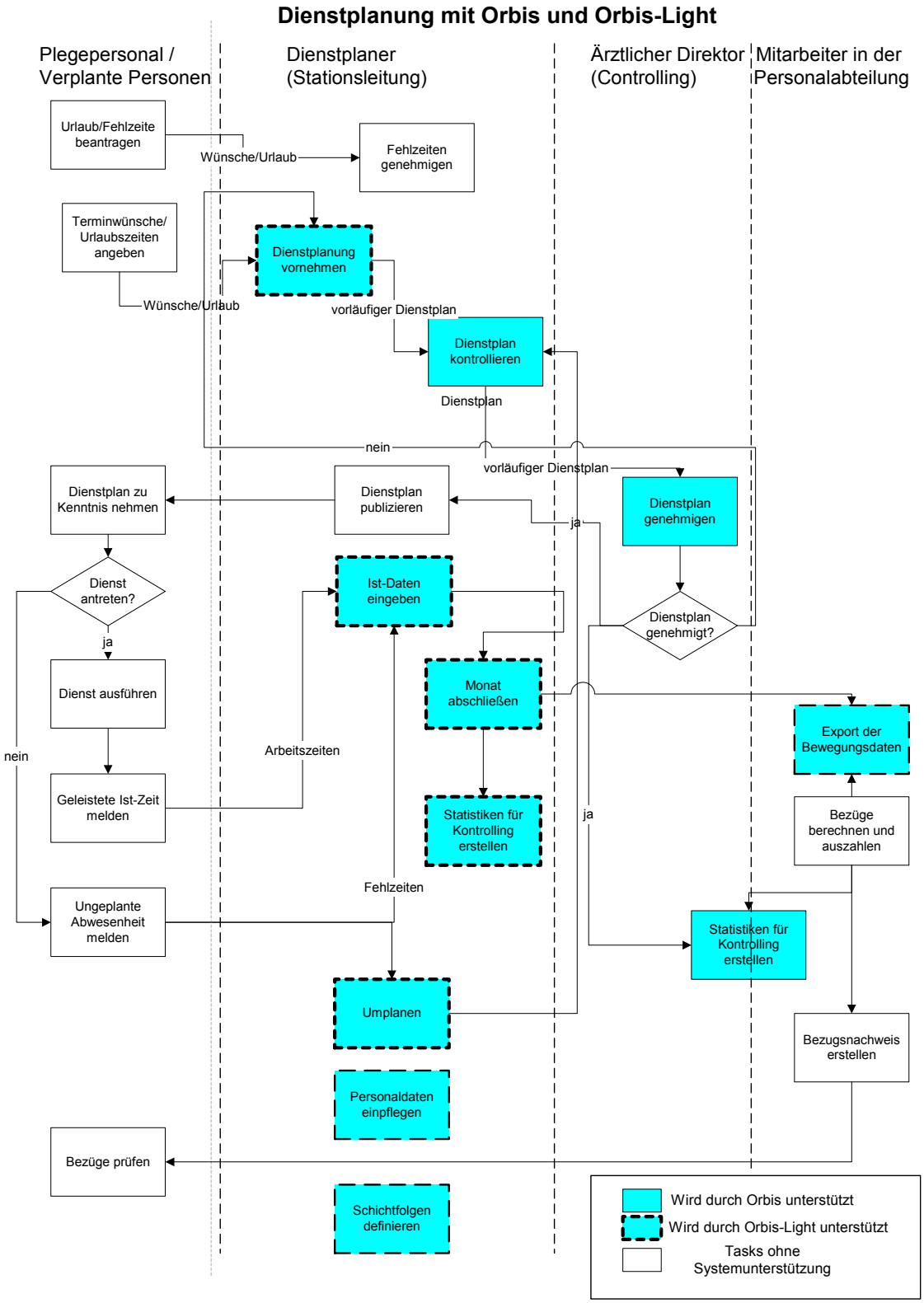
Phasen:

Die Nutzung des Systems lässt sich in folgende Phasen einteilen:

- Vorplanung
- Dienstplanung
- Arbeitszeiterfassung (+ Fehlzeiten)
- Abrechnung
- Zusätzlich Auswertung/Controlling

6.3 Business Process

Die folgende Grafik zeigt den Sieda Business Prozess der von Orbis und von Orbis Light unterstützt wird. Der Prozess wurde mit Visio modelliert, Swimlanes zeigen die Verantwortungsbereiche der einzelnen Personen. Es wurden sowohl die durch Orbis Light als auch die nur durch Orbis unterstützen Tasks (gestrichelteLinie) mit aufgenommen. Da OrbisLight eine Untermenge von Tasks von Orbis unterstützt gibt es keine neuen Tasks die nur Orbis unterstützt.



6.4 Task List

Zusätzlich zu den Tasks im BusinessProzess wurde eine Taskliste erstellt. Dabei bedeutet PA Personalabteilung, Pl die Planende Person, Pp die verplante Person und C das Controlling :.

Tasks	Orbis	Orbis-Light
Pa: Personaldaten einpflegen	X	X
Pl: Dienste & Schichtfolgen definieren	X	X
C: Dienstplan genehmigen	X	
Pl: Dienstplan kontrollieren	X	
Pl: Statistiken für Kontrolling erstellen	X	
Pp: Dienstplan zur Kenntinis nehmen	X	
Pp: Terminwünsche/Urlaubszeiten angeben	X	X
Pl: Dienstplan publizieren	X	X
Pl: Dienstplanung vornehmen	X	X
Pl: Ist-Daten eingeben	X	X
Pl: Monat abschließen	X	X
C/PA: Statistiken für Kontrolling erstellen	X	X
Pl: Umplanen	X	X
Pa: Beügsnachweise erstellen		
Pa: Bezüge berechnen und auszahlen		
Pp: Bezüge prüfen		
Pp: Dienst ausführen		
Pp: Geleistete Ist-Zeit melden		
Pp: Ungeplante Abwesenheit melden		

6.5 1. Primäre Persona: Stationsleitung

“Der Computer tut nie was ich will”

Annemarie Oberlies
- Schwester Annemarie

Ziele:

- Reibungsloser Ablauf der Pflegeschichten

Aufgaben:

- Dienstplan erstellen und abstimmen
- Monatsabrechnung an Personalabteilung schicken
- Pflegetätigkeiten übernehmen
- Lernschwestern anleiten

Hintergrund:

- Oberschwester auf Station 5C, Kardiologische Privatstation mit 50 Betten und 20 Krankenschwestern
- Seit 3 Jahren Oberschwester, davor Krankenschwester
- 45 Jahre, verheiratet, 2 Kinder
- Ausbildung zur Krankenschwester, Weiterbildung zur Pflegedienstleiterin
- Im privaten kein Umgang mit PCs. Der 17 jährige Sohn hat zu Hause eine PC mit Internetanschluß.
- Hat in Station 5 C Zugriff auf den PC des Sekretariats. Verwendet diesen auch gelegentlich für andere Aufgaben
- Sie ist die einzige in Station 5 C, die sich mit Orbis auskennt

Bedürfnisse:

- Einfach zu Benutzen
- Hilfe System
- Annemarie hat für die Dienstsplanung nur recht wenig Zeit, da sie auch pflegerische tätig ist und sich um die Lernschwestern kümmert

Scenario:

Schwester Annemarie hat Spätdienst. Sie will heute Abend die Dienstplanung für den nächsten Monat fertig stellen, da morgen Stationsbesprechung ist. Sie hat Orbis gerade gestartet, da wird sie von Schwester Monika zur Hilfe gerufen. Dem Patienten Mayer geht es schlecht, er ist heute operiert worden. Erst ½ h später kommt sie zurück an den Computer um die Dienstplanung fort zu setzen.

6.6 Identified Domains

Zur Produktlinienanalyse mit Usability Aspekten wird eine Variante der Produktlinien Scoping Methode PuLSE-Eco verwendet. Diese Informationen wurden durch eine Analyse der Orbis Dienstplan Dokumentation mit Hilfe der CaVE –Methode (Commonality and Variability Elicitation) und durch das durch Interviews und frühere Projekte gewonnene Domänenwissen angereichert

Das Orbis Dienstplansystem besteht aus folgenden Domänen/technischen Bereichen :

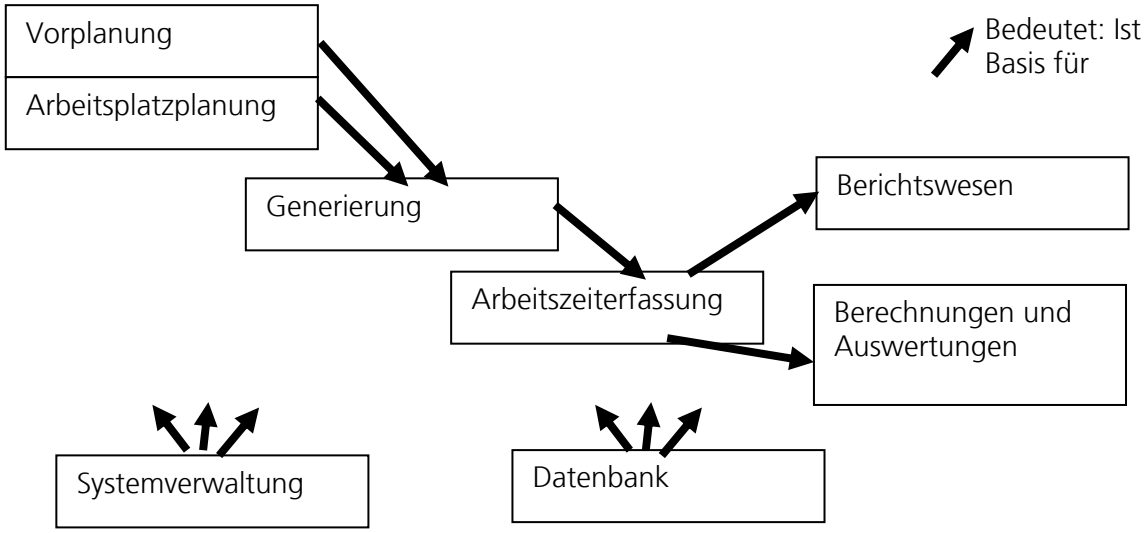


Abb :Domänenstrukturdiagramm

Die Domänen sind wie folgt beschrieben:

6.6.1 Domäne: Systemverwaltung

Informationsquelle: Orbis Handbuch
Funktion: Die Systemverwaltung schafft die Voraussetzung für die unterschiedlichen Benutzer, mit ORBIS®-Dienstplan zu arbeiten. Die Benutzer müssen zusammen mit ihren Rechten und ihrer Sicht auf die Organisation in das System eingegeben werden
Domänengrenzen:
In:
Out:
Higher Level Domänen:
Lower Level Domänen: Datenbank
Domänen auf gleichem Level: -
Subdomänen: Allgemeine Verwaltung, Konten, Konfiguration, Personaldaten, Benutzergruppen
Übergeordnete Domäne: -
Funktionen: ....Konten verwalten, Konfiguration Einstellen, Personaldaten eingeben, Benutzergruppen anlegen...
Daten: ..... Alle Systemverwaltungsdaten incl. Konten und Personaldaten
Systemcharakteristiken

Existierende Artefakte: ja...
System relationen: wo/wie implementiert?
Produkt: alle

6.6.2 Domäne: Vorplanung

Informationsquelle: Orbis Handbuch
Funktion: Unter die Vorplanung fällt die Erfassung aller Fehlzeiten, Dienste und Arbeitsplatzwechsel, die bereits vor dem Beginn der Dienstplanung für den Planungszeitraum und darüber hinaus bekannt sind. Die Vorplanung erfolgt dabei für eine frei wählbare Gruppe
Domänengrenzen:
In:
Out:
Higher Level Domänen: -
Lower Level Domänen: Datenbank, Systemverwaltung
Domänen auf gleichem Level: Arbeitsplatzplanung, Generierung, Erfassung
Subdomänen: allgemeine Vorplanung, Sollbesetzung
Übergeordnete Domäne: -
Funktionen: .Erfassung von Diensten und Fehlzeiten, Definition von Schichtfolgen, Sicherstellen der Sollbesetzung
Daten: Dienstplan
Systemcharakteristiken

Existierende Artefakte: ja...
System relationen: wo/wie implementiert?
Produkt: alle

6.6.3 Domäne: Arbeitsplatzplanung

Informationsquelle: Orbis Handbuch
Funktion: Zuordnung von Mitarbeitern auf Arbeitsplätze.
Domänengrenzen:
In:
Out:
Higher Level Domänen:
Lower Level Domänen: Datenbank, Systemverwaltung
Domänen auf gleichem Level: Vorplanung, Generierung, Erfassung
Subdomänen: -
Übergeordnete Domäne: -
Funktionen: . Zuordnung von Mitarbeitern auf Arbeitsplätze
Daten:

Systemcharakteristiken

Existierende Artefakte: ja...
System relationen: wo/wie implementiert?
Produkt: alle
Do

6.6.4 Domäne: Automatische Dienstplangenerierung

Informationsquelle: Orbis Handbuch
Funktion: Liegen sämtliche Daten für die Dienstplanung vor, so kann die automatische Dienstplangenerierung durchgeführt werden. Bei der Dienstplangenerierung wird anhand der vorgegebenen Vorplanungsdaten ein Dienstplan erzeugt, der die vorgegebenen Bedingungen bestmöglich erfüllt
Domänengrenzen:
In:
Out:
Higher Level Domänen: -
Lower Level Domänen: Datenbank, Systemverwaltung
Domänen auf gleichem Level: Arbeitsplatzplanung, Vorplanung, Erfassung
Subdomänen: ConSolve, Generierung, Genehmigungen
Übergeordnete Domäne: -
Funktionen: Generierung, Genehmigung, Überarbeitung,
Daten: Dienstplan
Systemcharakteristiken

Existierende Artefakte: ja...
System relationen: wo/wie implementiert?
Produkt: alle

6.6.5 Domäne Arbeitszeiterfassung

Informationsquelle: Orbis Handbuch
Funktion: Bei der Arbeitszeiterfassung werden die Abweichungen der Istzeiten von den geplanten Sollzeiten erfasst.
Domänengrenzen:
In:
Out:
Higher Level Domänen:
Lower Level Domänen: Datenbank, Systemverwaltung
Domänen auf gleichem Level: Arbeitsplatzplanung, Vorplanung, Generierung
Subdomänen: -
Übergeordnete Domäne: -
Funktionen: .Automatische oder Manuelle Erfassung, Online Prüfung
Daten:

Systemcharakteristiken

Existierende Artefakte: ja...
System relationen: wo/wie implementiert?
Produkt: alle

6.6.6 Domäne: Berechnungen und Auswertungen

Informationsquelle: Orbis Handbuch
Funktion: Nach dem Abschluss der Arbeitszeiterfassung werden alle für die Lohn- und Gehaltsabrechnung erforderlichen Daten zur Verfügung gestellt. Dies geschieht über das Berichtswesen und über eine Standardschnittstelle zum Abrechnungssystem. Die erfassten Arbeitszeiten der Mitarbeiter können außerdem statistisch ausgewertet werden
Domänengrenzen:
In:
Out:
Higher Level Domänen:
Lower Level Domänen: Datenbank
Domänen auf gleichem Level: -
Subdomänen: Auswertungen, Monatsabrechnung, Urlaubsplanung, Ausfallstatistik
Übergeordnete Domäne: -
Funktionen: Darstellung der tatsächlich geleisteten Arbeitsstunden, Statistiken, Abrechnungen, Übergabe der Daten zum Abrechnungssystem
Daten: Dienstplan
Systemcharakteristiken

Existierende Artefakte: ja...
System relationen: wo/wie implementiert?
Produkt: alle

6.6.7 Domäne: Berichtswesen

Informationsquelle: Orbis Handbuch
Funktion: Mit Hilfe des Berichtsgenerators, können unter schiedlichen Berichte visualisiert, gedruckt und in eine Vielzahl von Formaten (z.B. doc, rtf, xls, html) exportiert werden

Domänengrenzen:

Higher Level Domänen: -
Lower Level Domänen: Datenbank
Domänen auf gleichem Level: -
Subdomänen:
Übergeordnete Domäne: -
Funktionen: Ausgabe von Berichten
Daten: -

Systemcharakteristiken

Existierende Artefakte: ja...
System relationen: Schnittstelle zu Crystal Reports
Produkt: alle

6.6.8 Domäne: Datenbank

Informationsquelle: Orbis Handbuch
Funktion In der Datenbank werden alle Dienstpläne, Personaldaten und weiteren Einstellungen gespeichert

Domänengrenzen:

Higher Level Domänen: **-**
Lower Level Domänen:
Domänen auf gleichem Level: -
Subdomänen:
Übergeordnete Domäne: alle
Funktionen: Datenhaltung
Daten: alle

Systemcharakteristiken

Existierende Artefakte: ja...
System relationen: SQL Server, Oracle, Sybase
Produkt: alle

6.7 Priorisierung

Die Tasks wurden anhand der in 4.2.6 angegebenen Faktoren priorisiert, es ergibt sich folgendes Bild:

Priorisierungsfaktor	Personaldaten einpflegen	Dienste & Schichtfolgen definieren	**Dienstplan genehmigen**	**Dienstplan kontrollieren**	**Statistiken für Kontrolling erstellen**	Terminwünsche angeben	Dienstplan publizieren	Dienstplanung vornehmen	Ist-Daten eingeben	Monat abschließen	Statistiken für Kontrolling erstellen	Umplanen	Fehlzeitenplanung und Erfassung
Genügend Gemeinsamkeiten	1	2	0	0	0	3	0	3	3	1	1	2	0
Genügend (systematische) Variabilität	2	1	0	0	0	1	0	3	1	2	1	2	0
Wichtig für die Business Goals	1	2	1	1	0	2	1	3	3	1	2	2	1
Genügend Abdeckung durch das System (kein reiner User Task ohne das System zu verwenden)	2	3	0	0	1	1	1	3	3	2	2	3	0
Priorität	**6**	**8**	**1**	**1**	**1**	**7**	**2**	**12**	**10**	**6**	**6**	**9**	**1**

Die Tasks wurden vorpriorisiert, in einem Meeting wurde diese Priorisierung dann durchgegangen. Dabei wurden 8 Priorisierungen (Orange markiert) geändert. Es wurden 5 Tasks (in der letzten Reihe grün markiert) ausgewählt:

- „Dienste und Schichtfolgen definieren
- „Terminwünsche angeben“
- „Dienstplanung vornehmen“
- „Ist-Daten eingeben“
- “Umplanen”

6.8 Detaillierte Task Beschreibung

Von 4 der 5 priorisierten Tasks wurden in einem Interview detaillierte Taskbeschreibung erstellt. Die Interviewergebnisse sind im folgenden Beschrieben. Während der Interviews stellte sich heraus, dass Ist-Daten eingeben und Umplanen fast identische Aktivitäten sind, deshalb wurde für Umplanen keine detaillierte Task Beschreibung erstellt.

6.8.1 Dienste und Schichtfolgen definieren

Wer führt die Aufgabe durch?
Wer ist daran noch beteiligt?

In manchen Fällen die Personalabteilung

Meistens aber Stationsleitung (SL) oder Planungsgruppenleitung (PGL)
= Oberschwester

Meistens sehr gutes Domänenwissen
"finden sich sofort darin wieder"
aber geringe EDV-Kenntnisse

Wann wird die Aufgabe durchgeführt?

Dienste und Schichtfolgen werden in der Phase der Einführung aufgrund bestehender Dienste definiert.

Neue Schichtfolgen müssen erstellt werden bei

Neueinstellungen, die spezielle Dienstzeiten benötigen

Umstrukturierung

Zu welchem Zweck wird die Aufgabe durchgeführt?

Dienste müssen in Schichtfolgen (= regelmäßiger Turnus von Diensten) eingebunden werden

Mitarbeitern (MA) muss ein Schichtrhythmus zugeordnet werden.

Welche Voraussetzungen müssen erfüllt sein, damit die Aufgabe durchgeführt werden kann?

Für Schichtfolgen müssen Dienste und Organisationsstrukturen definiert sein.

Es sollten allgemein so wenig wie möglich verschiedene Dienste und Schichtfolgen definiert werden.

Stoßzeiten der Organisationseinheit (z.B. Mittagessen an Patienten austeilen) müssen berücksichtigt werden.

Welche zusätzlichen Hilfsmittel werden für die Durchführung der Aufgabe noch benötigt?

Historisch bedingte Daten (Dienste, Schichtfolgen) bei Neueinführung

Welche Schritte umfasst die Erfüllung der Aufgabe?

Organisationseinheit (OE) auswählen

Dienste für OE anlegen

Auf gleicher OE Schichtfolgen definieren

Wann gilt die Aufgabe als erfüllt?

Allen MA ist eine Schichtfolge zugeordnet (außer Aushilfskräften)

Welche Schwierigkeiten können bei der Durchführung der Aufgabe auftreten? Welche Fehler werden typischerweise gemacht?

Schichtfolgendialog: Die Schichtfolge muss pro Feld im Kalender (?) ausgewählt werden. Einfacher wäre eine Vorauswahl, die mit Drag and Drop in die Felder gezogen wird.

Offset des Schichtfolgenturnus wird durch Beginn in Kalenderwoche definiert und dann rückwärts projiziert. Besser: „Starte Turnus mit Dienst x (x=1..n bei n.stelligen Turnus)

Schichtfolge kann auf unterschiedlichen organisatorischen Ebenen definiert werden. Führt manchmal zu Verständnisproblemen.

Welche Daten sind für die Aufgabe notwendig? Sind diese im Beispieldatensatz enthalten?

Organisationsstruktur

MA

Sind enthalten.

Welche Funktionen werden benötigt?

(Organisationseditor)

Diensteditor

Schichtfolgeneditor

Kalenderansicht

Alternativ: Assistent

6.9 Terminwünsche und Urlaubszeiten angeben

Wer führt die Aufgabe durch?
Wer ist daran noch beteiligt?

SL oder PGL beantragt Terminwünsche und Urlaubszeiten

Pflegedienstleitung genehmigt

Wann wird die Aufgabe durchgeführt?

i.d.R. am Ende des Jahres fürs Folgejahr

BAT: eine bestimmter Anteil der Urlaubszeiten muss bis zum 31.12. des Vorjahres festgelegt sein.

Wird bei Schülern sehr häufig gemacht wegen Unterrichtszeiten und wechselnden Einsatz über Organisationseinheiten

Zu welchem Zweck wird die Aufgabe durchgeführt?

Wird zur langfristigen Vorplanung der Dienste und Arbeitsplätze eingesetzt.

Welche Voraussetzungen müssen erfüllt sein, damit die Aufgabe durchgeführt werden kann?

Dienste und Schichtfolgen müssen definiert sein

Welche zusätzlichen Hilfsmittel werden für die Durchführung der Aufgabe noch benötigt?

Urlaubsanträge, -listen

Terminwünsche der MA

Welche Schritte umfasst die Erfüllung der Aufgabe?

- Zeitraum auswählen für den vorgeplant wird
- OE auswählen
- Personen auswählen
- Art des Eintrags bestimmen (Terminwunsch, Urlaubswunsch)
- Im Kalender zuordnen
- Genehmigung des vorläufigen Eintrags durch PDL

Wann gilt die Aufgabe als erfüllt?

Wird kontinuierlich durchgeführt

Kann nur für Zeiträume durchgeführt werden, für die kein abgeschlossener Dienstplan vorliegt.

Welche Schwierigkeiten können bei der Durchführung der Aufgabe auftreten? Welche Fehler werden typischerweise gemacht?

Welche Daten sind für die Aufgabe notwendig? Sind diese im Beispieldatensatz enthalten?

Enthalten

Welche Funktionen werden benötigt?

Dialog Vorplanung

Evtl Gantt-Chart

Personalfilter

6.9.1 Dienste und Schichtfolgen definieren

Wer führt die Aufgabe durch?
Wer ist daran noch beteiligt?

In manchen Fällen die Personalabteilung

Meistens aber Stationsleitung (SL) oder Planungsgruppenleitung (PGL)
= Oberschwester

Meistens sehr gutes Domänenwissen
"finden sich sofort darin wieder"
aber geringe EDV-Kenntnisse

Wann wird die Aufgabe durchgeführt?

Für jeden Monat ca. 6-8 Wochen im voraus

Zu welchem Zweck wird die Aufgabe durchgeführt?

Mittelfristige und mehr feingranulare Planung der Dienstzeiten (gegenüber langfristiger Vorplanung)

Außer Freizeitausgleich werden keine Fehlzeiten erfasst (gegenüber Ist-Daten)

Welche Voraussetzungen müssen erfüllt sein, damit die Aufgabe durchgeführt werden kann?

Eine Vorplanung ist empfehlenswert aber nicht zwingend für den einzelnen Monat

Allerdings ist die Vorplanung zwingend für das ganze Jahr (???)

Welche zusätzlichen Hilfsmittel werden für die Durchführung der Aufgabe noch benötigt?

- Urlaubsanträge
- Anträge auf Freizeitausgleich
- Dienstwünsche

Welche Schritte umfasst die Erfüllung der Aufgabe?

- Festlegung der Sollbesetzung
- Setzen der variablen (=weichen) und der unumstößlichen (=harten) Randbedingungen
- Generierung des Dienstplans nach
  - Sollbesetzung
  - Rahmendienstplan

- o Vorrang der Kriterien festlegen
- o Generator anwerfen
- o Am Monatsende: abschließen

Wann gilt die Aufgabe als erfüllt?

Das Optimum ist erreicht, wenn die Besetzung gewährleistet ist

Alternativen:
- – Manuelle Erstellung
- – Erstellung gemäß Rahmendienstplan
- – Generator

Welche Schwierigkeiten können bei der Durchführung der Aufgabe auftreten? Welche Fehler werden typischerweise gemacht?

- o Guter Rahmendienstplan ist Voraussetzung für effizienten Generatorlauf
- o Schwierigkeit: Sortierung durch Drag and Drop
- o Verständnisschwierigkeit mit der Unterscheidung von Vorplanung, Dienstplanung und Arbeitszeiterfassung (Masken sehen zum Teil sehr ähnlich aus)

Welche Daten sind für die Aufgabe notwendig? Sind diese im Beispieldatensatz enthalten?

Sind vorhanden

Welche Funktionen werden benötigt?

- o Sollbesetzungsbeschreibung
- o Dienstplantabelle

6.9.2 Ist-Daten eingeben

Wer führt die Aufgabe durch?
Wer ist daran noch beteiligt?

Stationsleitung (SL) oder Planungsgruppenleitung (PGL)

Wann wird die Aufgabe durchgeführt?

Tagesaktuell bei Abweichungen vom Dienstplan

Zu welchem Zweck wird die Aufgabe durchgeführt?

Erfassen von kurz- und mittelfristigen Abweichungen vom Dienstplan

Berechnen von Mehrarbeitszuschlägen und des Uerlaubskontos

Welche Voraussetzungen müssen erfüllt sein, damit die Aufgabe durchgeführt werden kann?

Ermittlung der *unständigen Bezüge*

Welche zusätzlichen Hilfsmittel werden für die Durchführung der Aufgabe noch benötigt?

- Grundsätzlich keine Genehmigung erforderlich
- Angeschlossener, genehmigter Dienstplan

Welche Schritte umfasst die Erfüllung der Aufgabe?

- Arbeitszeiterfassung, Personalfilter
- Maske zum einfügen der Abweichungen
- Am Monatsende: Anforderung des Abschlusses
  - Abrechnung
  - Genehmigungsvorgang

Wann gilt die Aufgabe als erfüllt?

Wenn der Monat abgeschlossen und genehmigt ist (???)

Welche Schwierigkeiten können bei der Durchführung der Aufgabe auftreten? Welche Fehler werden typischerweise gemacht?

Verständnis: Fehlzeit bezieht sich auf die Station nicht auf den MA (???)

Welche Daten sind für die Aufgabe notwendig? Sind diese im Beispieldatensatz enthalten?

vorhanden

Welche Funktionen werden benötigt?

Fenster Zeiterfassung

Document Information

Title:	UseLine: Process Description and Case-Study
Date:	Juni 2004
Report:	IESE-074.04/E
Status:	Final
Distribution:	Public