2008

End User Development of Information Artefacts: A Design Challenge for Enterprise Systems

Michael Spahn
University of Siegen, michael.spahn@sap.com

Christian Dorner
University of Siegen, christian.doerner@uni-siegen.de

Volker Wulf
Fraunhofer-Institut für Angewandte Informationstechnik FIT and Universität Siegen, volker.wulf@uni-siegen.de

END USER DEVELOPMENT OF INFORMATION ARTEFACTS: A DESIGN CHALLENGE FOR ENTERPRISE SYSTEMS

Spahn, Michael, SAP Research, SAP AG, Bleichstr. 8, 64283 Darmstadt, Germany, michael.spahn@sap.com

Dörner, Christian, University of Siegen, Hölderlinstr. 3, 57076 Siegen, Germany, christian.doerner@uni-siegen.de

Wulf, Volker, University of Siegen, Hölderlinstr. 3, 57076 Siegen, Germany, volker.wulf@uni-siegen.de

Abstract

Delivering the right information at the right time to the right persons is one of the most important requirements of today's business world. Business users have to deal with changing information needs, but suffer from their inability of adapting the used information systems to support their changing needs. Enterprise Resource Planning (ERP) systems, which act as the main source of information in many enterprises, are still too complex to be adapted to current information needs and working practises by business users themselves. Business users face considerable challenges when trying to use tools to create individual information artefacts to support and facilitate their working tasks. Targeted at improving the design of such tools, we started our design process with a qualitative empirical study, presented in this paper. The study discloses main problem areas business users face and identifies design challenges from an End User Development (EUD) point of view. The study was conducted as a combination of semi-structured interviews and a participatory design workshop, which were analysed to derive design constraints for EUD tools.

Keywords: Information Systems Development, Empirical Studies, End User Development, Design

1 INTRODUCTION

One of the most important challenges of industrial software engineering is the development of flexible standard software products, matching the needs of ideally all customers. As it is impossible to completely capture all requirements related to a software system during design time, adapting software during use time is an immanent problem (Henderson & Kyng, 1991). Many research papers dealt with the subject of flexible systems and most of them concluded that the adaptation of software is necessary during its life cycle, e.g. (Fischer et al., 2004). Requirements concerning information systems (IS) of an organization change frequently (Henderson & Kyng, 1991). Due to a lack of resources and expertise, especially small and medium sized enterprises (SMEs) suffer from their inability to customize used enterprise software to their needs. They are forced to adapt themselves to the possibilities offered by the used enterprise software or to delegate adaptations to IT professionals, resulting in long and costly adaptation processes (Brehm et al., 2001; Markus & Tanis, 2000). Developing software which is supporting end users in performing adaptations on their own, improves the achievable efficiency and effectiveness of SMEs by increasing the fit between the offered functionality of the software and the needed functionality by the users (Roth & Scheidl, 2006). As organizational and technological development are closely correlated, the inability of users to adapt the technical systems used, is limiting the organizational development possibilities of organizations as well (Wulf & Rohde, 1995). As a consequence the ability of organizations to innovate and to differentiate themselves to gain unique competitive advantages is limited. The inherent challenge is to reduce the expertise tension, existing in a two-dimensional continuum of job-related domain knowledge and system related development knowledge (Beringer, 2004). Enterprise resource planning (ERP) systems are building the backbone of modern enterprises, but little research has been done addressing a reduction of the expertise tension in this domain. Traditional IS literature is addressing problems of ERP adoption, implementation, and use on a higher abstraction level from a managerial adoption and diffusion perspective. In contrast to this, we are approaching this challenge from a user-centred End User Development (EUD) perspective. EUD can be defined as *"[…] a set of methods, techniques, and tools that allow users of software systems, who are acting as non-professional software developers, at some point to create or modify a software artefact"* (Lieberman et al., 2006). As a detailed discussion of EUD approaches would go far beyond the scope of this paper, we refer to Spahn et al. (2008), providing an overview of EUD approaches and application possibilities.

The problem area focused in this paper is the creation and adaptation of information artefacts. A generally claimed main value of ERP systems is their ability to act as a central data repository that can be mined by users in a self-service manner. Users of ERP systems use the ERP system to access important business data directly via the provided user interface, or they extract data using queries and reports. In many cases just accessing predefined sets of data is not sufficient for the users. To support their working tasks, users need to create more specific queries and building tools consuming and processing data in a way especially tailored to their needs. Many users construct spreadsheet solutions to facilitate their operational tasks. Some advanced users even use simple databases like Microsoft Access to create individual tools with. All these activities of creating information artefacts can be seen as a kind of EUD activity. Especially technically inexperienced users face considerable challenges due to the complexity of the ERP data model or the complexity of available development tools. Traditional programming and design environments are focusing on providing very sophisticated, and consequently complex, solutions, overlooking needs for simple, easy to apply and use solutions. To lower the entrance barrier for users to develop information artefacts, simplified EUD tools need to be developed. As EUD is not an unproblematic, self-evident approach, which can simply be "applied", existing problems in the addressed domain have to be investigated in the context they occur, and specific EUD methods and tools have to be developed to tackle the identified problems. In this paper we present the results of empirical studies, conducted within German SMEs as a preliminary step of creating an EUD tool for the lightweight composition of information artefacts. We investigate work practices of users and the problems they face to get insights on the main problem areas that have to be addressed by according EUD tools (cf. chapter 3). We analyze the results on a meta-level to identify existing composition processes of information artefacts that have to be supported and the relevant building blocks used. As identified building blocks need to be composed in an intuitive and lightweight way within EUD tools, we conducted a participatory design workshop to get insights on how users intuitively design information artefacts using a simple design paradigm like boxes and wires (cf. chapter 4). The obtained results are summarized in chapter 5 and enriched with first conclusions on what aspects could be addressed to achieve complexity reduction in appropriate EUD tools. We think that the possibilities of using EUD tools, like very lightweight composition approaches, in complex enterprise system environments are currently under-researched in both computer science and information systems and thus require further investigation. With regard to the addressed research field the paper contributes by providing a user-oriented view highlighting existing problem areas in information artefact creation. Additionally, the paper provides first suggestions on how to address the identified problems. These suggestions can help to make informed design decisions for developing EUD tools.

2 THE END USER DEVELOPMENT POINT OF VIEW

The implementation of data warehouses and Business Intelligence (BI) tools are often proclaimed to build a suitable base with regard to information self-service and end user computing. Pinnington et al. (2007) present a field study of challenges end user computing faces in business intelligence enabled enterprise system environments. They critically discuss the generally claimed main value of enterprise systems as a central data repository that can be mined by users using self-service business intelligence functionality. They argue that the sole implementation of business intelligence technology is not sufficient to expose data in a manner suitable for end user computing and suggest that some aspects cannot be ameliorated but must simply be endured. The question of how to improve this situation is left open.

This raises the general question of how to design flexible information systems and tools suitable for end users. Fischer et al. (2004) propose Meta Design as an approach targeted at preventing users to be forced to be just consumers of closed solutions. Users should be supported to be creative and collaboratively develop evolving systems as co-developers. Therefore Fisher et al. suggest the development of "underdesigned" systems, which are not complete solutions, but tools and environments allowing users to build complete solutions with. As a consequence users have to be provided with suitable building blocks and a design environment to combine these building blocks to a desired solution. The meta-design approach is supported by Stiemerling et al. (1997), as they identify tailorability as one of the main challenges in the design of user interfaces and interactive systems suitable for end users. They show how tailorability can be combined with evolutionary and participative software engineering to enable flexibility in specifically such problem areas really of concern for users. As a consequence it is very important to understand users to provide the right building blocks and composition possibilities.

An important prerequisite for creating EUD tools is to identify the critical problem areas users have to deal with, and what would be an intuitive solution strategy for users which should be supported by especially coined EUD tools. Qualitative empirical studies concerning user design behaviour were used by Pane et al. (2001) in preparation for designing an EUD programming environment. They analyzed textual descriptions of a desired program behaviour made by non-programmers to discover underlying abstract programming models implied by these descriptions. The results were then used to create a simple programming language, especially designed with regard to the identified abstract programming models, intuitively used by the users. In their later work, Pane and Myers (2006) generalized this approach and developed a Natural Programming design process, which treats usability as a first-class objective in programming system design by following these steps: (i) identify the target audience and the domain, (ii) understand the target audience, (iii) design the new system based on this information, and (iv) evaluate the system to measure its success.

In preparation of such a "natural" design process we conducted a semi-structured interview series, targeted at identifying the main problem areas of users which have to be addressed. The results were complemented by conducting a participatory design workshop, targeted at identifying “natural” user design behaviour, which is related to the creation of information artefacts that are based on an ERP system as a central data repository. The results are presented in the following chapters.

3 EMPIRICAL STUDY – PART I: PROBLEM AREAS

To get a better understanding of the targeted domain and users, we examined users in German SMEs, who are using SAP ERP systems as their main source of information. Users engaged in analytic tasks were chosen as subject of interest as they are already confronted with the creation of information artefacts and thus are capable of giving insight in current work practices and existing pain points. The presented results can be used to identify main problem areas that have to be considerably simplified in order to be able to create lightweight EUD tools for the creation of information artefacts.

3.1 Methodology, Setting, and Design of Evaluation

The series of semi-structured interviews presented in this section is based on qualitative research methods (Kvale, 1996) and was conducted in two steps. The first step aimed at getting a general overview of the five German SME partners involved. Semi-structured face-to-face interviews were conducted in an exploratory way with CEOs and employees in managerial positions to get an insight in formal organizational structures and lines of action. In a second step we extended the set of interviewees to non-managerial employees doing operative work, to get a deeper insight into operational tasks and work practices. All interviews were conducted at the site of the respective company in the familiar surrounding of the work place to limit de-contextualization effects. The companies addressed were two smaller handicraft companies (12 and 26 employees), two mid-sized companies from production industry (120 and 155 employees) and one larger software vendor (500 employees). The diversity in size and business purposes of the addressed companies allowed an exemplary view on a cross section of SMEs. 14 employees of these companies have been interviewed. The interview duration was between 60 and 120 minutes each. All interviews were recorded and transcribed in an abbreviated manner. The analysis of the first interview series focused on common aspects to create a general understanding of the companies, their organization, functional areas and problem domains. The analysis of the second interview sessions focused on more specific aspects of the domain of analytic tasks and the creation of information artefacts. The interview results were structured with regard to the focused aspects to ex-post generated categories (Pandit, 1996). The presented extract of the obtained set of information in the following section was chosen in a way to present the main issues identified.

3.2 Results

IT infrastructure. All companies use an SAP enterprise resource planning (ERP) system to support their operative tasks and Microsoft Office products to support communication tasks (e-mail, text processing) and analysis (spreadsheet application). Spreadsheets are used for consolidating data, analyzing data and delivering analytic results, thus making spreadsheet applications the main “composition environment” used by users to create information artefacts with. The “building blocks” users compose their information artefacts from are predefined queries exposed by the ERP system. All companies avoid using individually developed software and stay as close to the standard of the used software systems as possible to reduce costs and efforts for adaptations and maintenance. A manager of an IT department told us: *“We are not as incapable as you may think. Of course we could develop some small Access databases or Excel macros to support users, but we try to avoid this to reduce the amount of solutions that have to be maintained. [...] Currently we run the business mainly with just SAP and Excel.”*[1]. As the IT departments avoid developing individual solutions, they leave users in the situation to

[1] The statements of the interviewees have been freely translated from German to English by the authors.

arrange themselves with the standard functionality offered by the used systems. Maintenance and technical support of the systems is outsourced and only accessible for the IT departments. The IT departments themselves only offer limited support, e.g. by helping with the development of individual queries – as far as sufficient system knowledge exists.

Repetitiveness of tasks. Analytic or data-centric tasks executed by users are the main reason for users to create information artefacts to support them in these tasks. Such tasks can be distinguished of being repetitive or nonrecurring. The creation of a daily sales report is an example of a repetitive task, whereas analyzing cost structures for a single project is a very special and non recurring analytic task. For the reason of efficiency, repetitive tasks should be automated, but as users are unable to create information artefacts extracting all needed data from the ERP system automatically, many repetitive tasks remain manual work. An IT manager said: *"Our ERP system accumulates the turnover values on a per month basis. To get the daily turnover values we access this value every morning at the same time, and manually write it to an Excel file. By subtracting the recorded accumulated values we calculate the daily turnover."*

Needed flexibility. Tasks like determining the sold quantities of the best selling products can be done using a static report. In contrast to this, analyzing the origin of differences in the planned and actual sold products requires exploring a lot of data and creating ad-hoc defined spreadsheets for analysis. The user does not know exactly prior to the analysis what information will be needed during the analysis to be able to answer the question or solve a problem. In this case the analysis process can be characterized as being exploratory. One user told us: "*Sometimes I click my way through the system and check many different things, just to get a hint on what might have caused the deviation.*" As the information demand cannot be anticipated in advance users need to be able to browse the available information space and create information artefacts in an ad-hoc manner by themselves.

Consolidation of data. Consolidating data from different ERP modules, designed for different functional areas of an enterprise, to a single view, can be cumbersome as users need to collect the data from different ERP queries, export them to a spreadsheet and then do the consolidation manually. An example is given by a user responsible for checking credit limits of customers: *"I sometimes have the problem, that I have to access four or five things in order, to get the things I need [...] for example the annually credit limit check of our customers. Therefore I need master data and data from SD* [SAP ERP module for sales and distribution], *some data from financial accounting and I don't get them by pressing a button."* This user would have much appreciated a single view on the data relevant for his decision, but the ERP system does not offer mechanisms usable for him to create such a view.

Including data from different sources. The need for combining ERP system internal data with system external data exists. An example for this is the wish of a user responsible for global procurement to be able to automatically include current currency exchange rates from a web site into his calculations. Another example is the inclusion of manually codified parameters from a spreadsheet to improve forecast accuracy. For instance, the range of stock calculated by the ERP system does not reflect customer orders that have not yet been placed, but are likely to arise soon because of seasonal effects or the launch of new products. As a consequence stock quantities available to promise and material to order from suppliers need to be recalculated in order to be more realistic. This is done by extracting quantities for goods on stock, actual orders and reservations and combining these with additional information based on experience and estimations from a spreadsheet. The responsible user told us: *"To plan order quantities I divide the year in certain time periods by experience. [...] I am able to estimate when the demand will be high and low, which colours will sell best and if product launches will cause additional demand."* To support this task, the user created a complex spreadsheet-based planning model which is manually updated by extracted facts from the ERP system.

Work time spent. The time users spend on analytic or data-centric tasks, differs with the type of their everyday work. As an example, the manager of a HR department estimated to spend 15% of the overall work time on analytic tasks, an assistant of a sales manager estimated to spend over 50% of the overall work time on analytic tasks, and an employee responsible for procurement even estimated to spend 67% of the overall work time on analytic tasks. In many cases a huge amount of this time is spent on finding, accessing, extracting, consolidating and post-processing of data. The ability to create specialized information artefacts, usable as tools to automate extraction and processing of data could reduce the manual efforts, but users do not know how to develop such solutions. As a consequence, providing EUD tools supporting users in the process of fast and easy creation of information artefacts could save a lot of working time.

Finding data. With regard to efficiency, finding and extracting the needed data from the ERP system is one of the most criticized issues. *"Sometimes the problem is, that I simply do not know where certain information is stored in the system. I can not find it and therefore can not combine it with other data. [...] There's this restriction that you are bound to a limited set of information. [...] The knowledge, which data is stored where, should be easily accessible."* Another user reported a common workaround used to find data within the system: *"If the data is master data I know that in most cases I can jump into the field, press F1 and read from the technical help where the data is stored. But there are things, you can't find this way and even Mr. X from IT can't find them."* An employee of the IT department told us: *"I had to include a certain deletion flag in a query. I didn't know where to find it and so I searched for it. To use the search by name function I had to guess the field name of the deletion flag. [...] I finally found deletion flags... about 70 deletion flags... and didn't have the slightest idea which one was the right one. To find out, I tried. Two windows. One to set and unset the deletion flag. Another to run a query against one of the found flags to see if the results change if I change the flag value."*

Need for system knowledge. As a consequence, knowing what data exists, finding this data and using it e.g. in queries strongly depends on the individual system knowledge of each user. The help system of the ERP system is not rated to be helpful in these cases. In reply to a question about the usage of the ERP help system a user told us: "*I don't want to say anything about that. [...] there is probably nothing in it that is comprehensible. The system is probably only written from experts to experts.*" The knowledge about which data exists and where it is stored is not explicitly documented in the observed companies and only available as distributed implicit knowledge which can only be discovered by asking colleagues or the IT department, which is limiting the possibilities of the individual business user. A user describes: *"Sometimes I create my own queries in SAP, although in effect I don't know the needed basics. I play around and ask IT if needed. [...] I know how to select some master data and link it with some turnover values. That's it. [...] I'm sure I'm using less than three percent of the things I could use. One could have such more possibilities, if even one would know that they exist."* Due to the fact that IT departments support a lot of different users from many functional areas, the biggest amount of knowledge about what business data is stored where is available in the IT departments. This knowledge is not documented and shared as common knowledge among the employees. The employee of an IT department told us: *"We know SD and MM* [SAP ERP module names] *very well. There are some things everybody knows, who is getting data from SD, like Mrs. X. She simply knows that sales order header data is stored in table 'VBAK'. We do not document such things."* As the knowledge of the IT departments mainly results from already solved problems, the solution of new problems can be cumbersome even for the IT department: *"We do not know everything. HR asked us to do some queries, but we don't know anything about the HR tables. We have to talk with HR, so that they can show us which data they mean. Perhaps they already know some tables. The rest, we have to figure out on ourselves."*

3.3 Conclusions

As our studies revealed, it is common for business users to create individual information artefacts supporting them in their operative tasks – or at least try to develop such information artefacts. Abstracting from details, we analyzed the observed creation processes of information artefacts on a meta-level with regard to the used building blocks and composition environments. We identified two levels or layers of composition the users acted on: (i) composition of queries using the embedded query designer of the ERP system as composition environment and tables of the ERP system as building blocks, and (ii) composition of task-specific information artefacts using a spreadsheet application as composition environment and queries as building blocks. Within the first layer users create building blocks to be used in the second layer, where building blocks are composed to the desired information artefacts. Each layer requires different skills of users. The creation of queries confronts users to a much greater extent with the complex data model of the ERP system as the consumption of already built queries. As users face frequently changing information needs, adapting artefacts at both layers might be needed in order to create the desired results. Building queries requires good system knowledge, and adaptations within the first layer expose inexperienced users to a set of problems not solvable by themselves. There are no tools supporting users in searching and composing needed data from tables in a way rated to be appropriate for inexperienced business users. Users think in business terms and know what data is stored inside the ERP system, but their business knowledge does not allow them to locate the data within the data storage of the ERP system. In some limited cases users are able to locate data using the help system of the ERP system, but the existing mechanism does not seem to assist users in a suitable way. Knowledge about where to find data and how to combine data is only available as implicit knowledge of experienced users and is not explicitly documented and shared among users (e.g. using simple knowledge management systems like wikis). The collaboration between users of different skill levels and the active transfer of knowledge is not addressed by especially designed tools dedicated to end user support and collaboration.

Another observed aspect is the low degree of the realized automation of repetitive tasks. The building blocks used within the second layer of composition do not establish a permanent connection between the ERP system and the information artefact, making it impossible to refresh the data without manually repeating the process of exporting data and composing it. Thus, users are not able to create any information artefact offering interaction on live data of the ERP system. Least of all, an automated integration of external information sources, like data available on the web, is not feasible.

4 EMPIRICAL STUDY PART II: DESIGN BEHAVIOUR

Analyzing the interview results on a meta-level, two layers of information artefact composition, with different kinds of used building blocks, could be identified. A prerequisite for building lightweight EUD design environments for composing such building blocks in a way being intuitive for users is the identification of a simple and easy to use design paradigm for composition. According to our aim of collecting basic insights on which aspects need to be addressed in order to create a very lightweight composition approach, we investigated the effects of confronting users with a simple design paradigm like boxes and wires (e.g. used by Mashup tools like Yahoo! Pipes). The boxes and wires model was chosen because it enables visual modelling while using a minimum of different design entities. To get first insights in how users interact with such a simplistic modelling approach, we complemented our interview study with a participatory design workshop, putting users in the role of designers. By leaving the boxes and wires model underspecified in a way that no concrete instructions were given to the users on how to formally specify the meaning of the used design elements, we were able to observe users' natural design behaviour, while they were intuitively using the design elements. The design behaviour and the created artefact were analyzed to get a better understanding of how users intuitively approach this design task.

4.1 Methodology, Setting and Design of Evaluation

A participatory design workshop (PDW) requires users to collaborate actively in a solution-oriented design process and puts them in the role of co-designers, which can be seen as a type of user-centred prototyping (Muller, 2003). By giving the participants a very limited set of design elements with underspecified semantics, they were able to use the design elements with self-defined semantics, which seem "natural" to them. Environments that are more "natural" are closer to the way user think about their tasks, enabling them ideally to formally express their ideas in the same way as they think about them (Pane & Myers, 2006; Pane et al., 2001).

The PDW was held with three employees of one mid-sized industry partner (approximately 150 employees), who already acted as interviewees in the conducted interview series. The specific company and employees were chosen as workshop participants, as they had an excellent mixture of motivation and experience in information artefact creation to be able to even tackle more complex design challenges in the PDW. The work of two employees (assistant of sales manager and procurement manager) consists to a high degree of analytic tasks and the creation of information artefacts. The work of the third employee (manager of IT department) includes the support of users in accessing data for analytical tasks and information artefact creation. As designing information artefacts, by using an underspecified semantic might confuse the users, two software engineers participated in the workshop that could be asked for assistance. The workshop was coordinated by two moderators and documented by two observers. The observers took notes during the workshop and videotaped the actions within the design space. They also used a digital voice recorder to record the whole workshop and took pictures of all created artefacts and notes.

At the beginning of the workshop an analytical scenario taken from the actual working context of the users was collaboratively specified, which should exemplarily be solved in the design process. After the scenario was set, the boxes and wires design paradigm to use was introduced to the users. Boxes could be used to represent data or operations. A box could be logically connected to another box by connecting the boxes with a line. The connection could be done by connecting the output port of one box with the input port of another box, which defines the information flow between the boxes. Which data or operation is represented by a box should be annotated inside of the box. The design space was represented by a big blank white paper. To include boxes, small white slips of paper, which were preprinted as boxes containing input and output ports, could be used. To design their solution, users could write in the boxes to specify their content or function, arrange the boxes on the design space, and connect boxes with lines. Coloured markers could be used to separate different semantic meanings of lines and annotations.

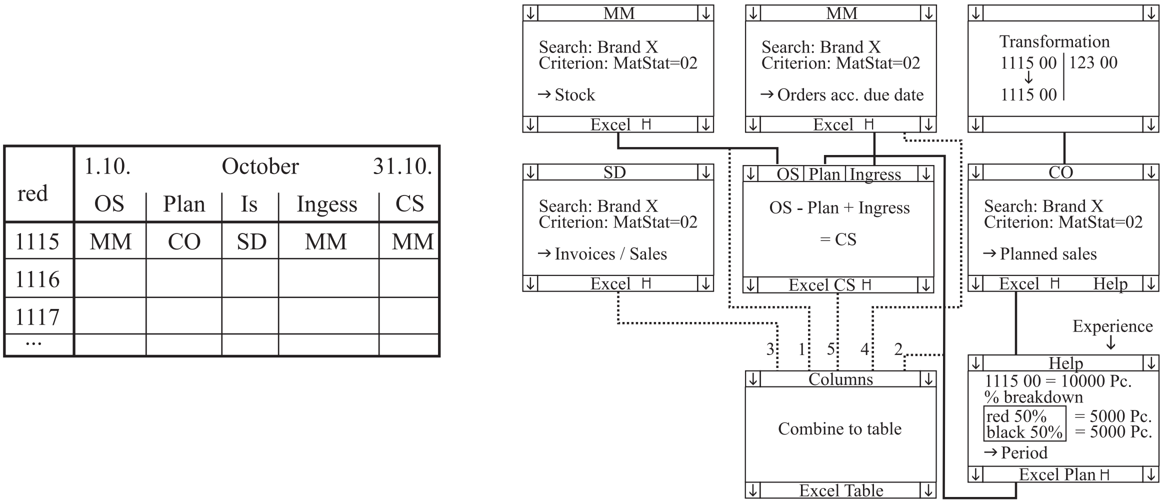
The evaluation of the workshop focused especially on deriving insights on the "natural" design behaviour of users and to which extent they are able to use the simplistic design paradigm to specify an information artefact tackling a practical problem. At this, the underspecified semantics of design entities additionally gave first vague insights on how a design environment could be organized with regard to (i) how users specify the needed data (i.e. needed building blocks), (ii) what kind of data representation is intuitively assumed, (iii) which kind of operations are performed on data, and (iv) how operations are formally expressed. The design artefact collaboratively created by the users was used as a subject of analysis as well as the video material and notes taken from the audio material, which were aligned with the notes of the two observers, to complete the overall picture.

4.2 Results

To have a clear vision of the desired results, the users first prepared a sketch of the desired result as a table. The columns of the table were annotated with the modules of the SAP ERP system the users use and from which the data of individual columns should be taken from. The left part of Figure 1 shows a reconstructed version of the table sketched by the users. The design artefact used to specify the desired outcome is depicted in a simplified way in the right part of Figure 1. As a detailed description of the solved problem and the design process would go far beyond the scope of this paper, we focus on describing the main design behaviours observed.

The actual design process was started by introducing a box for every SAP module data needed to be extracted from. The input area of the boxes were annotated with the according SAP module, while the centre of the boxes were annotated with selection criteria which have to be applied in order to get the desired subset of relevant data. Selection criteria were for example the brand of goods or a desired state of the material. At the bottom of the box a short explanation of the data of interest was added.

The output port of a box was labelled with the desired output format (e.g. "Excel" and the number of output columns). Special boxes were introduced to specify functions for transformation or calculation, like extracting a part of the product number or calculating the current stock of goods. Transformations and calculations were either specified by giving a formula or by giving an example of the desired outcome. The data flow was specified by connecting output ports of boxes with input ports of other boxes.



red	1.10.	October			31.10.
	OS	Plan	Is	Ingess	CS
1115	MM	CO	SD	MM	MM
1116					
1117					
...					

Figure 1. Sketch of result desired by workshop participants (left). Solution design created by workshop participants (right).

A specially coloured box labelled with "experience" was introduced to express that the data was "stored" in the head of one of the users and is manually entered into a spreadsheet to make it technically available. The output ports of all boxes, which should contribute columns to the desired result table, were connected to the input port of a box representing the result. The input port of the result box was labelled with "columns". The connecting lines ending in this input port, were especially coloured and annotated with numbers, representing the number of the table column the input data should be transferred to. The centre of the result box was labelled with the desired functionality "combine in table".

4.3 Conclusions

The artefact designed by the users is not formal and precise enough to be transformed to an executable artefact, but nevertheless it gives insight in how design elements are intuitively used by users and what semantics seem to be "natural" to them. Having to deal with data as building blocks (boxes), users think of data as being a table, organized in rows and columns. The concept of specifying main entities as boxes within the design space was understood very fast and without any problems by the users. As the users had no concrete instructions (e.g. a formal syntax) on how to specify the exact set of data, which should be represented by a box, this was a topic of discussion between the users at the beginning of the design process. When discussing about the data in a business context the users used business terms to describe the meaning, but when discussing the specification of the data within the box, the users thought about ways to transform that meaning into a form "understandable by the machine" using their system knowledge. The users decided to specify the data by giving a short description on how they would access the data in their used ERP system. They specified the module containing the data, the key entity for search and further filters to apply in order to get the desired subset of data. As they thought of data being a table, they finally specified which columns of the table they need for further processing. This transformation process of business terms to system knowledge is inherently difficult especially for inexperienced users with little system knowledge. A "natural" EUD tool should support users in specifying data in a more business-oriented, conceptual way, avoiding the need of mentally transforming business knowledge to system knowledge. Remarkable is the fact that users exactly thought about what columns they extract from the tabular data, but never thought about specifying which rows of that tabular data, taken from different boxes, correspond to each other. Thinking in terms of relational databases the users did not think about foreign key relationships and joins at all and left such mechanisms to a certain “magic” automatically realized by the underlying system. It seemed natural to them that the system knows which rows belong together. A high degree of creativity was observed as users thought about how to transform data and do calculations with it. They decided to use special boxes representing these operations and specified the operations by giving a formula or an example of the needed transformations. While specifying a formula is close to a “machine understandable” format, giving an example demonstrated that users did not know how to encode the needed transformations into a more “machine understandable” format. In contrast to this, the wiring of the boxes to create the overall solution was done by the users without any serious problems. Overall, the usage of boxes and wires as design paradigm did not lead to severe problems, whereas problems were observed in situations when users tried to transform business knowledge into a “machine understandable format” using system knowledge.

5 SUMMARY AND OVERALL CONCLUSIONS

After having identified main problem areas, users face in analytic tasks and information artefact creation in section 3, the results of the participatory design workshop can be used as indicators of what can be seen as intuitive and “natural" for the observed users with regard to solution design. Combining this information allows concluding some basic implications of how to design tools for supporting users in creating information artefacts in a way tailored to their needs and their natural design behaviour. The results may not provide general validity, but are valid with regard to the observed partner companies.

As the results of the first part of the empirical study revealed, users are regularly confronted with changing information demands and therefore need to find, extract, manipulate and consolidate data from the ERP system. To facilitate their tasks, users create information artefacts, but these are not able to automate data delivery from the ERP system to refresh data for recurring task execution, indicating the need for an improved inclusion mechanism of live data in the information artefacts. A spreadsheet application was used by the users as composition environment for individual information artefacts. At this, two layers of composition activities could be distinguished, requiring different skills and degrees of system knowledge of the users and should be addressed by supporting artefact creation on different levels of complexity. Especially the creation of new building blocks (queries) confronts users with the complex data storage structures of the ERP system, which should be avoided by offering a simplified and more business-oriented data model. Finding relevant data in the data structures was rated to be difficult and the help system rated to be not understandable, indicating the need for better search capabilities and an improved documentation. The fact, that collected knowledge of advanced users is only implicitly available as personal knowledge, suggests, that knowledge should be expatiated to be accessible by inexperienced users. Additionally the collaborative development of information artefacts, involving inexperienced and advanced users should be considered to be supported actively.

The design methods offered by a composition environment should consider the “natural” design behaviour of the addressed user group. As the participatory design workshop revealed, users are able to use a simple visual design model for the composition of information artefacts, like boxes and wires. The semantics of the design elements should consider that users intuitively think about data as being organized as a table and offer appropriate building blocks and operations. Specification of data should be close to the business terms used by users, avoiding the need of transforming business knowledge to system knowledge to be able to specify data. Technical details of data composition, like specification of joins using foreign key relationships should be hidden from users. The specification of individual operations should be able in a way, which is well understood by users (e.g. by using formulas).

Compared to the currently established design environments, EUD design environments offering a very simplistic and lightweight approach could lower the entrance barriers for usage and through this enable even inexperienced users to design information artefacts. Through this more employees are able to begin adapting their working environment to their needs and thus limit the gap between needed and offered functionality of the used systems. We suggest that EUD environments targeted at supporting users in creating information artefacts should radically improve simplicity, even if simplicity implies lack of flexibility or limiting application to only selected areas of business. We suggest that a significant simplification (in the sense of EUD) can be achieved by focusing on complexity reduction with regard to the following issues: (i) data abstraction, (ii) searching and browsing data, and (iii) orchestrating data. We want to focus on those three aspects briefly.

Data abstraction. As the first part of the empirical study revealed, users have problems finding and understanding data in the technically oriented data storage structures of ERP systems, and problems composing data using technically oriented query tools. As the second part of the empirical study revealed, users intuitively specify data elements by using concepts known from their work context, and are able to build conceptual solution models using such elements. To close the gap between the technically oriented data storage of ERP systems and the business-oriented understanding of users, EUD environments should abstract data from its technical storage and present the information space in a business-oriented conceptual way, exposing elements related to the work context of business users. Organizing data in such a way simplifies the searching, the browsing, and the understanding of data.

Searching and browsing data. Just abstracting technical data to conceptual data elements improves understandability, but does not fully answer the question of how to find the data. The information space of ERP systems might expose several tens of thousands of elements. Just offering search capabilities like keyword search causes the same problem as described by the interviewee in section 3.2, who was thinking about what keyword to use to find a special deletion flag and got 70 hits when performing the search. An important question a EUD environment has to answer is how to organize the data in a way that desired data can be found. Participants of the PDW used a multitude of conceptual ways of thinking about data (e.g. process step in which data is entered, organizational unit or employee responsible, SAP module processing the data). In some cases users might want to search and browse data using different taxonomies of data element organization (e.g. process or organizational unit). To enable such data organization structures, data elements have to be linked with information about their context. Furthermore data elements should be annotated with a business-oriented description, explaining the exact meaning of the data to avoid misinterpretation. Easy to use mechanisms like tagging data sources could be used to establish a search space tailored to the needs of individual organizations and their employees.

Orchestrating data. Comparable to the abstraction of data to a conceptual level, the specification of how to combine data elements should also be enabled on a conceptual level (e.g. abstracting from operations based on relational algebra). As the participatory design workshop has shown, business users are able to build conceptual models for combining data using a simple box and wires framework. A EUD environment should support such ways of conceptual query building and faces the challenge of defining a formal model transformable in technical query languages. The EUD design environment should also take into account, that data is intuitively interpreted by users as being organized as a table and that users are quite familiar with spreadsheet applications which are based on a tabular model. This familiarity could be exploited by offering spreadsheet-like functionalities, like the ability to add a new column by just entering a formula and have instant results. To enable a unified representation of data, all data elements should be transformed to a tabular representation, regardless of their technical origin (e.g. relational tables, data warehouse cubes or service call results).

6 OUTLOOK

The presented results are relevant for the development of EUD tools, supporting business users in creating solutions supporting and facilitating tasks of their work. In order to transform these insights into the design of a concrete tool, the conceptual results have to be reflected and concretized using suitable technologies. We currently develop a prototypical visual EUD design environment for the lightweight composition of information artefacts, based on the obtained results. The prototype will be evaluated in a real enterprise scenario to examine if and to which extent the EUD tool is able to reduce complexity and enable especially technically inexperienced users to successfully create solutions, supporting tasks of their work, in a way they were not able before.

7 ACKNOWLEDGMENTS

We thank our reviewers for their valuable feedback and constructive suggestions. The research was funded by the German “Federal Ministry of Education and Research” (BMBF, project EUDISMES, number 01 IS E03 C).

REFERENCES

Beringer, J. (2004). Reducing Expertise Tension. Communications of the ACM, 47 (9), pp. 39-40.

Brehm, L., Heinzl, A. and Markus, M. L. (2001). Tailoring ERP Systems: A Spectrum of Choices and their Implications. Hawai'i International Conference on System Sciences (HICSS '01), Maui, HI, USA, January 3-6.

Fischer, G., Giaccardi, E., Ye, Y., Sutcliffe, A. G. and Mehandjiev, N. (2004). Meta-Design: A Manifesto for End-User Development. Communications of the ACM, 47 (9), pp. 33-37.

Henderson, A. and Kyng, M. (1991). There's no Place like Home: Continuing Design in Use. In Design at Work: Cooperative Design of Computer Systems, Lawrence Erlbaum, Hillsdale, NJ, USA, pp. 219-240.

Kvale, S. (1996). Interviews: An Introduction to Qualitative Research Interviewing. Sage Publications, Thousand Oaks, CA, USA.

Lieberman, H., Paternò, F. and Wulf, V. (2006). End User Development. Springer, Dordrecht, The Netherlands.

Markus, M. L. and Tanis, C. (2000). The Enterprise System Experience: From Adoption to Success. In Framing the Domains of IT Research: Glimpsing the Future Through the Past, Pinnaflex, Cincinnati, OH, USA, pp. 173-207.

Muller, M. J. (2003). Participatory Design: The Third Space in HCI. In The Human-Computer Interaction Handbook: Fundamentals, Evolving technologies and Emerging Applications, Lawrence Erlbaum, Mahwah, NJ, USA.

Pandit, N. R. (1996). The Creation of Theory: A Recent Application of the Grounded Theory Model. The Qualitative Report, 2 (4).

Pane, J. F. and Myers, B. A. (2006). More Natural Programming Languages and Environments. In End User Development, Springer, Dordrecht, The Netherlands, pp. 39-58.

Pane, J. F., Ratanamahatana, C. A. and Myers, B. A. (2001). Studying the Language and Structure in Non-Programmers' Solutions to Programming Problems. International Journal of Human-Computer Studies, 54 (2), pp. 237-264.

Pinnington, B., Light, B. and Ferneley, E. (2007). Too Much of a Good Thing? A Field Study of Challenges in Business Intelligence Enabled Enterprise System Environments. 15th European Conference on Information Systems (ECIS '07), St. Gallen, Switzerland, June 7-9.

Roth, A. and Scheidl, S. (2006). End-User Development for Enterprise Resource Planning Systems. Informatik 2006, Dresden, Germany, October 2-6.

Spahn, M., Dörner, C. and Wulf, V. (2008). End User Development: Approaches towards a flexible Software Design. 16th European Conference on Information Systems (ECIS ‘08), Galway, Ireland, June 9-11.

Stiemerling, O., Kahler, H. and Wulf, V. (1997). How to Make Software Softer - Designing Tailorable Applications. Conference on Designing Interactive Systems (DIS '97), Amsterdam, The Netherlands, August 18-20.

Wulf, V. and Rohde, M. (1995). Towards an Integrated Organization and Technology Development. Designing Interactive Systems 1995 (DIS 95), ACM, Ann Arbor, Michigan, USA.