Towards a Real-time Usability Improvement Framework based on Process Mining and Big Data for Business Information Systems

Sharam Dadashnia, Tim Niesen, Peter Fettke, Peter Loos

Institute for Information Systems (IWi) at the German Research Center for Artificial Intelligence (DFKI) and Saarland University, Saarbrücken, Germany

{sharam.dadashnia, tim.niesen, peter.fettke, peter.loos}@iwi.dfki.de

Abstract

Workflow improvement nowadays plays an important role in the selection process of supporting software. This is especially true in the context of user-centric development, where the usability of business information systems is a crucial characteristic of differentiation. However, automatically measuring the usability of such systems as well as their dynamic enhancement has not been studied before. This paper describes an approach to improve the usability of web-based information systems in real-time. Different concepts are presented, which build on data gathering methods from web analytics to provide log mechanisms for user interactions at a detailed level and subsequently process this data by means of data analytics and process mining methods. Concepts are then integrated into a comprehensive framework representing the main contribution of this paper. We evaluate our framework with a software prototype based on in-memory technologies developed in cooperation with a major German software company. Furthermore, we report on findings of a user study that was conducted in an exemplary use case scenario demonstrating dynamic workflow improvements to validate our research in a real-world setting.

1 Introduction

While functionality has traditionally been the most important criterion for selecting business information systems, nowadays, the *usability* of such systems plays an increasingly important role. In consequence of the continuous dissemination of intuitive and user-friendly devices in the consumer sector, e.g. the Apple iPhone, the demand for easy-to-use software is also increasing in business environments. Therefore, it is crucial for software manufacturers to not only pay attention to a software's functionality but also to ensure a satisfying usability for end-users. Usability covers many aspects; one of the most important ones relates to *intuitive* and *efficient* use of a software component in order to effectively fulfill a given task. Nielsen et al. describe the importance of usability especially for web applications in terms of compliance with the existing style guides. According to those guides, the question of whether a user stays on a webpage or leaves it is strongly influenced by the appearance and ease of use of the corresponding system and not necessarily by its functionality or offer of services (Nielsen 2012). These aspects can be mapped to applications in the business context: as the usability of business information systems becomes an important distinguishing element, software manufacturers strive to develop products that provide enhanced usability. In that regard, also testing and evaluating the usability of a software system becomes an important aspect in the development process. As the testing is often conducted manually, the costs associated with usability evaluations are usually very high. Real-time usability improvement as presented in this paper can help to mitigate such costs and provide more efficient ways to assess the usability of software systems.

As for the status quo, it is not possible to test the usability of business applications automatically. Common practice is that the usability of software application is manually tested during the adoption stage and after the development phase. Further testing during the operational usage of the software is usually not conducted. Therefore, there is no efficient way to ensure good usability after this phase. Reasons for the change of usability requirements are highly correlated with changes within underlying business processes, e.g. to assure compliance of the implemented processes. Beyond this aspect, it is not possible to analyze the user behavior during the usage of an operational system. Against this background, the paper at hand aims at providing a framework to reflect usability aspects in order to combine mining methods and in-memory computing to dynamically improve the usability of software systems. While the improvement can be deployed globally, i.e. for all users, we focus on user-specific improvement, which allows for the improvement of individual usability aspects per user. The main goal is, thus, to develop concepts to improve the interaction between a user and a software system to enable a more intuitive, more efficient and quicker usage.

The problems discussed in the following were ascertained by analyzing practical use cases within business information systems deployed by a major German software manufacturer that has a strong focus on enterprise resource planning. Against the background of an increased usage of web applications in recent years, concepts and analytical methods from the field of web analytics form the basis for our usability improvement framework (Booth and Jansen 2008). In particular, the following two research questions will be addressed in the paper:

(RQ1) Which usability aspects can be improved dynamically with respect to the underlying business processes?

(RQ2) Which conceptual approaches are suitable for developing a comprehensive framework that provides an objective and transparent method for the improvement of software systems?

Our research follows a design-oriented approach (Hevner et al. 2004). We seek to develop an innovative artifact in the form of a methodical framework as well as a concept that can be practically applied in the domain of business information systems. The research was conducted within the scope of a joint research project with partners from both academia and software industry. We developed a prototype to demonstrate the applicability of our ideas, thereby following the ARIS methodology for information systems. In ARIS, the overall process of software design and implementation is divided into the three phases: *requirements definition*, *design specification* and *implementation* (Scheer 1994). The developed prototype was evaluated in a laboratory experiment, which was embedded in an exemplary use case scenario deduced from a real-world scenario together with the industry partner.

The paper at hand is structured as follows: After this introduction, section 2 describes related work to provide the context for the concepts presented in this paper. Section 3 then exposes our concepts for a dynamical improvement of workflows within business information systems and incorporates them into a framework. In section 4, a prototypical software implementation is presented. Afterwards, section 5 describes the evaluation results based on task executions. In Section 6, we provide a conclusions and an outlook on further research.

2 Related work

2.1 Usability and Business Process Management

In general, the usability of a product is an indicator on whether a user can use a product – for the scope of this paper, a business information system – to achieve a given objective in an *effective*, *efficient* and *satisfactory* way (ISO-Norm 1998). These are the main characteristics from the ISO definition of usability, which we take as the basis for defining a systems acceptance by a corresponding user. Certain usability aspects refer to the direct interaction between users and systems, i.e. the (graphical) user interface of a system. The aspects of *effectiveness* and *efficiency* indicate a process character as we define business processes as a series of logically related activities performed in a business context to attain a specified goal in an efficient and effective way (Scheer 1999). This is essential for the quality of processes in companies which use tools and methods in the context of business process management (vom Brocke and Rosemann 2010). Therefore, a clear connection between the usability aspects within business information systems and the underlying business processes can be assumed. The idea of measuring usability-related aspects regarding log data resulting from the execution of business processes was already applied in projects in the context of business process management (Thaler 2014; Thaler et al. 2015). However, the work of Thaler only allows for analyzing historical data and does not comprise real-time usability improvement techniques.

2.2 Process Mining and Web Analytics

With the increasing number of enterprise information systems storing relevant events from underlying business processes, a vast amount of structured event logs becomes available (van der Aalst 2014). Leveraging information within log files by means of data mining techniques is referred to as *process mining* (van der Aalst et al. 2004). Because process mining is based on the analysis of existing log files, it helps obtaining insights into the actual process behavior, which, in practice, may differ from the desired behavior modeled in the corresponding reference models. Regarding analysis methods, three sub-disciplines can be distinguished (van der Aalst 2012): *discovery* aims at generating process models from the sequences of events identified within log files. Comparing an existing reference model to the event log generated by executing the model is denoted *conformance*. Finally, *enhancement* seeks to improve existing process models by augmenting information gathered during the analysis. The field of *process discovery* has attracted most attention in recent research. Against this background, various algorithms drawing on formal, heuristic or machine-learning approaches have been proposed. Traditional process mining is conducted on historical data of past process executions. In this paper, however, it is necessary to gather and analyze data immediately after user interactions, i.e. in real-time. In the context of web applications, methods from the field of web analytics, for instance the page tagging method, are used to gather data and store them into a database instead of a server side log file (Peterson 2004).

2.3 Big Data

The term *big data* has become very popular in recent years to describe the exponential growth and availability of data from a variety of sources in structured as well as unstructured form (Kambatla et al. 2014). Big data is often described by characteristics referred to as the *four Vs*: *volume, variety, velocity* and *variability* (NIST 2014). The problem we face in the current context mainly relates to the volume aspect with respect to the user interaction data needed for the analysis and improvement of usability. Furthermore, in combination with a slump in prices regarding IT storage capacity for both sequential and random access memory, storing and analyzing of large amounts of data becomes technically feasible. Business information systems support a wide range of processes comprising a large amount of tasks for a given business environment. Thus, amounts of detailed log data correspond to the definition of big data regarding the aspect *volume*. Furthermore, the aspect *variety* is addressed regarding log files that are produced by different users of different systems within a company or sourced from different repositories. This usage data has to be stored in an adequate way to enable real-time analysis, which is assumed for the concepts presented in section 4.

3 Concept Development

3.1 Preliminary remark

The goal of the developed concepts is to generate initial methods to dynamically improve software-supported workflows for each user. The improvement of workflows is highly intertwined with usability aspects (see section 2.1), so we base our concepts on the definition of usability, especially with respect to the aspects of *effectiveness, efficiency* and *satisfaction*. The concept development follows three different phases. These phases are "Data Collection", "Calculate Metrics" and "Workflow Improvement". Figure 1 shows the basic framework components as well as their interactions and is divided in two sections: the left section represents user interactions with the software system through a user interface, the optimization of which is the main ambition of this contribution. Each user operating with the business information system produces usage data.

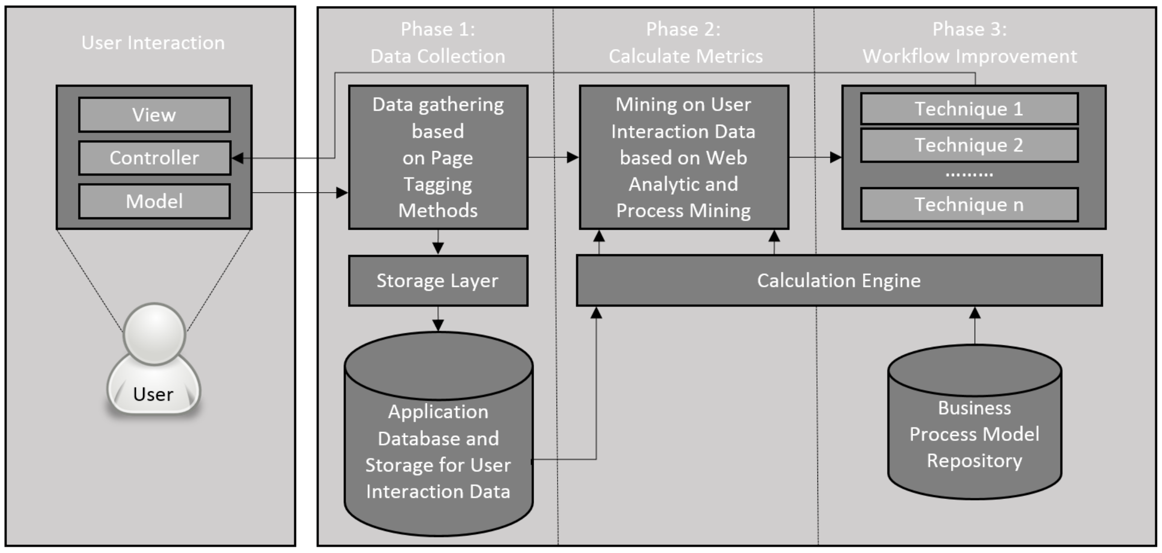

User Interaction
View
Controller
Model
User
Phase 1: Data Collection
Phase 2: Calculate Metrics
Phase 3: Workflow Improvement
Data gathering based on Page Tagging Methods
Storage Layer
Application Database and Storage for User Interaction Data
Mining on User Interaction Data based on Web Analytic and Process Mining
Calculation Engine
Technique 1
Technique 2
....
Technique n
Business Process Model Repository

Figure 1: Three-phase concept for real-time improvement

The right section depicts the mechanisms which store user interaction data collected by the page tagging gathering method (phase 1) into the application's database. For the calculation of the metrics (phase 2), we employed certain metrics from the field of web analytics and process mining. For the calculations based on process mining, an additional data storage which contains reference models, a so-called "business process model repository", is needed. These repositories are often used in companies and are organized by business process management methods (Gruhn and Schneider 1998).

The calculated metric values are used to derive decisions for dynamic workflow improvement. The improvement of the software system is implemented in the third and last phase of the framework. Within this phase, the developed concepts are applied to the business information system, which has a direct influence on the execution of the software's internal control flows that are related to the *Controller Layer* (Gamma et al. 1994). Because of the dynamic character of the improvement concept, the software system has to be extended on a technical level. This enables the software to respond automatically without the interaction of a software developer and to dynamically improve itself. The following paragraphs show the individual phases of the concept development and present them in more detail.

3.2 Phase 1: Data Gathering

In the first phase, the data produced by user interactions with the business information system is gathered for further analysis. For that purpose, the page tagging method was employed, which is widely used in the context of web applications (see the table below). As this method is based on JavaScript elements, it can easily be added to each website of the application system to collect a set of technical information about the requesting user system. This information (called *request properties*) is summarized in table 1. The first column *request property* describes the attributes that the page tagging method can collect from the request triggered by user interaction. The *description* column gives a brief summary of the respective property, while the third column (*value*) is an indicator of whether and how strongly the request property is considered useful for further analysis.

The *value* is differentiated into *strongly valuable* ●, *valuable* ◑ and, at first glance, *not valuable* ○. Within this first phase, we focused on gathering as much data as possible in order to obtain a broad range of interaction data that future analyses can draw on. Hence, the development of the framework constitutes an overview of currently existing concepts for further analysis techniques and metrics for different use cases.

All properties listed in table 1 are persistently stored in the application database, yielding an efficient access to analytical results in later phases of dynamic improvement. With respect to the value of properties and process mining methods applied in later phases, we identified the properties *userId*, *timeStamp*, *route* and *target* as very important because these data and the information provided in log data used for process mining analysis are highly correlated (van der Aalst et al. 2004).

3.3 Phase 2: Metrics

Based on the data gathered during the application's usage, we analyze the data and calculate certain metrics which are used in the web analytics context. These metrics form the basis for further analysis since they are extended by other metrics based on process mining (see section 3.4), thus, incorporating the information available through the log data of business processes.

Request Property	Description	Value
appId	The unique application identifier used to separate the log entries from different applications. Within a company, it is possible that more than one application is used as a productive system.	●
sessionId	A session identifier, which most commonly is a web browser in the area of web applications, is stored to identify clients.	◑
userId	Individual user identifiers are mandatory for user-specific improvement of the software. Therefore, a pseudonymized persistent identifier is used to enable continuous improvement beyond the created session.	●
timeStamp	Every entry in the table is enriched with a timestamp to ensure the recording of the user interaction in a time-based manner.	●
deviceType	The user of an application has several options to access the web application, e.g. via desktop computer, mobile device or tablet. This property stores which device type is used to access the application.	◑
browser	The browser property stores which browser is used by the software user.	○
browserLanguage	The browser language is stored to get further information about a user's context.	●
operatingSystem	In addition, the device's operating system is stored.	○
route	Another important property is the route on which the user navigates through the application. This property stores the user's geographical location of access.	●
target	Target describes the function, which is executed at last. This is e.g. a screen change within the application. Target functions typically indicate the end of a process.	●
targetView	Because the applications are usually implemented following the model-view-controller-pattern and we want to improve the application with assistance of the controller, the target view of the application is stored.	●
duration	This property records the duration of an user session.	○
posX	posX stores the actual x coordinate of the mouse on the screen	○
posY	posY stores the actual y coordinate of the mouse on the screen	○
pageX	pageX stores the x value of the screen resolution.	○
pageY	pageY stores the y value of the screen resolution.	○

Table 1: Data gathering method

The metrics in table 2 represent a selection of best practice metrics used in the context of web analytics applications (Booth and Jansen 2008). Certain metrics, for instance demographic information, are not relevant in the context of business information systems because these systems are not produced for an anonymous market but for users in a certain company. Therefore, this metric is not important in terms of usability improvement. These metrics depict an initial step to analyze the data.

Metric	Description	Data gathering property
Visitor Type	The type of visitor is important to improve the usability personalized for a single user or a user group. The user can be assigned to a certain user group.	*sessionId, userId*
Visit Length	The duration of a visit can be calculated as the duration to finish a task on a screen of the software.	*timestamp, duration*
Visitor Path	This metric identifies the chosen sequence by a user to accomplish a certain task within the business information system.	*route, target, targetView*
Top Pages	Calculates a ranking of top-most selected pages.	*route, target, targetView*
Errors	Errors on the client side are collected and aggregated into a single metric.	*deviceType, browser, browserLanguage, operatingSystem, posX, posY, pageX, pageY*

Table 2: Metrics

3.4 Phase 3: Development of workflow improvement techniques

Within this section, we describe the developed concepts, which are based on existing data gathering methods and metrics. Additionally, aspects from the area of process mining are used to explore the implicit knowledge, existing in log data originating from executing business processes, which can, in turn, be used to improve and personalize the software usability. The usage of the user interaction data in form of logs produced by the underlying business information system are highly correlated with the log files used in the context of process mining. Against this background we can use e.g. the metric of *visitor path* derived from the instance logs to calculate probabilities for the prediction of the user behavior while processing a given task. The developed concepts are described as follows. First, we present the basic idea of the concept along with a motivation. In a second step, within the corresponding table, data gathering properties are listed along with existing metrics and the employed process mining method. Every table comprises three rows, where *PG* denotes the properties gathered, *M* denotes metrics and *PMA* denotes the process mining approach employed.

3.4.1 Preloading of content

Within business information systems, many data has to be provided on demand when a user needs to execute a certain task. Against the background of an increasing amount of application data that has to be processed in those systems, resource bottlenecks can occur. Software companies have already counteracted these performance problems by using more efficient technology approaches, e.g. in-memory computing. However, within many software systems, there is still a problem in terms of data transfer from a database to the user interface, especially in the context of enterprise resource planning applications when lists have to be aggregated from stored data. Existing approaches within these applications try to cache these data, however, this is mostly done for specific known problems and not dynamically in terms of user behavior. To face this problem, we present a technique to preload content that is needed in a subsequent screen to ensure a smooth transition from one screen to another. The following table provides further details on the technique *preloading of content,* which describes this idea.

	Preloading of content
PG	*appId:* Important to identify the used application (in case several applications are used simultaneously). *sessionId:* Used to identify exactly one single client for whom the improvements are calculated. *userId:* Necessary for providing a user-specific improvement of the software. *timestamp:* Needed to calculate certain metrics like the task completion time. *route:* Used to reproduce the user behavior for a given task within the application. *target:* Provide the basis for metrics like the most popular views of an application. *targetView:* Used in the context of the model-view-controller pattern to gather the technical representation of the view in order to get the corresponding controller. *duration:* Forms the basis for the calculation of the overall task execution time.
M	*Visitor Type:* For the personalized improvement, an obvious user is necessary. *Visitor Length:* For planning and triggering the data load process at the right time. *Visitor Path:* Main metric used for analyzing user behavior within an application and for conformance checking.
PMA	The gathered logs provide the input for the *conformance checking* method. Another input for the conformance checking is a business process, which is held in the process model repository. Thus, it is possible to predict the next steps within the application. So, another metric which holds the probability to switch to a next function of the application can be calculated. An assumption for this approach is that the function of the next screen is accessed within the same workflow execution. Furthermore, the approach only operates if certain iterations of each corresponding business process are executed to provide a base of log data for the calculation.

Table 3: Preloading of content

3.4.2 Dynamic Quick-Links

The second technique describes the problem that business application systems provide a user with many transactions of process business related tasks. It is possible for a user to deposit favored transactions for their own assigned tasks. However, proposals for the respective users are not provided. User tasks can change sporadically and an intelligent mechanism can help to reorganize the most often used and most necessary transactions for a user. The concept of *dynamic quick links* provides the user with a list of most frequently accessed pages or transactions in the form of a link to the respective screen or transaction. Therefore, using specific transaction numbers is not necessary anymore to execute certain tasks, leading to an application that is easier to use.

	Dynamic Quick-Links
PG	*appId, sessionId, userId, timestamp, target, targetView*. Remark: The explanations are the same ones mentioned in Table 3.
M	*Visitor Type, Visitor Length.* Remark: The explanations are the same as in Table 3. *Top Pages:* This metric depicts the most frequent application views per user and the corresponding function calls.
PMA	The generated user interaction data that is stored in the log database is used to apply certain calculations regarding the *conformance checking* method. This method uses the identification of functions and features that were performed e.g. in a final step. Assuming that no direct actions are required between the first and the last function, the user could jump directly to the last step of the application. The user can skip the functions between the first and the last step, which are not necessary to reach a process goal by using the generated quick-link. The check of whether a function is necessary to reach the goal of a business process and to generate the corresponding output is performed by the conformance check with the underlying process from the model repository.

Table 4: Dynamic Quick-Links

3.4.3 Functionality Overhead

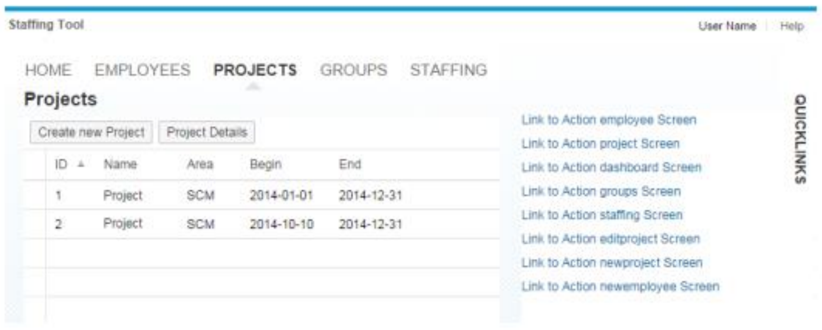
The third concept concerns the overhead of functionality in business information systems. Users are often provided with more functionality than they need to complete certain tasks. This is because business information systems, in most cases, support a complex business process. Thus, an overhead of functionality mainly occurs in the area of standard software. The aim is to provide a user only those functionalities, which he actually needs to process a specific task. The sequence of different screens or buttons that are not used for that task within an application should be hidden from the user. By doing so, a complex application can be simplified for a specific user by providing context-sensitive information. A positive side effect is that the stored business processes can continuously be checked for a given task and can be continuously improved based on this generated information. This concept addresses very individualized user tasks on a fine-grained level, as opposed to the role concept in customizable standard software systems. Role concepts in general separate the users in certain groups with specific permissions and individually adapted graphical user interfaces (buttons, dropdowns etc.). However, the role concept is a static concept, which is implemented during the customizing phase of a software system. In contrast to that, the *functionality overhead* concept considers a much higher level of detail: It analyzes process instance data to determine the actually user interactions and to identify functionality that is never used by a certain user (“overhead”).

	Functionality Overhead
PG	*appId, sessionId, userId, timestamp, route, target, targetView*. Remark: The explanation is the same as the one mentioned in Table 3.
M	*Visitor Type, Visitor Length, Visitor Path, Top Pages.* Remark: The explanation is the same as in Table 3.
PMA	This technique also uses the *conformance checking* approaches from process mining. Thus, it is possible to make a comparison of the actual state and the desired state of the system by applying the mentioned metrics on the actual execution of the process. Functionalities which occur in the business processes and are held in the model repository but are not executed in the actual processes are identified by this method. However, it should be noted that the output of the functionality modeled in the process from a repository is also confirmed in the actual model.

Table 5: Functionality Overhead

4 Prototype

The implementation of the concept developed in this paper needs a business information system as a basis. For this reason, a use case was implemented to provide this necessary basis. The prototype was implemented to evaluate the developed concept *dynamic quick-links* and to demonstrate its effectiveness. The use case is part of a process to organize employees, projects and their mutual assignment. This tool is a so-called *staffing tool*. The tool supports an underlying business process; hence, it fulfills the requirements to apply the concepts of the presented framework. The tool itself contains projects and employees and helps project managers to organize and staff projects in an efficient way. It also displays the projects that have a demand for employees and the employees that have free capacities to work on a project. The prototype was implemented by using state-of-the-art technologies and strongly builds on in-memory databases. The prototype is divided in a front-end component, using the model-view-controller pattern, and a back-end system, which provides the business logic as well as the application storage. To implement the front-end of the prototype, we used the web application framework SAP UI5 (SAP SE 2014a) while the back-end component was implemented in SAP HANA XS, which acts as the application layer in the overall architecture (SAP SE 2014b). With respect to the requirements of real-time data processing, performance is a crucial factor, which in turn favors the usage of in-memory databases (Plattner and Zeier 2015). Therefore, we use the in-memory database SAP HANA (Sikka 2013). Figure 2 shows a screenshot of the latest version of the prototype. The screenshot shows the basic functionality to view and edit the details of project and employee data records. On the top of the screen, there is a navigation bar, allowing access to different views of the application. The rest of the screen is vertically divided into a list of quick-links on the right side (see concept of *dynamic quick-links*), which is dynamically updated every time a user requests a particular application view. The left side of the screenshot shows a list of current projects with the possibility to switch to a project's detail view, which shows further information.



ID	Name	Area	Begin	End
1	Project	SCM	2014-01-01	2014-12-31
2	Project	SCM	2014-10-10	2014-12-31

Link to Action employee Screen
Link to Action project Screen
Link to Action dashboard Screen
Link to Action groups Screen
Link to Action staffing Screen
Link to Action editproject Screen
Link to Action newproject Screen
Link to Action newemployee Screen

Figure 2: Screenshot of the Prototype

5 Evaluation

To evaluate our concept and its implementation, we used an evaluation scenario based on the use case implemented in the prototype. Therefore, we selected two persons with a background in project management: Both user were entrusted with the task of creating 50 datasets for current projects. For performing this task without a dynamic improvement of the software system, four steps were needed (*A*, *B*, *C* and *D*) for the insertion of each project or employee respectively. The number of four steps is due to the fact that a user (starting from *A*) had to "click trough" two screens (*B* and *C*), which were irrelevant for the given task (on screen *D*), i.e. the screens were displayed to the user but immediately skipped by them. After some initial interactions, the system was able to analyses the recorded interaction data. From this analysis, the system could derive instances of the *ABCD*-process, which revealed that the tasks *B* and *C* were not necessary to perform the given task. The system then could provide a dynamic quick link (from *A* to *D*), which lead to an improvement from four steps to two steps only and a tremendous saving of time.

For both users, the concept provided an improvement of the application and an improvement of the necessary expenditure of time regarding the executed task. Another finding within the evaluation was that the first user noticed the dynamic quick link already in the third iteration and used the quick link provided by the dynamical improvement from then on. The second user noticed the dynamical improvement after the sixth iteration and then began using the dynamic quick link. Hence, both users were using the application with a dynamical improvement in form of a personalized quick link. This evaluation only provides a first indicator for the further development of the prototype as it is not based on a representative amount of test persons.

6 Summary and Future Work

The developed concepts provide a basis for the evaluation and improvement of web-based business information systems' usability. The paper at hand presents a framework to integrate the fields of mining methods, big data technologies and business process management and to integrate corresponding methods in order to improve the usability in a dynamic and user-specific way. With respect to *RQ1,* we found that especially the usability aspect *efficiency* can be analyzed and, under certain limitations, dynamically be improved, resulting in time savings achieved in the evaluation scenario. Regarding *RQ2,* we found that conceptual approaches from the field of web analytics (*gathering methods* and *metrics*) and approaches from the field of process mining (esp. *conformance checking*) are suitable for developing an integrated framework with certain techniques (*preloading of content, dynamic quick-links* and *functionality overhead*) to improve software systems. Regarding the developed prototype, we also show that in-memory databases can also be used to solve software company's inherent problems like the improvement of self-developed software systems. Therefore, the presented framework shows that these concepts can help to improve the usability of business information systems in a dynamic way. The presented concepts are integrated within a comprehensive framework and provide a transparent method for the improvement of software systems.

However, our concepts as well as the implemented prototype still entail some limitations. The first limitation concerns the usage of the gathered data. In their current form, the concepts only consider a subset of the data, leaving potential for more detailed analysis and further concept development. For instance, the concept of dynamic quick-links does not consider any temporal aspects: regarding the use case scenario, if a user executes a given task *x* times, the system establishes a quick-link to the appropriate screen. If the task changes, causing the user to work on a different screen, an adequate quick-link is only provided after the user has executed this new task for *x+1* times. This means that temporal information regarding recent usage is not consider as a weighting factor for the ranking of the quick-link. As a second limitation, the prototype currently implements one of the proposed concepts (*dynamic quick-links*). Further investigations need to be done with respect to the integration of further concepts (*functionality overhead, preloading of content*) into the existing framework. In addition, the interoperability among the different concepts needs to be considered.

In the future, we plan on improving our research prototype. Against this background, we focus on a more detailed implementation of the current concepts and the development of additional concepts to extend the framework. In addition, the existing concepts will be evaluated by comprehensive user tests to gain more insights into the concepts' strengths and weaknesses. Additionally, the gathered data will be used to generate important information and statistics for the developers of business information systems. Therefore, aspects and metrics from software quality will be used to gain insights into the improvement potential of business information systems. Possible aspects are building on existing concepts of software improvement and integrating these steps into the application lifecycle of these software systems. It is also possible to provide this information in the context of a scrum-like organized development process to obtain early user feedback and be able to correct errors in a system in a very early stage of development.

7 References

Booth D, Jansen B (2008) A review of methodologies for analyzing websites. Handb Res web log Anal 141–162.

Gamma E, Helm R, Johnson R, Vlissides JM (1994) Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley

Gruhn V, Schneider M (1998) Workflow management based on process model repositories. In: Proceedings of the 20th international conference on Software engineering. pp 379–388

Hevner BAR, Esearch SYR, March ST, et al (2004) Design Science in Information Systems Research. MIS Q 28:75–105.

ISO-Norm (1998) INTERNATIONAL Ergonomic requirements for office work with visual display terminals (VDTs) - Part 11 : Guidance on usability.

Kambatla K, Kollias G, Kumar V, Grama A (2014) Trends in big data analytics. J Parallel Distrib Comput 74:2561–2573. doi: 10.1016/j.jpdc.2014.01.003

Nielsen J (2012) Introduction to Usability. http://www.nngroup.com/articles/usability-101-introduction-to-usability/. Accessed 3 Oct 2013

NIST (2014) NIST Big Data Interoperability Framework: Definitions. 2014

Peterson ET (2004) Web analytics demystified: A Marketer’s Guide to Understanding How Your Web Site Affects Your Business. Celilo Group Media

Plattner H, Zeier A (2015) In-Memory Data Management: Technology and Applications. 2nd edn. Springer

SAP SE (2014a) SAP UI5 Developer Guide. https://sapui5.netweaver.ondemand.com/sdk/#docs/guide/95d113be50ae40d5b0b562b84d715227.html. Accessed 25 Sep 2014

SAP SE (2014b) SAP HANA XS JavaScript Reference. http://help.sap.com/hana/sap_hana_xs_javascript_reference_en/index.html. Accessed 25 Sep 2014

Scheer A-W (1994) Business Process Engineering, 2nd edn. Springer Berlin Heidelberg

Scheer A-W (1999) ARIS -Business Process frameworks. Springer Berlin Heidelberg

Sikka V (2013) Re-thinking the performance of information processing systems. 2013 IEEE 29th Int Conf Data Eng 9–13. doi: 10.1109/ICDE.2013.6544809

Thaler T (2014) Towards Usability Mining. In: Lecture Notes in Informatics - Jahrestagung der Gesellschaft für Informatik (INFORMATIK-14), Big Data - Komplexität meistern. Springer,

Thaler T, Fettke P, Loos P (2015) Mining the Usability of Business Process Modeling Tools : Concept and Case Study.

van der Aalst W (2012) Process mining: Overview and opportunities. ACM Trans Manag Inf Syst 3:1–17.

van der Aalst W, Weijters T, Maruster L (2004) Workflow mining: discovering process models from event logs. IEEE Trans Knowl Data Eng 16:1128–1142.

van der Aalst WMP (2014) Process Mining in the Large: A Tutorial. In: Zimányi E (ed) Third European Business Intelligence Summer School. Springer International, Cham, pp 33–76

vom Brocke J, Rosemann M (2010) Handbook on Business Process Management 1, 1st edn. Springer