Service-oriented Development of Federated ERP Systems

Nico Brehm, Jorge Marx Gómez

Department of Computer Science, Carl von Ossietzky University Oldenburg,
Ammerländer Heerstrasse 114-118, 26129 Oldenburg, Germany
{nico.brehm, jorge.marx.gomez}@uni-oldenburg.de

Abstract. The paper presents a new architecture approach for the distribution of the application logic in ERP systems. The approach proposes the provision of software components which implement specific functionality as Web Services. The paper shows how these Web Services can be developed and provided by independent software vendors. The model advances the reusability of data types and reduces the necessity of data transformation functions in business process descriptions. Furthermore a first prototype implementation *(FERPxONE)* is presented and an example process for the generation of a simple diagram for the comparison of customers is given.

Keywords: Federated ERP systems (FERP), SOA, Workflow, Web Service, YAWL

1 Introduction

Because business processes do not stop at artificial borders which had been set by the structure of the functional organization of enterprises the integration of cross-department business processes has been becoming a more and more frequently discussed topic. Since now the IT support of enterprises still faces problems which result from the missing interoperability of available software systems or the non-availability of software functionality which matches individual requirements of single enterprise departments.

One of the main reasons which increased the demand of ERP system technology in the last two decades results from its data-centric view. This paradigm forms the basis for the internal architecture of ERP systems as well as the structure of the business functionality. ERP systems facilitate the view of the enterprises as a whole. The application of ERP systems mainly aims at the improvement of the collaboration between the different departments of an enterprise. An *ERP system* is a standard software system which integrates operational application systems of different enterprise sectors. The integration is based on a common data model for all system components. Modern ERP systems consist of many software components which are related to each other. Currently these components are administered on a central application server. In connection to the ERP system complexity several problems appear:

- The price- performance ratio is dependent to the potential benefit an ERP system is able to generate.
- Not all installed components are needed.
- High-end computer hardware is required for the application of complex ERP systems.
- Customizing is expensive because specialists are needed in order to manage the complexity of the overall system.
- Utilizing enterprises are dependent to one ERP system provider. This aspect is in particular a disadvantage for Small- and Medium-sized Enterprises (SME) because of their demand as regards to the necessary flexibility of their business processes.
- A migration from one ERP system to another is very time-consuming and expensive. Even a partial migration by means of the exchange of components requires manual effort for the adaptation of interfaces because software components are mostly not interoperable on the business abstraction level. Furthermore the composition of business application systems out of subsystems from different vendors often leads to a redundant keeping of data and a redundant application of functionality because the functionality is overlapping.

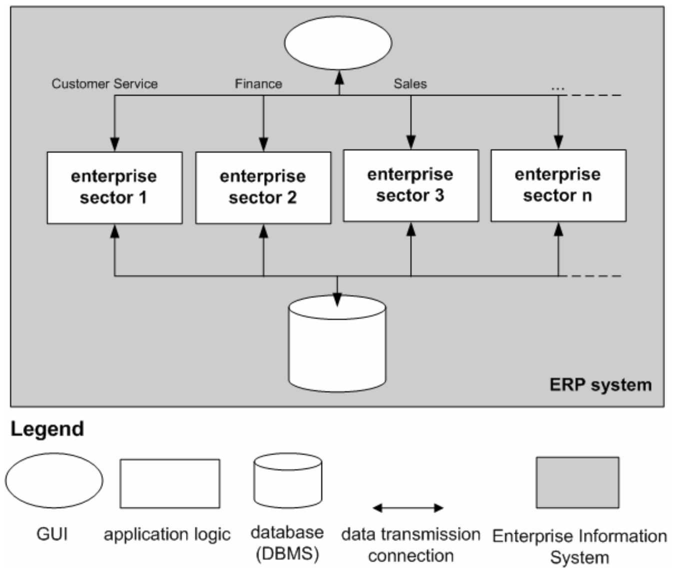


Figure 1: Architecture of a conventional ERP system

Due to the expensive installation [8] and maintenance proceedings of ERP systems normally only large enterprises can afford complex ERP systems which cover all enterprise sectors. In practice small- and medium sized enterprises (SME) deploy several business application systems in parallel. Often many of these systems approach the definition of an ERP system. However, the full potential of each system is not exploited.

Survey results
Because of the problem that the currently available ERP software products do not include all necessary functions companies have to run various systems in combination with each other. In order to prove this proposition the authors carried out an empirical investigation which was completed in the third quarter of 2006. This investigation was based on a survey where more than 600 German SMB with up to 250 employees answered a questionnaire. Amongst others the following figures could be extracted as results:

- In order to meet the overall core functionality requirements[1] the average SMB in Germany runs *four software products in parallel*.
- 47 percent of the interviewed enterprises complain about the serious problem of the redundant keeping of data. Other 30 percent agree that at least a part of their enterprise data is double-stored. Only 18 percent state that they do not have a problem with redundant data. 5 percent could not form an estimate.

Taking a look at the practical use of parallel business applications, problems like data inconsistencies in independent databases and increased communication expenses for function transitions in business processes become obvious. Furthermore it has to be mentioned that in most cases the software solutions come from different vendors.

The parallel operation of business application systems causes problems which jointly arise from insufficient system integration [1, 4].

A new solution to face these problems is the application of a *shared ERP system* which makes its components available over a network of services providers. This component ensemble (federated system) still appears as single ERP system to the end-user, however it consists of different independent elements which exist on different computers. Based on this construction it is possible for an enterprise to access on-demand functionality (business components) as services[2] of other network members over a P2P network. This approach solves the mentioned problems as follows:

- The total application costs conform to the effective use of software functions.
- Due to the separation of local and remote functions, no local resources are wasted for unnecessary components.
- Single components are executable on small computers.
- Due to decreasing complexity of the local system also installation and maintenance costs subside.

[1] The term core functionality refers to the support of business tasks in the areas of financial accounting, customer management, sales, payroll accounting, controlling, procurement, administration of inventory, personnel administration, production planning and control, quality management as well as research & development.

[2] In this term, a service is a software component that encapsulates one or more functions, has a well defined interface that includes a set of messages that the service receives and sends and a set of named operations [6].

- A cross-vendor standardization of extendable data models and functional specifications for ERP systems and the establishment of a unified architectural model (reference architecture), however, would change this situation for the benefit of the interoperability and the effectiveness of Enterprise Application Integration (EAI) solutions.

This motivation leads to the definition of a Federated ERP system which can be defined as follows:

Definition 1: A *federated ERP system* (FERP system) is an ERP system which allows a **variable, adaptive or dynamic assignment of business application functions to independent software providers**. The overall functionality is provided by an ensemble of standardized subsystems that all together appear as a single ERP system to the user. Different business components can be developed by different vendors.

In this paper we present a FERP system based on Web Services. The main idea follows the multi-layer paradigm of modern information systems which aims at the separation of the application logic from the presentation layer and the database layer. In our approach the application logic of ERP systems is encapsulated in a multiplicity of Web Services which can be provided either locally or remotely. The vision of this approach is to allow the application of business logic components in a distributed manner. In order to facilitate a vendor-independent development and provision of those components the approach considers the standardization of Web Services as well as GUI descriptions and database interactions. The standardization process is supposed to be advanced by a consortium of ERP vendors, utilizing enterprises and scientific institutions (*FERP standardization consortium*).

2 Development and marketing model

As described above the idea of FERP is the shared development of ERP functionality in a community of Web Service providers and workflow designers. Figure 2 shows the development and marketing model of this vision. The model consists of four layers. The standardization layer represents an initiative for the standardization of Web Service types. The development layer represents two types of actors. On the one hand workflow designers are responsible for the specification of ERP business logic by an orchestration of Web Services. On the other hand Web Service developers encapsulate business functionality in Web Services. Both groups are referencing the same standard in order to potentiate a matching of demand and supply at execution time of workflows. Workflow definitions can be considered as best practice processes which represent one part of the business logic of a standard software system.

The marketing layer is represented by a marketplace for workflow definitions and Web Service offerings. Because Web Service descriptions have to comply with predefined standards a concurrent offering of Web Services by a variety of providers and their dynamic use is possible. The utilization layer in this model is represented by a standard software system for utilizing enterprises which consists of a graphical user interface, a database and a Workflow management system [1]. This system allows the execution of workflows. The administrator of this system can choose workflow definitions from the market place and feed them into the workflow management system. Concrete Web Services are chosen automatically on execution time on the basis of standardized type descriptions. Finally a user is able to initiate the execution of workflows and by this to use the system.

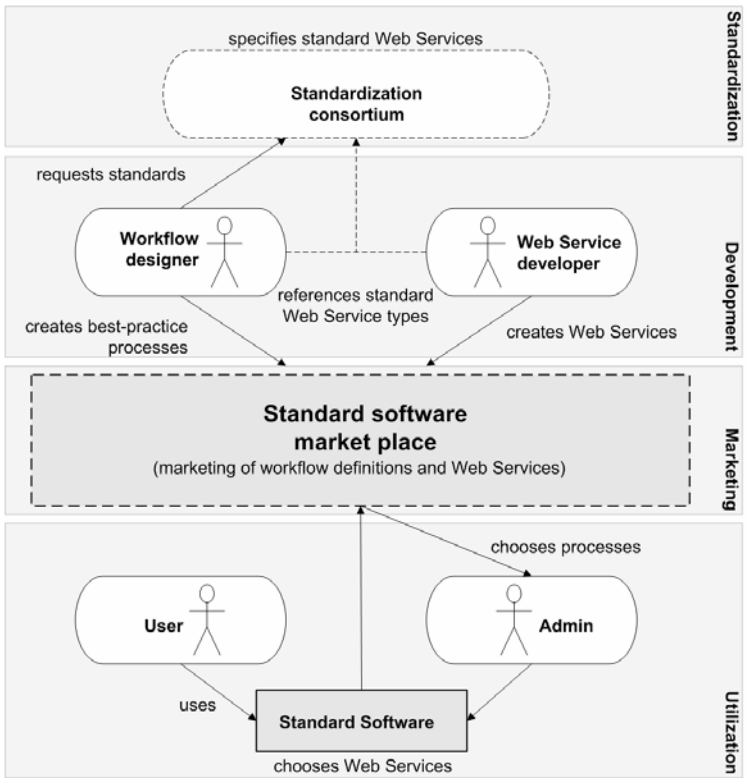

Standardization
specifies standard Web Services
Standardization consortium
Development
requests standards
Workflow designer
Web Service developer
references standard Web Service types
creates best-practice processes
creates Web Services
Marketing
Standard software market place
(marketing of workflow definitions and Web Services)
Utilization
chooses processes
User
Admin
uses
Standard Software
chooses Web Services

Figure 2: Marketing model for best-practice processes based on Web Services

3 Architectural vision statement

Figure 3 shows the vision of a Federated ERP system from the point of view of a utilizing enterprise. The basic FERP framework consists of a graphical user interface, a workflow management system, a central enterprise database and several business components which are encapsulated in Web Services. The workflow management system is responsible for the execution of business processes which include three different types of tasks. GUI-tasks describe generic user interfaces for the communication with end users. Database-tasks describe operations which are related to the data accesses. Web Service tasks include information about the dynamic invocation of Web Services.

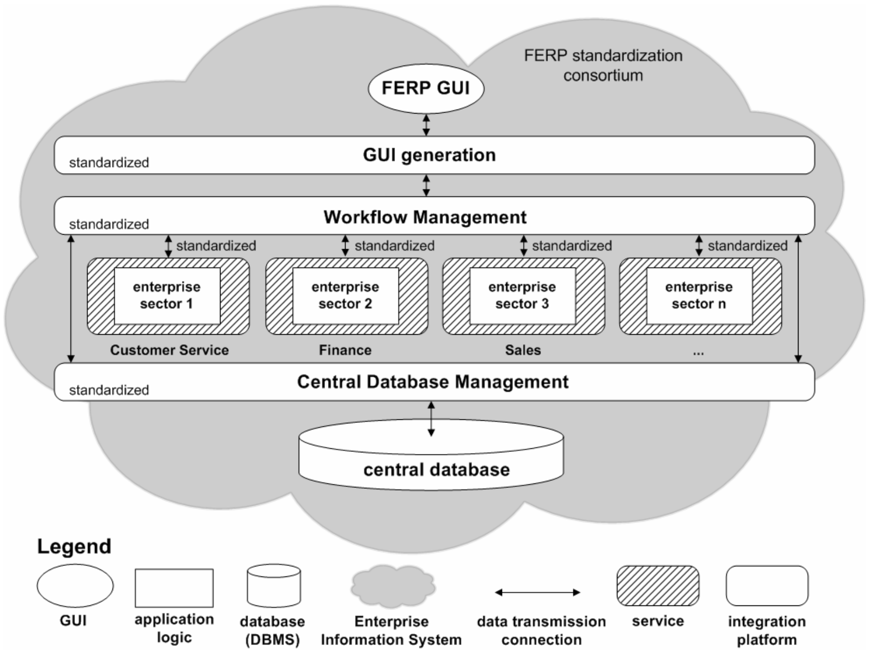


Figure 3: Vision of a Federated ERP system in practice

As shown in figure 3 FERP aims to closely connect various services which can be implemented by independent providers. This directly implies that the services must share common understanding of some important aspects, such as what they offer, what they assume, and how they handle sensitive information. Some aspects can be predefined in the architecture or system model and implemented, but many others need to be negotiated and agreed on during system construction, configuration, and operation.

4 Centralized architecture approach

Figure 4 gives a survey of the centralized architecture of a Web Service-based FERP system. The architecture consists of several subsystems which are interconnected. Because one of the main objectives of an FERP system is to integrate business components of different vendors, all components have to comply with standards. In this approach these standards are described as XML schema documents. In order to separate the three different layers of a typical layered architecture of conventional ERP systems each layer is assigned to its own standard. A more detailed description of the relationship between the components and the standard documents can be found in [1].

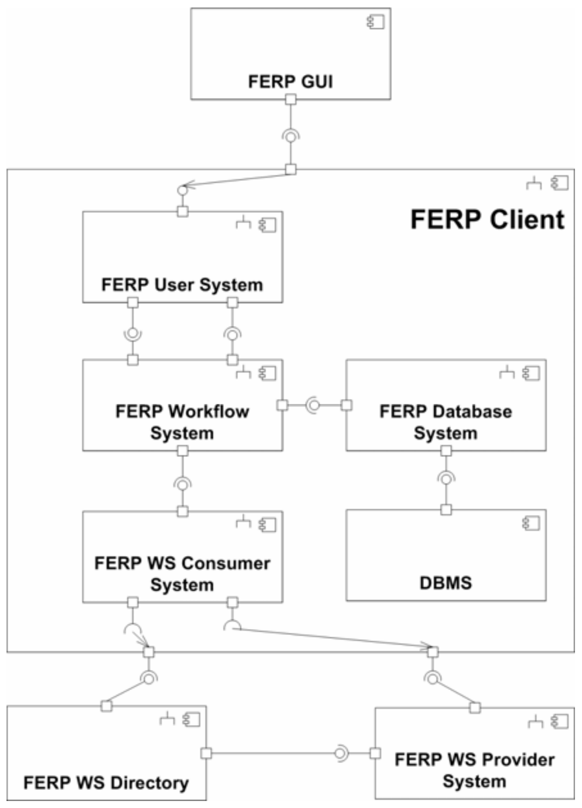


Figure 4: Centralized architecture approach of an FERP system with as UML 2.0 component diagram

The subsystems of the proposed architecture are the following:

FERP Workflow System (FWfS)
The FWfS coordinates all business processes which have to be described in an appropriate XML-based workflow language. A workflow in this context is a plan of sequentially or in parallel chained functions as working steps in the meaning of activities which lead to the creation or utilization of business benefits. Workflows implicitly contain the business logic of the overall system. The function types a workflow in FERP systems can consist of are the following:

- model based user interface functions, e.g. show, edit, select, control
- database access functions, e.g. read, update
- application tasks which are connected to Web Service calls

FERP User System (FUS)
The FUS is the subsystem which implements functions for the visualization of graphical elements and coordinates interactions with end users. This subsystem is able

to generate user screens at runtime. Screen descriptions which have to comply with the *FERP User Interface standard* are transformed to an end device-readable format, e.g. HTML in case of web browsers.

FERP Database System (FDS)
The FDS is the subsystem which implements functions for the communication with the FERP database. This subsystem is able to interpret XML structures which comply with the *FERP data standard*. The interface differentiates between two kinds of requests. Database update requests contain object oriented representations of business entities as XML trees. Database read requests contain XML expressions specifying portions of data to be extracted. In both cases the request parameters have to be transformed into different types of request statements that vary depending on the type of database management system (DBMS) which is used. Assumed that a relational DBMS (RDBMS) is used the underlying data model also has to comply with the FERP data standard which means that the corresponding table structure has to reflect the XML-Schema specifications respectively.

FERP Web Service Consumer System (FWCS)
The business logic of FERP systems is encapsulated in so called FERP business components which are wrapped by a Web Service. The FWCS is the subsystem which provides functions for the invocation of Web Services. All possible types of FERP Web Services are specified by the *FERP WS standard*. This standard contains XML schema definitions which describe Web Service operations as well as input and output messages. A Web Service references these types in its WSDL description. Furthermore this subsystem is able to search for Web Services which are defined by a unique identifier. By this it is possible that different Web Service providers implement the same business component type as Web Service. Beside the implementation of Web Service invocation and search functions this subsystem is responsible for the interpretation and consideration of non-functional parameters. Examples for those parameters are: security policies, payment polices or Quality of Service (QoS) requirements on the part of Web Service consumers. The architecture references the FERP pricing standard which is supposed to enable Web Service providers to specify Web Service invocation costs.

FERP Web Service Provider System (FWPS)
The FWPS is the subsystem which implements functions for the provision of Web Services which comply with the FERP WS Standard. The subsystem includes a Web Server which is responsible for the interpretation of incoming and outgoing HTTP requests which in turn encapsulate SOAP requests. The subsystem provides business components of the FERP system as Web Services. A connection to the *FERP Web Service Directory* allows the publication of Web Services. Furthermore this subsystem is responsible for the negotiation of common communication policies such as e.g. security protocols or usage fees with the requesting client.

FERP Web Service Directory (FWD)
The FWD provides an interface for the publication and the searching of FERP Web Services. The structure of this directory leans on the FERP WS standard. In this

standard Web Services are assigned to categories mirroring the predetermined functional organization of enterprises.

3 Hierarchical XML schema structure

The proposed architecture is dependent to the specification of different standards. The next paragraph focuses the standardization of FERP data types and Web Service operations in the context of FERP systems. Because of the complexity of enterprise data models and the difficulty to standardize a completed data model we propose a hierarchical standardization model which allows different abstraction levels. This model uses XML namespaces for the representation of hierarchical levels and XML schema documents for the definition of data types and their relationships. The reason for the usage of XML schema documents is their compatibility with Web Service Description Language (WSDL) which is the common standard for the description of Web Services and is already well supported by tools. The interoperability between FERP Web Services and the FDS is achieved by a transformation of XML schema-based data model descriptions to SQL-based data model descriptions. Web Service Descriptions in WSDL reference the FERP data standard by including the appropriate XML schema documents of the standard. In order to standardize the input and output messages of FERP Web Services we propose the usage of XML schema documents as well.

Figure 5 shows the hierarchical XML schema structure of an FERP system and shows the influence on the systems activities. The left hand side represents different enterprise sectors which are assigned to XML namespaces. This hierarchy can be compared to the internal structure of the application logic of conventional ERP systems which is often mirrored to the navigation structure of their GUI.

The upper half of figure 5 shows the relationships between XML schema documents and concrete Web Service descriptions. Standardized Web Service input and output messages (defined in *messageTypes.xsd*) build the basis for the standardization of Web Service types (described in *serviceTypes.wsdl*). The lower half of figure 5 shows the interactions between the different subsystems of the FERP system. The system internally creates a new XML schema document (*allTypes.xsd*) which includes a copy of all standardized data types that are used in process definitions. The system has a connection to the server of the FERP standardization consortium and will be notified in the case that the standard changed. Those changes are only allowed in terms of extensions. Thereby old versions will be supported during the whole lifetime of the standard.

The hierarchical structure provides a useful foundation for this requirement because it is already field-proved in the context of object oriented programming paradigms like polymorphism, generalization and specialization. The local XML schema representation will be transformed to a relational representation of the data model as SQL statement list. In addition to the schema transformation the FDS is able to transform SQL result sets to XML documents that comply with the FERP data standard in the case of *DATABASE_LOAD requests*. On the other hand XML documents will be transformed to SQL INSERT or UPDATE statements in the case of *DATABASE_STORE requests*. Both LOAD and STORE functions are provided by the FDS and can be used by the FWfS.

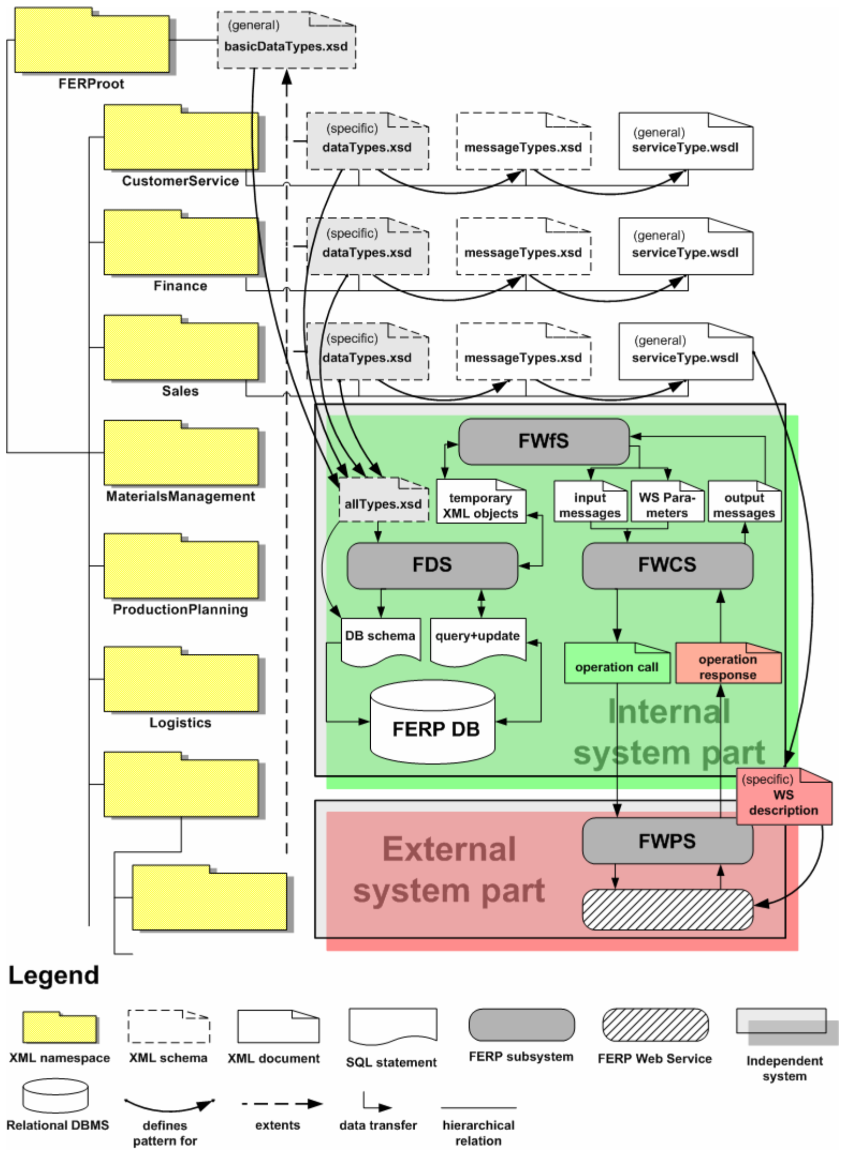


Figure 5: Hierarchical XML schema structure of an FERP system

Web Service calls are initiated by the FWfS as well (see figure 5). Therefore the FWfS sends a standardized XML representation of the appropriate input message to the FWCS. A second XML document contains configuration parameters which specify the concrete Web Service provider to be chosen by the FWCS. Those parameters include either a URL for a static Web Service call or requirements for a dynamic call like e.g. a maximum price. An alternative way for the specification of requirements for dynamic calls is a centralized mapping between Web Service types and requirements. Once the FWCS chose an appropriate Web Service provider it will repack this message to a *SOAP operation request* which includes the standardized name of the Web Service operation to be invoked. This request will be sent to the FWPS. After having finished the processing of the business logic the FWPS will return a SOAP operation response which includes a standardized response message. Figure 5 shows how this response message is going to be sent back to the FWfS that primarily initiated the Web Service call.

6 Example process

Figure 6 shows an example process model which is described in YAWL (*Yet Another Workflow Language*). In this process model tasks are assigned to one of the three following function types:

- Database communication (in figure 6 indicated as DB-task)
- End-user communication (in figure 6 indicated as GUI-task)
- Web Service communication (in figure 6 indicated as WS-task)

All other symbols comply with the graphical notation of YAWL. The example process model demonstrates a workflow for the display of the top-five customers. The example includes only one Web Service call which is responsible for the creation of a diagram. The Web Service receives the all contracts of all customers. Having finished the comparison the Web Service returns a diagram as Base-64-encoded PNG-file. The next workflow task visualizes this diagram.

The tasks in figure 6 refer to the communication with the connected subsystems. The interactions between these components are based on standardized communication languages. A workflow designer is supposed to refer to these standards in order to ensure the compatibility between workflow definitions and implementations of FERP client systems. The architecture of a FERP client system is shown in figure 4 and has to be considered as reference architecture. The standardization process has to cover the specification of standardized reference interfaces which are provided by these components and an inclusion of workflow task types which are directly associated to the invocation of interface operations. Furthermore the standardization of interaction protocols (by means of languages for the specification of commands to be executed) has to be done.

Figure 7 shows a screenshot of the graphical user interface of a first FERP system prototype (FERPxONE) which has been developed in order to verify the practicability of the proposed architecture approach. The left hand side represents the entry point for the navigation through a catalogue of workflows which have been registered in the FWfS and which can be started by a user. The right hand side shows the list of active tasks a user might select for completion. In the middle all user-relevant process outputs are shown. In this example the input form of the first workflow task of figure 6 is presented to the user.

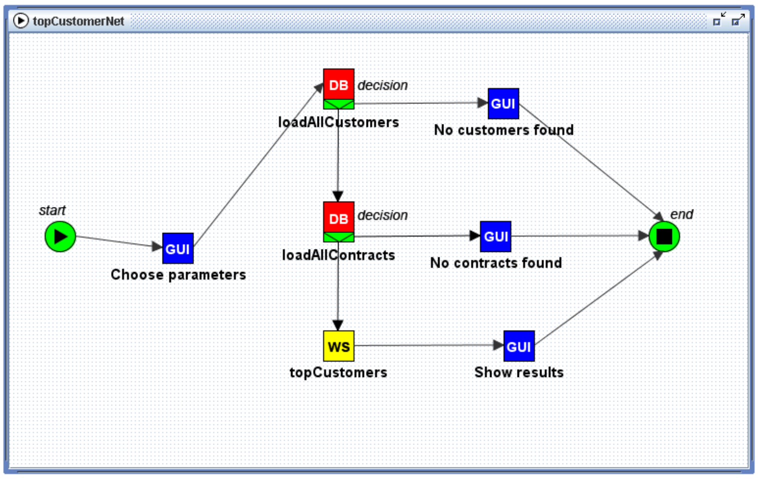
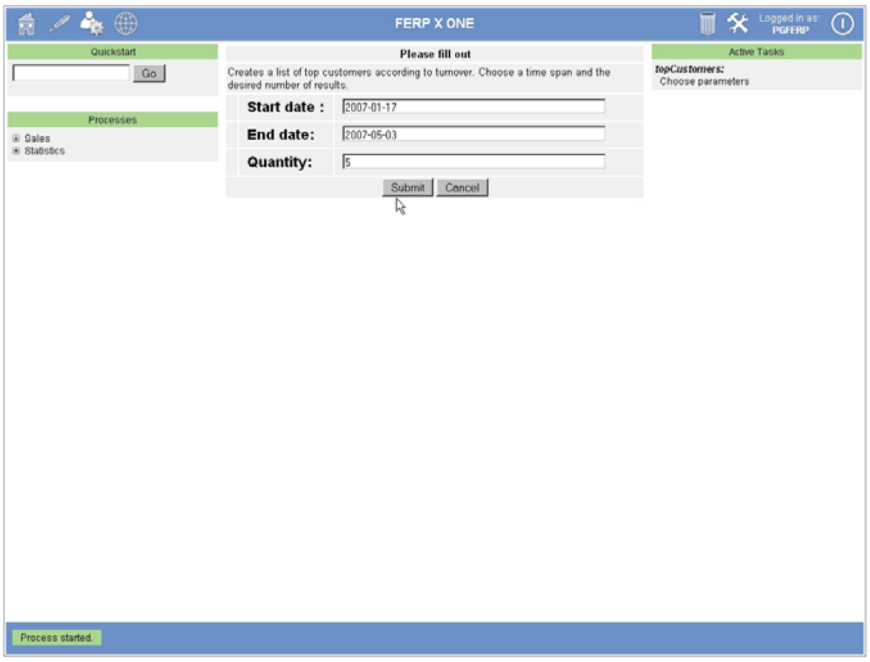


Figure 6: Process model in YAWL as simplified example for the display of the top-five customers



Please fill out

Creates a list of top customers according to turnover. Choose a time span and the desired number of results.

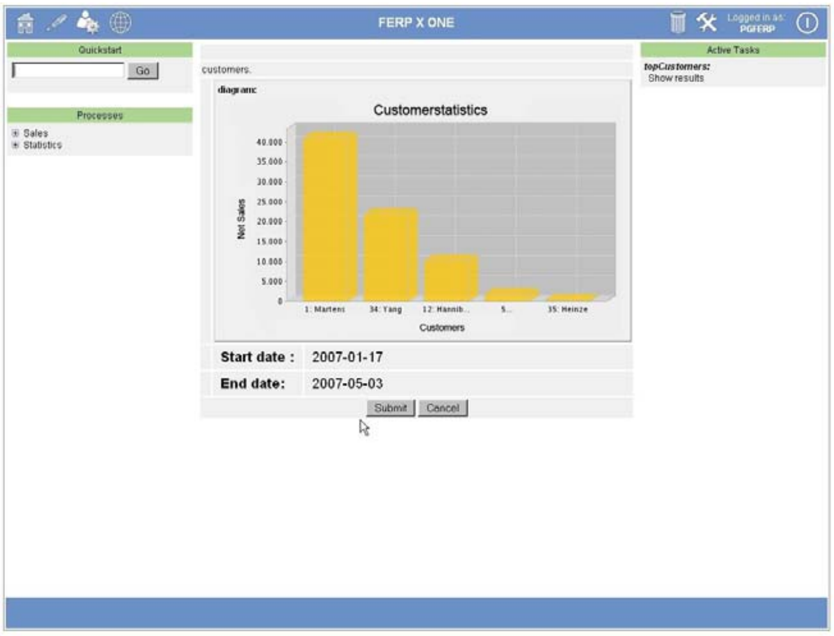
Start date :	2007-01-17
End date:	2007-05-03
Quantity:	5

topCustomers: Choose parameters

Figure 7: GUI view Parameter input

The necessary parameters in this case are a time period which marks the start and end dates of customer contracts and the number of top-customers which have to be shown in the opposing diagram. The next two workflow tasks will request the respective contracts by communicating with the FDS (see figure 6).The FDS will return an XML document which contains a list of all customer contracts.

The next workflow task encapsulates all relevant parameters[3] in a web service operation call where the standardized Web Service type is referenced and sends it to the FWCS. The FWCS sends a UDDI search request the FWD by submitting the Web Service type as name of the Web Service to be found. The FWD returns a list of references to Web Service descriptions as URLs. These descriptions include specific information about the price of operations calls as well as a reference to a WSDL document. The FWCS chooses one of the Web Services on the basis of a price comparison and sends an invocation request to the Web Service. The response message includes content of a PNG-File as Base64-encoded String which is decoded and stored in the FUS. Figure 8 shows how the result is presented to the user in the next step.



Start date : 2007-01-17

End date: 2007-05-03

Figure 8: Web Service result

7 Conclusions and Outlook

Comparing distributed ERP systems and ERP systems running on only one computer, the distributed systems offer a lot of advantages. Particularly small- and

[3] Relevant parameters are the Web Service type as string, the operation name and an information object which includes the input parameters for the Web Service invocation.

medium sized Enterprises (SMB) benefit from using shared resources. The example in paragraph 6 shows how the system is able to react on market changes by comparing the prices of Web Service operation calls. However, the design of distributed system architectures is subject to a number of problems [2, 3]. The paper addresses the problem of redundant data in business application systems of independent vendors presents a basis for the standardization of ERP system components that are provided as Web Services. A standardized data model builds the basis for message and service standardization. The hierarchical structure of the presented standard advances the reuse of existing data types. Furthermore we presented a reference architecture of FERP systems which reduces the necessity of data transformation functions in business process descriptions. The standardization of the syntactic level is only the first step. Semantic, Behaviour, synchronization and quality of Web Services must flow into the definition of an overall ERP system standard. The future work must pick up these problems to realize the vision of a loosely coupled ERP system which allows the dynamic outsourcing of applications [5, 7] and the combination of software components of different providers.

References

1. Brehm, N., Marx Gómez, J.: Web Service-based specification and implementation of functional components in Federated ERP-Systems. In: Abramowicz, W. (ed.): 10th International Conference on Business Information Systems. Springer, Poznan, Poland (2007) 133-146
2. Brehm, N., Marx Gómez, J.: Secure Web Service-based resource sharing in ERP networks. International Journal on Information Privacy and Security (JIPS) 1 (2005) 29-48
3. Brehm, N., Marx Gómez, J.: Standardization approach for Federated ERP systems based on Web Services. 1st International Workshop on Engineering Service Compositions, Amsterdam (2005)
4. Brehm, N., Marx Gómez, J.: Federated ERP-Systems on the basis of Web Services and P2P networks. International Journal of Information Technology and Management (IJITM) (2007)
5. Brehm, N., Marx Gómez, J., Rautenstrauch, C.: An ERP solution based on web services and peer-to-peer networks for small and medium enterprises. International Journal of Information Systems and Change Management (IJISCM) 1 (2005) 99-111
6. Cuomo, G.: IBM SOA "on the Edge". ACM SIGMOD international conference on Management of data. ACM Press, Baltimore, Maryland (2005) 840-843
7. Dan, A., Davis, D., Kearney, R., Keller, A., King, R., Kuebler, D., Ludwig, H., Polan, M., Speitzer, M., Youssef, A.: Web services on demand: WSLA-driven automated management. IBM SYSTEMS JOURNAL 43 (2004) 136-158
8. Vogt, C.: Intractable ERP: a comprehensive analysis of failed enterprise-resource-planning projects. Software Engineering Notes 27 (2002) 62-68