REAlist: Towards a Business Model Adapting Multi-Tenant ERP System in the Cloud

Dieter Mayrhofer[1], Alexandra Mazak[1], Bernhard Wally[1], Christian Huemer[1], Peter Regatschnig[2]

[1]TU Vienna, [2]eventus Marketingservice GmbH
[1]{mayrhofer, mazak, wally, huemer}@big.tuwien.ac.at,
[2]peter.regatschnig@eventus.at

Abstract. Enterprise Resource Planning (ERP) systems are nowadays adapted to the company-specific requirements by customization. Whereas smaller and general feature changes are performed quickly by parameterization, changes of a company's business activities may lead to enormous changes in the system's code. To overcome this problem, we introduce an approach of a flexible ERP system which can be adapted by specific business models for each individual company/tenant by using model engineering techniques. Our approach—*REAlist*—is built on top of a generic data structure based on the Resource Event Agent (REA) business modeling ontology and provided as a cloud based service. Business models for each company are described by a domain specific language called the REA-DSL. REAlist enables companies to incorporate business activity changes quickly and without extensive code changes.

1 Introduction

In the last five years cloud computing led to a paradigm change in IT industry. The cloud enables to deliver computation time, data storage, network capacity, or complete software. In our approach, we address the latter one also known as Software-as-a-Service (SaaS). In a SaaS solution applications are centrally maintained and can be used directly over the Internet. Furthermore it promises high availability as well as central automatic backups. The customer always gets the latest software version and can concentrate on using the software instead of maintenance issues.

As for any kind of software, also an ERP system can be provided through the cloud. Instead of delivering each customer his/her own instance of an ERP system, one instance is centrally hosted in the cloud for all customers together. Since all customers use the same instance, the ERP system needs to be a multi-tenant enabled system.

A central aspect of an ERP system is the internal process support as well as the industry neutral approach. Even though the business processes of various companies can look similar on an abstract level and are built upon the same core concepts, the specific detailed requirements differ a lot. Consequently, an individual and quick adaptation based on individual requirements needs to be done by customization. Traditionally, this can be realized by parameterizing or programming. Therefore, we introduce a generic and dynamic approach for company specific adaptations based on runtime configurable business models. Another aspect of this project is the multi-tenant capability of the system. Although for each individual client user interfaces are provided with individual characteristics, a common generic database is used.

For the purpose of company-specific adjustments we follow a model-driven approach. We describe standard business cases as reference models by using a graphical modeling language. In addition, company-specific adjustments can be made to the reference models with this modeling language. Based on these models, user interface and database mappings can be generated. The graphical modeling language is based on the REA (Resource-Event-Agent) ontology. REA is a universal language that enables clear communication and a common understanding for all stakeholders involved in the software development process.

The project name REAlist on the one hand reflects the reference to the REA ontology and on the other hand, it refers to the fact that reports (e.g., account balance lists) can be generated on the fly. We emphasize that we are *REAlistic* in that we do not aim for a fully featured ERP system. In a nutshell, our focus is on an architecture for an ERP system based on the REA ontology by making use of model-driven approaches and keeping multi-tenant SaaS challenges under consideration.

2 Problem Description

Business applications are usually summarized under the term Enterprise Resource Planning system (ERP system). An ERP system is used to support all business processes in an enterprise. Therefore, most business applications (modules) are connected by a common data base [1]. However, this data structure is not based on a common business modeling ontology. Typical modules are financial accounting and controlling, production planning, purchasing, logistics, sales, distribution, as well as human resource planning [1].

In general, ERP applications are limited to the essential business-specific processes in selected industries (e.g., automotive, manufacturing). Companies are still not capable of using a flexible system for adapting their business processes which can quickly change according to market demands. As mentioned, many of the ERP systems available on the market do not meet this requirement. Since, they are not based on model-driven approaches, they are too static. Customer-specific requirements, such as the integration of new business models, require a big change in the code and consequently changes in the underlying data structure. This can lead to inconsistencies and additionally hinder the analysis and traceability of data.

The principle of parameterization in traditional ERP systems is based on selection by which predefined functions and processes are offered to customers. With the help of parameterization (as an option of customizing) parts of the standard software are enabled or disabled by setting certain parameters [2, 3]. Starting from a comprehensive general model in the standard software, a concrete version of the standard software for a particular company is created through a top-down approach [4]. Many of the parameters are controlled by means of several thousands of database tables, which are queried at run time. Such an approach is very complex and it is difficult for user to understand the dependencies among those tables. Errors in the adaptation of predefined processes also lead to costly follow-up costs. Furthermore, the adaptations require employees with experiences in this field. The briefly shown customizing problems of ERP systems can be exemplified by the example of the ERP system Navision Microsoft Dynamics NAV[1]: Adjustments through expansion by code fragments in the ERP system for a medium sized Austrian company, which already consumed 10

[1] Formerly Navision

person-years of development, needed to be individually tested and debugged again. Another problem occurs when the predefined customizing functions do not fit the customer's requirements. In such cases, standard programs provide so called extension points. They enable developers to add customer-specific program code into the standard software. However, an integration using many different interfaces creates additional dependencies that are difficult to manage. Thus, the complexity of the system increases dramatically.

Despite the comprehensive ERP systems available on the market companies are far from to handle the flexibility of "daily business", even though it is more important than ever to rapidly and flexibly adapt business models to new needs. This fact is also supported by a study from 2011, conducted for the ERP vendor Intact Software[2] by the international market and opinion research institute YouGov.

3 Technical Goals

The objective of the REAlist project is the iterative prototypical realization of a generic and dynamic adaptable ERP system in the cloud based on the REA data structure, we have already created in former work [5]. In the REAlist-approach, business data, business models, reference models, as well as policies are stored in a single database—the REA-DB. This solution enables us to create a multi-tenant ERP system. The various desired functions of REAlist are depicted in Figure 1.

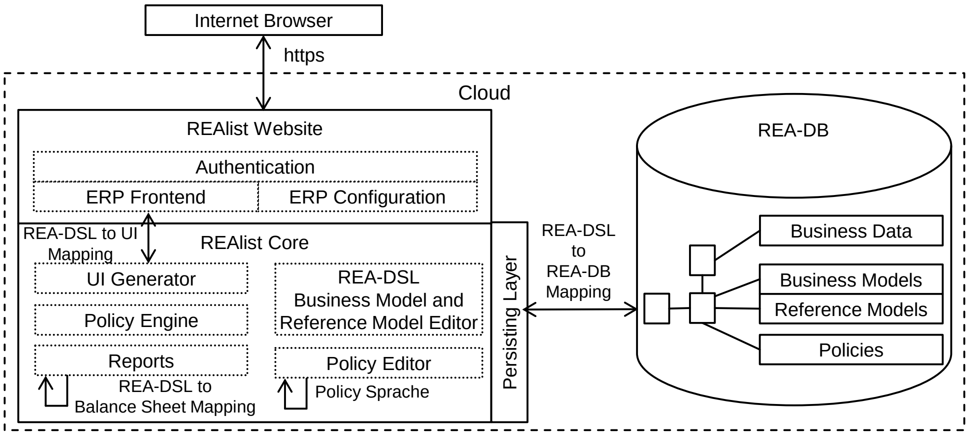

Fig. 1. Realist functionality

Based on this solution, customers are able to access their customized ERP system and record the current business case data through a dynamic user interface. The features and customizations of the ERP system are controlled by business models. These models can be created and changed with the business and reference model editor easily by the users (e.g., customers, vendor) themselves. Additionally, we provide a graphical domain-specific language to ensure a intuitive creation of business models. This language, the REA-DSL bases on the dissertation of Dieter Mayrhofer [6]. Furthermore, we provide predefined reference models to prevent users from constantly starting from scratch. They can be flexibly adapted for the creation of new business models. There are different approaches to modify reference models as described in [7, 8].

[2] http://www.intactsoftware.co.uk/erp-survey-results.html

Generally, business transactions are subjected to certain policies (e.g., tax rules, corporate rules) they have to adhered to. Depending on the individual context of business transactions or events, certain rules have to be applied. For example, sale events performed by an Austrian company to its customers are subjected to sales tax depending on the Austrian value added tax act (UStG 1994). In addition, there are internal policies introduced by the company itself (e.g., certain customer benefits). For example, if the volume of sale of a customer is achieved, the company grants a special bonus. One part of the REAlist project is to find a user-friendly way for defining such rules by using a policy editor. For this purpose, we create a policy language that can be processed automatically by the ERP policy engine. This means that new policies do not require any code changes in the ERP system. Our special attention is about ensuring that any REA objects and properties can be referenced by this policy language. Therefore, we consider it useful to place policies on the same level as the REA business models and store them in the REA-DB.

Another aim of the project is to create meaningful analyzes and statistics from the business data based on the REA event-driven data structure. We want to provide customers advantages over existing ERP systems, which only display historical data and therefore provide only limited conclusions on individual business cases. By using REAlist customers are able to retrace which payments have been made for which activities (events), as well as on which activities the balance sheets are based on. This means that the process behind the accounting ratio can be fully traced. Furthermore, we want to provide benchmarks and key performance indicators (KPIs) based on the REA event data. In a future step, it is even conceivable that balance sheets can be automatically generated at any time.

4 Conclusion

In the REAlist project, we are following the model engineering methodology to enable customizing for the ERP system by using REA as a generic data structure. We apply business models and reference models based on the REA-DSL for defining customizations and we create a policy language to make it able for customers to define certain rules and restrictions which can then automatically be executed.

5 Acknowledgements

This work is supported as part of the BRIDGE program of the Austrian Research Promotion Agency (FFG) under grant number 841287—a joint research effort of the Vienna University of Technology and the eventus Marketingservice GmbH.

References

1. Gadatsch, A.: Geschftsprozess-Management. Volume 4th edition. (2005)
2. Gronau, N.: Industrielle Standardsoftware Auswahl und Einfhrung. (2001)
3. Leweng, H.P., Lanninger, V., Thome, R.: Betriebliche Standardanwendungssoftware. In: Das Wirtschaftsstudium (WISU). Volume 2. (2004) 219227
4. Wedeking, H., Gnzel, H.: Top-Down vs. Buttom-upAdaption von betriebswirtschaftlicher Standardsoftware. In: Datenbanksysteme in Bro, Technik und Wissenschaft (BTW), GI-Fachtagung. Springer (1999) 449–455

5. Mayrhofer, D., Huemer, C., Regatschnig, P.: REA-ERP: Challenges of using REA in an ERP System. In: 7th International Workshop on Value Modeling and Business Ontology (VMBO 2013). (2013) 1–4
6. Mayrhofer, D.: Business Model Driven Data Engineering. PhD thesis, Vienna University of Technology (2012)
7. vom Brocke, J.: Design Principles for Reference Modeling - Reusing Information Models by Means of Aggregation, Specialisation, Instantiation, and Analogy. In: Reference Modeling for Business Systems Analysis. Idea (2007) 47–76
8. Hofreiter, B., Huemer, C., Kappel, G., Mayrhofer, D., vom Brocke, J.: Inter-organizational Reference Models - May Inter-organizational Systems Profit from Reference Modeling? In: both, Business System Management and Engineering in Springer LNCS volume 7350 and Proceedings of the International Workshop on Business System Management and Engineering (BSME 2010), in conjunction with the 48th International Conference on Objects, Models, Components, Patterns (TOOLS 2010), Malaga, Spain, Springer (2012) 1–16