PERGAMON

Information Systems 28 (2003) 673–690

www.elsevier.com/locate/infosys

ERP modeling: a comprehensive approach

Pnina Soffer, Boaz Golany*, Dov Dori

Faculty of Industrial Engineering and Management, Technion-Israel Institute of Technology, Technion city, Haifa 32000, Israel

Received 20 February 2002; accepted 18 October 2002

Abstract

We present a generic reverse engineering process, aimed at developing a model that captures the available alternatives at different application levels of an Enterprise Resource Planning (ERP) system. Such a model is needed when ERP systems are aligned with the needs of the enterprise in which they are implemented. In order to support the ERP implementation process, the model should describe the entire scope of the ERP system's functionality and the alternative business processes it supports, as well as the interdependencies among them. We analyze the desired properties a modeling language should satisfy to be applied in constructing an ERP system model. This analysis, which follows the Cooperative Requirements Engineering With Scenarios classification framework in its adapted ERP modeling form, results in a set of criteria for evaluating modeling languages for this purpose. Using these criteria, we evaluate the Object–Process modeling Methodology and apply it for generating a detailed ERP system model. The generic process and detailed criteria we develop can serve for comprehensive ERP modeling, as well as for obtaining a model of other process-supportive off-the-shelf systems that are of generic and configurable nature.

Keywords: Enterprise resource planning; Reverse engineering; Object–process methodology; Modeling languages

1. Introduction

Enterprise Resource Planning (ERP) systems are off-the-shelf software packages that support most of the key functions of an enterprise, such as logistics, sales, and financial management. These systems are generic, and the functionality they provide can serve a large variety of enterprises.

The implementation of an ERP system involves a process of customizing the generic package and aligning it with the specific needs of the enterprise. The alignment process simultaneously defines the software configuration and the enterprise business solution. Due to the need to adapt the enterprise to the software package rather than the other way around, it often results in redesigned business processes [1,2]. Enhancing the system's functionality through software customizations, although not desired [3], is sometimes required. The business process alignment, affected by various environmental aspects, such as existing information systems prior to the ERP implementation [2] and organizational culture [4], bears crucial consequences on the success of the ERP implementation project and on the future business practice of the enterprise [5].

*Corresponding author. Tel.: +972-4-8294512; fax: +972-4-832-5194.

E-mail addresses: pnina@tx.technion.ac.il (P. Soffer), golany@ie.technion.ac.il (B. Golany), dori@ie.technion.ac.il (D. Dori).

PII: S0306-4379(02)00078-9

This work is motivated by the need to provide support for the alignment process in enterprises whose business processes are unique and form their competitive edge. Such enterprises do not necessarily wish to standardize all their processes due to ERP implementation, as is often the case. Common tools that support the alignment process refer to predefined "best practice" models and configurations [6–8], and therefore do not support the needs of these enterprises. Preserving their unique processes may require these enterprises to make software customizations in the ERP system, and take risks of a long implementation time and high costs for maintenance and upgrades in the future [3]. However, the functionality of ERP systems is in many cases rich enough and capable of supporting business processes that are not included in the "best practice" solutions.

A requirement-driven alignment approach [9–11] enables the enterprise to identify the ERP configuration that satisfies stated requirements that do not necessarily match any predefined "best practice" solution, and the gaps—the requirements that are not satisfied by the system.

Such an approach matches specifications of the enterprise requirements with a model, specifying the ERP system capabilities. This requires the model of the ERP system capabilities to represent the entire scope of options available in the ERP system in a manner that enables matching with the enterprise requirements. A common basis of semantics and representation is needed for both the ERP system model and the enterprise requirements.

The enterprise requirements, discussed in detail in [11], are obtained through a requirements engineering process and represented formally in order to be matched with the ERP system capabilities. Some ERP systems, such as SAP and Baan, have embedded enterprise models, which represent their capabilities and "best practice" solutions [6,8]. The model embedded in an ERP system may serve as a basis for matching the system with the requirements. Thus, it makes sense to represent the requirements using the same modeling approach. However, some ERP systems have no embedded business model, and others apply different modeling conventions, addressing different aspects of the enterprise. The Architecture of Integrated Information Systems (ARIS) [6,12], applied in the reference models of SAP, incorporates five representation views. These include a function view, decomposing activities in a top-down manner, a business process view, represented as Event-driven Process Chains (EPC), a resource view, representing organizational units and other resources, a data view represented by Entity-Relationship Diagrams (ERDs), and an output view, representing physical inputs and outputs. The Dynamic Enterprise Modeling (DEM) [8], applied in the Baan reference models, incorporates a business control view, which represents business functions, their structure and interaction, an organizational structure view, an enterprise structure view, showing geographically distributed inner supply chain, and a business process view, represented as Petri Nets [8].

Due to the lack of a standard modeling language, the enterprise requirements must be represented differently when implementing different ERP systems in order to be matched with the ERP system capabilities. Furthermore, in order to be matched, the issues addressed by the enterprise requirements must correspond to the issues addressed by the modeling method applied in the package, which is being implemented. This does not make much sense, since the issues addressed by the enterprise requirements are independent of the specific ERP package implemented.

Rather than relying on the models embedded in the various ERP packages, our goal was to establish a proper ERP model and determine what modeling language is most appropriate for representing the ERP system capabilities to be matched with the enterprise requirements. For this purpose the desired model should represent business processes and their corresponding underlying information objects [11], and capture the entire scope of alternative options provided by the ERP system and their interdependencies.

Constructing an abstract model of an existing system is a process of reverse engineering. Reverse engineering processes are generally composed of three main activities: restructuring, comprehension, and production of an abstract model of the system under investigation [13]. Restructuring,