The Journal of Systems and Software 85 (2012) 607–625

Contents lists available at SciVerse ScienceDirect

The Journal of Systems and Software

journal homepage: www.elsevier.com/locate/jss

Controversy Corner

Automatic execution of business process models: Exploiting the benefits of Model-driven Engineering approaches

J. Fabra[a,*], V. De Castro[b], P. Álvarez[a], E. Marcos[b]

[a] Department of Computer Science and Systems Engineering, University of Zaragoza, María de Luna 1, E-50018 Zaragoza, Spain
[b] Kybele Research Group, Rey Juan Carlos University, Tulipán S/N, E-28933 Móstoles, Madrid, Spain

ARTICLE INFO

Article history:
Received 4 May 2010
Received in revised form 15 July 2011
Accepted 12 September 2011
Available online 12 October 2011

Keywords:
Business processes
Model-driven Engineering
Model transformation
Model execution
Service-oriented development

ABSTRACT

The business goals of an enterprise process are traced to business process models with the aim of being carried out during the execution stage. The automatic translation from these models to fully executable code which can be simulated and round-trip engineered is still an open challenge in the Business Process Management field. Model-driven Engineering has proposed a set of methodologies with which to solve the existing gap between business analysts and software developers, but the expected results have not as yet been achieved. In this paper, a new approach to solve this challenge is proposed. This approach is based on the integration of SOD-M, a model-driven method for the development of service-oriented systems, and DENEB, a platform for the development and execution of flexible business processes. SOD-M provides business analysts with a methodology that can be used to transform their business goals into composition service models, a type of model that represents business processes. The use of the Eclipse Modelling Framework and the ATLAS Transformation Language allows this model to be automatically transformed into a DENEB workflow model, resulting in a business process that is coded by a class of high-level Petri-nets and is directly executable in DENEB. The application of the proposal presented herein is illustrated by means of a real system related to the management of medical images.

1. Introduction

In recent years, the rapid development in the field of Web service technologies and Service-Oriented Computing (SOC) has motivated an interest in the area of Business Process Management (BPM) (Watson, 2008). BPM has broadened to become a set of technologies and standards for the design, execution, administration and monitoring of business processes, and also as a means to deal with frequent changes in business and value chains (Watson, 2008; Havey, 2005). In conceptual terms, a business process is a defined set of activities that represents the steps required to achieve a business objective (OMG, 2011). In BPM terminology, an activity can be seen as the work of a person, an internal system, a service provided by an external entity or the process of a partner company.

The motivation in BPM and Service Oriented Architectures (SOA) areas has also provoked the emergence of several languages for the design and implementation of such processes. The Web Services Business Process Execution Language, WS-BPEL (BPEL, 2011), is a standard language widely used for the specification of executable business processes. However, WS-BPEL and most of the current existing approaches for the specification of processes have some relevant problems: it is difficult for business analysts to use them in the early stages of the development process (Verner, 2004) (analysis and modelling stages), their dependence on a specific implementation technology (more specifically, on the Web service technology) and, finally, the lack of formal semantics that permit process analysis. These problems increase the gap between business analysts and software developers, which represents a serious limitation in the BPM field.

A new beta version of the Business Process Model and Notation standard, BPMN 2.0 (OMG, 2011), has recently been published as a future means to reduce this gap. The informal formalization of the execution semantics for all BPMN elements and the independence of specific implementation technologies proposed by the specification will solve the first two problems associated with previous approaches. Nevertheless, this promising proposal lacks formal semantics and needs time to mature. The aforementioned gap is, therefore, still open.

In order to overcome these limitations, the Web Engineering field has proposed a set of methodologies which make it easier to develop service-oriented systems based on current technologies. Model-driven Development (MDD) is a development methodology which is principally characterised by the use of models as a product (Selic, 2003). MDD is a subset of Model-driven Engineering (MDE) (Schmidt, 2006) because, as the E in MDE suggests, MDE

* Corresponding author. Tel.: +34 976 762 874; fax: +34 976 761 914.
E-mail addresses: jfabra@unizar.es (J. Fabra), valeria.decastro@urjc.es (V. De Castro), alvaper@unizar.es (P. Álvarez), esperanza.marcos@urjc.es (E. Marcos).

0164-1212/$ – see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.jss.2011.09.022

goes beyond pure development activities and encompasses the other model-based tasks of a complete software engineering process (Ameller, 2009). Model-driven Architecture (MDA) (Miller and Mukerji, 2003) is the particular MDD methodology defined by the Object Management Group (OMG) and therefore relies on the use of OMG standards. More specifically, MDA offers an open, vendor-neutral approach with which to master technological changes and provides a conceptual structure for: (i) specifying a system independently of the platform that supports it; (ii) specifying software platforms; (iii) choosing a particular platform for the system; and (iv) transforming the system specification into the corresponding code for a particular target platform (that is, moving the system specification to the technological one).

Two fundamental ideas are behind the Model-driven Development of software systems. First, MDD methodologies provide a set of models with which to cover the different stages involved in the design of a system. These models are consistent with each other and include traceability between different abstraction levels of the design. Second, the models are explicitly separated from implementation technologies and execution platform details. This last characteristic allows developers to reuse software components and thereby avoids the need for repeated investments in time and money. Both ideas have been used to define Model-driven Architecture (MDA) approaches which can specify the functionality of each type of software systems (Mellor et al., 2003): MDA for service-oriented information systems, for business processes, for embedded systems, etc.

However, after several years of the popularization of MDA proposals and the appearance of many MDD approaches for software development, the usefulness of these approaches is questionable (Selic, 2003). The main controversy is related to the lack of support in the automatic generation of code from models. Ideally, technological frameworks should be able to transform models into executable code and then execute the resulting code without human intervention (Selic, 2008). This ability has an influence on several stages of the lifecycle of a software system, principally with regard to its development and in the analysis prior to its deployment. Automatic code generation has traditionally been restricted to the generation of code skeletons and fragments, requiring human intervention to complete them. In order to overcome the limitation imposed by this kind of automatization, modeling and implementation languages should converge to facilitate the generation of complete programs. Moreover, the code generated could also be used for the automatic verification of models on a computer. This signifies checking that the requirements fulfill the business goals and that there are no undesirable properties in the models. From an empirical point of view, model verification can be based on the simulation of generated programs (Selic, 2003). Nevertheless, more powerful and formal analysis methods and techniques should be integrated into MDA approaches to verify a system's model (model checking or logical inference, for instance). Code generation therefore emerges as a key element if the usefulness of MDA solutions is to be achieved in different stages of the development of a software system.

This paper presents the integration of the Service-Oriented Development Method, SOD-M (De Castro et al., 2009), and the platform for the Development and Execution of iNteroperable dynamic wEB processes, DENEB (Fabra et al., 2011). From the point of view of MDD, SOD-M defines a model-driven method for the development of service-oriented information systems. In this method, the key concept is the use of composition service models (which are equivalent to business process models) to represent the Platform-independent Model (PIM) level of a system. On the other hand, from an execution point of view, DENEB provides an environment for the deployment and execution of business processes. Our integration proposal therefore consists of automatically translating SOD-M composition models into business processes that are directly executable by DENEB. This solution solves the gap between business analysts (their business goals have been represented in SOD-M models) and software developers (their implementations of models are automatically generated). With regard to existing approaches our proposal presents three relevant contributions in connection with the resulting business processes: firstly, they are completely generated without human intervention; secondly, they can integrate a wide variety of implementation technologies (not only Web services); and, finally, they have formal semantics (a class of high-level Petri-nets has been used to code DENEB processes), and could therefore be analyzed by using formal analysis techniques (Ibáñez et al., 2008a, 2008b). From a technological point of view, both the model transformation and the code generation utilities required by our proposal have been developed using the results of projects supported by Eclipse (Eclipse, 2011), the ATLAS Transformation Language (ATL) (Jouault and Kurtev, 2005) and the Eclipse Modeling Framework (EMF) (Biermann et al., 2008).

The remainder of this paper is structured as follows. Section 2 presents the main work related to the BPM modeling area, the execution of business processes and code generation from business process models. In Section 3, the foundations of the two approaches to be unified, SOD-M and DENEB, are introduced. The proposed model transformation process, the metamodels involved and the plug-in implementations are then sketched in Section 4, in which the process of moving from the composition service model in SOD-M to the workflow model in DENEB is depicted by first presenting the model implementation for both the SOD-M and DENEB approaches. The EMF pluging implementation, the model transformation and the mapping implementation using Ecore and ATL are then described. The application of the proposal is illustrated in Section 5 by means of a real use case related to the management of medical images. Finally, Section 6 concludes the paper and shows the directions of our future work.

2. Background

In this section the role of MDA in the BPM–SOA combination is presented, and the key points in which MDA helps and improves in the whole process are described. Several BPM standards have been proposed for their application to the different levels of MDA in order to enhance its benefits. The most relevant ones, and the most important transformation approaches to generate executable business process models, are also depicted here.

2.1. The role of MDA in the BPM and SOA combination

Service-oriented development has recently become one of the major research topics in the field of software engineering, which has even led to the appearance of a new and emerging discipline called Service Engineering (SE). Service Engineering aims to bring together the benefits of SOA and BPM initiatives. The BPM and SOA combination has been proposed as the best approach by which to achieve a closed alignment between business processes and IT resources. This combination allows the rapid development of agile processes which can respond to changes in business requirements.

However, BPM and SOA represent two different initiatives. BPM is mainly a management discipline and strategy which endorses the idea that a business can be modelled in terms of its end-to-end processes which cut across organizational and system boundaries. These processes are then represented in a manner that computers can understand and process (Brunswick, 2008; Frye, 2006). On the other hand, SOA looks for a better business process alignment with service protocols, legacy applications and software components, this is, it comprises an organizational paradigm which is supported