A VISION FOR AGILE MODEL-DRIVEN ENTERPRISE INFORMATION SYSTEMS

N. R. T. P. van Beest, N. B. Szirbik and J. C. Wortmann
Department of Business & ICT, Faculty of Economics and Business, University of Groningen
Landleven 5, P.O. Box 800, 9700 AV Groningen, The Netherlands

Keywords: Enterprise information system, Model-driven development, Flexibility, Agility, Workflow, ERP.

Abstract: Current model-driven techniques claim to be able to generate Enterprise Information Systems (EISs) based on enterprise models. However, these techniques still lack – after the initial deployment – the long-time desired flexibility, which allows that a change in the model can be immediately and easily reflected in the EIS. Interdependencies between models are insufficiently managed, requiring a large amount of human intervention to achieve and maintain consistency between models and the EIS. In this position paper a vision is presented, which describes how model-driven change of EISs should be structured in a coherent framework that allows for monitoring of interdependencies during model-driven change. Therefore, proposing fully automated consistency and pattern checks, the presented agile model-driven framework will reduce the amount of required human interventions during change. As a result, the cost and time span of model-driven EIS change can be reduced, thereby improving organizational agility.

1 INTRODUCTION

This is a position paper, which presents a vision of how model-driven change of Enterprise Information Systems (EISs) – after the initial deployment – should be structured in a coherent framework. EISs are interactive systems that offer support for certain parts of the workload of employees and comprise a vast array of functionality. Depending on the industry context, two of the most typical categories of EISs are realized via the deployment of Enterprise Resource Planning (ERP) systems and Workflow Management (WfM) systems (Van der Aalst et al., 2002; Cardoso et al., 2004; Olson, 2004).

However, when both an ERP and a WfM system are used in the same organization to support processes that have overlapping areas or are related via data exchanges and task triggers, the flexibility of the WfM system could be lost and changes can become extremely costly in terms of budget and manpower (Reijers, 2006).

In an attempt to increase flexibility, while reducing development, implementation and integration efforts, model-driven development (MDD) emerged as a possible solution using the Model-Driven Architecture (MDA) (Kleppe et al., 2003; OMG, 2003). Nowadays, MDD approaches exist, which enable a partial automatic generation of EISs based on models of the organization (OMG, 2003; Atkinson et al., 2003). Although these techniques are currently used in the first deployment phase of an EIS, the flexibility with respect to later changes still remains a desired feature.

In this paper, an agile MDD framework will be proposed, which increases model-driven EIS flexibility and, therefore, organizational agility. The proposed framework comprises a formal validation of interdependencies, in order to reduce the amount of human intervention. Furthermore, it is envisioned that the framework will enable the automatic preservation of consistency of the EIS during change.

Throughout this paper, EIS refers to that software component, which is specific for an organization. It does not contain the hardware, the network or the standard applications. The remainder of this paper is structured as follows. Section 2 discusses the background concerning ERP and WfM integration. Next, section 3 presents the contribution of current MDD techniques to increase the flexibility related to EIS development. Section 4 explains the framework that envisages how to achieve flexibility after deployment. Finally, section 5 concludes the paper.

van Beest N., Szirbik N. and Wortmann J. (2009).
A VISION FOR AGILE MODEL-DRIVEN ENTERPRISE INFORMATION SYSTEMS.
In *Proceedings of the 11th International Conference on Enterprise Information Systems - Information Systems Analysis and Specification*, pages 188-193
DOI: 10.5220/0002001201880193

2 BACKGROUND

There are many real-life situations where an EIS is formally based on both database schemas and executable process models that are interrelated. This situation is typical for organizations that use an ERP system and – on top of that – one or more WfM systems that work together with some parts of the ERP system (Szirbik et al., 2004). The operational link between these working systems can be implemented easily in a lightweight manner via a loose connection, by just having the same human user for both systems. A more costly alternative is to couple the systems in a heavyweight manner, by building a full-featured integration (via formal interfaces and semi-automated procedures). In this situation the process execution has to be integrated in the ERP system and the data from the ERP database has to be used and changed directly by the WfM systems. However, these integration attempts have been always marked by one or more of the following problems: the difference between explicit / implicit models of data and process used in both systems, the redundancy of representation, concurrent transactions and loss of flexibility (Reijers, 2006).

The ERP system is deployed from its characteristic data perspective (Olson, 2004) and the WfM system is deployed from its characteristic process perspective (Van der Aalst et al., 2002). The ERP system always has an explicit data representation (its database schema) but only an implicit process perspective that is merely employed as a useful attachment for managers (Ami et al., 2007). Moreover, one of the well-known drawbacks of ERP is process rigidity after deployment (Botta-Genoulaz et al., 2005).

The WfM system, on the other hand, has an explicit process model, but there is no strict requirement for a data model. Nonetheless, the use of some data is inevitable for any workflow implementation. Furthermore, a database can exist externally of the enactment engine, and provides a formal interface with this database (Ceri et al., 1997). Taken in isolation, a WfM system commonly enacts a number of process models, each controlling a multitude of running cases. These process models can be changed rather easily (Stohr et al, 2001) if attention is given to the versioning of the cases that are executed during the change. Sometimes, manual intervention is required (Van der Aalst, 2001).

Several problems can arise after integration. For example, conditional execution of activities (XOR splits in the process) depends on certain data. Based on that data, the condition is evaluated for each case and a decision is made whether the activities are executed or not for that particular case. Nevertheless, if this data is changed by a concurrent process, the activities that are executed after the XOR-split may be based on an incorrect condition and there is no trigger to notify the erroneous execution of these activities.

However, the main problem of ERP and WfM integration is the well-known problem of frozen process representations. After the integration, the result is often a redundantly specified system. That is, some database areas will be specified in both the ERP and the WfM system. In addition, some process enactment logic is replicated in the ERP system as well. This is necessarily so, because the ERP system always contains some process logic that overlaps with the explicit process definition of the workflow. Similarly, the WfM system should be ‘aware’ of the entities that are used in the WfM system and which already exist in the database structure of the ERP system.

This redundancy between process and data specifications frequently results in problems, especially if there is no clear awareness and there are no procedures in place to tackle it (Rayhupathi et al., 2008). Necessary changes in the process model of the WfM will induce costly changes of the business logic in the ERP system. In some cases, a process change includes a change of the data structure used by the process. If this data is also represented in the ERP database, this leads to changes to the database structure and, implicitly, to adaptation of the business logic.

As a result, any WfM system that is linked to a legacy system – in this case the ERP system – makes the change in the process model of the WfM system almost impossible, especially if the degree of integration is high (Reijers, 2006).

The problems that arose during ERP and WfM integration did not disappear with the new trends in MDA-based EIS and MDD-change. The point is, that the integration, specification redundancy, and especially consistency of data and process models is affected when change of data models or process models (or both) is effectuated in order to support change in an organization and its business processes.

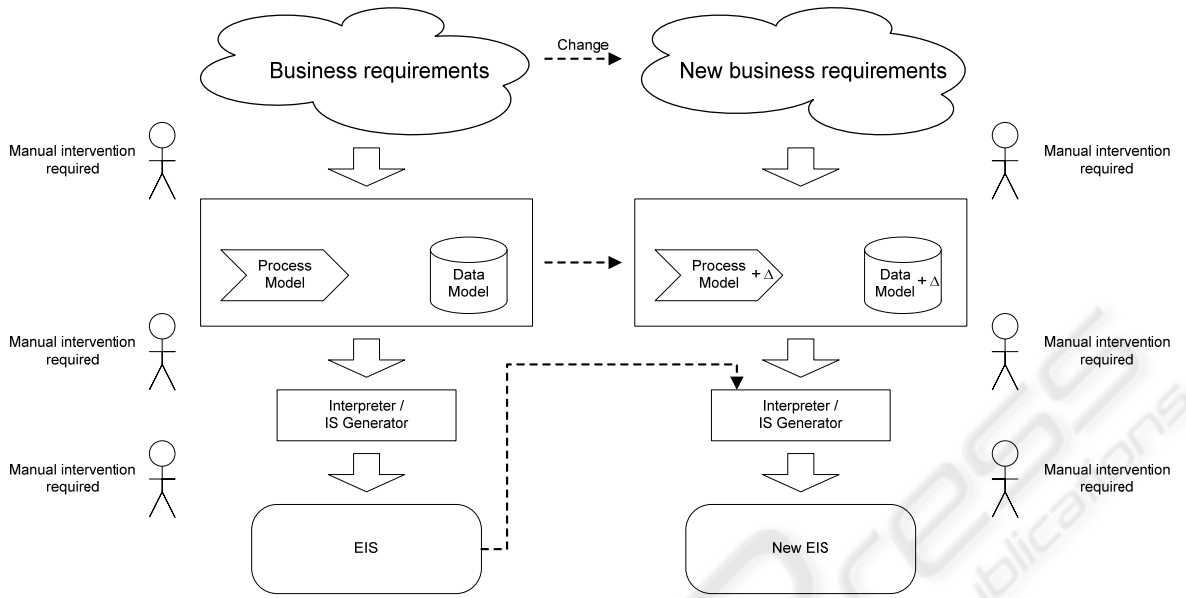

Figure 1: Schematic representation of a flexible model-driven EIS. Initial generation (right side) and subsequent change (left side).

3 BESPOKE FLEXIBILITY

MDD emerged as a possible solution to increase flexibility by using MDA. Nowadays, MDD approaches exist, which enable a partial automatic generation of EISs.

In this paper, flexibility of an EIS is referred to in two different ways. Firstly, it can be seen as the MDD-based capability of a system to be specified according to the exact requirements of the business process and structure. That is, the software should be bespoke and provide a one-on-one match with reality and the business requirements. Secondly, flexibility can be defined as the capability of an existing EIS to be adjusted to changing business requirements as fast and inexpensive as possible.

If during initial design time the process description is seamlessly integrated with the required data, redundancy issues can be resolved initially, as no longer two distinct systems have to be integrated. Using MDD, a complete system specification can be provided based on the requirements (see the left-hand side of Figure 1) (OMG, 2003). With respect to modeling abstraction, MDA concerns three different layers of abstraction, including Computation Independent Model (CIM), Platform Independent Model (PIM) and Platform Specific Model (PSM). However, the view of this paper abstracts from these layers and refers only to an enterprise model specification, which consists of both a process and data specification (this holds for every layer in MDA).

A change in the business requirements after the initial deployment of the system requires a change in the process model of the system or the data model of the system (as indicated by the delta notation in the right hand part of Figure 1). In a perfect situation, the new business requirements should be fully represented by both the process and data model, which are continuously consistent with each other. However, incomplete integration of processes and data may result in some operational problems. For example, the process may require data, which does no longer exist. Similarly, the process may use data, which is no longer accurate. Finally, the process may execute incorrect activities due to a (correct) change of data elsewhere in the process.

In most MDD approaches, process and data models are kept separately on purpose, in order to change one without affecting the other. As a result, interdependencies are not explicitly described or identified in current MDD languages (France et al., 2006; Meijler et al., 2006). However, as a result of the natural interdependencies between processes and data, a change in the process model may affect the data model. In addition, this enforced change of the data model may also affect other parts in the process, which are not accounted for. Essentially, whenever an artifact changes (that is, a process or a

data object), this may affect some or all of its interrelated artifacts. As a result, the cascade of change spirals out of control. This problem is referred to as the rampant roundtrip problem (Hailpern et al., 2006; Rayhupathi et al., 2008). Consequently, a change in a model-driven developed EIS is still time-consuming and costly. For that reason, the interdependencies need to be described explicitly (by using novel syntactic constructs) in order to prevent data-process inconsistencies in the models and, therefore, lack of coherence in the generated EIS.

Nonetheless, past research has primarily focused on model-driven change in the context of change that is required due to shifts in the metamodel (Hearnden et al., 2006; Garcés et al., 2008) (e.g., a shift from UML 1.5 to UML 2.0). However, change, which is pushed by new business requirements, remains largely ignored. In the remainder of this paper, change will refer to the latter category.

The "ultimate" (not necessarily attainable) goal of MDD can be identified as creating the ability to change the EIS by redrawing a part of a model and executing the model interpreter/generator only once without any human intervention. This goal implies that "perfect" (consistency ensures that none of the mentioned problems occur) integration of both the data and process model is essential.

As shown in Figure 1, the current MDD techniques require the manual intervention at three stages in the process. First, a considerable amount of human intervention is required to translate the business requirements – which may be a fuzzy description of the business process, organizational structure and resources, and the way employees of an organization perform their work – into the models to be used for either initial system development, either for later change. Second, human intervention is required to create consistency between the models and make them appropriate to generate an EIS. Third, human intervention is required during and after system generation and validation, as some parts have to be reworked later due to consistency and interdependency issues. The interdependencies between data and process models may trigger a cascading effect of model changes and exception handling during the interpreting runs, all of which need to be identified eventually and solved manually. Nevertheless, all the parts in Figure 1 that are fully automated will increase the level of flexibility. Therefore, a perceptible gain in terms of flexibility can be achieved just by decreasing the amount of required human intervention, without necessarily aiming for complete automation.

4 THE AGILE MDD VISION

The new envisaged framework considers both a formal process model and a formal data model, which are integrated by a formal mapping. Furthermore, the framework contains a number of additional steps compared to the existing framework, as shown in Figure 1. An overview of the envisaged framework is represented graphically in Figure 2.

This framework contains a different procedure of model-driven transformation of business requirements to an EIS. In this case, the dependencies between data and processes should be identified in an early stage during requirements analysis.

Next, these business requirements should be translated into both a data model and a process model. The design of the models is performed using an MDD modeling toolset, which allows the system architect to create both a data model and a process model. The MDD modeling toolset should contain syntax-driven constraints that enforce the explicit definition of the relation between data objects and activities as defined in the process model. That is, each data object defined in the data model should in some way be related (or mapped) to one or more activities in the process model.

The next step – which was already included in the currently existing framework – should concern the individual syntax check of the process model and the data model. For instance, process models are checked on deadlocks, livelocks etc.

Subsequently, interdependencies between the process and data model should be checked on consistency. All interdependencies will be automatically identified during the design of the models. This way, consistency between activities and data structures can be ensured. Furthermore, after a (manual) change to the process model, the MDD editor should be able to track and pinpoint all dependencies, showing possible alternatives and solutions in case of inconsistencies. However, the final decision is to be left to the system architect. After a change of the data model, the affected processes will be automatically detected and a suitable change is proposed.

The process definition of the agile approach should make use of the concept of workflow patterns (Russell et al., 2006). After the automated checks on the models, those workflow patterns, which potentially result in erroneous execution of the business process, must be identified. That is, every occurring instance of data-dependent workflow patterns (like

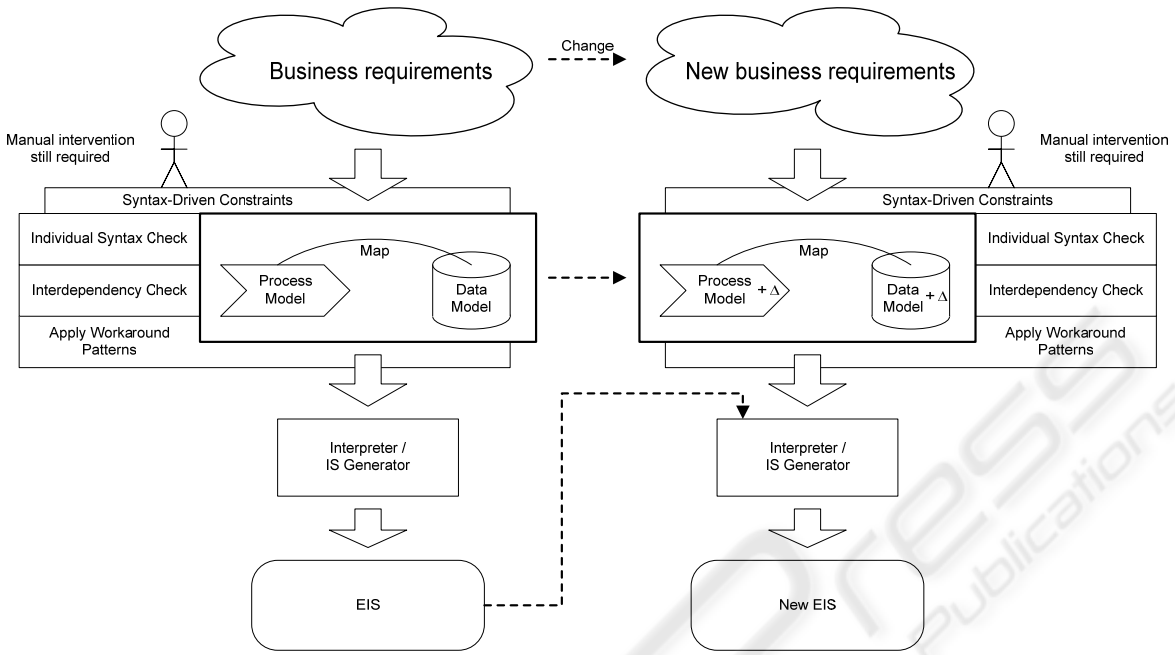

Figure 2: Schematic overview of the “agile MDD vision”.

workflow constructs, which need external data to proceed) are to be detected in the context of data interconnected processes (parallel processes, which use the same data at a particular point in time).

Prior to the conversion the models into an executable EIS by the interpreter, the data-dependent workflow patterns are to be replaced using formally defined workaround patterns. Workaround patterns specify an implementation solution, which is to be used by the interpreter with respect to a certain troublesome workflow pattern. In this way, the system architect will no longer be required to keep track of all potentially contradictory specifications, as these are automatically identified and solved. Finally, the models will be prepared to be executed by the interpreter.

The left-hand side of Figure 2 represents the MDD procedure in case of initial deployment. The right-hand side presents the MDD procedure with respect to change of the EIS as a result of a change in business requirements. However, Figure 2 shows that the model-driven change procedure is identical to the procedure for initial deployment. The automated checks concerning syntax and inter-dependencies represented on the left-hand side are replicated on the right-hand side. Therefore, the only major issue that affects flexibility, which was not discussed, is migration.

Two approaches can be considered with respect to migration. The first approach is based on a phasing out scheme of running cases that rely on old process definitions. However, the main disadvantage of this approach is that if changes occur frequently, many versions of the process have to be kept. Moreover, if process instances phase out too slowly, an explosion of versions may occur.

The second migration approach (big bang) is a complete migration of all process instances after deployment of new models. Every running case is, therefore, migrated to the new process definition. The main disadvantage of this approach is that some manual intervention may still be required, despite automatic process and data migration. Moreover, big bang deployments are costly in terms of manual intervention.

As it appears, this poses a trade-off. In the situation of process models that comprise a large amount of short-lived cases, it is better to use the phase-out method. On the other hand, if a few cases with a long throughput time are running at a certain moment in time, then a big bang migration is preferred with some possible manual intervention. Although migration of running cases is a serious issue to take into consideration, it is not the focus of this discussion. The vision presented in this paper focuses on the modeling part of MDD and not on migration.

5 CONCLUSIONS

Although many MDD techniques claim to be able to generate EISs based on enterprise models, these techniques still lack the long-time desired flexibility, which allows that a change in the model can be immediately and easily reflected in the EIS. In this paper a position is presented about how the flexibility of an MDD EIS can be achieved via a so-called agile framework. In addition to existing MDD approaches, this agile framework proposes three additional validation checks based on a fully integrated data and process description. The envisaged framework leads to a need for a practical mapping formalism between process and data.

The application of the framework will benefit those organizations that tend to change their business processes quite often. As a result of the automated consistency and pattern checks, it is expected that flexibility increases, by reducing the amount of required human interventions during change. Therefore, deployed EISs will not act as a constraint on organizational agility.

Foreseen further work can be described as follows. The observation of the process of EIS change engineering requires the identification of troublesome workflow patterns (albeit syntactically and semantically correct, due to data coupling the execution of these patterns may result in consistency errors) and the development of workaround patterns. All these are directed towards the issue of data interconnected processes and data dependent workflow patterns. By collecting and systematizing these patterns, it is aimed to build a theoretical and practical knowledge base that can stand as a foundation for future MDD technologies that provide the needed flexibility for EISs.

REFERENCES

Van der Aalst., W.M.P., 2001. Exterminating the Dynamic Change Bug: A Concrete Approach to Support Workflow Change. *Information Systems Frontiers*, 3, 3, pp. 297-317.

Van der Aalst, W.M.P., Van Hee, K.M., 2002. *Workflow Management: Models, Methods and Systems*. MIT Press, Cambridge, Mass.

Ami, T., Sommer, R., 2007. Comparison and evaluation of business process modelling and management tools. Int. J. of Services and Standards, 3, 2, pp. 249-261.

Atkinson, C., Kühne, T, 2003. Aspect-Oriented Development with Stratified Frameworks. *IEEE Software*, 20, 1, pp. 81-89.

Botta-Genoulaz, V., Millet, P.A., Grabot, B., 2005. A survey on the recent research literature on ERP systems. *Computers in Industry*, 56, 6, pp. 510-22.

Cardoso, J., Bostrom, R.P., Seth, A., 2004. Workflow Management Systems and ERP Systems: Differences, Commonalities, and Applications. *Information Technology and Management*, 5, pp. 319–338.

Ceri, S., Grefen, P., Sanchez, G., 1997. WIDE - a distributed architecture for workflow management. In *RIDE '97, 7th Int. Workshop on Research Issues in Data Engineering, High Performance Database Management for Large-Scale Applications*, pp. 76.

France, R.B., Ghosh, S., Dinh-Trong, T., Solberg, A., 2006. Model-Driven Development Using UML 2.0: Promises and Pitfalls. *Computer*, 39, 2, pp. 59-66.

Garcés, K., Jouault, F., Cointe, P., Bézivin, J., 2008. Adaptation of Models to Evolving Metamodels. *Technical Report, Institut National De Recherche En Informatique Et En Automatique*, ISSN 0249-0803.

Hailpern, B., Tarr, P., 2006. Model-driven development: The good, the bad, and the ugly. *IBM Systems J.*, 45, 3, pp. 451-461.

Hearnden, D., Lawley, M., Kerry, R. 2006. Incremental Model Transformation for the Evolution of Model-Driven Systems. *In MoDELS 2006 Proc. of the 9th International Conference*, 4199/2006, pp. 321-335.

Kleppe, G., Warmer, J., Bast, W., 2003. *MDA Explained: The Model Driven Architecture: Practice and Promise*. Addison-Wesley,

Meijler, T.D., Postmus, D., Wortmann, J.C., 2006. Towards Model-driven Evolvability of Enterprise Information Systems. *In 10th IEEE Int. Enterprise Distributed Object Computing Conference (EDOC'06)*, pp.413-416.

Olson, D.L., 2004. *Managerial issues of enterprise resource planning systems*. New York: McGraw-Hill.

OMG, 2003. MDA Guide Version 1.0.1. www.omg.org.

Rayhupathi W., Umar A., 2008. Exploring a model-driven architecture (MDA) approach to health care information systems development. *Int. J. of Medical Informatics*, 77, 5, pp. 305-314.

Reijers, H.A., 2006. Workflow Flexibility: The Forlorn Promise. In WETICE'06, *15th IEEE Int. Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises*, pp. 271-272.

Russell, N., ter Hofstede, A.H.M., van der Aalst, W.M.P., Mulyar., N., 2006. *Workflow Control-Flow Patterns: A Revised View*. BPM Center Report BPM-06-22, BPMcenter.org.

Stohr, E. A., Zao, J. L., 2001. Workflow Automation: Overview and Research Issues. *Information Systems Frontiers*, 3, 3, pp. 281-296.

Szirbik, N.B., Wortmann, J.C., 2004. Bridging The Gap Between ERP And WfM In Planning Using Agents. *In Proc. of the Int. IMS Forum*, Cernobbio Italy, pp. 317-324.