Model-Driven Tool Integration with ModelBus[1]

Christian Hein, Tom Ritter and Michael Wagner

Fraunhofer Fokus, Kaiserin-Augusta-Allee 31, 10589 Berlin, Germany

Abstract. ModelBus is a tool integration technology which is built upon Web Services and follows a SOA approach. ModelBus facilitates the orchestration of modeling services which represent particular functionality of tools acting on models. This demonstration paper summarizes the key concepts of the ModelBus technology and presents an outline of a scenario where ModelBus has been applied in an industrial context.

1 Introduction

In large distributed model driven development processes it is a difficult undertaking to maintain a conglomerate of different analysis, development, and documentation tools. Keeping data up to date and consistent across a whole development team is another important challenge. Using an extensible IDE with a set of plug-ins does only partially do the job. Firstly, there are still a number legacy tools which need to be part of a tool environment and secondly, the coordination of the interaction and workflow in a development team needs to be addressed. Standards like MOF are concerned with the problem of data storage and handling but not of data distribution.

Regarding tool integration a lot of theoretical work has already been done. In the late 80s Wasserman described the five dimensions of integration: Platform, Presentation, Data, Control and Process [1]. Thomas and Nejmeh extend the approach of Wasserman by experiences with framework services and integrated environments [2]. In 1989 a reference model has also be introduced [3] with the aim to provide a common standard for the development of software engineering environments. This reference model has also been used by the European Computer Manufacturers Association (ECMA) [4]. The general idea of services to provide operations on models has been described in [5,6].

A lot of technologies have been used in order to facilitate tool integration. The most common technologies are CORBA [7], Web Services [8], ESBs [9] or Java RMI [10]. New technologies like IBM's Jazz [11] do not only focus on tool integration but also on team collaboration. Tools like MOFLON [12] or CDO [13] see models as the main development artifacts and propagate a model-driven tool integration. However, no listed approach deals with all the problems of a model driven tool integration with-

[1] The work in this paper is partially supported by the European Commission via the MODELPLEX project, co-funded under the "Information Society Technologies" Sixth Framework Programme (2006-2009).

Hein C., Ritter T. and Wagner M. (2009).
Model-Driven Tool Integration with ModelBus.
In *Proceedings of the 1st International Workshop on Future Trends of Model-Driven Development*, pages 35-39
DOI: 10.5220/0002174800350039

in a complex development process in which diverse requirements must be addressed like process automation, distribution, user management, model consistency or model scalability.

ModelBus is a solution which aims at overcoming the limitation of existing approaches. It allows the integration of heterogeneous tools covering the whole development lifecycle and comprising custom-made, proprietary tools as well as COTS tools. With the help of specific adapters it is possible to connect tools to the ModelBus allowing them to share data via models and functionality via services. In the context of the EU Project Modelware [14] a first prototype has been developed under the term ModelBus. Due to some lacks in the design and implementation a revised ModelBus has been designed. This new ModelBus is hosted as an open source project at www.modelbus.org. This paper summarizes key concepts of this new ModelBus and it points out the major benefit of its SOA based approach. In addition, it outlines a development scenario which is been used in an industrial use case of the Modelplex [15] project.

2 ModelBus

The main goal of the ModelBus framework is to bring model awareness to Service Oriented Architectures (SOA) [9]. This is archived by extending Web Service interfaces on the Application Level to create Modeling Services. These Modeling Services support advanced features necessary for distributed programs dealing with models like scalability (load on demand, incremental loading), consistency, metamodel and model discovery as well as dependency, version and transaction management. However, this is done transparently for the application developer.

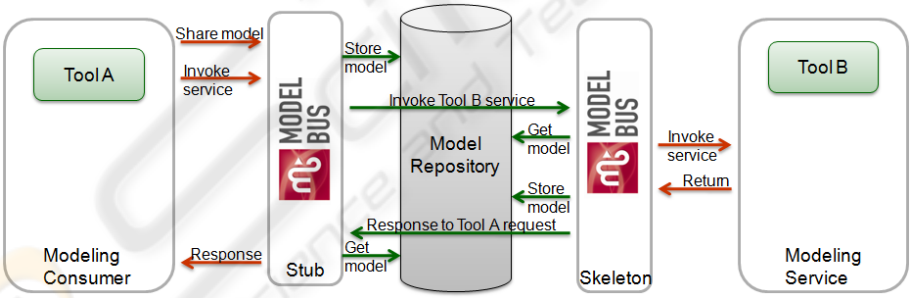

Fig. 1. ModelBus Interaction Pattern.

ModelBus provides an interaction pattern in order to enable model sharing in a distributed and heterogeneous model-driven development process. Figure 1 depicts the general interaction pattern in a ModelBus integration scenario.

The key concept of model sharing in ModelBus is realized via a model repository. This repository interface is open and allows straight forward addressing of models via URLs. This addressing schema also results in simple service interfaces, because only model references instead of models are transmitted. Repository vendors can imple-

ment this interface in order to be ModelBus conform. ModelBus itself is delivered with a built-in model repository, which supports versioning, partial check-out of models and coordinates the merging of model versions and model fragments.

In a model-driven development process a number of recurrent tasks must be performed like model transformation or model verification. To increase the efficiency of the underlying development process it is vital to reduce the amount of manual developer intervention needed and to introduce as much automation as possible in particular for long running tasks like model simulation. For this purpose ModelBus allows the definition of single tasks as modeling services and to orchestrate them together with other modeling services. These orchestrations can automatically be triggered either by user commands or by other orchestrations.

One of the major problems when dealing with complex models is to keep track of all incoming and outgoing references (e.g. profiled UML models or diagram models). As models grow larger and larger it becomes more and more difficult to coordinate the workflow in a way that permits concurrent work on different parts of a model. The built-in ModelBus model repository addresses this challenge. Moreover, the ModelBus notification system can be used to stay informed about the changes applied to a model by other developers.

Currently a number of tool adapters are already available. The set of ModelBus-connected tools will be extended by developing new corresponding ModelBus adapters. The organization, the execution and the control of the development process via BPNM [16] has been implemented utilizing Intalio [17]. Through the adaptation of Adaptive Repository [18] an alternative repository implementation has become available.

3 Example Development Process

The ModelBus technology outlined in the section above has been used for the realization of a distributed development scenario, which has been applied by partners in the Modelplex project. In this process several hundred developers of different development teams working in different locations are collaborating (see Figure 2). Specific requirements had to be obeyed regarding access control and model scalability implied by the complexity of the models involved in this scenario. Furthermore, certain activities of the development process have had to be automated. This has been realized using orchestration and notification provided by ModelBus.

In the example development process two development tools are used; Intalio Designer for BPMN modeling and Rational Software Architect (RSA) for UML modeling. In addition to this, a BPMN-to-UML transformation based on ATL is used as well as model verification based on OCL, to verify BPMN models according to design guidelines like naming conventions. All models, development artifacts, model transformations, and verification rules are stored within the ModelBus repository. For the automation of the verification and transformation a COTS tools is used. The orchestration is deployed as BPEL to an application server in which the ModelBus repository, the verification, and the ATL transformation engine are running (in Fig.2. see *Node Paris 2*).

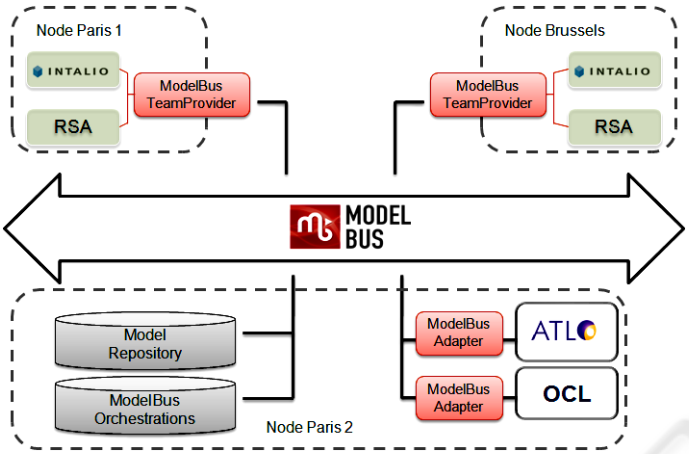

Fig. 2. Distributed Development Process.

References

1. Wassermann, A.: Tool Integration in software engineering environments. In The International Workshop on Environments (Software Engineering Environments), volume 647 of Lecture Notes in Computer Sciences, pages 137-149, Springer-Verlag, Berlin, September 1989, Chinon, France
2. Thomas, I., Nejmeh, B.: Definitions of Tool Integration for Environments. IEEE Software, 9(2):29-35, March 1992
3. Earl, A.: Principles of a reference model for computer aided software engineering environments. In F Ling, editor, The international Workshop on Environments (Software Engineering Environments), volume 647 on Lecture Notes in Computer Sciences, pages 115-129, Springer-Verlag, Berlin, September 1989, Chinon, France
4. ECMA Document – Reference Model for Project Support Environemnts. http://www.ecma-international.org/publications/techreports/E-TR-069.htm
5. Blanc, X., Gervais, M.-P., Sriplakich, P.: Model Bus. Towards the Interoperability of Modelling Tools, Proceeding of the European workshop on Model Driven Architecture: Foundations and Applications (MDAFA'2004), June 2004, Linköping University, Sweden, selected for : Lecture Notes in Computer Science (LNCS) « Model Driven Architecture: Revised Selected Papers », Volume 3599/2005, Springer
6. Blanc, X., Gervais, M.-P., Sriplakich, P.: Modeling Services and Web Services: Application of ModelBus to appear in the 2005 International Conference on Software Engineering Research and Practice (SERP'05), 2005.
7. OMG Document – CORBA – Common Object Request Broker Architecture, http://www.omg.org/technology/documents/corba_spec_catalog.htm
8. W3C specification, http://www.w3.org/TR/ws-arch/
9. Chappell, David A.: Enterprise Service Bus. Theory in Practice. O'Reilly Media; ISBN 978-0-596-00675-4
10. Java Remote Method Invocation - http://java.sun.com/javase/technologies/core/basic/rmi/index.jsp
11 IBM's rational Jazz, http://www-01.ibm.com/software/rational/jazz/

12 Amelunxen, C,Klar, F., Königs, A., Rötschke, T., Schürr, A.: Metamodel-based Tool Integration with MOFLON, 30th International Conference on Software Engineering, New York: ACM Press, 05 2008, ACM Press, 807-810.
13. Connected Data Objects, http://wiki.eclipse.org/CDO
14. ModelWare Project. www.modelware.org
15. ModelPlex Project. www.modelplex.org
16. OMG Document - Business Process Modeling Notation http://www.omg.org/cgibin/apps/doc?dtc/06-02-01.pdf
17. Intalio http://bpms.intalio.com/
18. Adpative Repository, http://www.adaptive.com/resources_papers/repository.html