

Support for Model-centric and Architecture-centric Development - with workflow-based composition

Arne-Jørgen Berre,
Distributed Information Systems, SINTEF
P.O.Box 124, Blindern
0314 OSLO, NORWAY
E.mail: Arne.J.Berre@sintef.no

Position statement for ECOOP'2003 WS12 "Third International Workshop Composition Languages"

1. Introduction

This position statement describes the key results and issues of the European Union supported COMBINE Project (COMponent Based Interoperable Enterprise system development) that ran from June 2000 to January 2003. A main result is the COMBINE Framework, which is supporting the establishment and running of a Component Centre within an organisation. One goal of the Component Centre is to support CBSE using the principles of the OMG's Model Driven Architecture^{TM1} (MDA).

In the context of composition the main position is that this is possible to derive in an MDA-style from the description of UML-based activity models, derived from earlier work on workflow modeling.

2. The COMBINE Component Centre for MDA

2.1 Overview

The COMBINE Framework includes an Integrated Environment with various tools, related to an Enterprise repository, Execution environment and workflow support as shown in figure 3. Modelling Toolset supports mapping between models and code-generation for the interfaces of software components through various platform specific models. The Execution Environment can support various platform- technologies, the initial targets being J2EE/EJB and later Web Services.

¹ Model Driven Architecture is a registered trademark of the Object Management Group Inc.

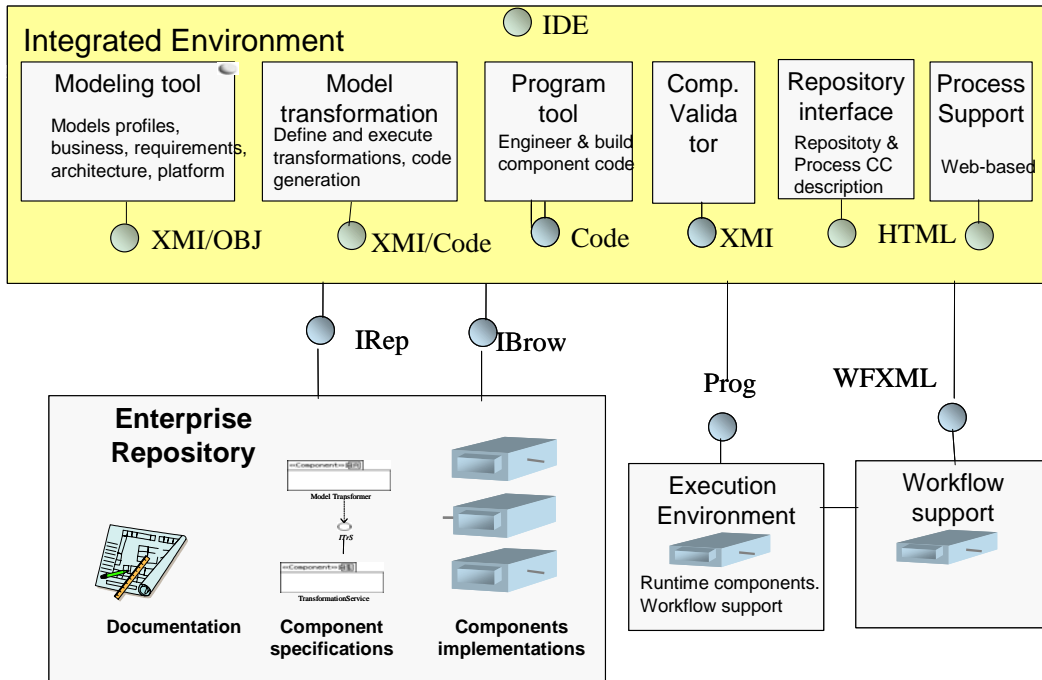


Figure 1 – COMBINE Framework – Parts and interfaces

Figure 3 shows the defined interfaces between the various parts included in the COMBINE Framework. The interfaces are as much as possible based on standard models, such as XMI and HTML, but also have some non-standardised interfaces, such as the Irep repository interface and the workflow XML interface. The initial Integrated Environment for the various tools is the Eclipse Open Source Java IDE, but this can later be substituted with an integration with another IDE, such as Microsoft Visual Studio.

2.2 Detailing the Component Centre

2.2.1 The COMBINE Model Architecture

The COMBINE methodology supports the development of a set of models, defined in the middle of figure 6 as the COMBINE Model world. These are the Business Model, the Requirements Model, the Architecture Model and the

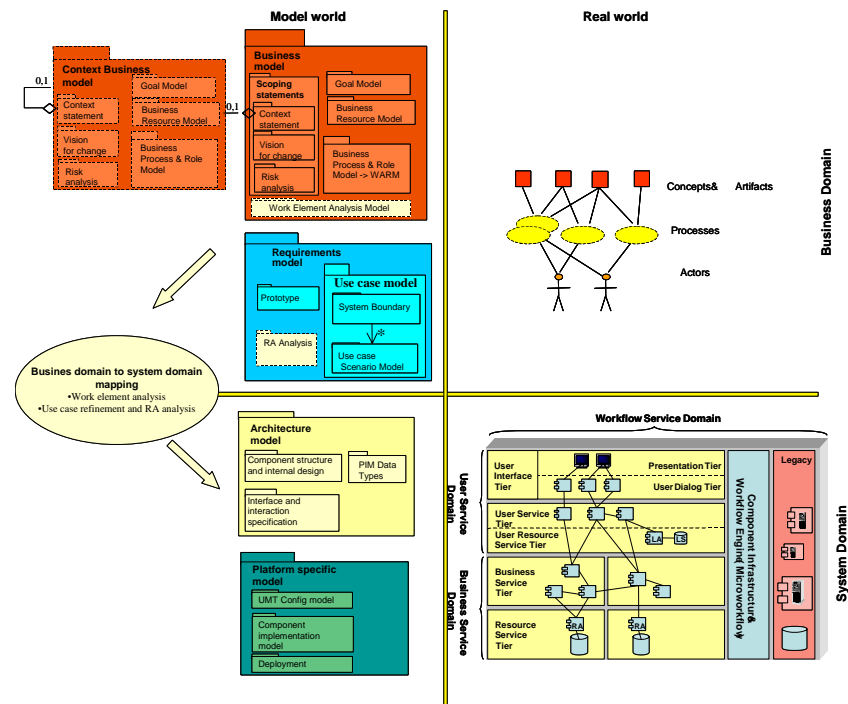


Figure 2 The COMBINE Model Architecture

Platform Specific Model for a Product (component-based business system or individually useful component). Each of these models is defined in UML according to a set of UML profiles (based on Metamodels) derived from the UML Profile for EDOC, the MDA and UML2.0 technology adoption processes. The focus of the architecture model is to support the platform independent specification of Enterprise systems using the concepts of the COMBINE Component Architecture as shown to the lower right. The COMBINE partners have been, and will continue to be, active in the OMG standardisation process around MDA, UML Profile for EDOC and UML2.0.

2.3 UMT – UML Model Transformation tool

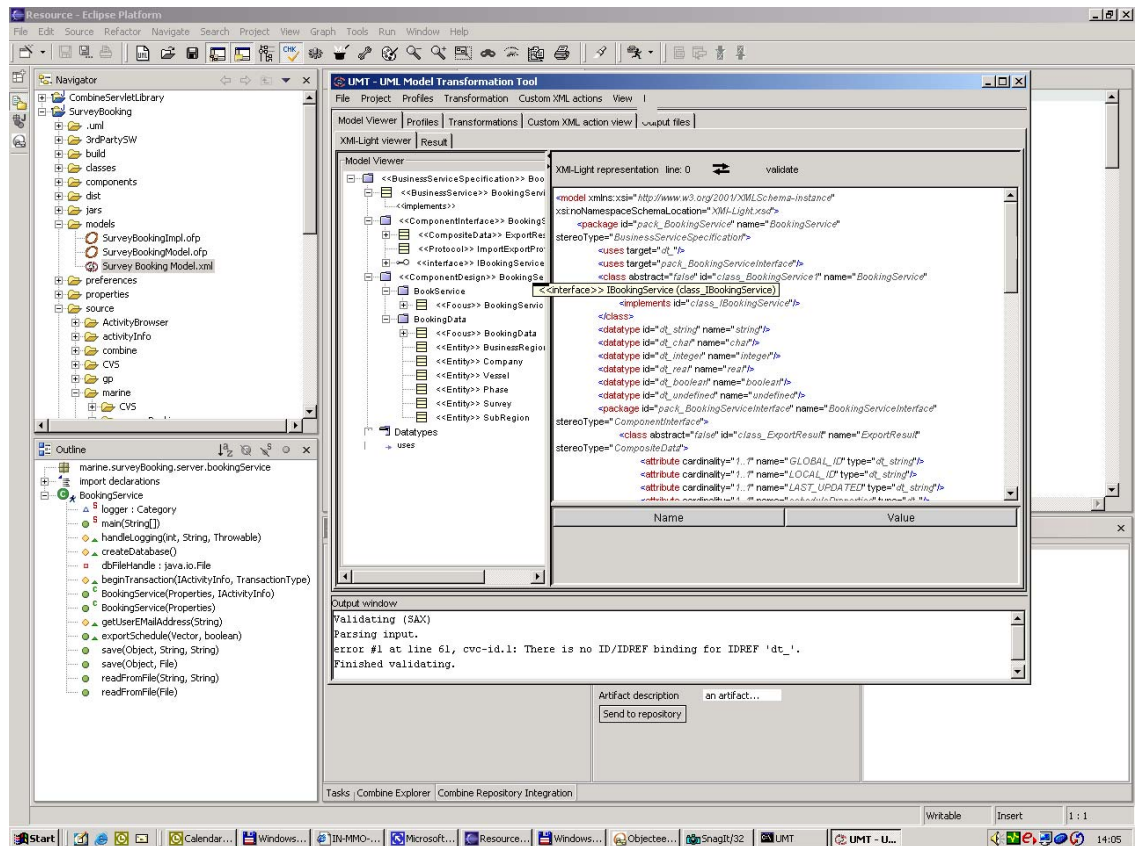


Figure 3 Eclipse IDE – with integrated UMT Model Transformation tool

UMT is a tool to support model transformation and code generation based on UML models in the form of XMI.

XMI models are imported by the tool and converted to a simpler intermediate format, which is the basis for validation and generation towards different target platforms.

The intermediate format is an XML/HUTN format, which in UMT is called XMI-Light. The end-user of the tool doesn't need to be concerned about this format. However, the developers of code generators (emitters) need to know its details.

The main usage scenario is to import a UML XMI model and generate code for the desired target platform

The COMBINE component architecture is about how components are described and interrelated. It has several dimensions to it, including a logical system dimension, a system platform dimension and a business dimension.

When speaking of component architecture in this document, we mean the logical system dimension of the component architecture. The concepts defined as component architecture provide the vocabulary for describing platform-independent models (PIM) within COMBINE's model-driven architecture. The PIM can be mapped to platform specific models (PSM) and implementations via transformation rules. The Component Centre will support the transformations from PIM to PSM and implementation platforms. Specifically, it will support transformation from a PIM component architecture (a logical design) to a PSM focused on J2EE/EJB and Web Services, which can then be mapped to specific execution environments, such as the JBOSS J2EE Application Server.

The component architecture specifically focuses on:

- defining a distributed application reference model,
- defining the concepts that are pertinent for building a system of components within COMBINE,
- and relating these concepts to the UML metamodel, i.e. stereotypes that can be used when modelling COMBINE components and systems.

COMBINE's reference architecture defines four logical distribution tiers: user interface, user service, business service and resource service, as depicted in Figure 4. It is supported by various parts of the COMBINE execution environment. Initially the target platform is J2EE/EJB, through the use of the JBOSS EJB Application Server for the Business services, the J2EE RMI and JMS for communication services, and the INESC microWorkflow component services for service-oriented workflow support.

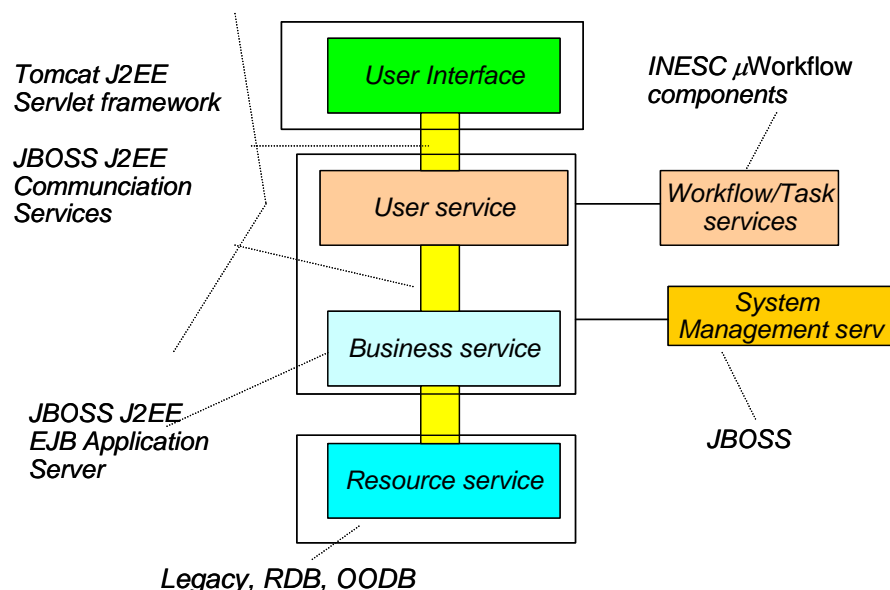


Figure 4 Combine reference architecture and execution environment

Micro-workflow is an integration middleware that allows the smooth implementation of business processes into an execution environment that decouples the work from the flow. (This approach is currently being enhanced to service and component composition in the ongoing ACE-GIS project, see later). The Work Analysis Refinement Modelling (WARM) allows this smooth implementation.

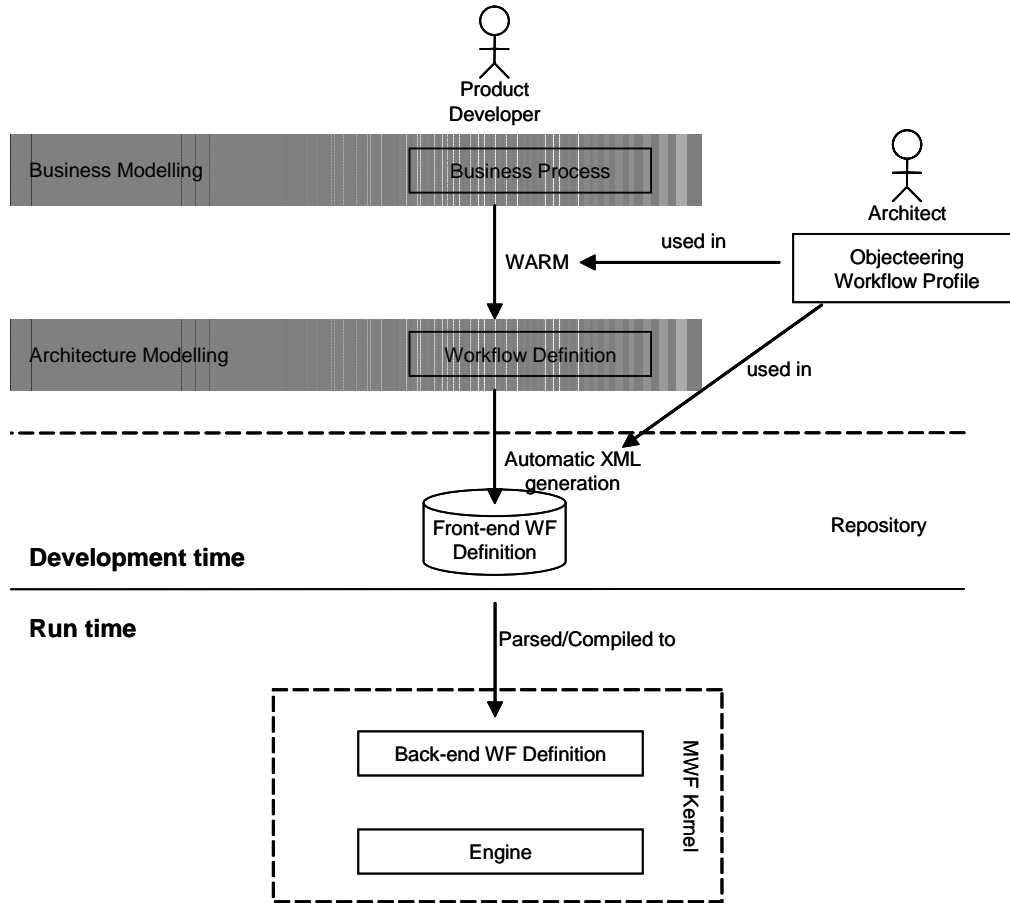


Figure 5 Micro-Workflow Architecture

The Figure above represents the relationships between WARM and Micro-Workflow Architecture. The Figure represents two different dimensions: workflow life cycle time and workflow development roles. The former considers development time versus runtime and the latter considers product developer versus architect.

Work Analysis Refinement Modelling applies during workflow development life cycle time while Micro-Workflow Architecture applies during workflow runtime – where an engine executes a XML workflow back-end definition that is the transformation of a XML front-end workflow definition. The existence of front- and back-end definitions

allows a cleaner separation of development time from runtime and it also supports different front-end definitions, turning the workflow robust to changes of high-level workflow definition languages.

The product developer defines workflows by distinguishing the parts of the business process that should be performed by human from the parts that can be automated, and should be performed by computers. As the result of product development activity it is defined which part of the business will be supported by information technology. Product developer uses a set of concepts, UML stereotypes, which are supported by the Objecteering tool, and allow the automatic generation of a front-end workflow definition in XML. The architect defines the meta-model of the WARM and implements Objecteering Workflow Profile to perform the automatic generation of XML front-end workflow definition from UML Workflow Definition.

The advantages of using micro-workflow are:

- It is a open-source product independent of specific software providers
- It implements the standard workflow definition languages for workflow definition and manipulation
- It supports web services choreography and composition languages
- It is divided into a definition front-end and an execution back-end in order to survive the evolving scenario of choreographic languages for web services.
- It has a modular architecture that allows the option integration of workflow features

3. Issues in Model-centric and Architecture-centric Development

The results of the COMBINE project can be viewed as a proof of concept of the validity of the MDA approach, based on its use in a pilot and demonstration example. However, it has also uncovered some issues related to the practical realization of MDA technologies, in particular:

- MDA support for service and component composition – through enhanced UML activity diagram modeling
- Distributed model management
- Specification of model transformations
- Model support for extra-functional aspects (QoS etc.)
- Establishment of standard type libraries for platform independent modeling

This issues need to be resolved in order for effective MDA-based development to succeed.

4. Conclusion and future work

Some of the identified issues are being addressed in further projects and development activities.

The ACE-GIS project (Adaptation and Composition for E-Commerce and GIS Services) is a project that further addresses the use of activity modeling as a basis for an MDA approach to service and component composition, based on the principles of workflow and business modeling, with micro workflow support, from the COMBINE project.

OMG is working on a strategy for MDA, where a set of ongoing RFP (Request for Proposals) are addressing model transformations and the modeling of extra functional aspects. However, now activities has so far been identified for distributed model management or a standard type library for platform independent modeling,

Bibliography

www.opengroup.org/~combine

www.acegis.net

www.omg.org/mda