Eclipse Modeling Workflow Engine (MWE) Component Proposal

Introduction

The Eclipse Modeling Workflow Engine (MWE) Project is a proposed open source component under the Eclipse Modeling Project (EMP) / EMFT.

Background

The Eclipse Modeling Project contains several components for working with models in the context of model-driven development. These include concrete basic frameworks and metametamodels, concrete syntax tooling, model serialization and parsing, as well as tools for checking and transforming models (into other models and into code). All of these components are essential for a practical MDD tool chain.

What is missing, however, is something that binds these tools together. Specifically, a component is required that can be used to orchestrate a model processing chain, such as reading the source model, checking constraints, transforming into another model, checking constraints on that new model, and then generating code. Because of this missing piece, the various components under the Eclipse Modeling Project, including in the Generative Modeling Technologies (GMT) project, are not easily pluggable and composable.

We call this missing component a *modeling workflow engine*. The steps in such a workflow shall be called *workflow components*.

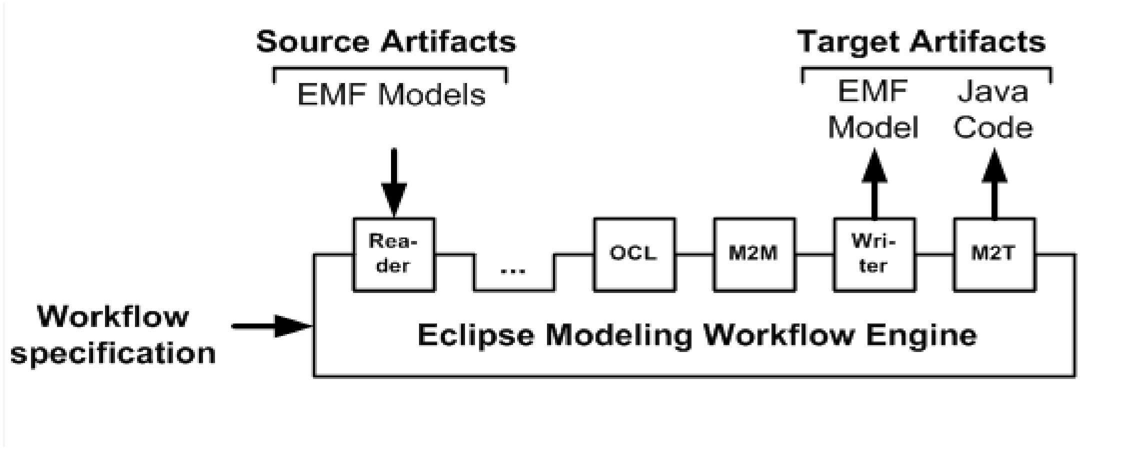
Component Description

The Modeling Workflow Engine component will deliver artifacts in two phases.

Phase 1 consists of the following artifacts:

- An extensible framework for integrating the orchestration of model processing workflows. Primarily this will consist of:
  - A well-defined API that defines how workflow components can integrate into a workflow.
  - A concrete implementation of the workflow engine that can be used to run a model processing workflow.
- Various kinds of tools such as:
  - An editor for defining workflows.
  - A way of executing the workflow from within Eclipse (Run As…) as well as from ant.

- A small general purpose library of components, such as
  - reading EMF models
  - writing EMF models
  - creating/cleaning directories
  - etc.



It is expected that most of the runtime-relevant model processing components in EMP will provide adapter workflow components to facilitate integration with the workflow engine. Key component providers (among them INRIA and oAW) have already agreed to provide these components once the APIs and the engine are available.

Please note that this is a relatively small component with a clearly defined scope. We expect results to be available soon after component creation.

Phase 2 will add the following aspects to the workflow engine:

- A way to attach metadata to workflow components that describe what they do and what the input and output data looks like.
- A repository of workflow components.
- An inference engine that automatically determines which chains of components can achieve a certain goal.

Existing Approaches

Two GMT (http://www.eclipse.org/gmt) projects currently offer some functionality close to the objectives of this project: AM3 and oAW.

The org.eclipse.gmt.am3.tools.ant plugin available from the AM3 project provides a set of ant tasks enabling the specification of complex modeling workflows. There are tasks to load and save models as well as to invoke the ATL engine to transform or query models. Arbitrary control flow (e.g., looping, calling other ant scripts) can be achieved using the general purpose tasks provided by ant and other projects like ant-contrib. With this approach, adding a workflow component consists in adding an ant task, for which Eclipse already provides extension points. However, the nature of ant-based workflow specifications is generally not abstract enough for an inference engine.

This solution is therefore compatible with the goals of phase 1, but is unsuitable in its present state for phase 2.
Please see http://wiki.eclipse.org/index.php/AM3_Ant_Tasks for further details.

openArchitectureWare comes with a workflow engine that can execute arbitrary workflows. Workflow execution happens in two phases: the first phase verifies all the workflow steps w/r/t to their configuration parameters. Only if this validation phase is successful (i.e., there are no configuration errors) is the second phase started. This second phase is the execution itself. Workflows are described using XML files. A workflow can be separated into several workflow files and workflow files can call each other, thereby allowing for modularized generators. oAW also comes with a custom editor that helps in writing correct workflow specifications (provides code completion, e.g.). In order to run arbitrary code as part of a workflow, the class has to implement a *WorkflowComponent* interface provided by oAW.
Please see http://www.eclipse.org/gmt/oaw/doc/4.1/r05_workflowReference.pdf for further details.

Initial Contributions

The following code will be contributed to the component, and corresponds to the goals of phase 1:

- oAW Community:
  - The oAW workflow engine
  - A workflow editor.

Scope definition

The proposed workflow engine is a very small endeavor. We need to describe what the proposed goal of MWE is and also what falls out of the scope of this initiative. The objective is not to propose a general scheme to link together various modeling components interactively. For example, a possible requirement scenario that clearly falls outside the scope of MWE may be the following: "During the creation of a model using a graphical modeler, subsequent model transformations can be triggered using model change events. Thus, an interactive, real-time integration between modeling tools can be achieved". In a much more limited way, the workflow engine is aimed as providing facilities as a batch-processing style of interaction, describing a clear sequence through a model processing tool chain. Although a workflow can be started interactively from within a modeling tool, there is no planned real-time, event based integration in this MWE initiative.

Organization

We propose that this component be undertaken as part of the Eclipse Modeling Project. It will have dependencies on EMF and the Eclipse Platform (optional, only needed for the tooling).

Proposed Component Lead

Bernd Kolb (b.kolb@kolbware.de), openArchitectureWare team member

Initial committers

The following companies will contribute committers to get the project started:

- openArchitectureWare
  - Sven Efftinge (sven@efftinge.de)
  - Arno Haase (arno.haase@haase-consulting.com)
  - Bernd Kolb (b.kolb@kolbware.de)
  - Markus Voelter (voelter@acm.org)
- INRIA
  - Freddy Allilaire (freddy.allilaire@univ-nantes.fr)

Interested parties

The following organizations have expressed interest in the project:

- Software Composition and Modeling Laboratory, University of Alabama at Birmingham www.cis.uab.edu/softcom

Participation

Critical to the success of this project is participation of developers in the modeling application development community. We intend to reach out to this community and enlist the support of those interested in making a success of the MWE project. We ask interested parties to contact the MWE newsgroup to express interest in contributing to the project.

Revision History

- First Draft, October 26, 2006.
- Second Draft, December 3, 2006.
- Third Draft, April 3, 2007.
- Final proposal, April 19, 2007