Model-level Execution
Preliminary Architecture Evaluation



István Gansperger, Róbert Kitlei, Gergely Dévai, Máté Karácsony,
Gábor Ferenc Kovács, Tamás Kozsik, Boldizsár Németh, Attila Ulbert

ELTE-Soft Nonprofit Ltd.

July, 2015

Contents

[1 Introduction 1](#_Toc425712577)

[2 Architecture summary 2](#_Toc425712578)

[2.1 High-level view 2](#_Toc425712579)

[2.2 Translation Chain 3](#_Toc425712580)

# Introduction

The purpose of this document is to give a high-level view of Model Executor architecture.

# Architecture summary

## High-level view



The user creates EMF-UML2 models via Papyrus. The model changes are propagated through the translation chain to create Java source files that are compiled to class files. When model execution is initiated, a Java Virtual Machine is instantiated to load these class files in and to start execution. During interactive model execution, the user is exposed to the Moka user interface. Moka breakpoints and state machine animation is implemented by Java breakpoints that are set via the Java Debug Interface.

In addition to Eclipse integration, a command line interface is also provided that uses the same translation chain (but in non-incremental way) and runs the model without involving the Java Debug Interface.

## Translation Chain



The model change events are propagated to IncQuery that updates the query results and creates change notifications. These notifications are handled by setting dirty flags (changed/new/deleted flags) on (to be) generated Java files. When translation is triggered (currently at model save), all dirty files are (re)generated by first building temporary translation models (on a per file basis) using the query results. The translation models are turned to Java text via Xtend templates and debug symbols are also generated.