

Model eXecution Framework (MXF)

Docuware for Creation Review

Copyright © 2009 Michael Soden, Hajo Eichler



ikv++ technologies ag



Table of Contents

- Executive Summary
- Background & Motivation
- Project Scope & Plan
- Code Contribution Outline
- Relation to other projects
- Mentors
- Initial Committers
- Interested Parties, Community Feedback

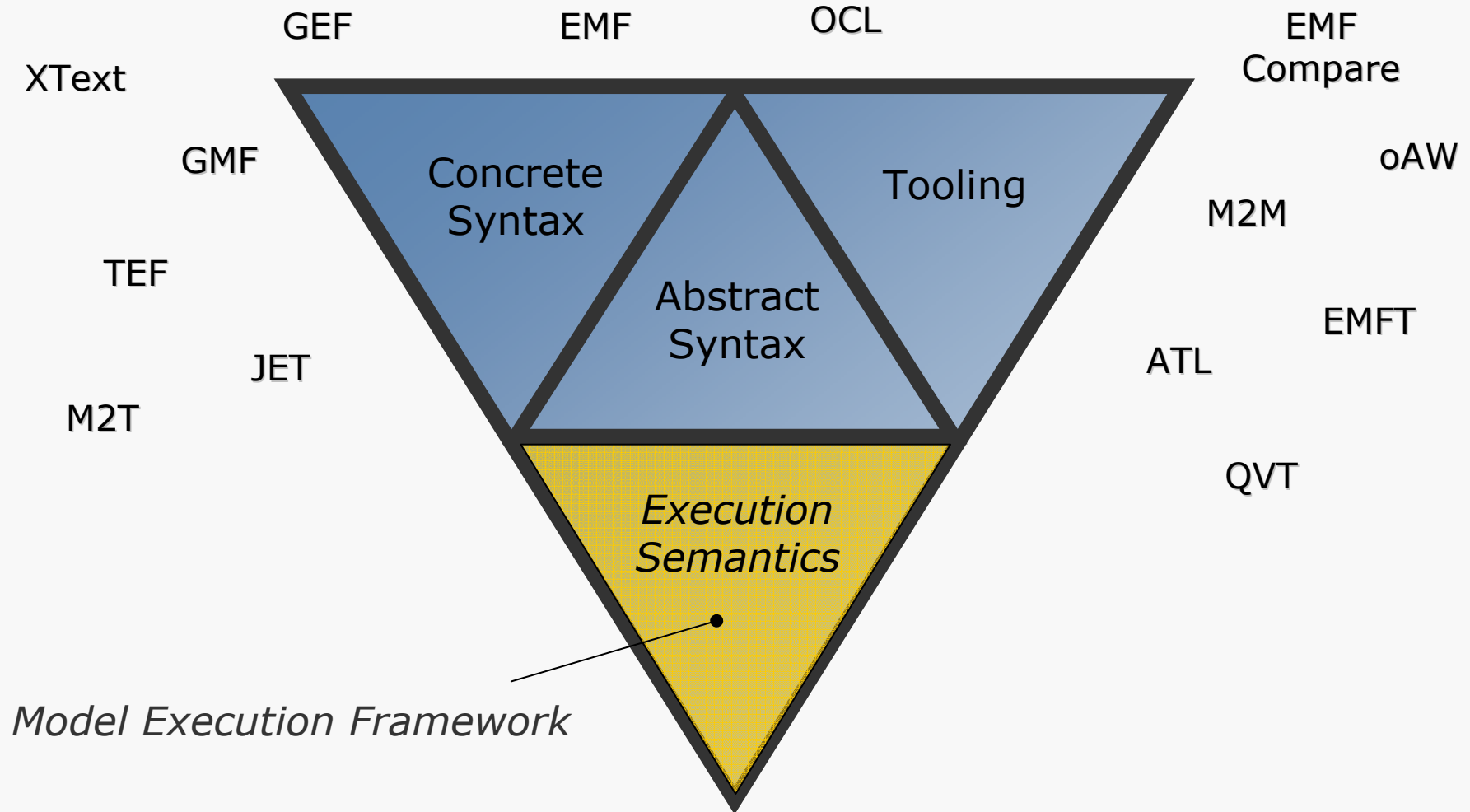


Executive Summary

- MXF aims to provide a framework for development, execution and debugging of EMF models with operational semantics
- Codebase of the project is the M3Actions framework that has been developed at the Humboldt University Berlin



Background: Language Development with EMP



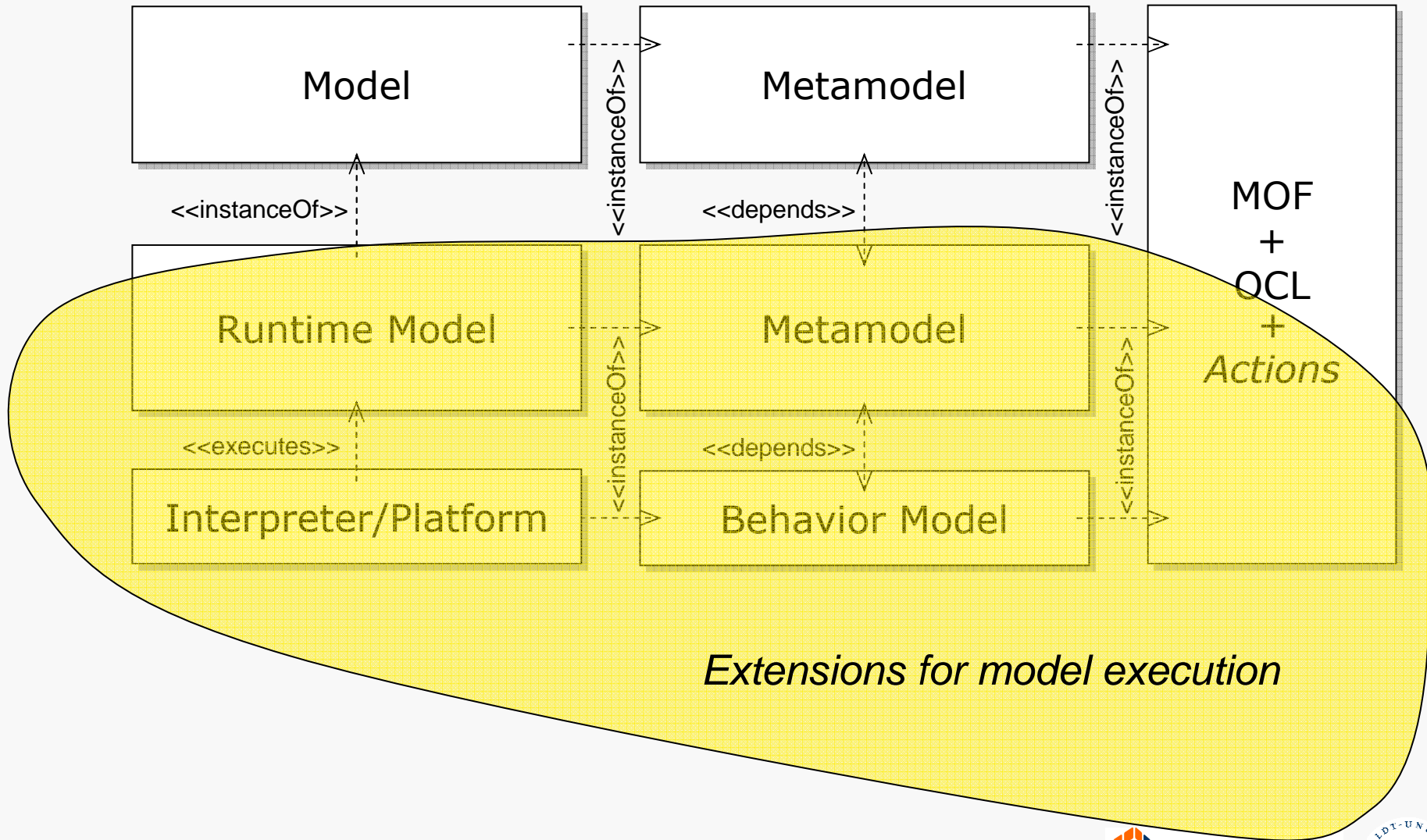
Motivation: Model Execution Semantics

- Operational semantics are model transformations
 - Special purpose language to express execution semantics (“action scripts”)
 - Focus on continuous model updates instead of single source-to-target transformation
 - Modeling operational semantics enables simulation of models without generating/implementing code

- For EMF models, execution semantics are currently either expressed as M2M transformations or coded in Java
 - No execution framework supports development (except eclipse debugging framework)
 - No modeling of execution semantics



Concepts

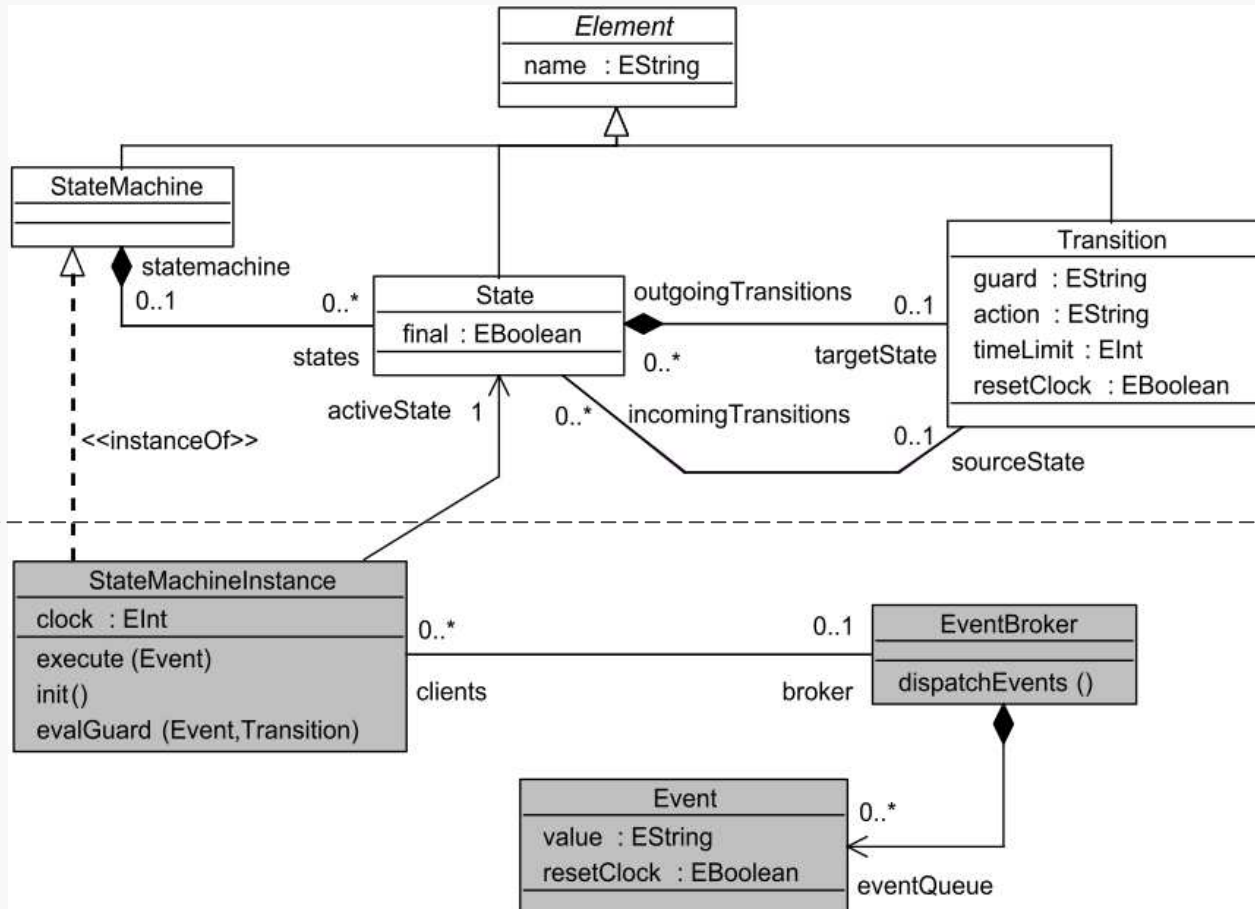


M3Actions: Operational Semantics for EMF

- M3Actions is the codebase of the MXF project
 - It provides an action language called *MActions* supporting graphical modeling of operational semantics. The notation is based on UML Actions/Activities and OCL for queries and constraints
 - *MAction* definitions are annotated inline as operations in an EMF model that can be interpreted (no code generation necessary)
 - Explicit instantiation concept supports definition of meta-layers, e.g. used to separate runtime model from language model



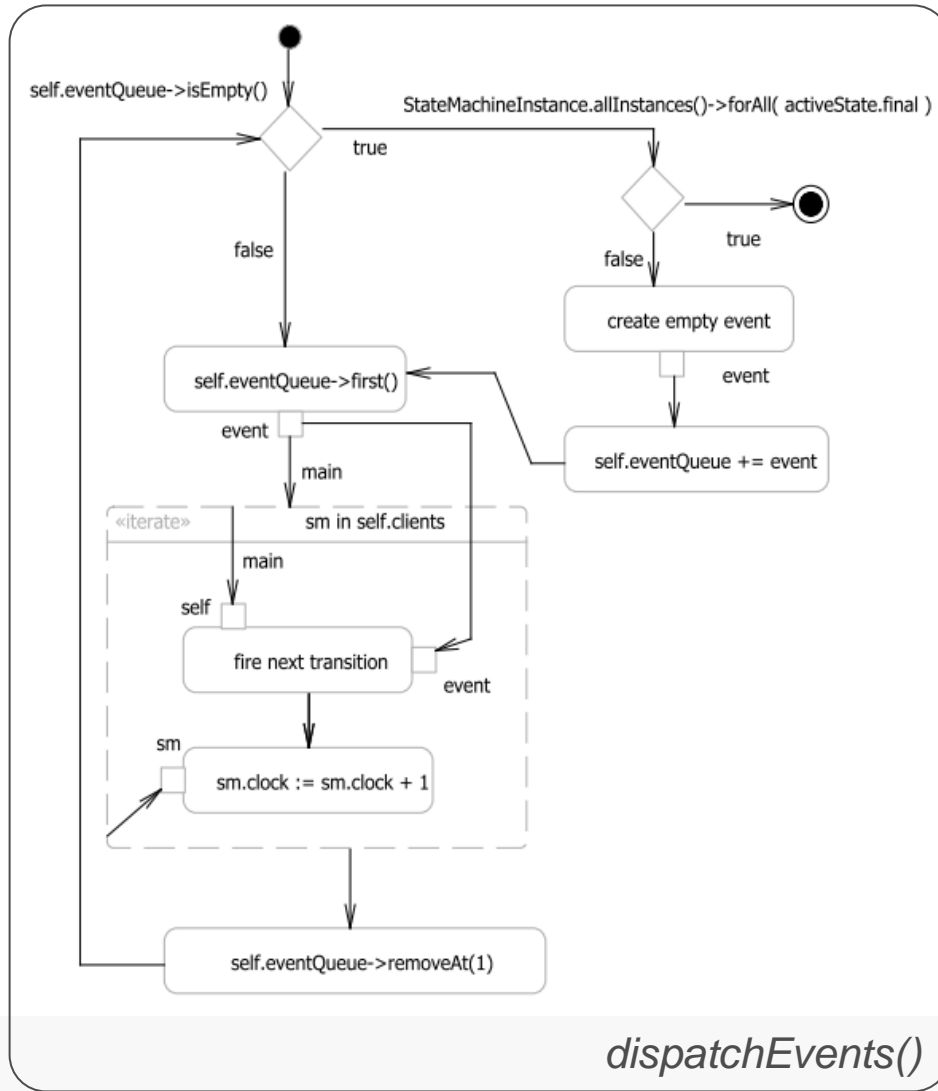
Example Metamodel: *Timed Automata*



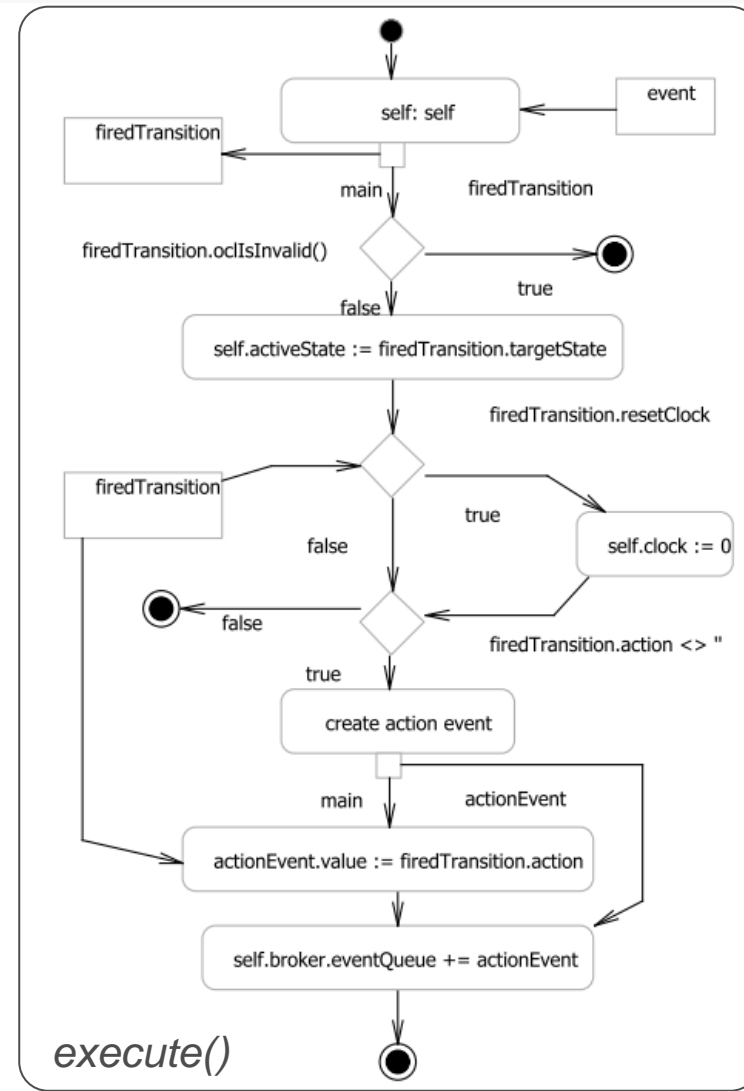
*Abstract syntax
model*

Runtime model

MActions behavior for the example



dispatchEvents()



execute()

MXF Project Scope

- Editor for language definitions
 - Graphical editor capable of specifying the action semantics of EMF models. The definitions include the abstract syntax of a language, the runtime model (both EMF models) as well as the behavior model.
 - Concepts are comparable to GMF mapping model, where the metamodel classes are related (via instance-of) to runtime model classes for execution (instead of semantic model to notation model mapping)
- Interpreter, debugger, common execution infrastructure
 - The interpreter/debugger provides reflective interpretation/debugging of behavior definitions without code generation. It works over the common execution infrastructure that defines concisely step/step-into and update behavior, and provides hooks for additional tooling (e.g. trace recording, execution listeners, UI updates, etc.)
 - Multi-threading will be supported, including a set of concurrency patterns for the language definitions. The execution infrastructure shall further define a concept of simulation time.



MXF Project Scope (contd.)

- Execution Trace API and implementation
 - A generic trace model API and adapter concept to record execution runs. The trace model is defined by a generic trace ecore model
 - Sample implementation of a trace recorder that performs recording of runs either file-based or via database.
- Generator to produce specific simulators
 - While the generic interpreter provides direct execution, the generator derives implementation code that works in the common execution infrastructure. It optimizes e.g. the reflectively interpreted OCL and can serve as basis for custom implementations, e.g. for a GMF editor of the language being defined.
 - In the long term, we want to develop GMF adapters that might be produced semi-automatically for developing language-specific debuggers on top of GMF editors



MXF Project Plan

- First step is to move current development from sourceforge.org to eclipse CVS
 - Renaming of plugins, classes and metamodels
 - better integrate with existing projects (e.g. Ecore tools)
- Milestone 1 (approx. until October 2009)
 - revise and stabilize existing code base according to eclipse quality standard
 - Setup more generic common execution infrastructure, extension points, etc.
 - complete debugger implementation (multi-threading)
- Collaborate and discussion further with interested parties
 - continue integration talks (e.g. KerMeta, TopCased, etc.) for execution infrastructure and other code contributions
- Plan for after October:
 - start with realization of code generators for specific generators
 - schedule identified extensions from other interested parties



Code Contribution Outline

- The M3Actions open-source project is hosted at sourceforge.org (<http://sourceforge.net/projects/m3actions/>)
 - License is Eclipse Public License (EPL)
 - Active contributors are the committers listed for the project (no other contributions)
 - Documentation and publications available at <http://www.metamodels.de>
- Source code movement
 - All plugins will be moved, changing their prefix from „hub.sam.mof“ to „org.eclipse.mxf“
 - .core models and code parts will be renamed from „M3Actions“ to MXF; classes with that prefix accordingly
- No trademarks are associated with the existing code
 - Copyright is owned by the authors: Michael Soden, Hajo Eichler
 - Licensing: 3rdParty plugins/libraries are solely EMP/eclipse plugins under EPL



Relation to existing eclipse projects

- EMF, OCL, GMF projects
 - base of the implementation, project might feedback extensions to these projects

- EcoreTools
 - For all editors we plan to integrate with EcoreTools such that runtime models and behavior definitions can be defined as additional view on an pre-existing ecore model

- Debugging Framework
 - The model execution framework will be build on top of the debugging framework, presumably providing extensions (e.g. GMF source lookup implementation)



Mentors

- Ed Merks (Macro Modeling)
- Sven Efftinge (itemis AG)



Initial Committers

- Michael Soden (ikv++ technologies)
 - Works as lead architect at ikv and has founded the M3Actions project. Michael has years of experience with eclipse and .NET technology and development of model driven solutions using EMP.

- Hajo Eichler (ikv++ technologies)
 - He works as quality assurance manager for all model driven development at ikv and is one of the experts for the open source project *medini QVT* (released under EPL). Hajo has years of experiences in development of eclipse based projects.



Initial Committers (contd.)

- Markus Scheidgen (Humboldt University Berlin)
 - Markus has several year of experience as professional software engineer in fields like web applications and modeling of distributed systems. He headed the development of several open-source eclipse plug-ins, including TEF, a framework for the automatic creation of textual eclipse modeling-based editors. He recently received a doctoral degree for his thesis on modeling of computer languages and automated development of language tool support based on EMP



Interested Parties & Feedback

- List of interested parties
 - ikv++ technologies ag - Dr. Marc Born (born@ikv.de)
 - Humboldt University Berlin - Prof. Joachim Fischer (fischer@informatik.hu-berlin.de)
 - itemis AG - Wolfgang Neuhaus (wolfgang.neuhaus@itemis.de)
 - Fraunhofer FOKUS - Tom Ritter (Tom.Ritter@fokus.fraunhofer.de)
 - Obeo - Stephane Lacrampe (stephane.lacrampe@obeo.fr)
 - CEA LIST - Sèbastien Gérard (sebastien.gerard@cea.fr)
 - Anyware Technologies - David Sciamma (david.sciamma@anyware-tech.com)
 - INRIA - Didier Vojtisek (didier.vojtisek@irisa.fr)
 - FeRIA - Marc Pantel (Marc.Pantel@enseeiht.fr)
 - Atos Origin - Raphaël Faudou (raphael.faudou@atosorigin.com)
 - University of Agder - Terje Gjøsæter (terje.gjosater@uia.no)

- Feedback from the community
 - Exclusively positive feedback, much interest in joint work on the issue
 - increasing list of interested parties (from initially 3 to 11 by now)

