FROM RESEARCH TO INDUSTRY

cea tech

EXECUTABLE MODELING WITH FUML AND ALF IN PAPYRUS: TOOLING AND EXPERIMENTS

Sahar Guermazi*, Jérémie Tatibouet*, Arnaud Cuccuru*,
Ed Seidewitz+, Saadia Dhouib*, Sébastien Gérard*

* CEA LIST - LISE lab
+ Model Driven Solutions

list

Models 2015 – EXE Workshop

INSTITUT CARNOT CEA LIST

digiteo

OUTLINE

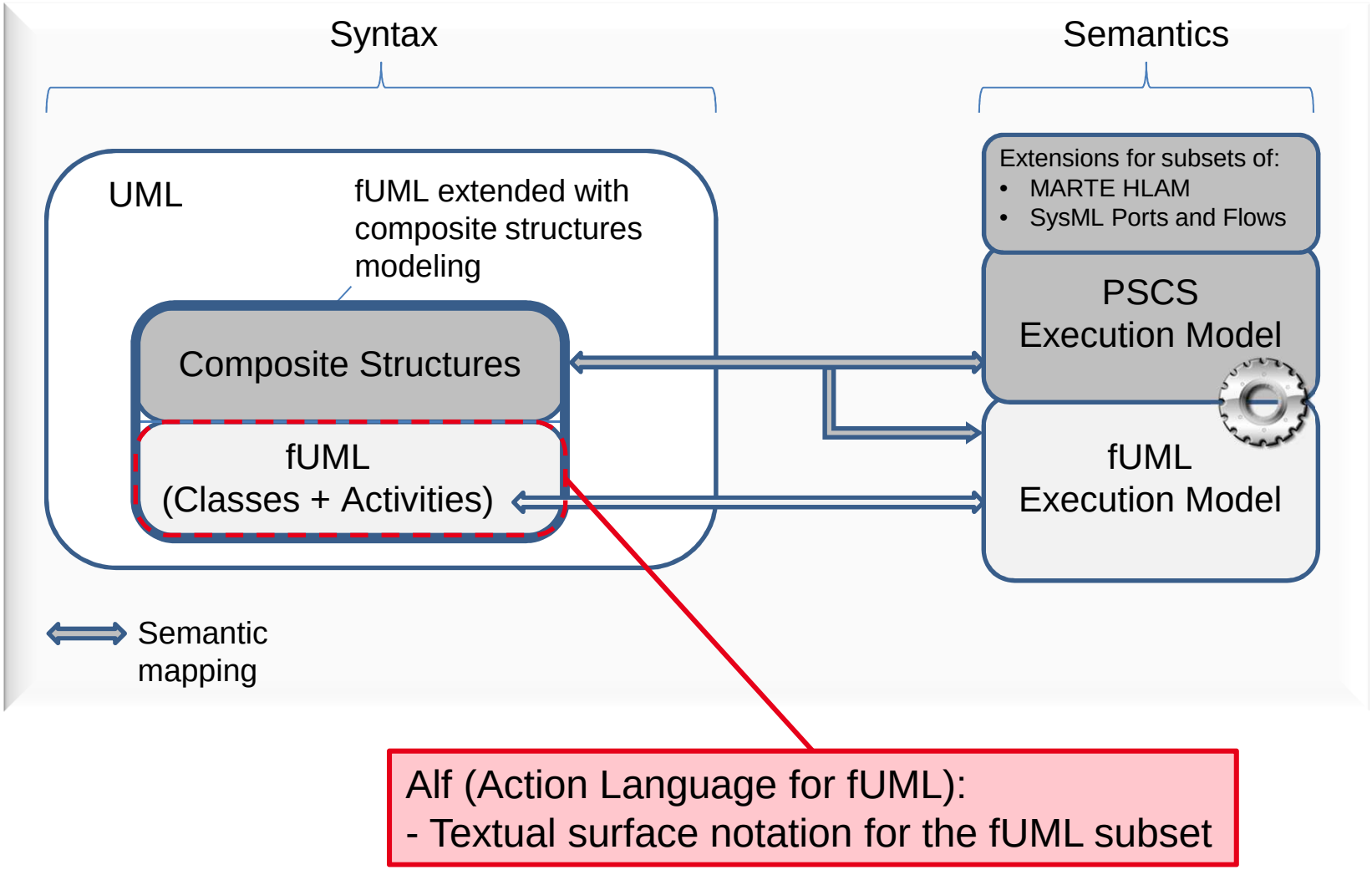
. REMINDER ON FUML / ALF, AND PAPYRUS ECOSYSTEM

. DESIGNING FUML-BASED EXECUTION ENGINES

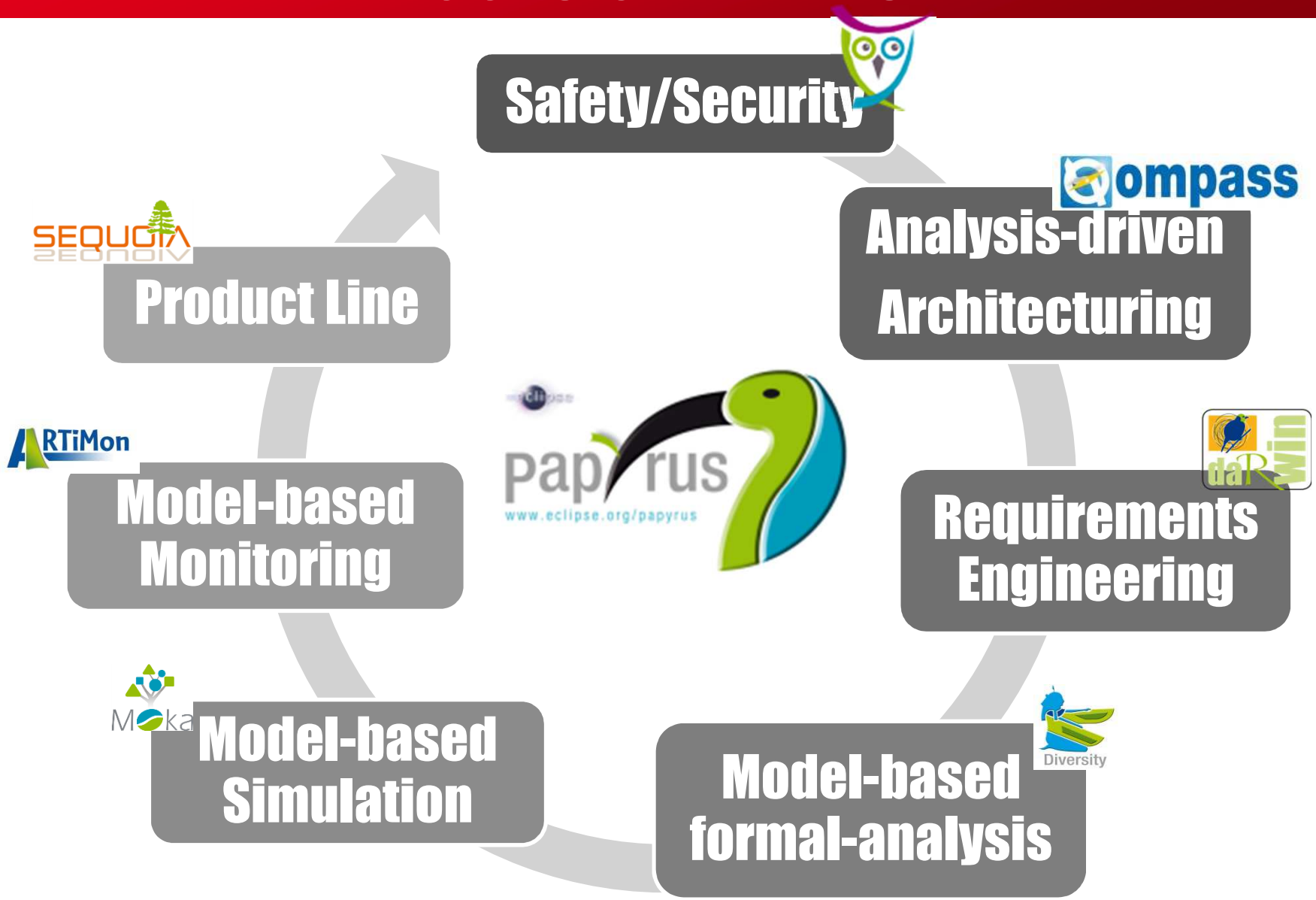
. COMBINING ALF AND UML (AND ITS PROFILES)

. CONCLUSIONS AND FUTURE WORKS

REMINDER ON OMG STANDARDS: FUML, PSCS AND ALF



MDE TOOLS OF THE LISE LAB



DESIGNING FUML-BASED EXECUTION ENGINES

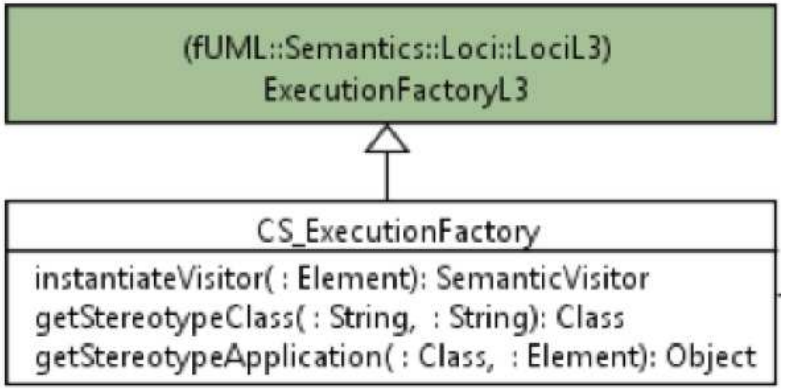
INTRODUCTION

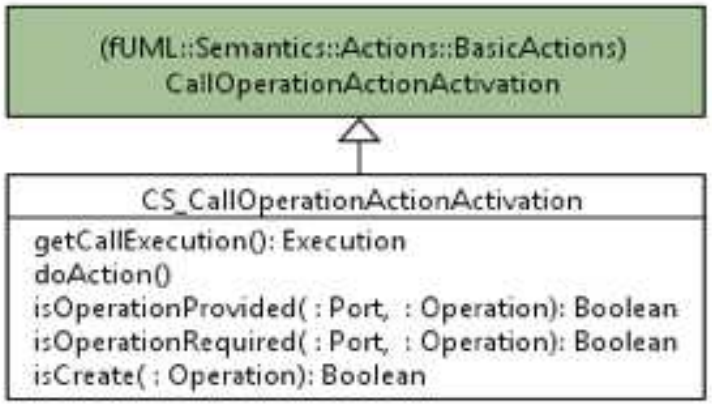
- **Moka's main goal: Provide a generic execution envt. for Papyrus**
  - Should be reusable for any user or domain-specific use of the tool
- **Moka is based on the fUML execution model:**
  - A domain specific flavor of Papyrus/Moka comes down to a specialization of the execution model for that domain
  - Object-oriented execution model => Specialization based on inheritance and polymorphism
- **4 key challenges to address:**
  - Extensibility
  - Control and observation
  - Time support
  - Connectivity with external tools

EXTENSIBILITY

- **Problem statement**
  - A DSL implemented as a UML profile may require abstract syntax elements that are out of the scope of the fUML subset.
  - It may also introduce stereotypes specializing fUML syntax and semantics.
- **Key fUML aspects:**
  - Execution model designed with the Visitor and Factory patterns
- **Proposed solution:**
  - Extend the fUML visitors and factories!

- **Experiments**
  - **PSCS Execution Model**





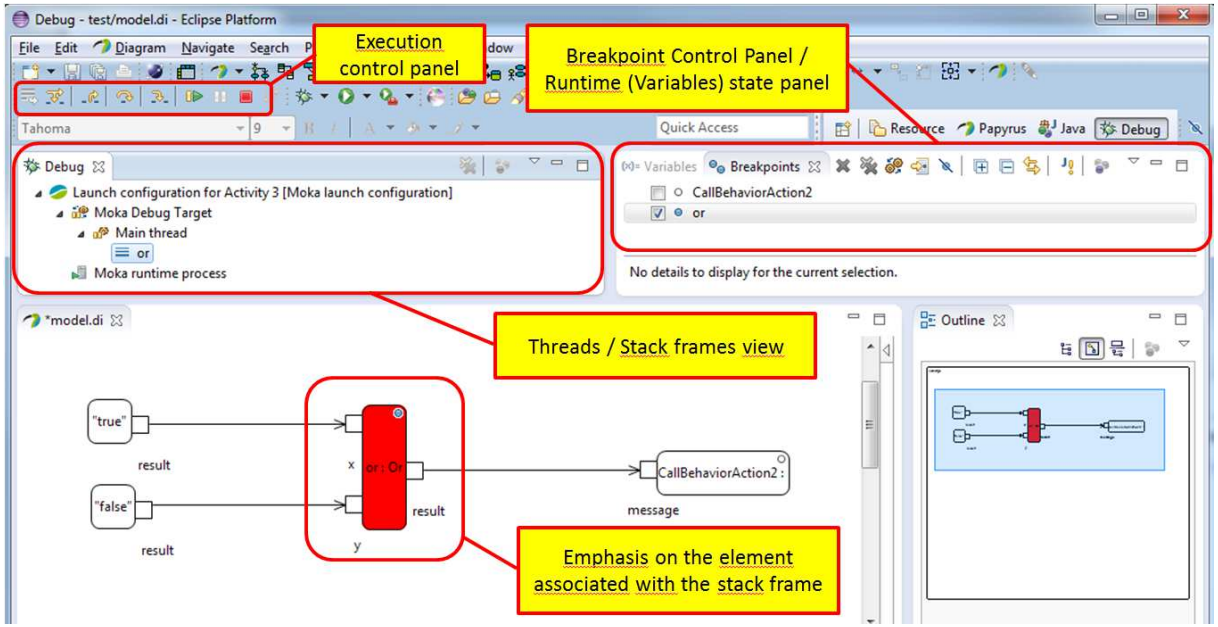
- **Limitations:**
  - **Factory pattern not used everywhere in the fUML execution model**
  - **No mechanisms so far to “elegantly” deal with multiple profile applications**

CONTROL AND OBSERVABILITY

- **Problem statement**
  - Modern execution / simulation tools should provide users with facilities to control (start/stop, suspend/resume, step by step, etc.) and observe (diagram animation, tracing) executions
- **Key fUML aspects:**
  - These are tooling concerns: Out of the scope of fUML
  - The fUML execution model simply enforces partial execution orders through token propagation rules (semantics of activities)
- **Proposed solution:**
  - Reroute the token propagation flow through explicit control and observation entities (Control delegation)
  - Can be done using Extensibility mechanisms

CONTROL AND OBSERVABILITY

- **Experiments**
  - Connection with the Eclipse Debug Framework
  - Tracing



- **Limitations:**
  - Currently lacks a systematic methodology for identifying the points where control needs to be extracted

TIME SUPPORT

- **Problem statement**
  - The fUML is a good basis for a simple simulation process (model, execute, observe, and refine)
  - OK for checking logical correctness – KO for timing aspects
- **Key fUML aspects:**
  - The fUML execution model is time-agnostic
- **Proposed solution:**
  - In widespread simulation tools, time is usually managed by an explicit control entity (scheduler-like)
  - => Rely on Control delegation

TIME SUPPORT

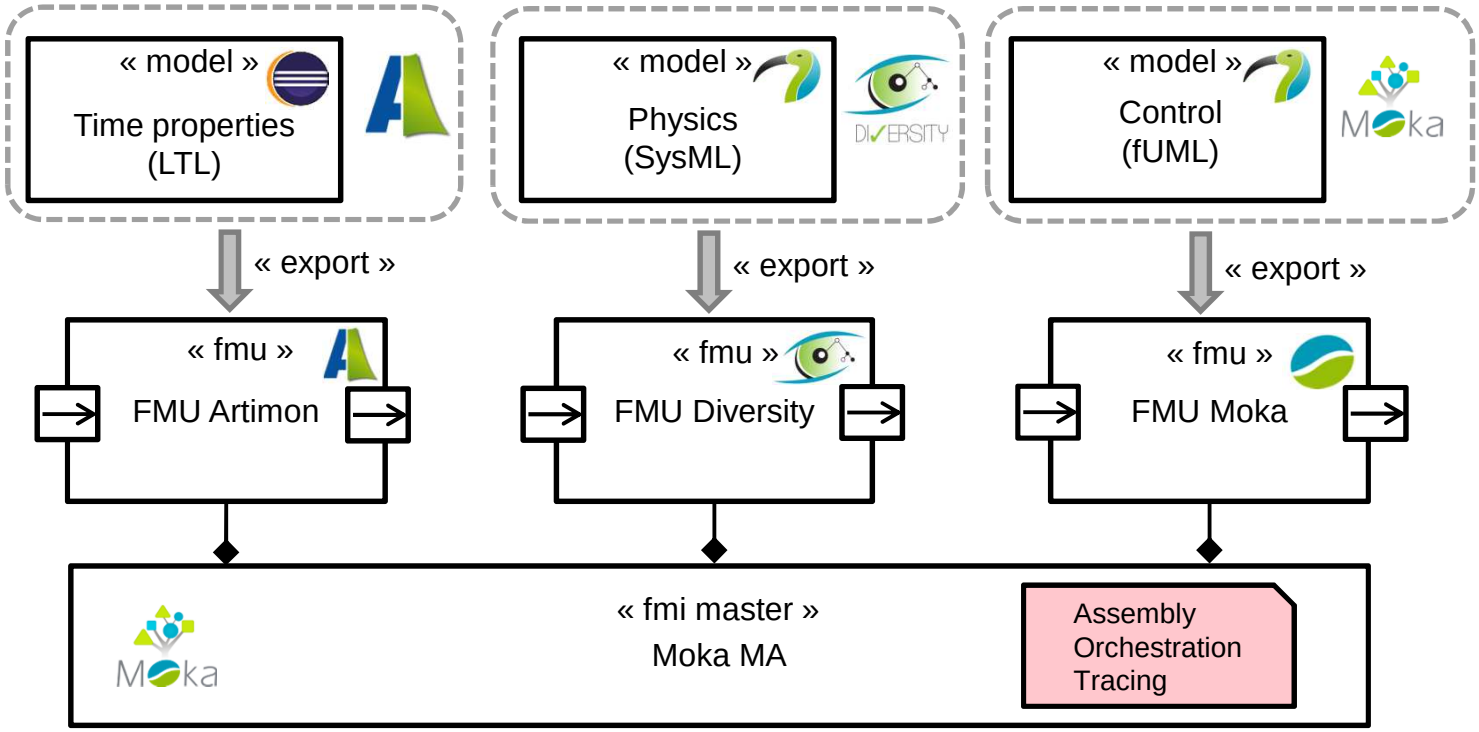
- **Experiments**
  - **Integration of a Discrete Event Scheduler**
  - **Application to the timed simulation of BPMN processes (demo video https://www.youtube.com/watch?v=ddogjaCtEbE)**

- **Limitations:**
  - **We only experimented on the integration of the discrete event time model.**
  - **Other experiments are required to determine if the approach is valid for other time models.**

CONNECTIVITY WITH EXTERNAL TOOLS

- **Problem statement**
  - Complex systems involve multiple engineering disciplines (mechanical, electrical, computer science).
  - Require multiple simulation tools, and co-simulation facilities
- **Key fUML aspects:**
  - The fUML subset includes Classes and Opaque Behaviors
  - Class instances are represented by Objects
  - Opaque Behaviors are bound to tool-specific implementations (so-called Opaque Behavior Executions)
- **Proposed solution:**
  - Wrap the connection with external simulations in specific Objects and Opaque Behavior Executions

- **Experiments**
  - Support for FMI 2.0 standard (Import of FMUs, Master algorithm)



- **Limitations:**
  - This is a work in progress.

COMBINING ALF AND UML (AND ITS PROFILES…)

USE CASES AND EXPERIMENTS
(CF. ARTICLE FOR MORE DETAILS AND REFERENCES, AND DEMO BY ED SEIDEWITZ AT THE OCLTM WORKSHOP)

- **Mixing graphical and textual notations:**
  - **Combining at the Unit Level (Classes, Activities, Packages)**
    - Key challenge: Synchronization of both views
  - **Combining at the Expression and Statement Level**
    - Key challenge: Modularity of the grammar implementation

- **Combining with profiles:**
  - **Experiment: Refactoring of MARTE's Value Specification Language as an extension of Alf**
  - **Extending the Alf grammar is a complex task…**

CONCLUSION

- **The Papyrus team develops open-source tools for executale modeling and simulation, based on OMG standards fUML* and Alf**
  - **Moka: an extendible framework for the execution of UML(-based) models**
  - **Alf integration: coupling textual with graphical notations for efficient executale modeling**

- **Two kinds of challenges and limitations have been identified**
  - **Tooling issues**
    - To be adressed as part of the Papyrus development roadmap
  - **Specification issues**
    - To be considered by the OMG's Executable UML WG