Business Informatics Group

Vienna University of Technology

Towards xMOF: Executable DSMLs based on fUML

www.modelexecution.org

Tanja Mayerhofer, Philip Langer, Manuel Wimmer

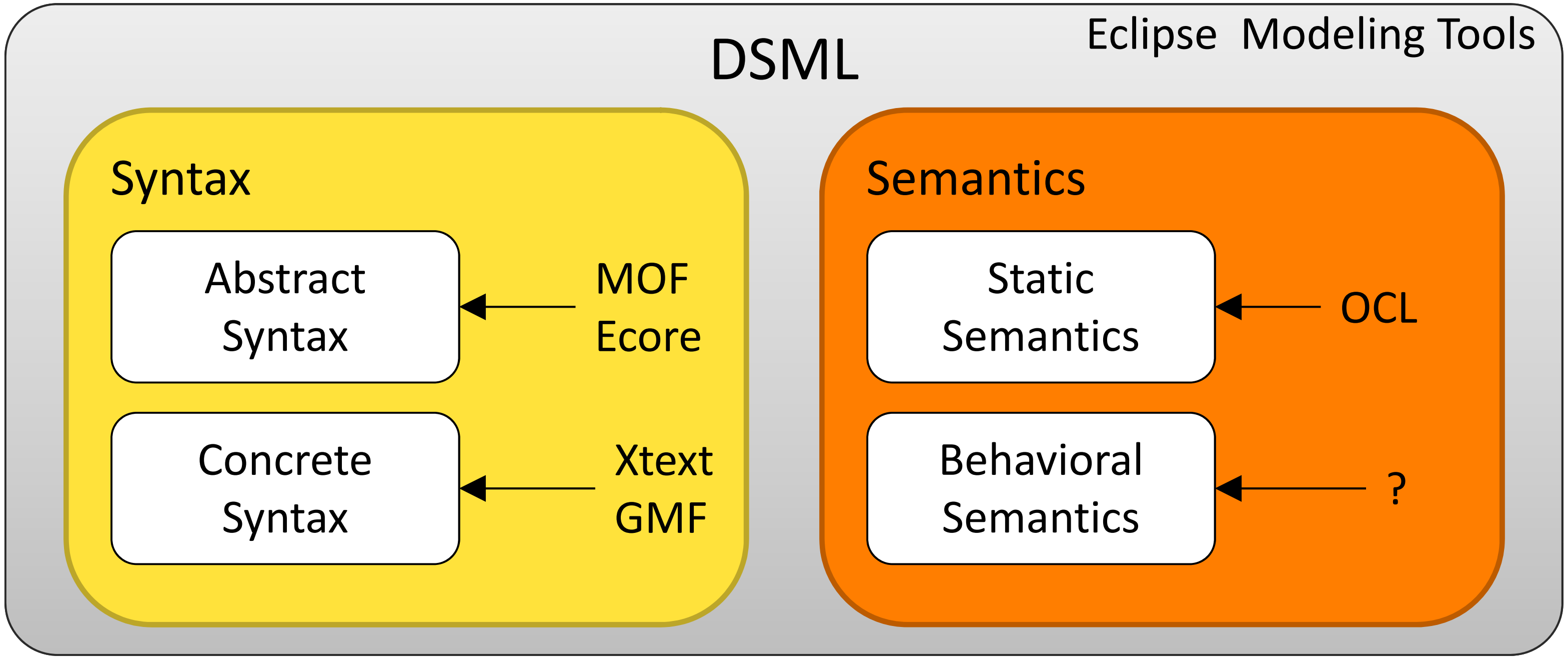
Business Informatics Group

Institute of Software Technology and Interactive Systems
Vienna University of Technology

Favoritenstraße 9-11/188-3, 1040 Vienna, Austria
phone: +43 (1) 58801-18804 (secretary), fax: +43 (1) 58801-18896
office@big.tuwien.ac.at, www.big.tuwien.ac.at

Domain-Specific Modeling Languages

- Success of MDE depends on availability of means for defining DSMLs



- No standard means for specifying behavioral semantics of DSML exist

→ **Efficient development of model execution facilities impossible**

Semantics of Domain-Specific Modeling Languages

- **Denotational / translational semantics**

  Examples:

  - *Abstract State Machines*
    K. Chen, J. Sztipanovits, S. Abdelwalhed, E. Jackson. Semantic anchoring with model transformations. In Proc. of ECMDA-FA'05, pages 115-129, 2005.
  - *Maude*
    J. E. Rivera, F. Duran, and A. Vallecillo. On the behavioral semantics of real-time domain specific visual languages. In Workshop Proc. of WRLA'10 @ ETAPS'10, pages 174-190, 2010.

  Pros:

  - Execution and analysis tools can be reused

  Cons:

  - Mapping model into target language is complex
  - Results have to be mapped back

Semantics of Domain-Specific Modeling Languages

- **Operational semantics**

  Approaches:

  - *Graph transformations*
    G. Engels, J. H. Hausmann, R. Heckel, and S. Sauer. Dynamic meta modeling: A graphical approach to the operational semantics of behavioral diagrams in UML. In Proc. of UML'00, pages 323-337, 2000.
  - *Action language*
    Kermeta, MXF, Smalltalk, Eiffel, xCore, Epsilon Object Language

→ **Use fUML as action language**

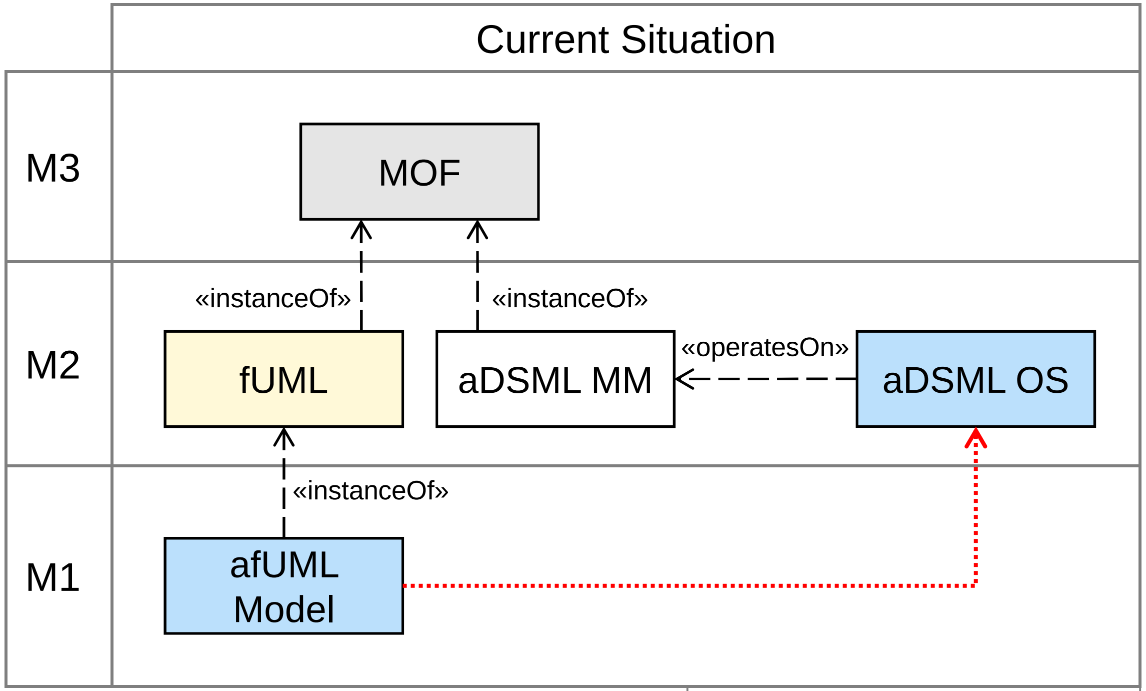
Foundational UML (fUML)

- **OMG standard** V1.0 released 02/2011
- Specification of behavioral semantics of **foundational UML subset**
  - **Structural kernel**: class, association, data type, etc.
  - **Behavioral kernel**: behavior, event, signal, etc.
  - **Activities**: activities, parameters, nodes , flows
  - **Actions**: communication, object, structural feature, link actions
- **Operational semantics** approach specifying a fUML virtual machine

➔ **UML activity diagrams can be executed**

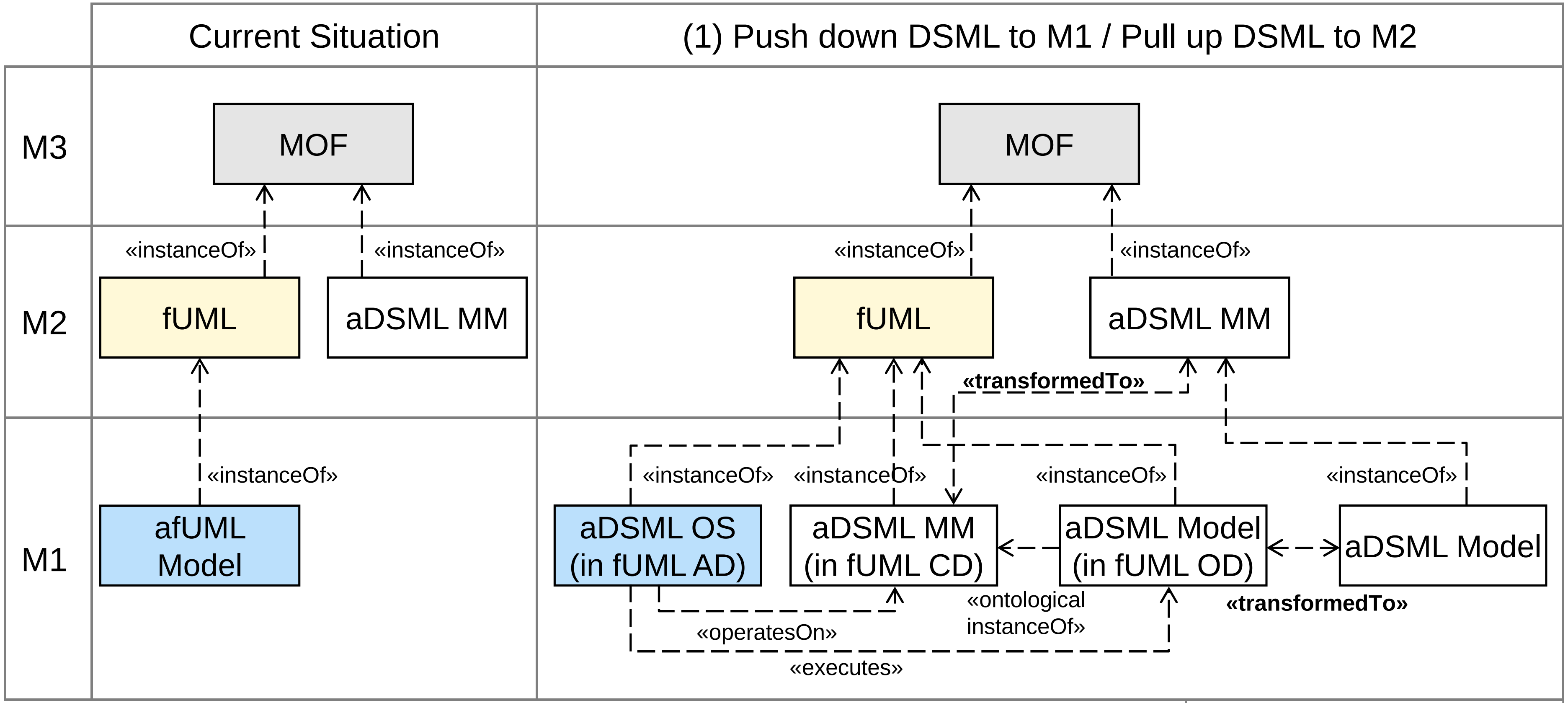
Specifying Semantics with fUML

- **Level mismatch** for specifying semantics of DSML using fUML activities



MM	...	Meta-Model
OS	...	Operational Semantics

Specifying Semantics with fUML



MM	...	Meta-Model
OS	...	Operational Semantics
AD	...	Activity Diagram
CD	...	Class Diagram
OD	...	Object Diagram

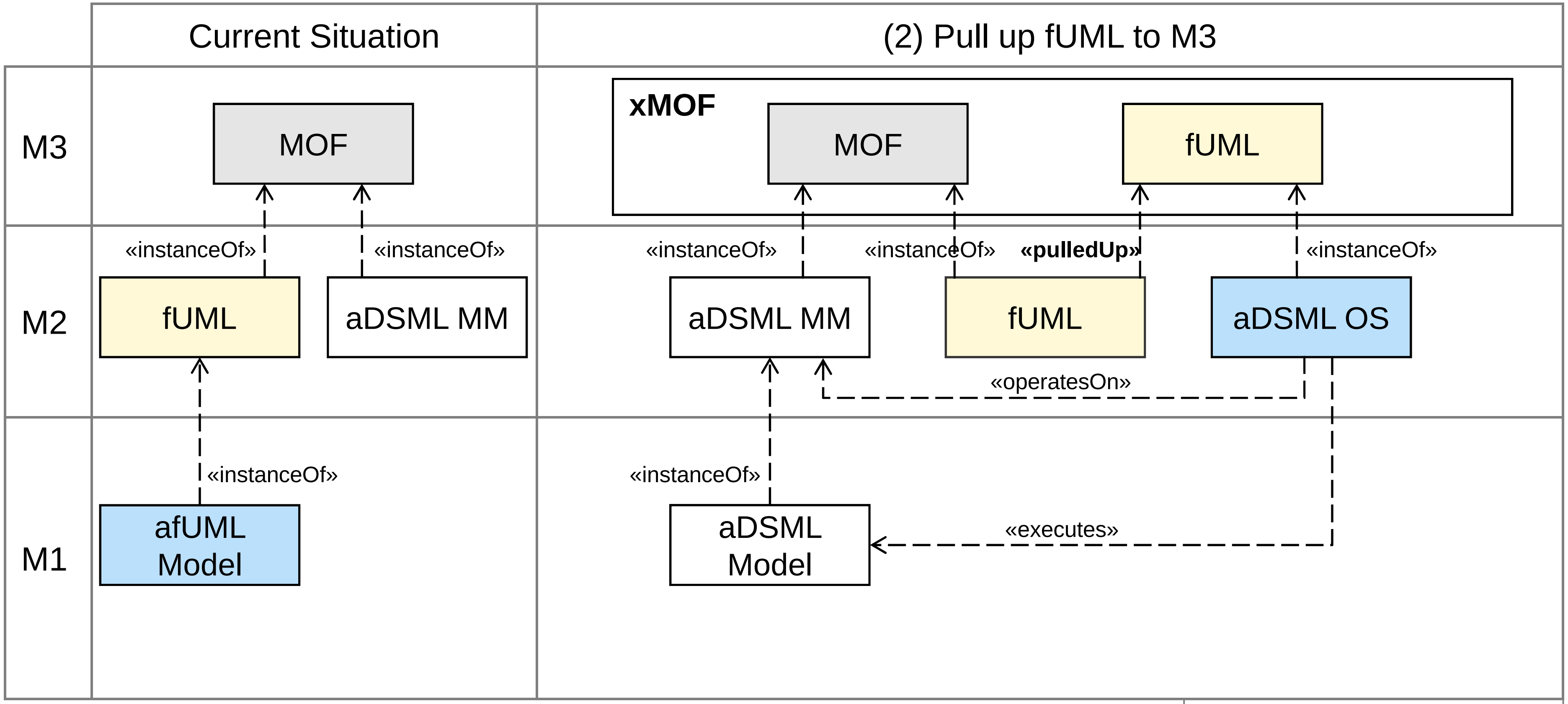
Pros

- Approach can be implemented using existing tools

Cons

- High effort for transformation needed
- UML environment has to be used instead of metamodeling environments

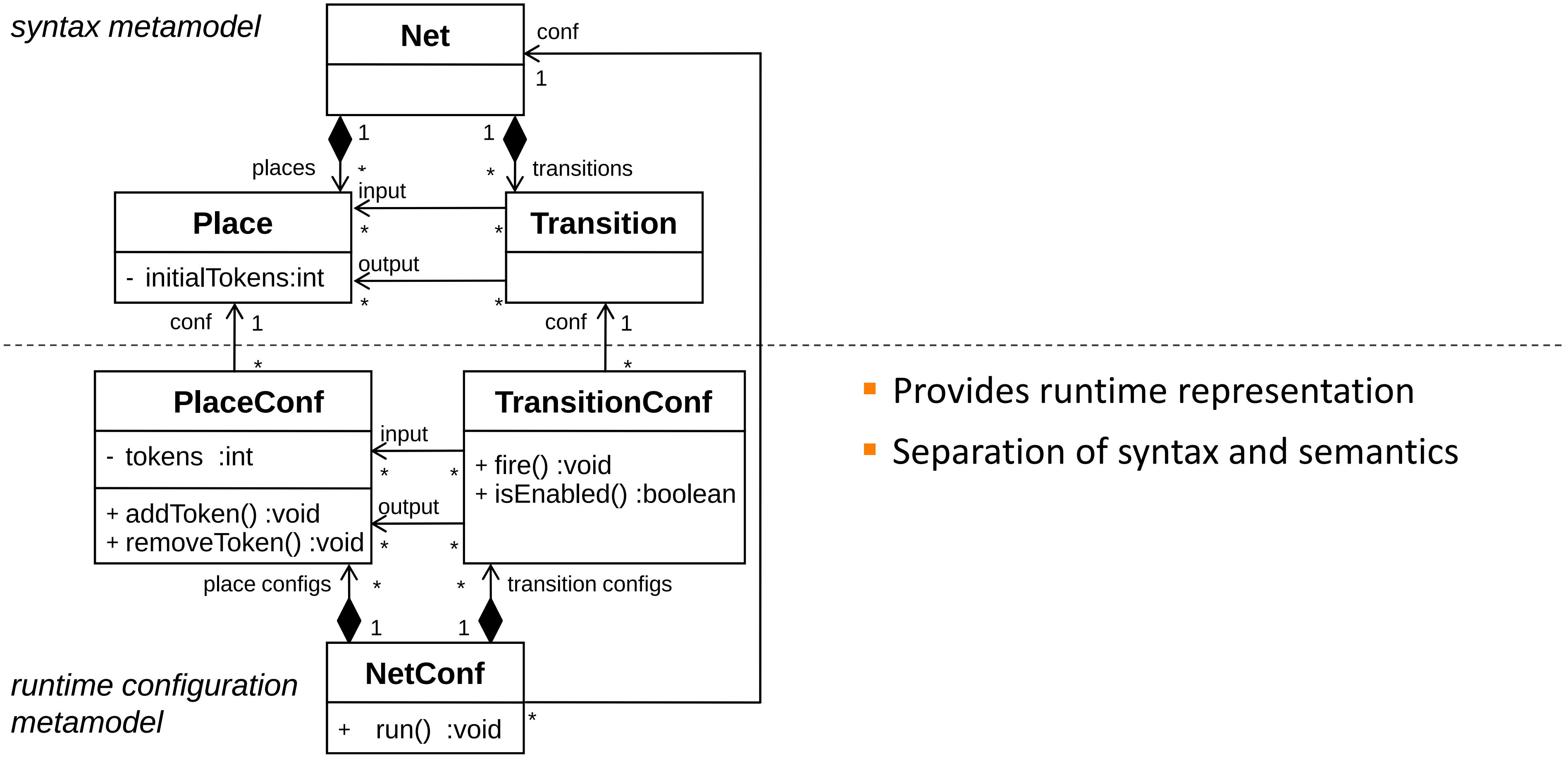
Specifying Semantics with fUML



MM	...	Meta-Model
OS	...	Operational Semantics

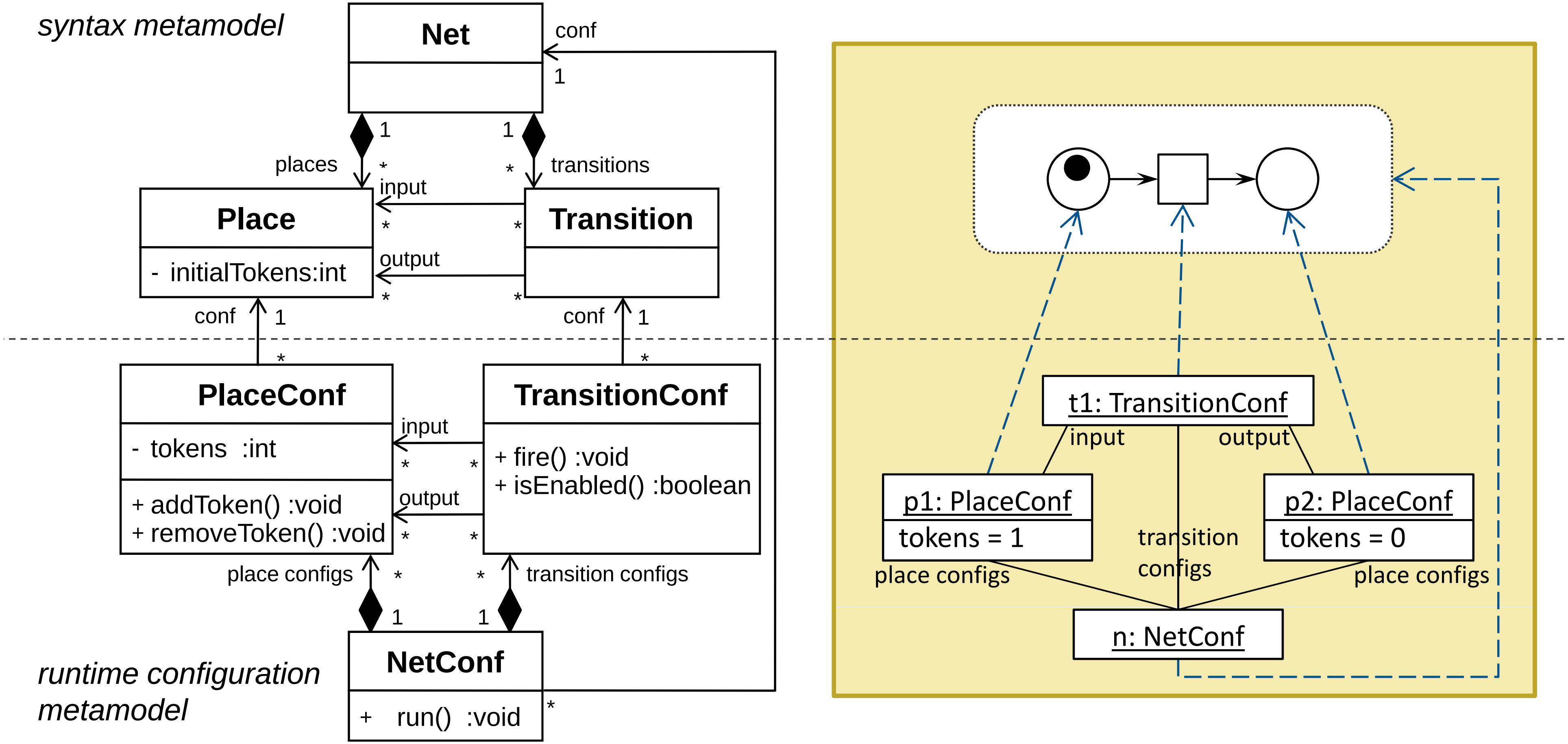
- Integrated metamodeling language **eXecutable MOF** (xMOF)
  - Abstract syntax: MOF
  - Behavioral semantics: fUML

Example: Petri Net

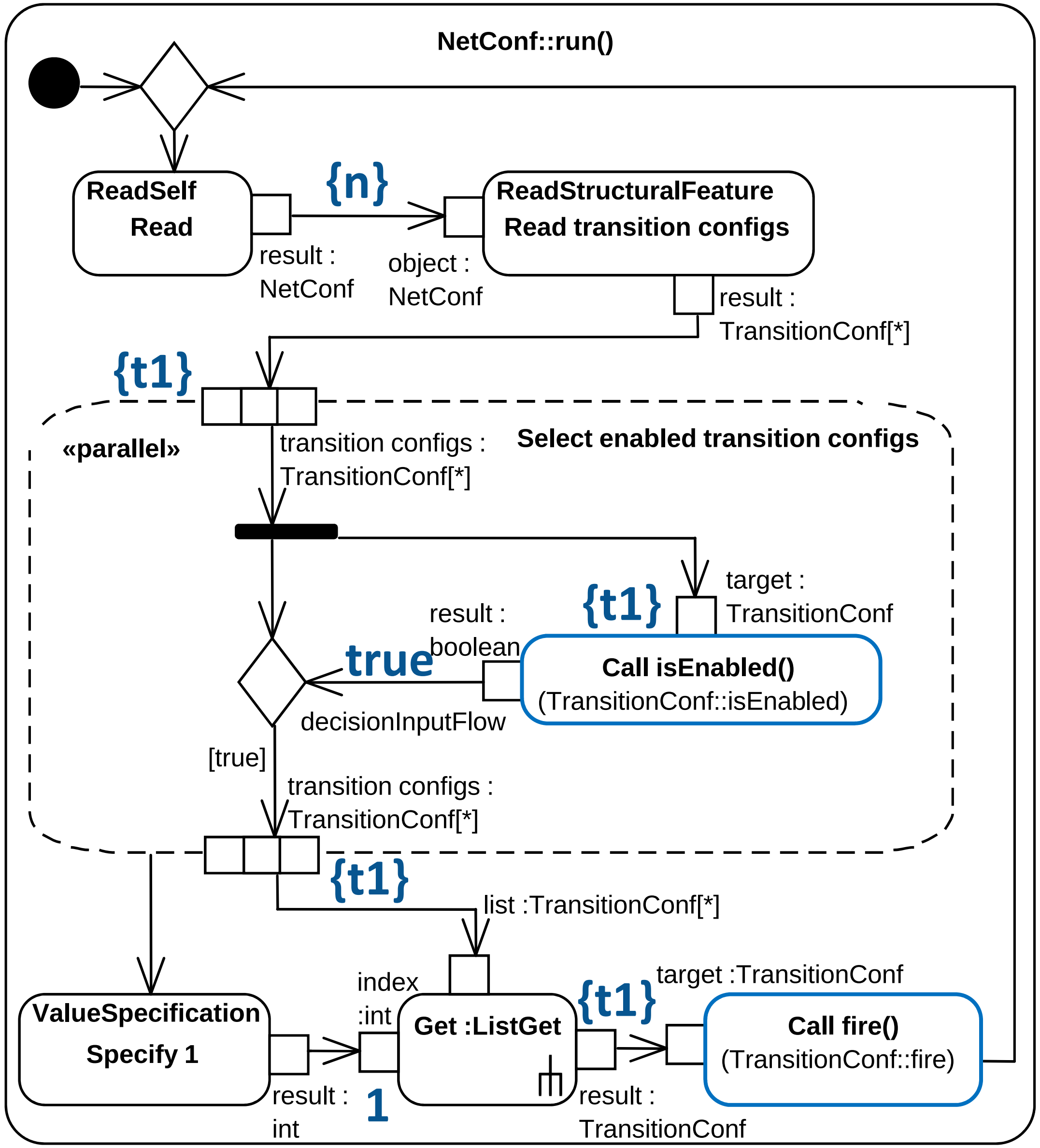


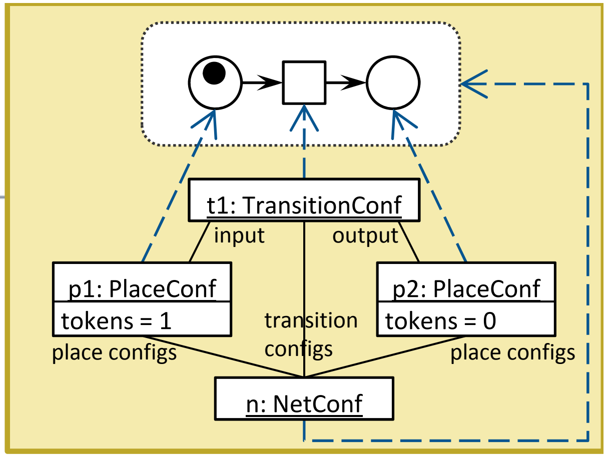
- Provides runtime representation
- Separation of syntax and semantics

Example: Petri Net

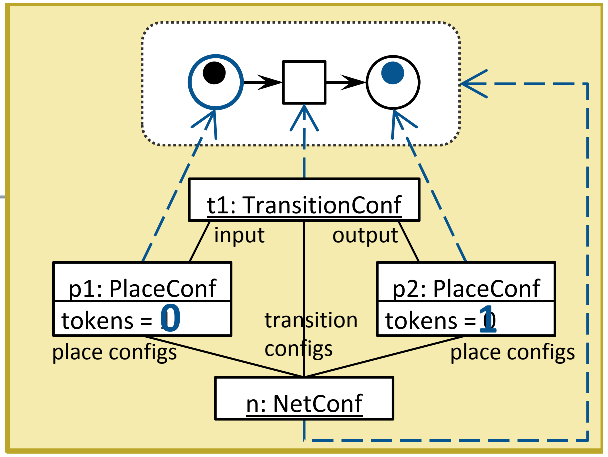


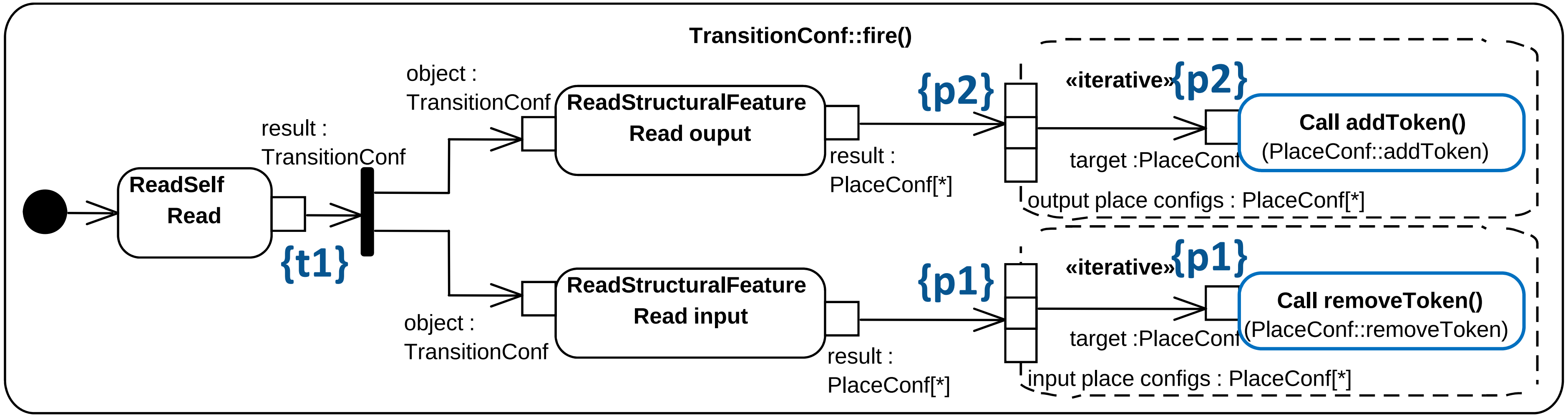
Example: Petri Net





Example: Petri Net





Extensibility of Semantics

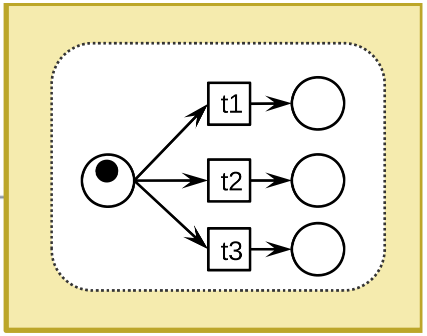
- Model everything down to the very last detail may not be feasible
- May require **utilization of libraries** which are not available for the fUML virtual machine

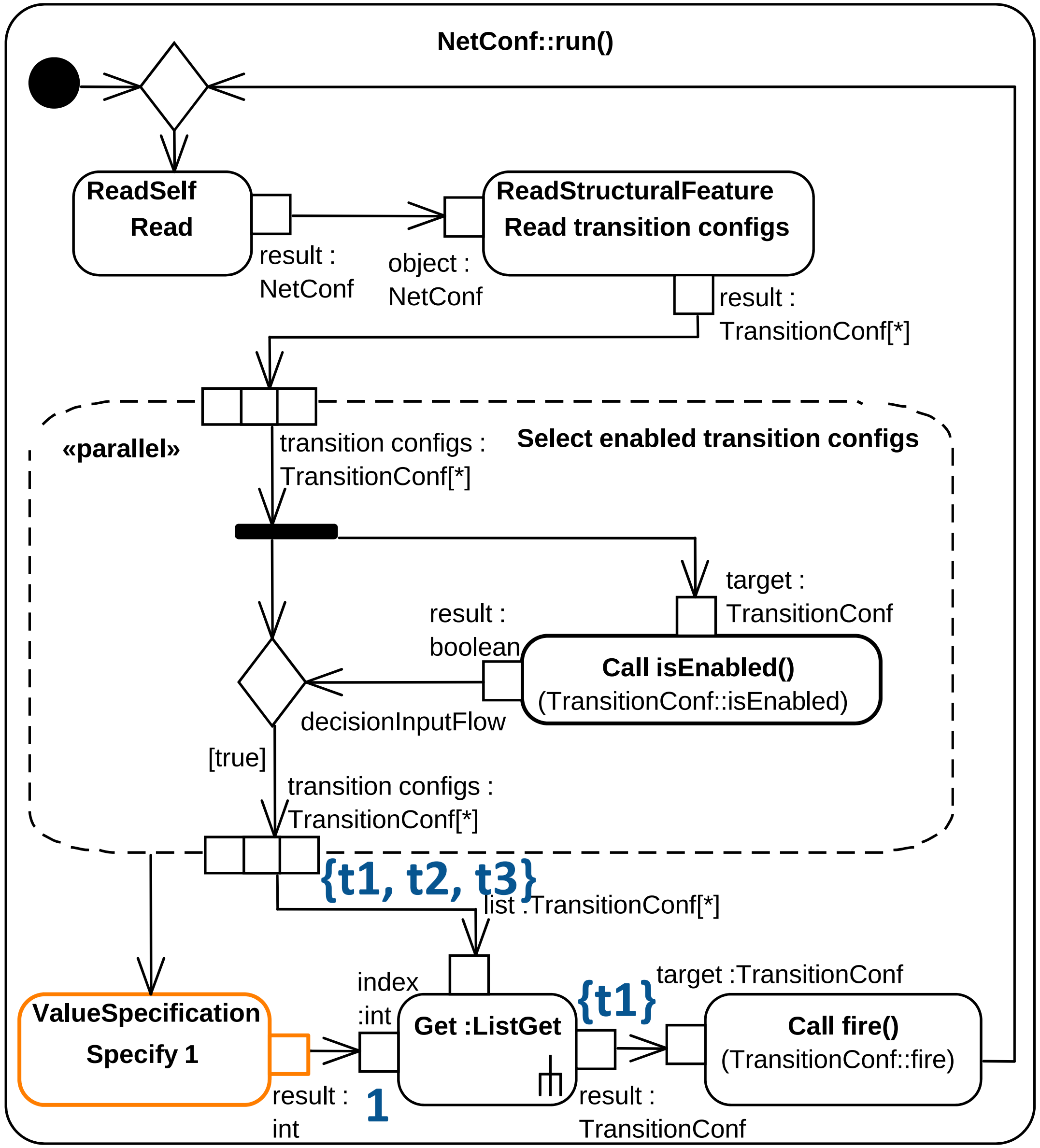
  Examples:
  - Complex mathematical calculations
  - Control of external resources

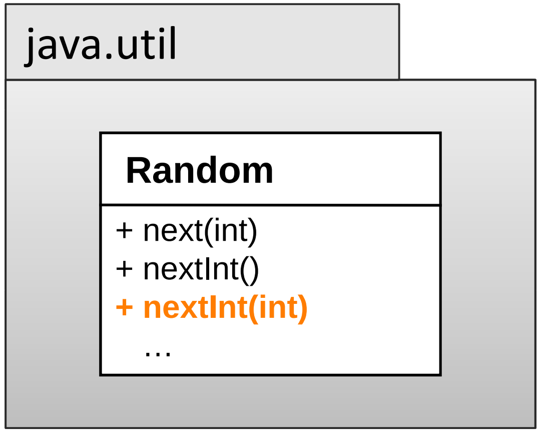
➔ **Integration of external libraries with fUML virtual machine**

- No extension of fUML metamodel and virtual machine
- Transparent usage of external libraries

Example: Petri Net







Extensibility of Semantics

1. Import of external libraries
   1. **Reverse engineering** of library for extracting API classes
   2. **Import of classes** into fUML model specifying the semantics of a DSML
   3. Create empty **activities** for each operation acting as **place holder**

2. Integration of external libraries at runtime

   Call of library operation

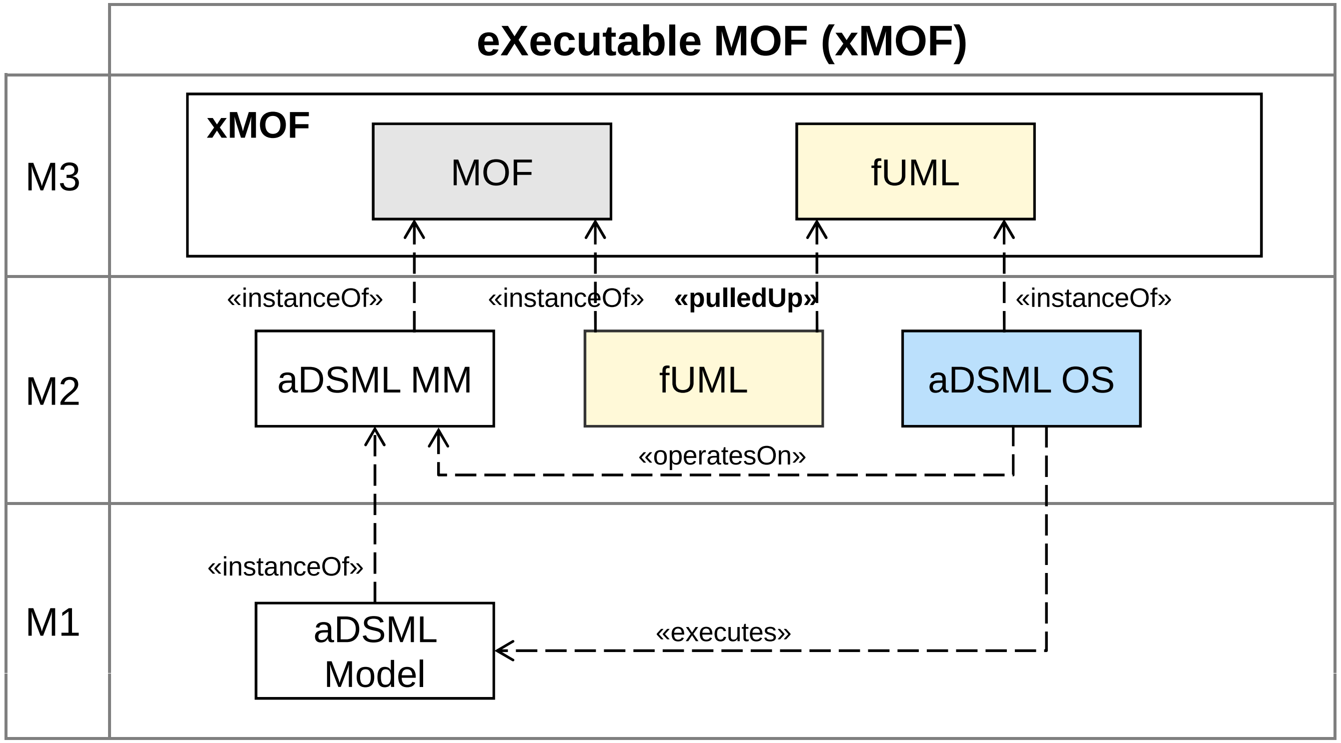
   1. **Suspend** execution at entry of place holder activity
   2. **Forward invocation** to actual operation of external library
   3. **Integrate result** into runtime model of execution

   Instantiation / Modification of library instances

   1. Maintain mapping between fUML instances and library instances

Conclusion

- Integrated metamodeling language **eXecutable MOF (xMOF)**
- Specification of **behavioral semantics** using **fUML**
- Usage of **external libraries**



Future Work

- **Implementation** of xMOF
  - Provide means for using fUML as semantics specification language
  - Conduct **case study** to show feasibility of our approach
- (Semi-)**Automatic generation** of model execution facilities
  - **Analysis** of execution: trace model
  - Runtime **observation** and **control**: event model, command API
- **Reusability** of semantics specifications
  - Definition of ***kernel semantics***, e.g. data flow, control flow, signal sending
  - **Composition** of *kernel semantics* for semantics specification of DSML
  - **Specialization** of existing semantics specification for variation point / profile

Thank you!

Debugging and Testing Models Based on fUML

www.modelexecution.org