NWPT 2015

HØGSKOLEN I BERGEN

BERGEN UNIVERSITY COLLEGE

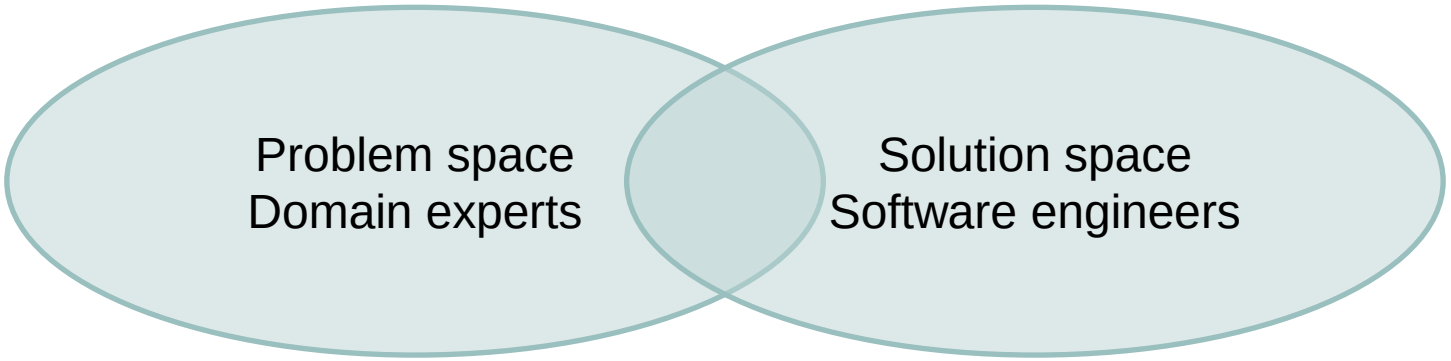
Runtime Verification of Executable Models

Fernando Macías – fernando.macias@hib.no
Adrian Rutle – adrian.rutle@hib.no
Volker Stolz – volker.stolz@hib.no

Motivation

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

- Modelling offers one more level of abstraction above programming
- Close the gap between domain experts and software engineers
  - › Different views of the system
  - › The solution can be specified in the problem space



Two types of models

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

Structural models

- The metamodel defines a type of structure
- The model represents a particular structure
- Semantics given by a set of instances (snapshots)

- E.g: Class diagrams

Behavioural models

- The metamodel defines a process language
- The model represent a process
- Semantics can be expressed as **model transformations**

- E.g: BPMN, Petri nets

Verification of Behavioural Models

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

- Testing
  - › Applied on small parts of the model
  - › Cumbersome in big models
  - › Not exhaustive
- Model checking
  - › Exhaustive and strong
  - › Bad scalability
- Runtime verification

Runtime Verification of Behavioural Models

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

- Useful when the system is too complex to be analysed thoroughly[1]
- Can be performed over simulations or the actual deployed system
- Based on:
  - Temporal properties: Invariants, implications of present/past events in future events, global properties (e.g: termination)
  - Monitors: Check properties against running instances

1. Leucker et al. *A brief account of runtime verification*.

Executable Modelling

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

- Definition of models with enough information to be executed
- Two alternatives
  - › Interpreted: The model itself is run in a custom runtime environment. The instances are evolved through model transformations[1]
  - › Compiled: The model is transformed into a machine-readable representation, e.g: imperative code[2]

- Focus on definition of interpreted process models

1. Guermazi et al. *Executable Modeling with fUML and Alf in Papyrus*
2. Dévai et al. *UML Model Execution via Code Generation*

HØGSKOLEN
I BERGEN

BERGEN UNIVERSITY COLLEGE

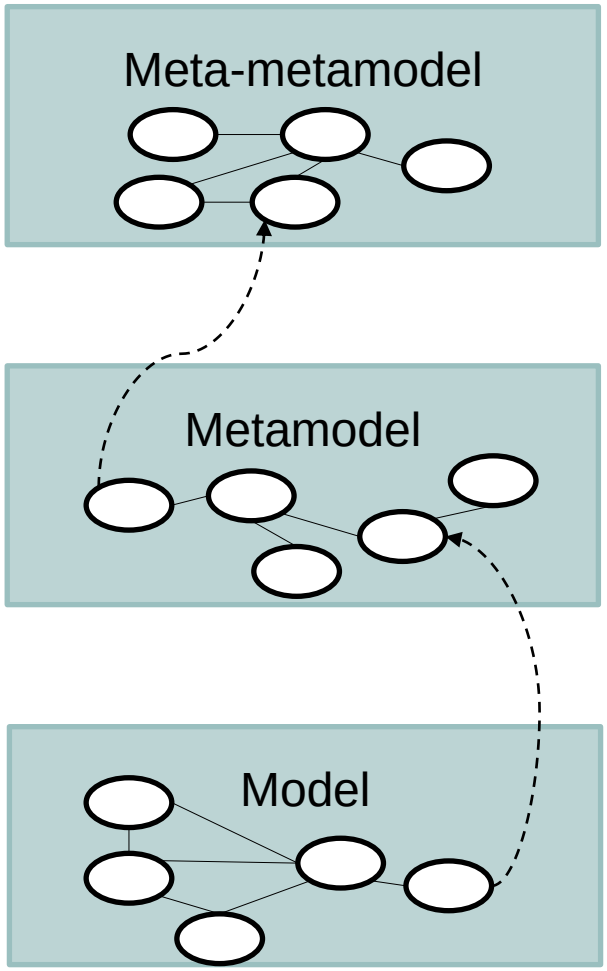
Runtime Verification of Executable Models

Fernando Macías – fernando.macias@hib.no
Adrian Rutle – adrian.rutle@hib.no
Volker Stolz – volker.stolz@hib.no

Hierarchy for Executable Modelling

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

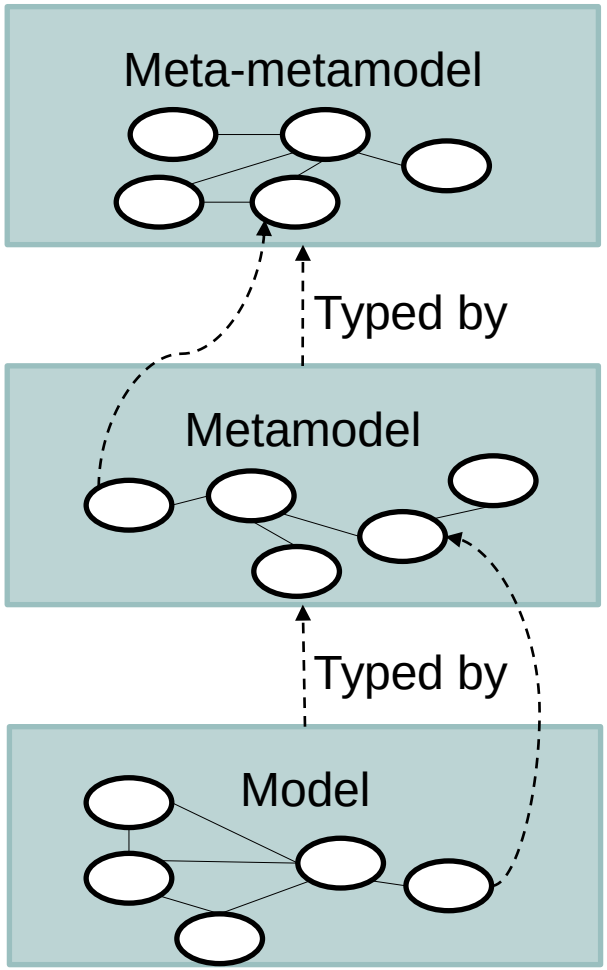
- The standardized solutions are EMF (MOF) and UML
- Both have a bigger focus on structure
- Limited number of levels
- In complex architectures, the levels have to be collapsed
  - › Convolution of models
  - › Bad maintainability



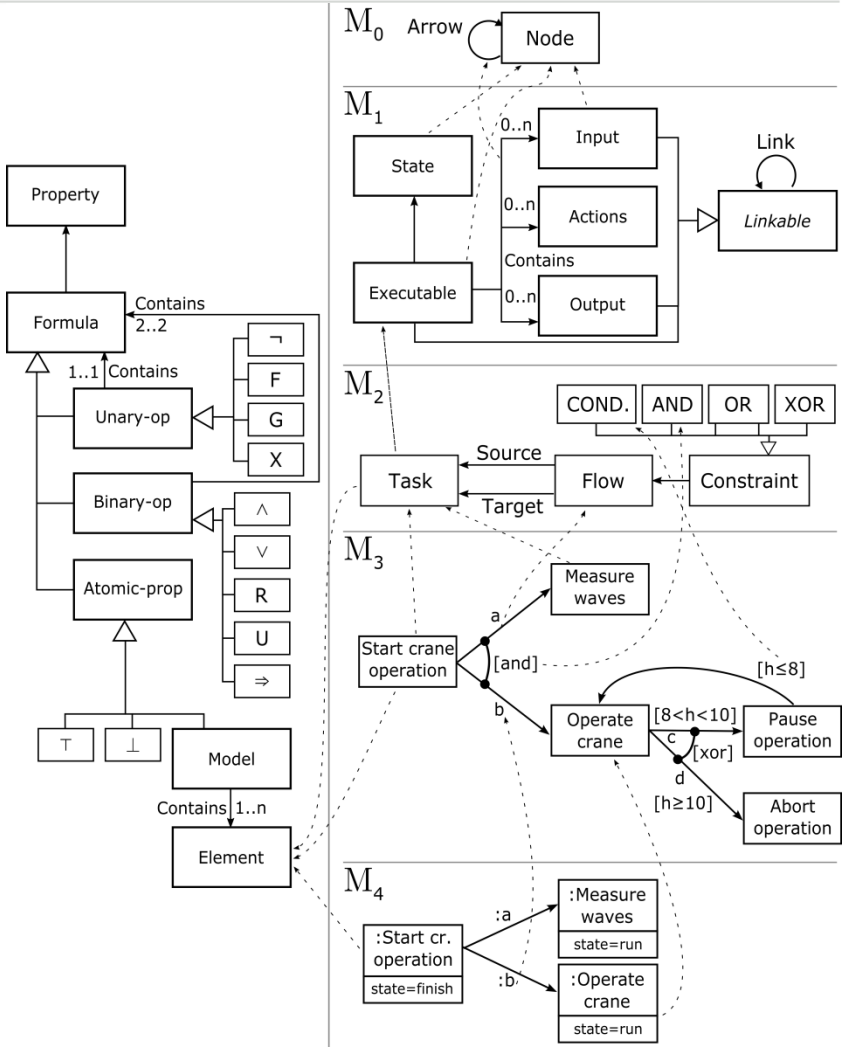
Hierarchy for Executable Modelling

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

- The standardized solutions are EMF (MOF) and UML
- Both have a bigger focus on structure
- Limited number of levels
- In complex architectures, the levels have to be collapsed
  - › Convolution of models
  - › Bad maintainability



Hierarchy for Executable Modelling

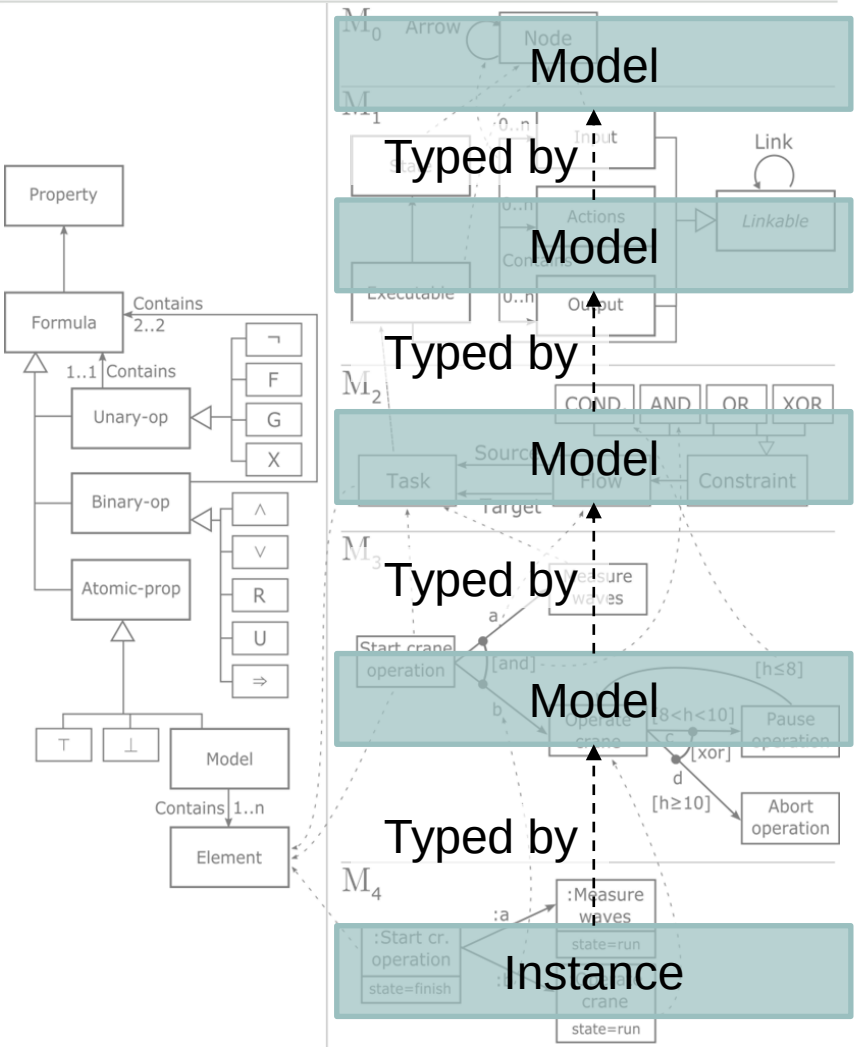


Hierarchy for Executable Modelling

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

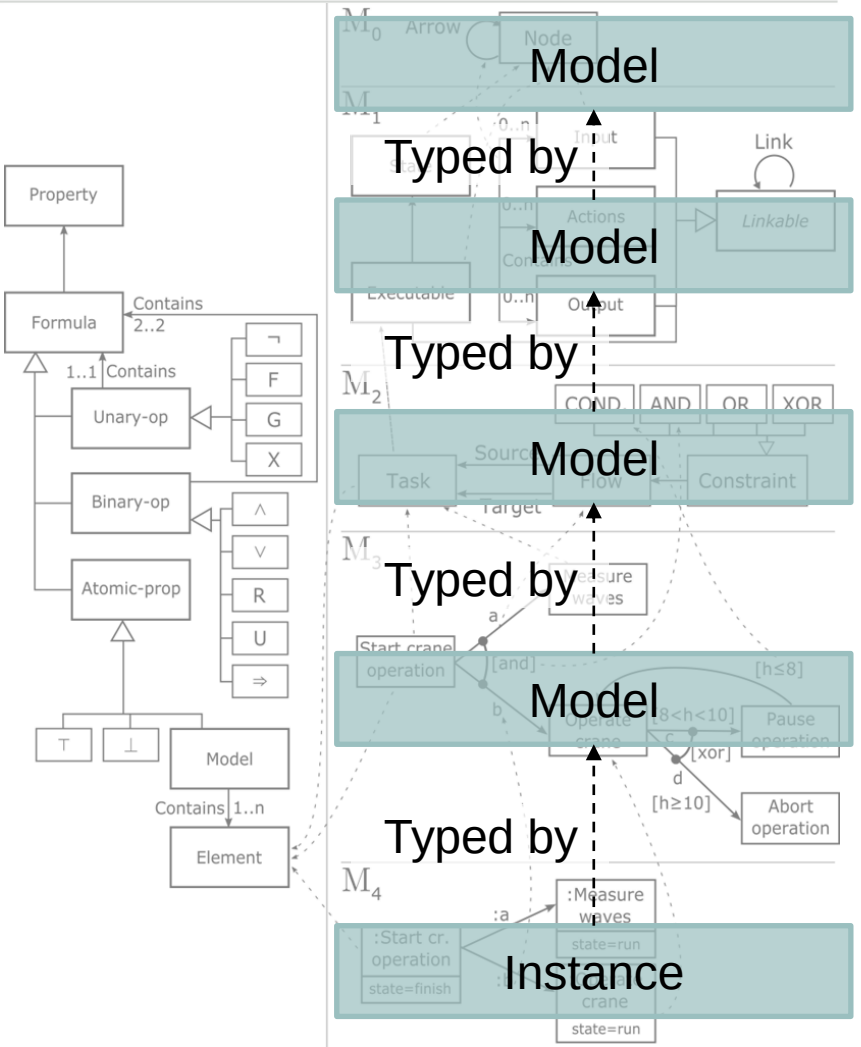
- Our hierarchy exploits the concept of **Multilevel Modelling**

"Enabling modelling at an arbitrary number of meta-levels" [1]



1. De Lara et al. *When and How to Use Multilevel Modelling*

Hierarchy for Executable Modelling

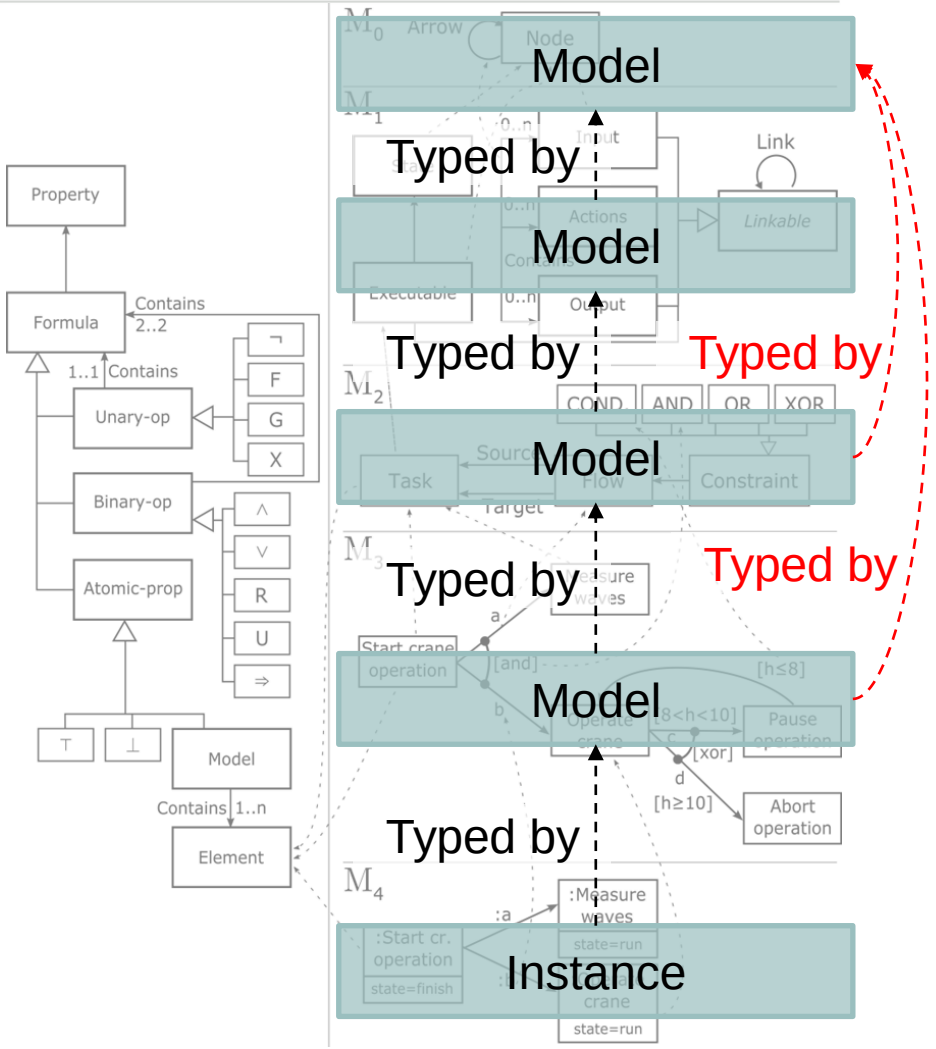


Hierarchy for Executable Modelling

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

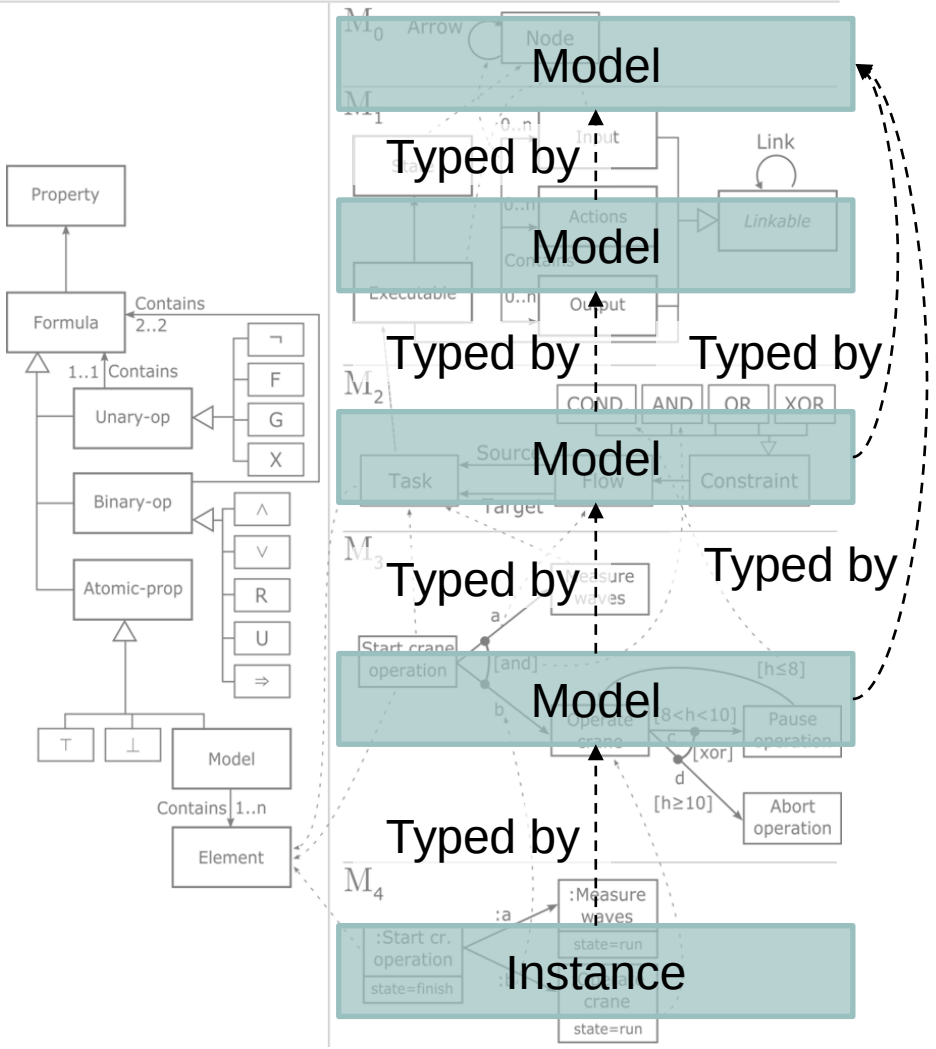
- Our hierarchy exploits the concept of **Deep Metamodelling**

"An element in a model can be typed by another element several models above"

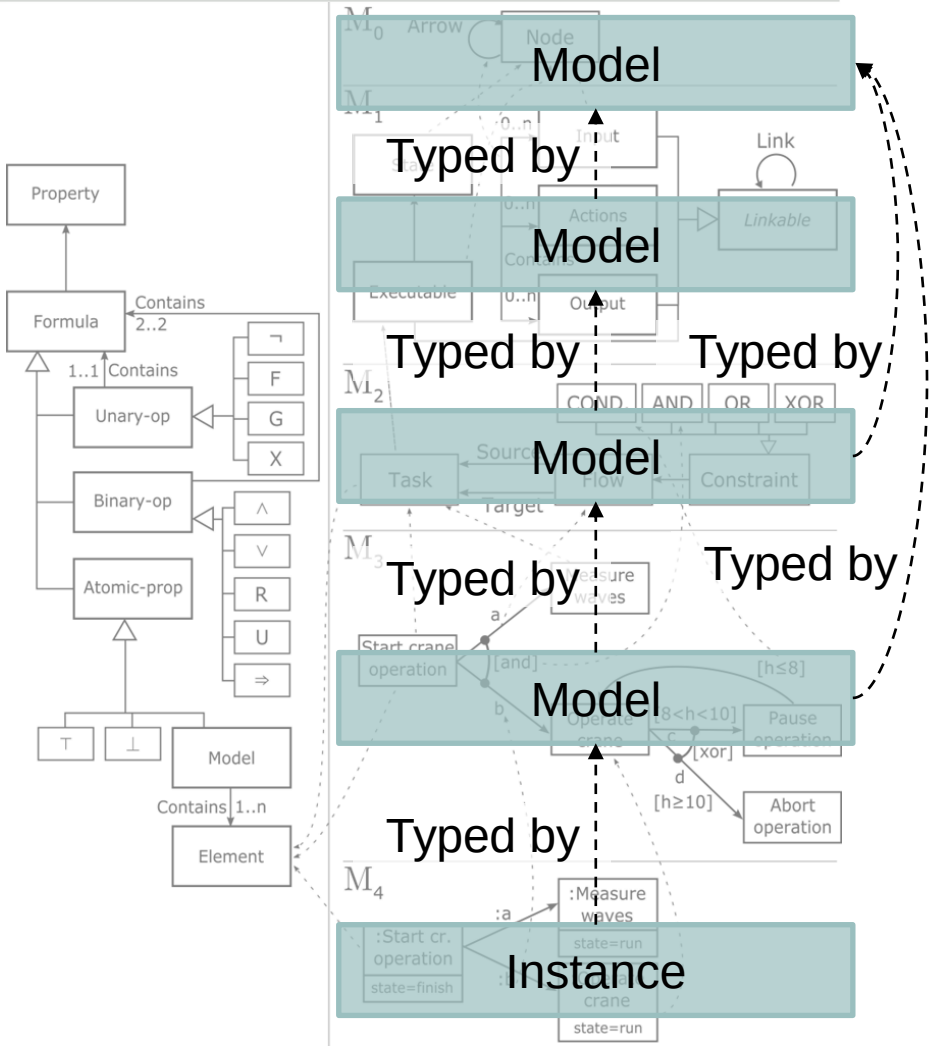


Hierarchy for Executable Modelling

- This hierarchy allows to
  - › Define custom executable modelling languages
  - › Create models according to those languages
  - › Run the instances with default semantics
  - › Customize semantics
  - › Simulation
  - › Deployment
  - › Runtime verification over the running instances



Property Specification Language

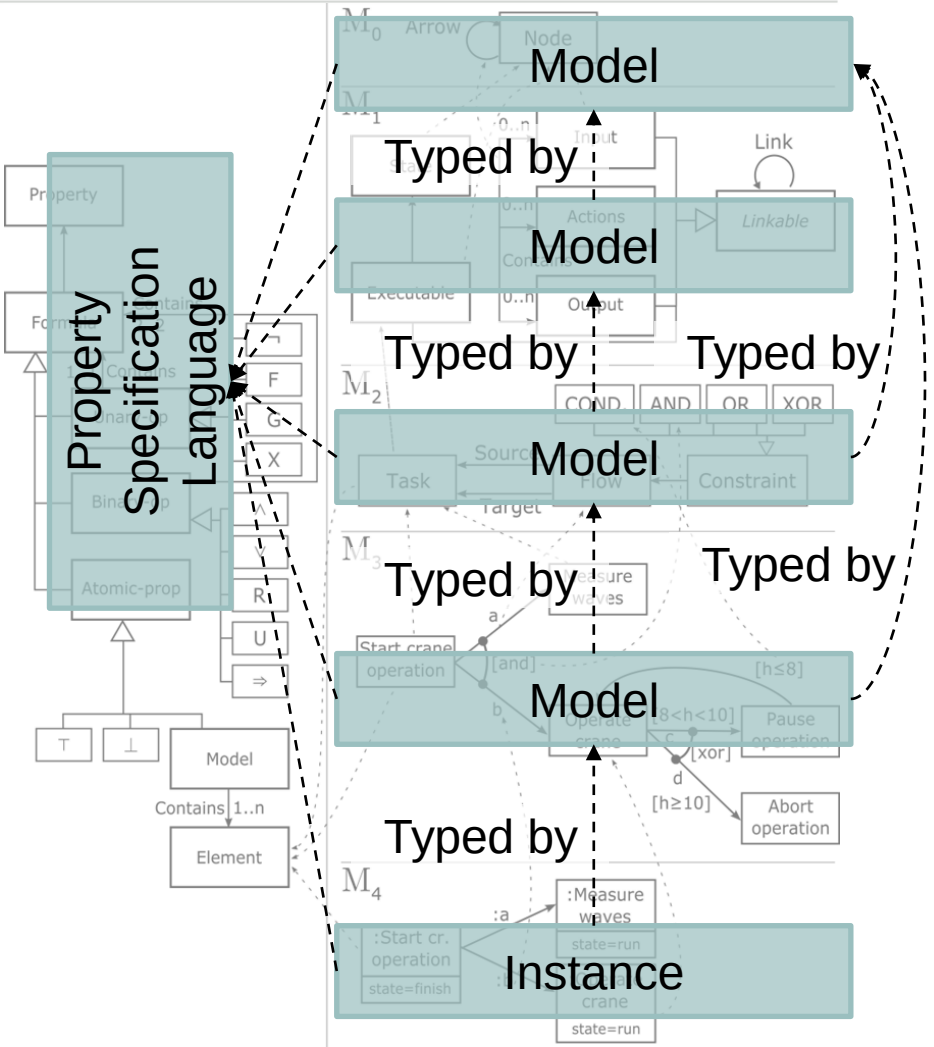


Property Specification Language

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

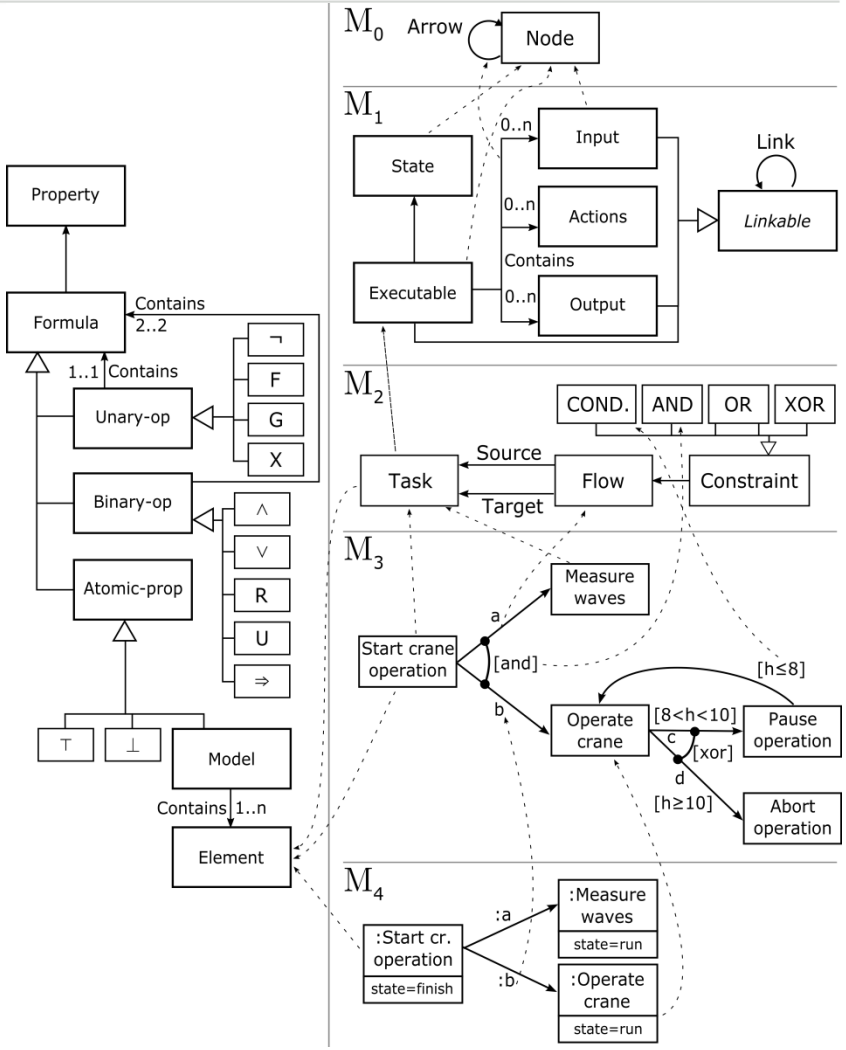
- Using the concept of **Linguistic Extension**

"Instantiation within a linguistic modelling language used to specify the models at all metalevels of the ontological stack" [1]

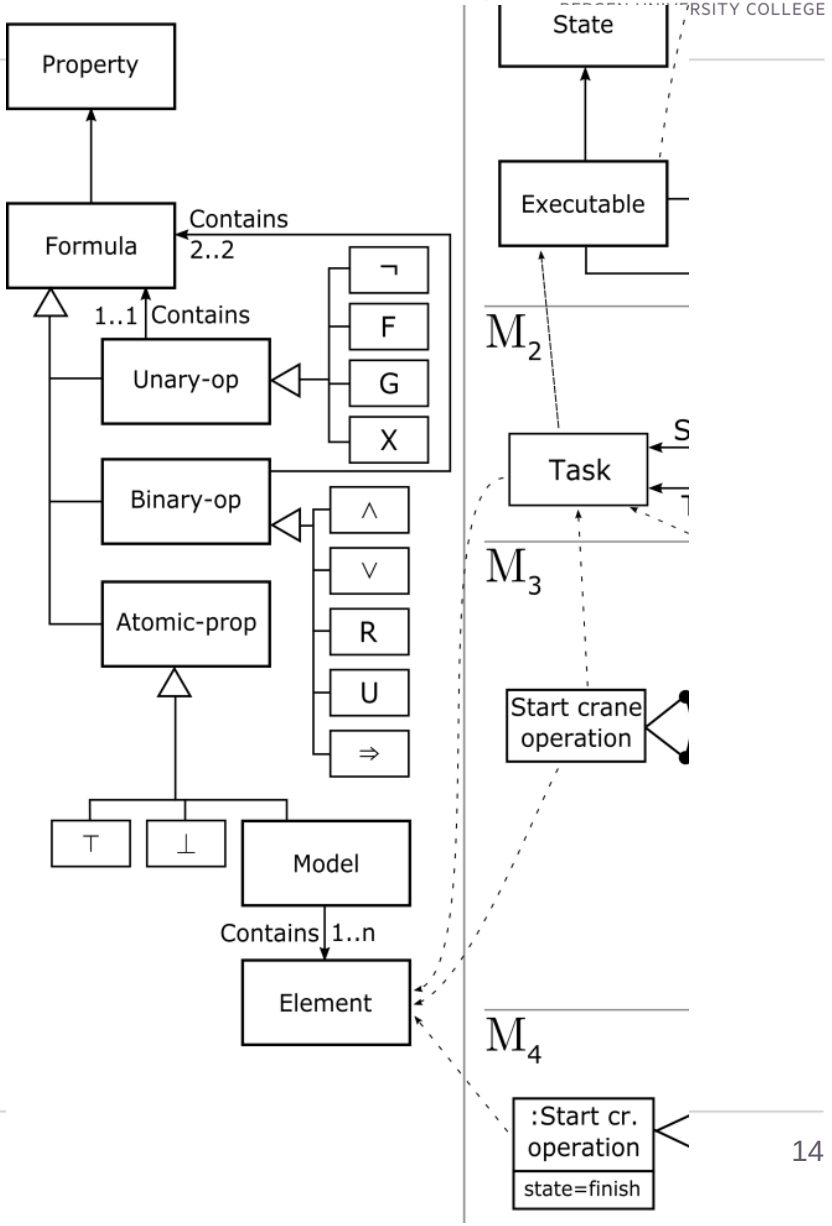


1. Rossini et al. *A formalisation of deep metamodelling*

Property Specification Language

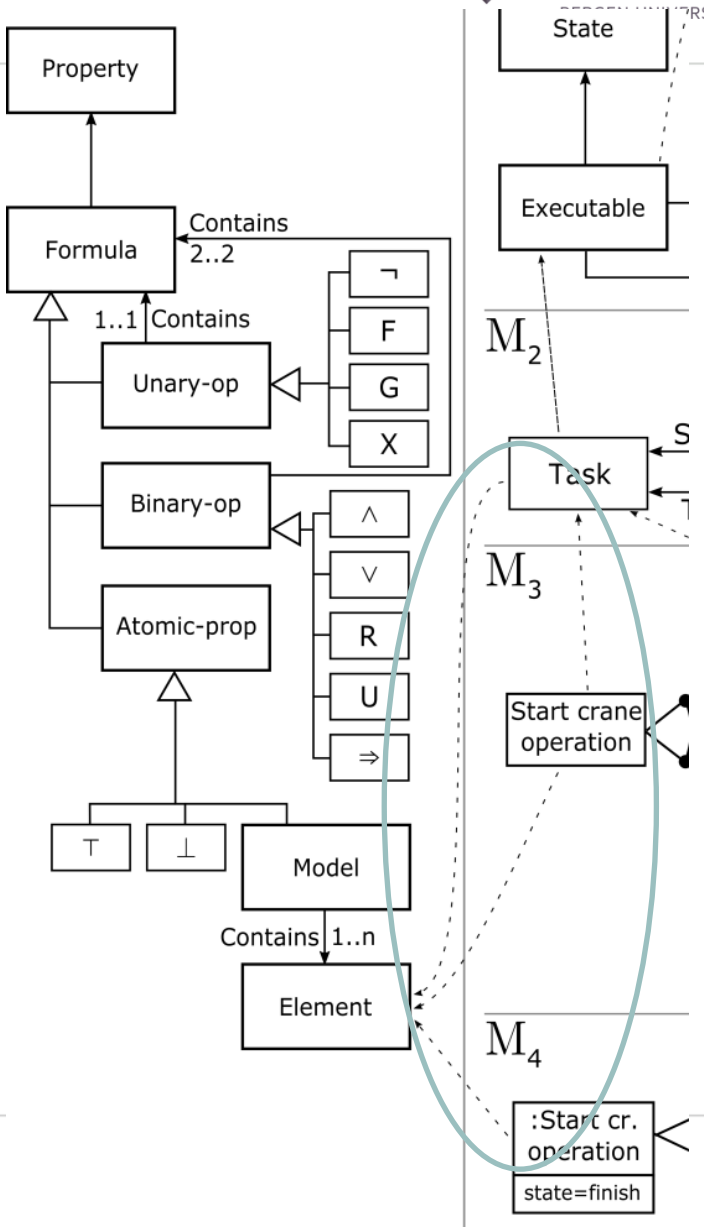


Property Specification Language



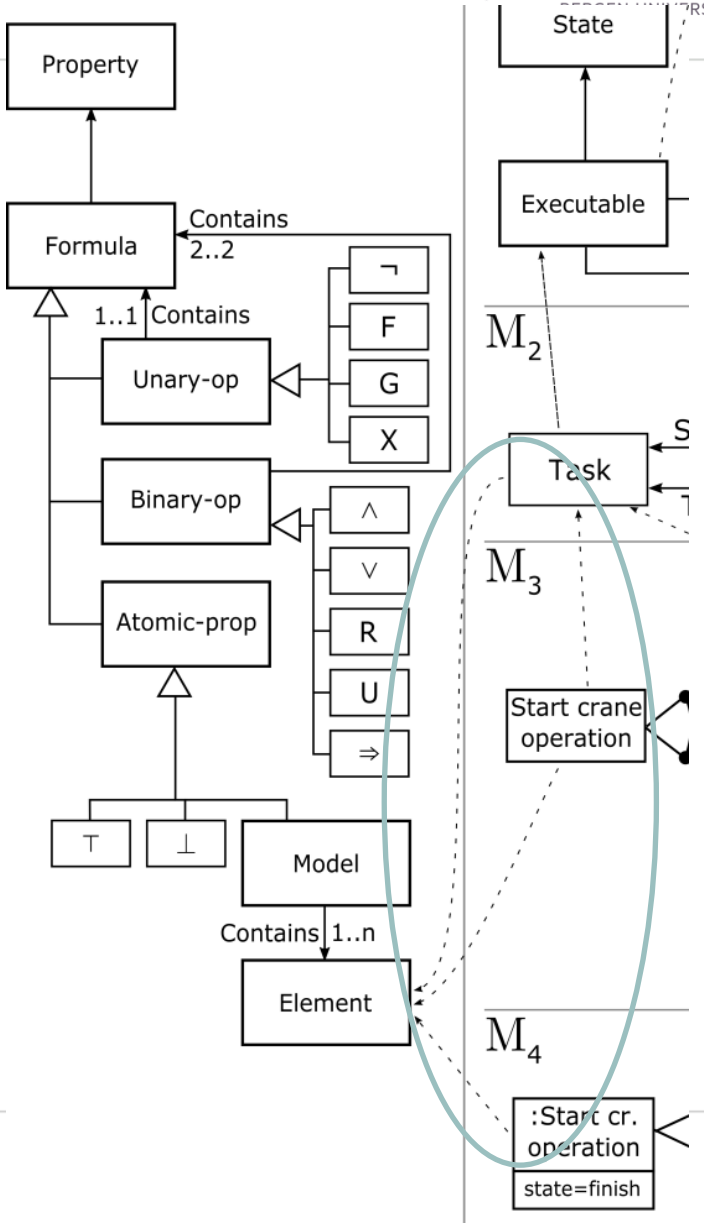
Property Specification Language

- Linguistic Extension allows to create properties connected to model elements
- Temporal properties expressed over types and instances of the models



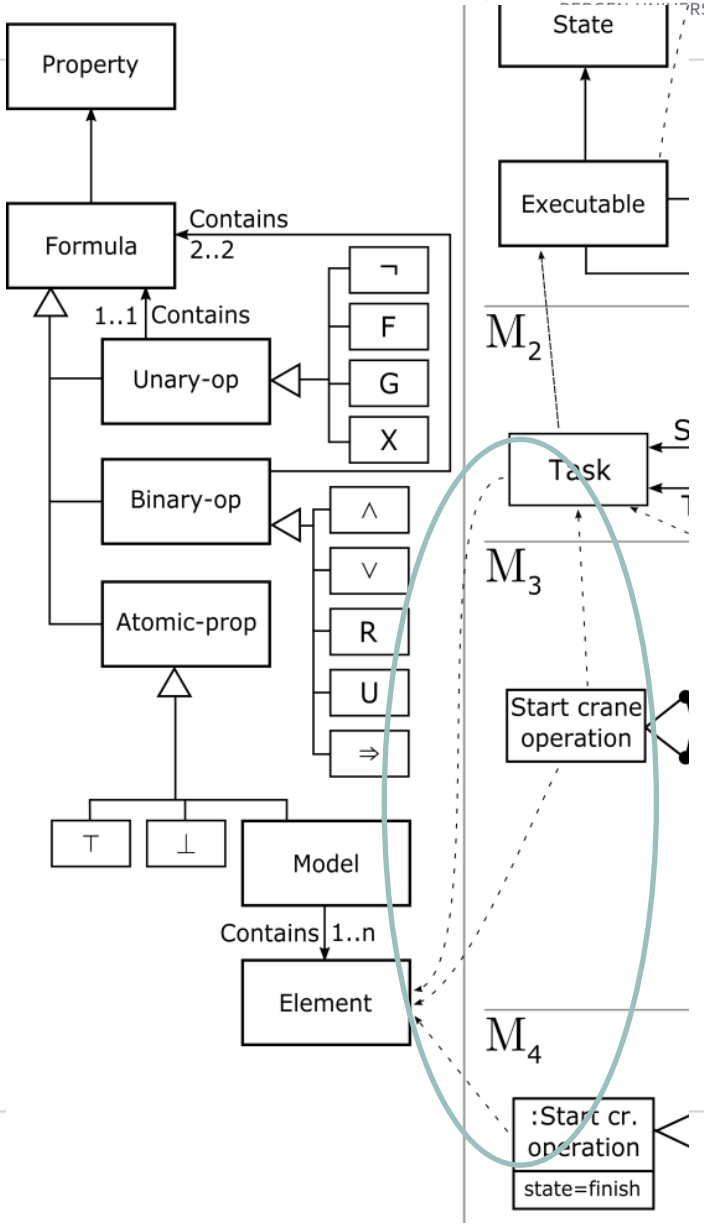
Property Specification Language

- Linguistic Extension allows to create properties connected to model elements
- Temporal properties expressed over types and instances of the models
- Possibility to define **cross-level** properties

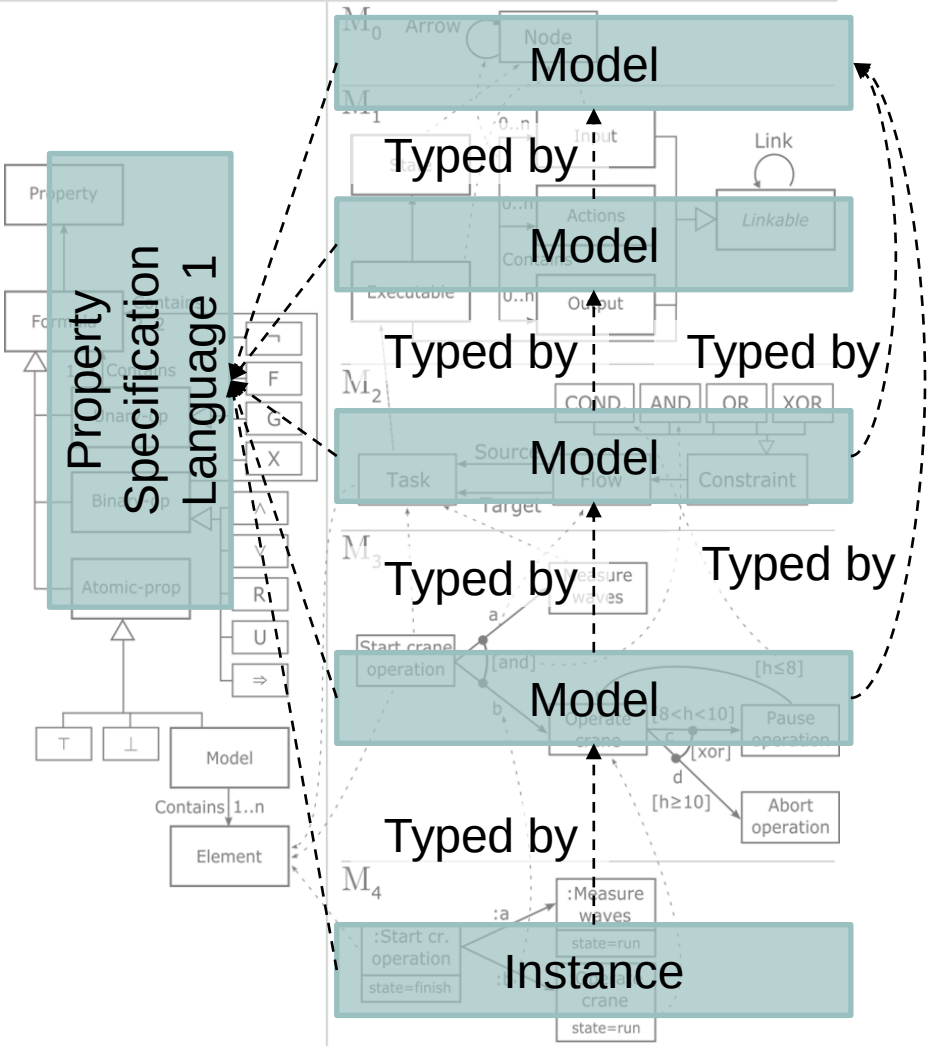


Property Specification Language

- Linguistic Extension allows to create properties connected to model elements
- Temporal properties expressed over types and instances of the models
- Possibility to define **cross-level** properties
- Possibility to link to **several instances**

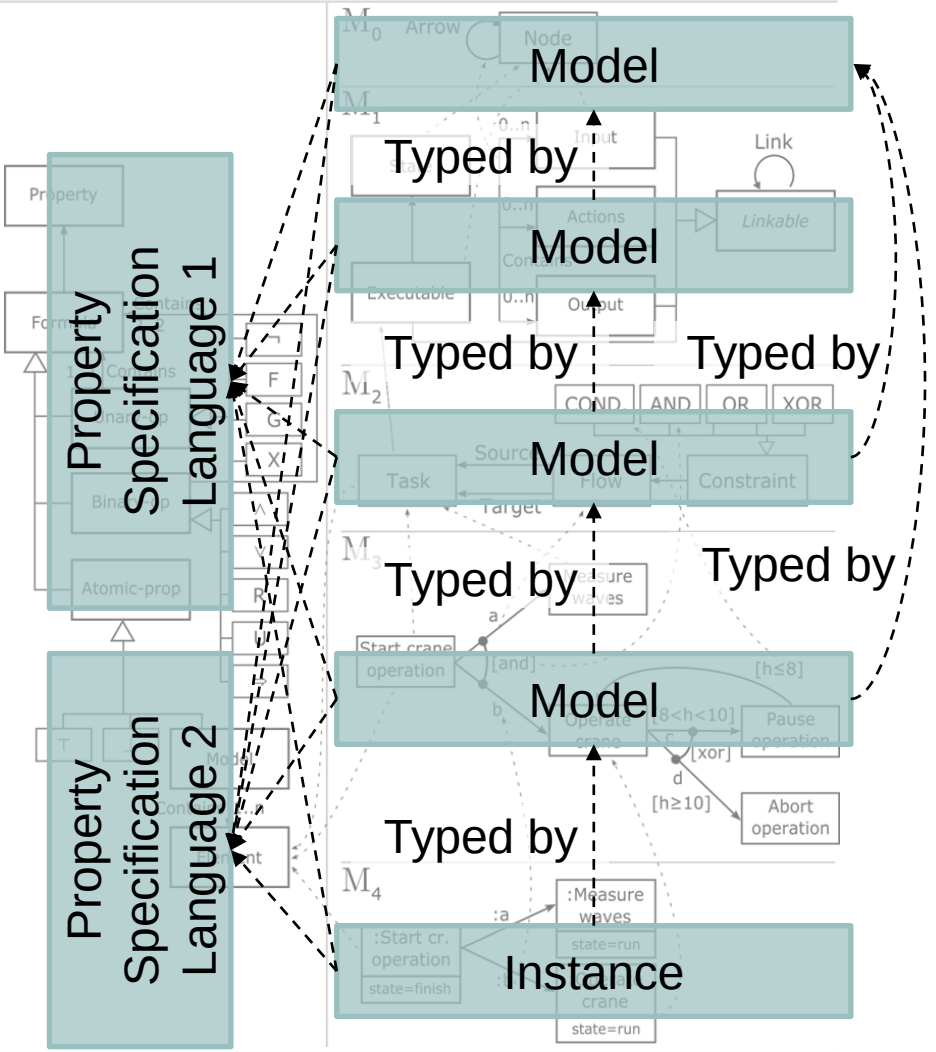


Property Specification Language



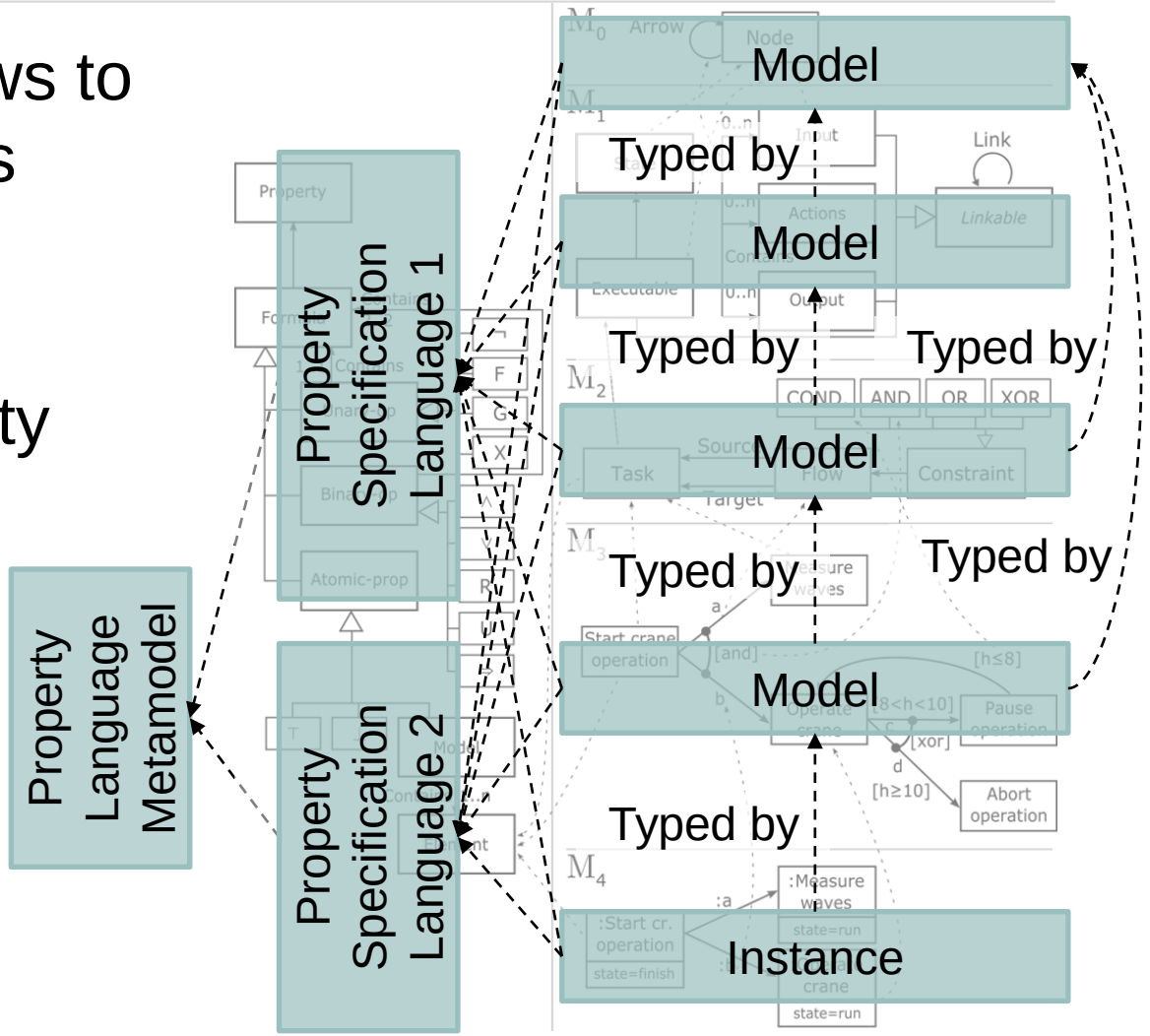
Property Specification Language

- The hierarchy allows to add new languages (e.g. TLTL, SALT)



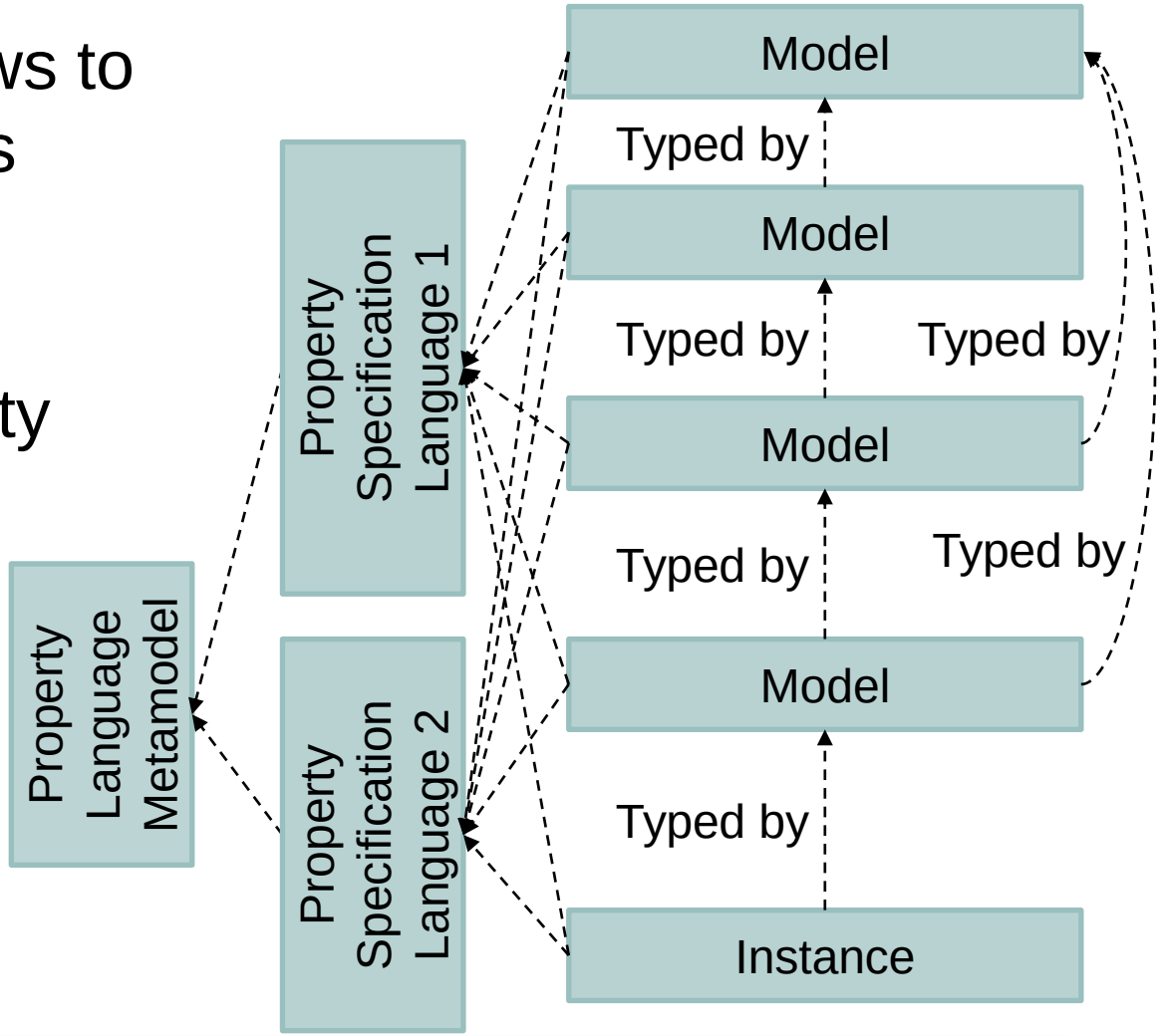
Property Specification Language

- The hierarchy allows to add new languages (e.g. TLTL, SALT)
- Possibility of a hierarchy of property languages



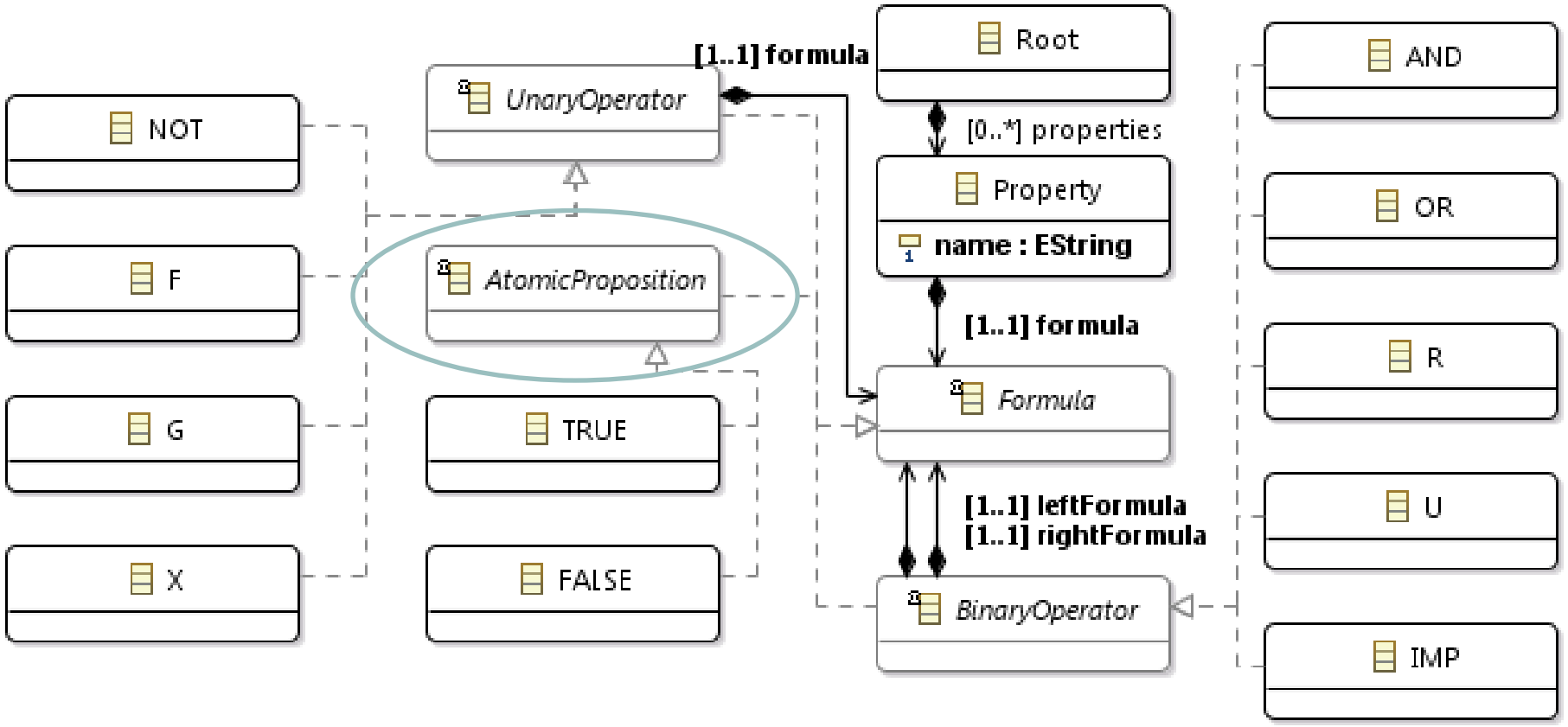
Property Specification Language

- The hierarchy allows to add new languages (e.g. TLTL, SALT)
- Possibility of a hierarchy of property languages



Property Specification Language

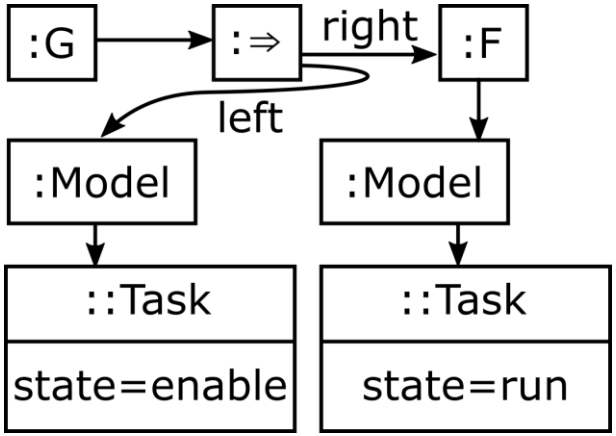
HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE



Syntax

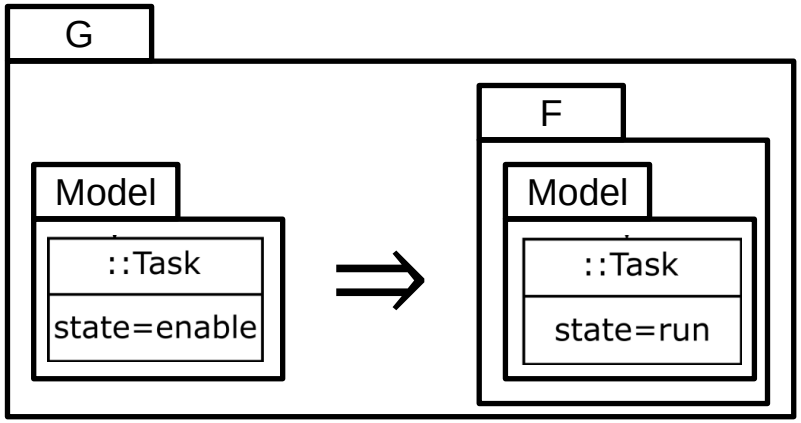
HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

Abstract syntax



- Internal representation of the model
- In graph-based models, nodes and relations among them

Concrete syntax



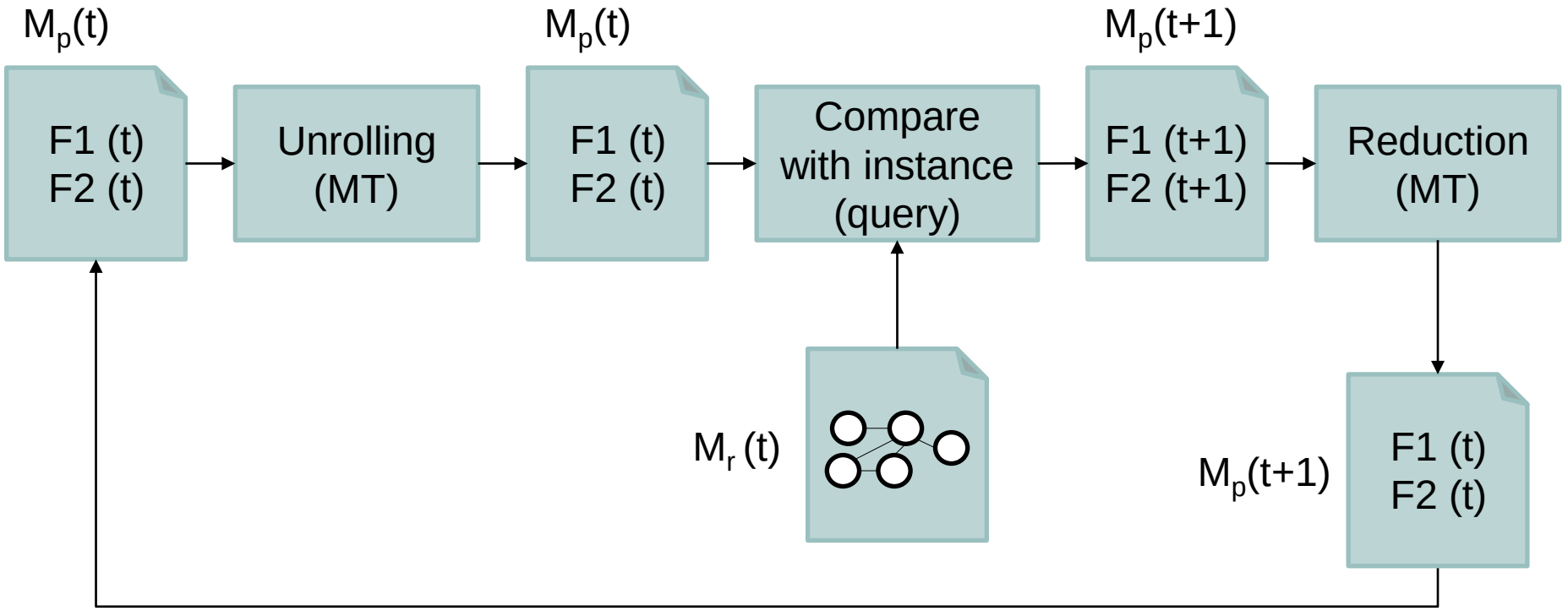
- Created to be human readable
- Synchronized with the abstract syntax
- Text, diagrams...

Semantics

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

- LTL temporal operator unrolling
  - $\varphi U \psi \equiv \psi \vee (\varphi \wedge X(\varphi U \psi))$
  - $F\psi \equiv \psi \vee XF\psi$
  - $G\psi \equiv \psi \vee XG\psi$
- LTL Next operator (X) processing
  - $X\psi(t_n) \equiv \psi(t_{n+1})$
- LTL reduction
  - $GG\psi \equiv G\psi$

Semantics



Repeat until every property has been reduced to ⊤ or ⊥

Semantics

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

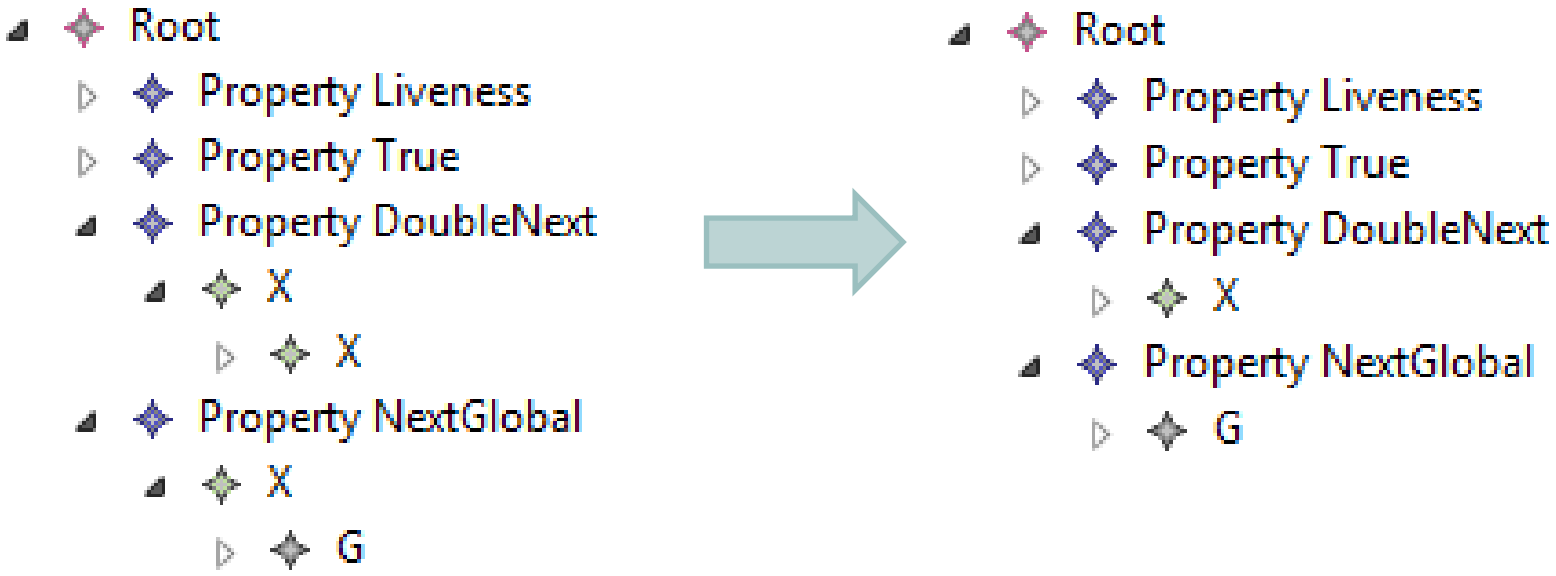
- ATL/EMF implementation

```
abstract rule processX {
  from
    input : mmProperties!X
  to
    output : mmProperties!UnaryOperator (
      formula <- input.formula.formula
    )
}
```

Semantics as model transformations

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

- ATL/EMF implementation



Future work

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

- Integrate our hierarchy of models and languages into an existing framework (GEMOC), or…
- … create a multilevel modelling editor for EMF models
- Add new languages for the specification of temporal properties
- Seamless and automatic linking of property specification languages with any model in the hierarchy

Summary

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

- Introduction of flexible hierarchy for executable modelling
- Definition of abstract syntax, concrete syntax and semantics for temporal properties on behavioural models
- Runtime Verification of temporal properties on interpreted models. No need for compilation/translation
- Usage of deep metamodelling concepts to achieve a customizable hierarchy

Summary

HØGSKOLEN I BERGEN
BERGEN UNIVERSITY COLLEGE

- Introduction of flexible hierarchy for executable modelling
- Definition of abstract syntax, concrete syntax and semantics for temporal properties on behavioural models
- Runtime Verification of temporal properties on interpreted models. No need for compilation/translation
- Usage of deep metamodelling concepts to achieve a customizable hierarchy

Thank you for your attention!