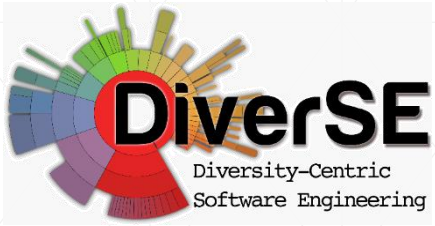


Melange: a Meta-language for Modular and Reusable Development of DSLs

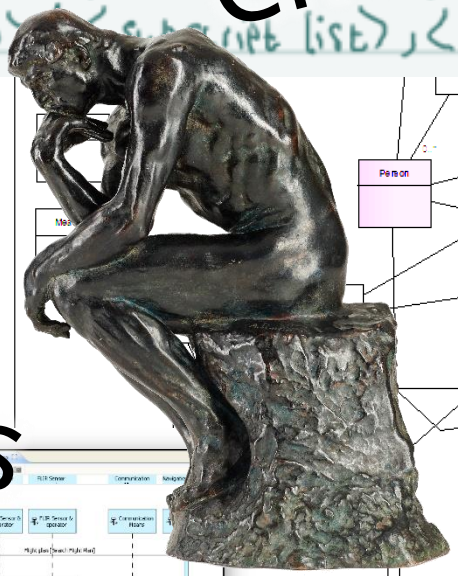
Thomas Degueule, Benoit Combemale, Arnaud Blouin, Olivier Barais, Jean-Marc Jézéquel

Static Analyzes

Compilers

Checkers

Interpreters



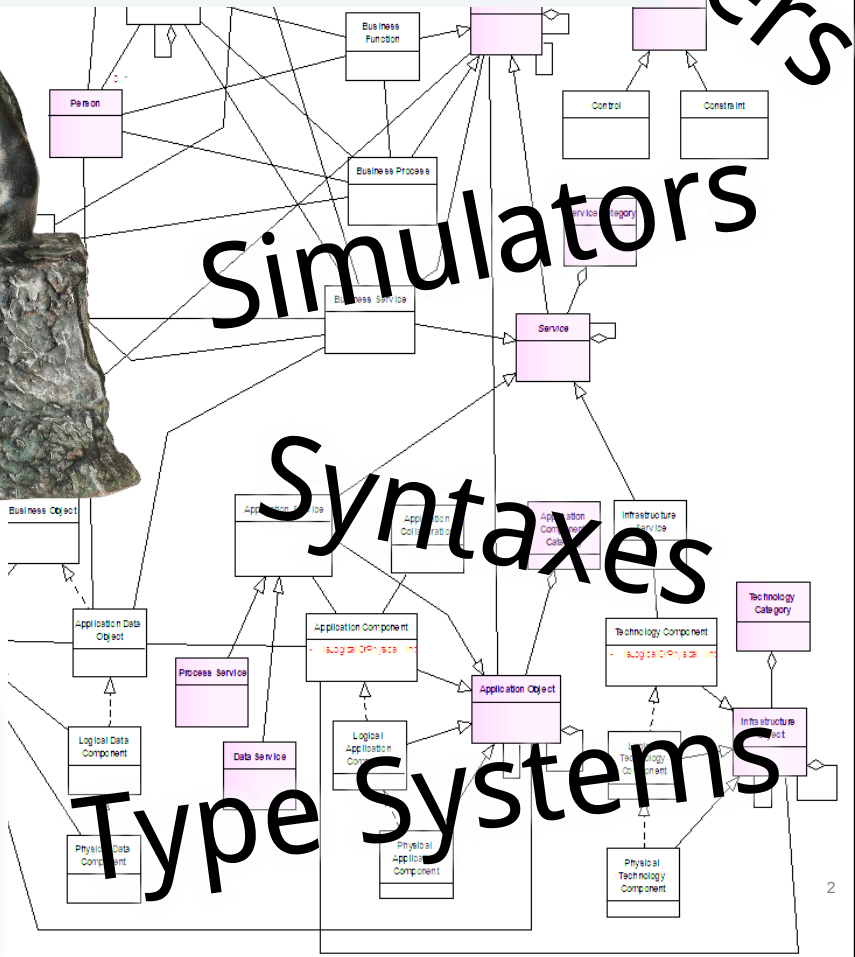
$$\frac{}{\Gamma, x : \tau \vdash x : \tau} Var$$

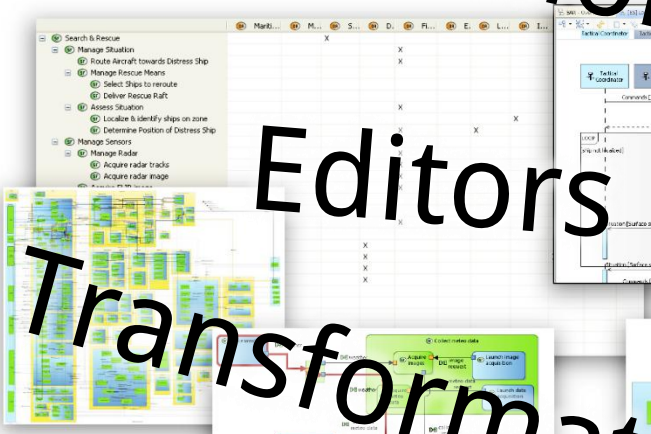
$$\frac{\Gamma, x : \tau_p \vdash t : \tau_r}{\Gamma \vdash (x : \tau_p.t) : \tau_p \to \tau_r} Lam$$

$$\frac{\Gamma \vdash t_f : \tau_p \to \tau_r \quad \Gamma \vdash t_p : \tau_p}{\Gamma \vdash t_f\ t_p : \tau_r}$$

Abstractions

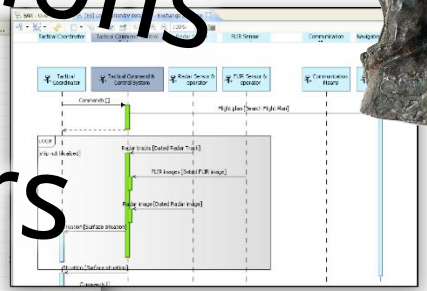
Simulators



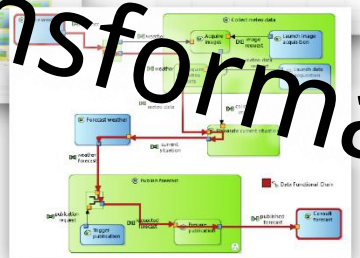


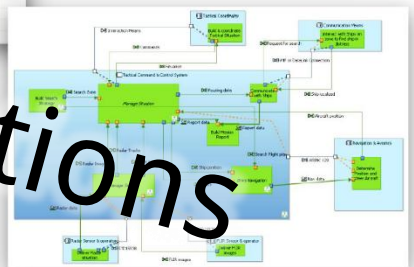
Editors

Syntaxes



Transformations



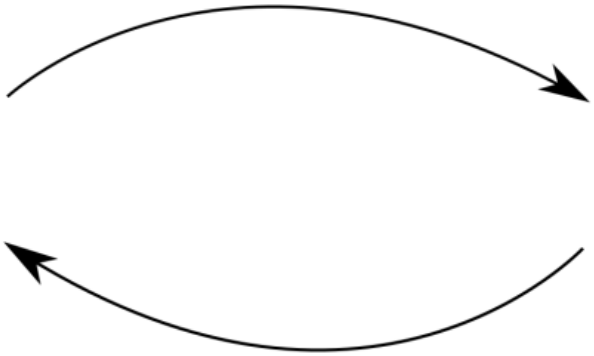


Type Systems

DSLs specificities

- Closely evolve with the domain
- Extended, shrunk, customized, etc.
- Quick prototyping, small development teams

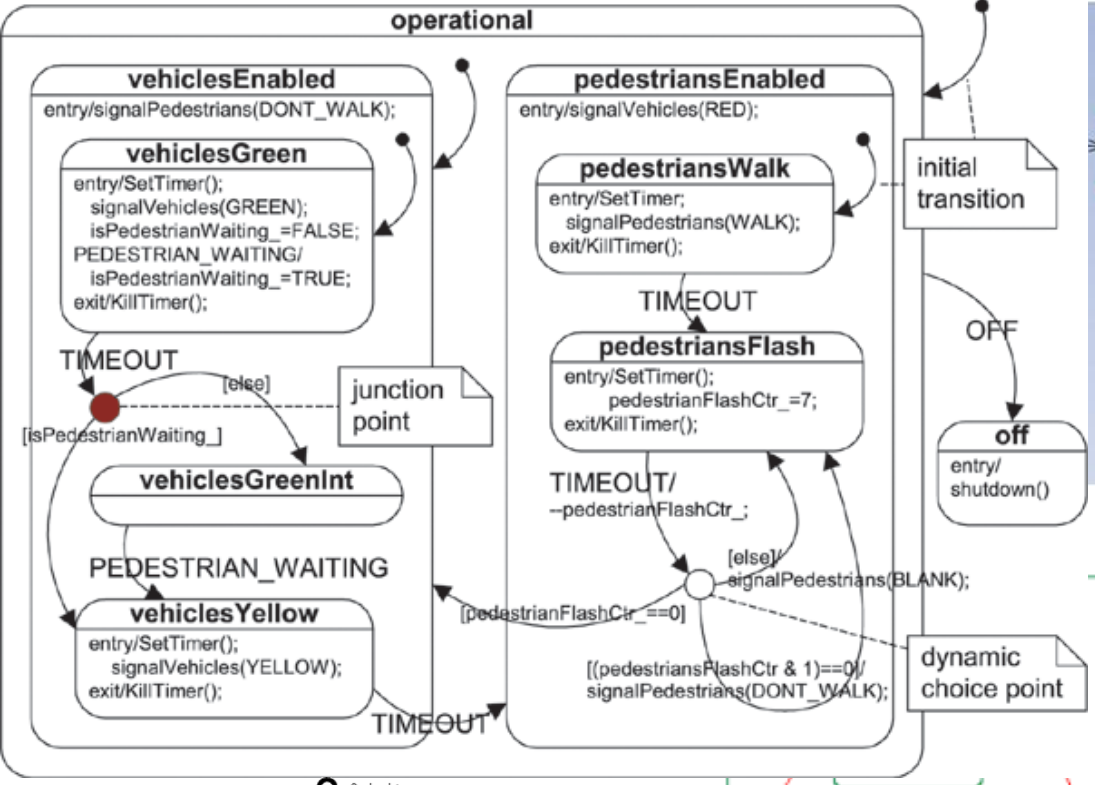
DSL User
(Domain expert)

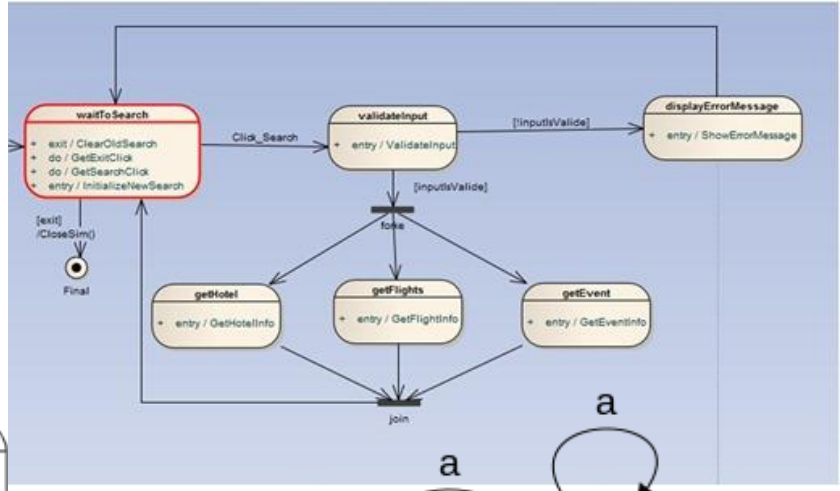


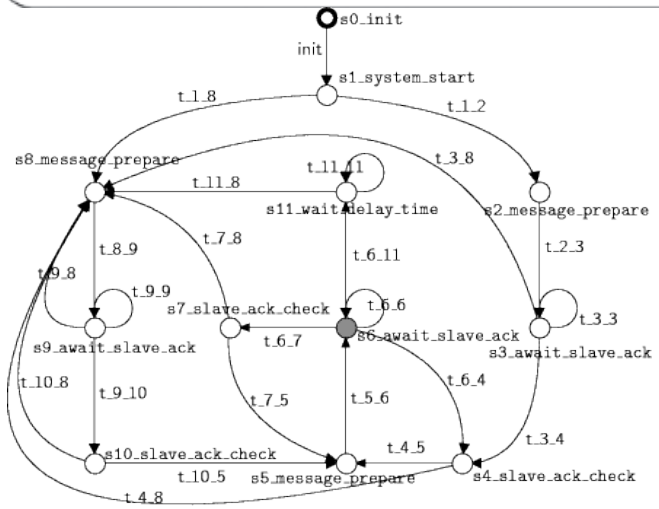
DSL Designer
(Language expert)

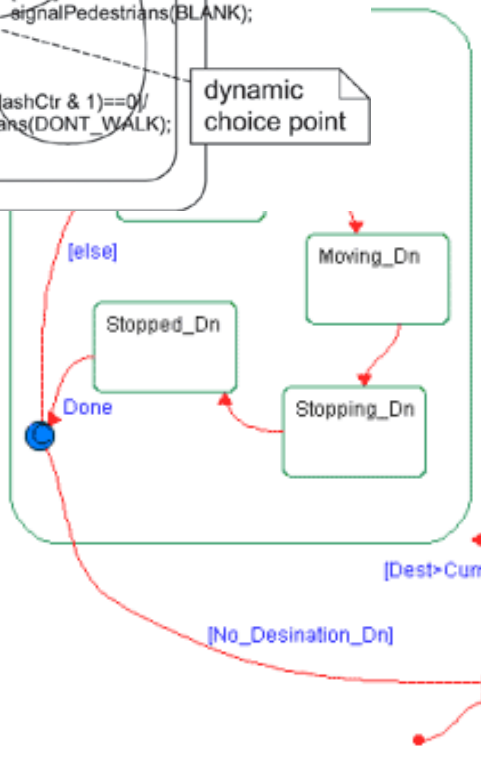
Software Language Engineering

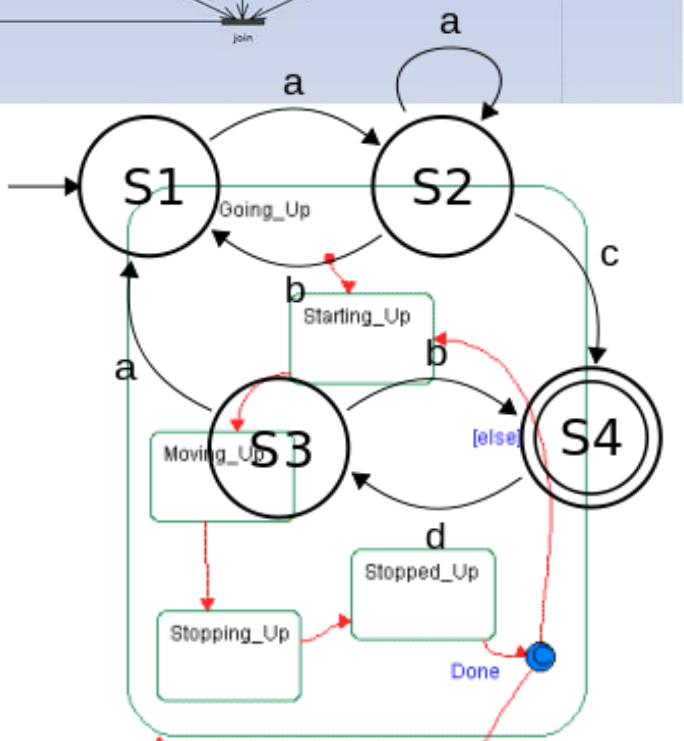
- Breathing software engineering into languages
- In this talk: **reuse**, **modularity**
  - Goal: reduce engineering costs
- Instead of starting from scratch, can we reuse previously-defined language artifacts?
- But, is *reuse* relevant in a *domain-specific* context?



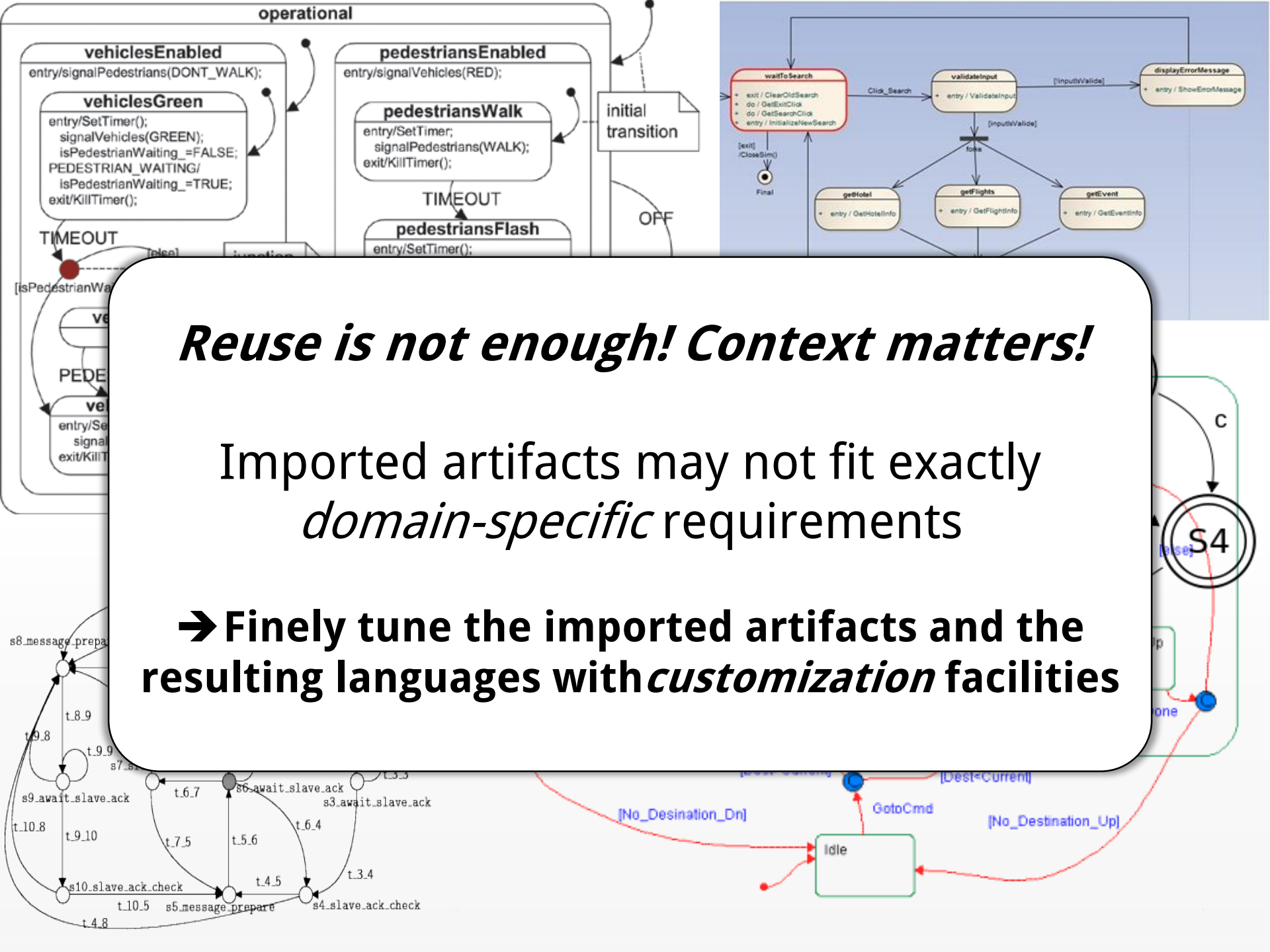








Crane *et al.*, *UML vs. classical vs. rhapsody statecharts: not all models are created equal*, SoSyM, 2007



Reuse is not enough! Context matters!

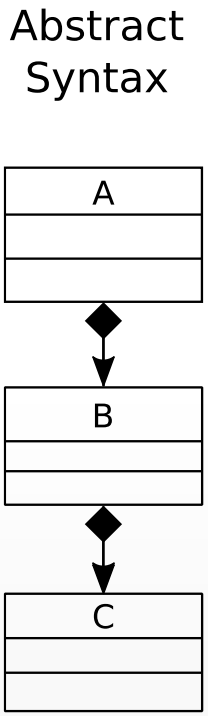
Imported artifacts may not fit exactly *domain-specific* requirements

➔ Finely tune the imported artifacts and the resulting languages with *customization* facilities

APPROACH OVERVIEW

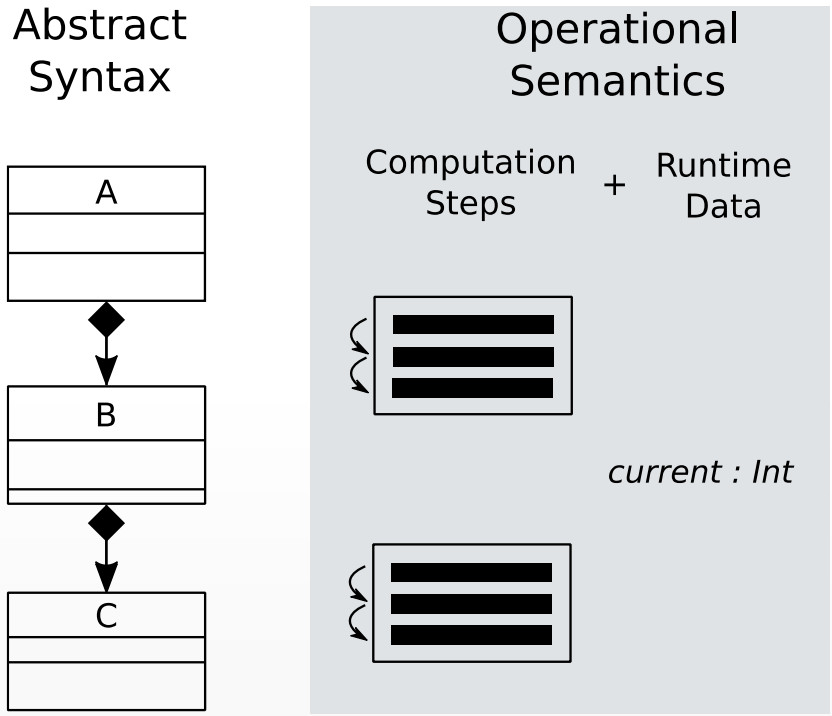
HYPOTHESIS ON LANGUAGE DEFINITION

- A metamodel specifies the *AS*



LANGUAGE DEFINITION

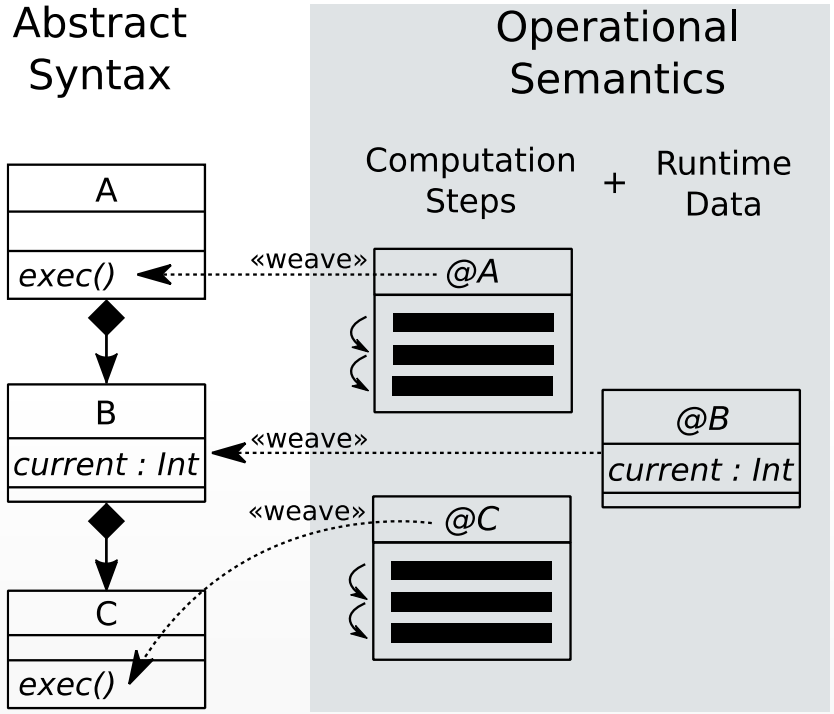
- A metamodel specifies the *AS*
- *Sem* consists of computation steps and runtime data



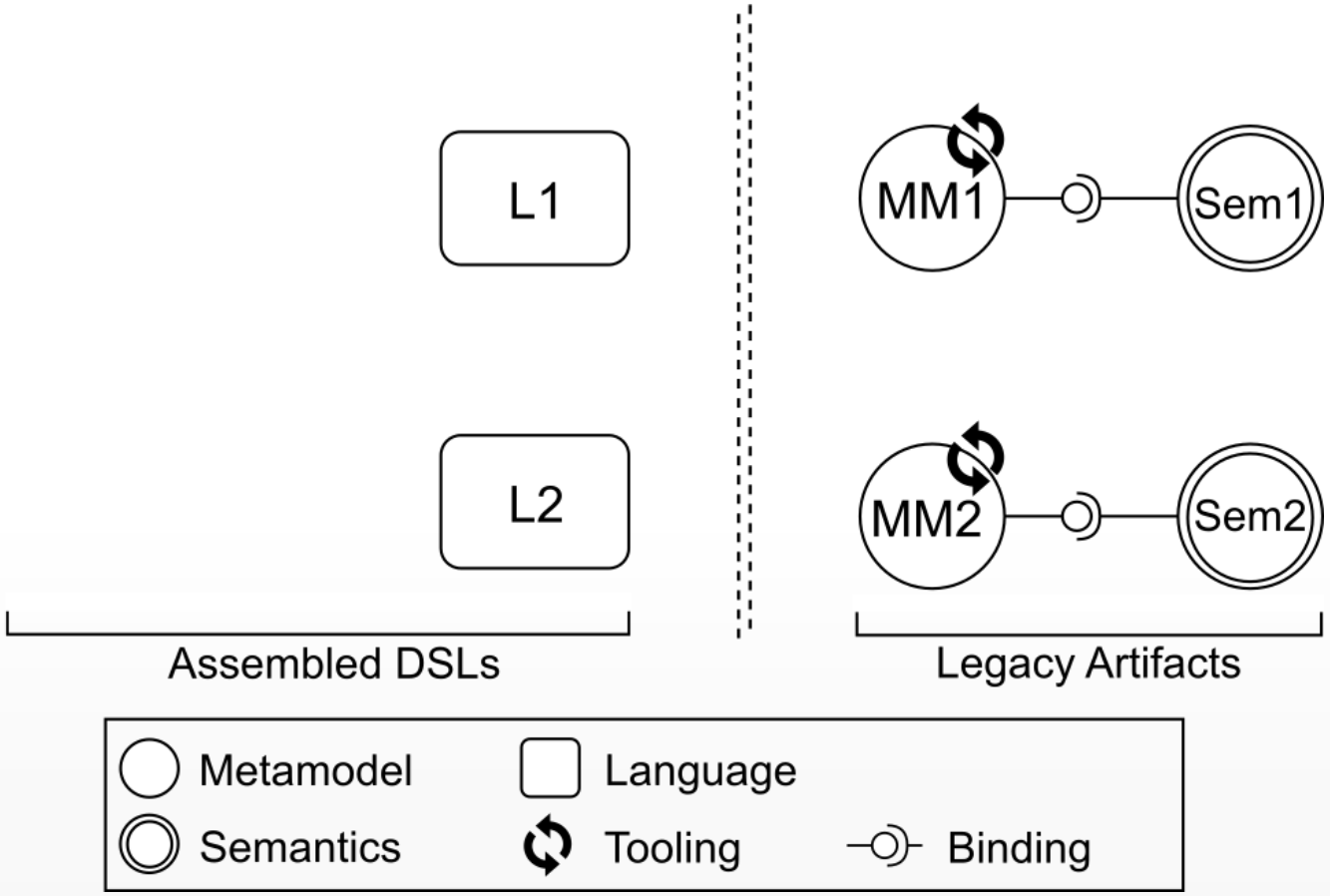
LANGUAGE DEFINITION

Jézéquel *et al.*, *Mashup of metalanguages and its implementation in the kermeta language workbench*, SoSyM, 2013

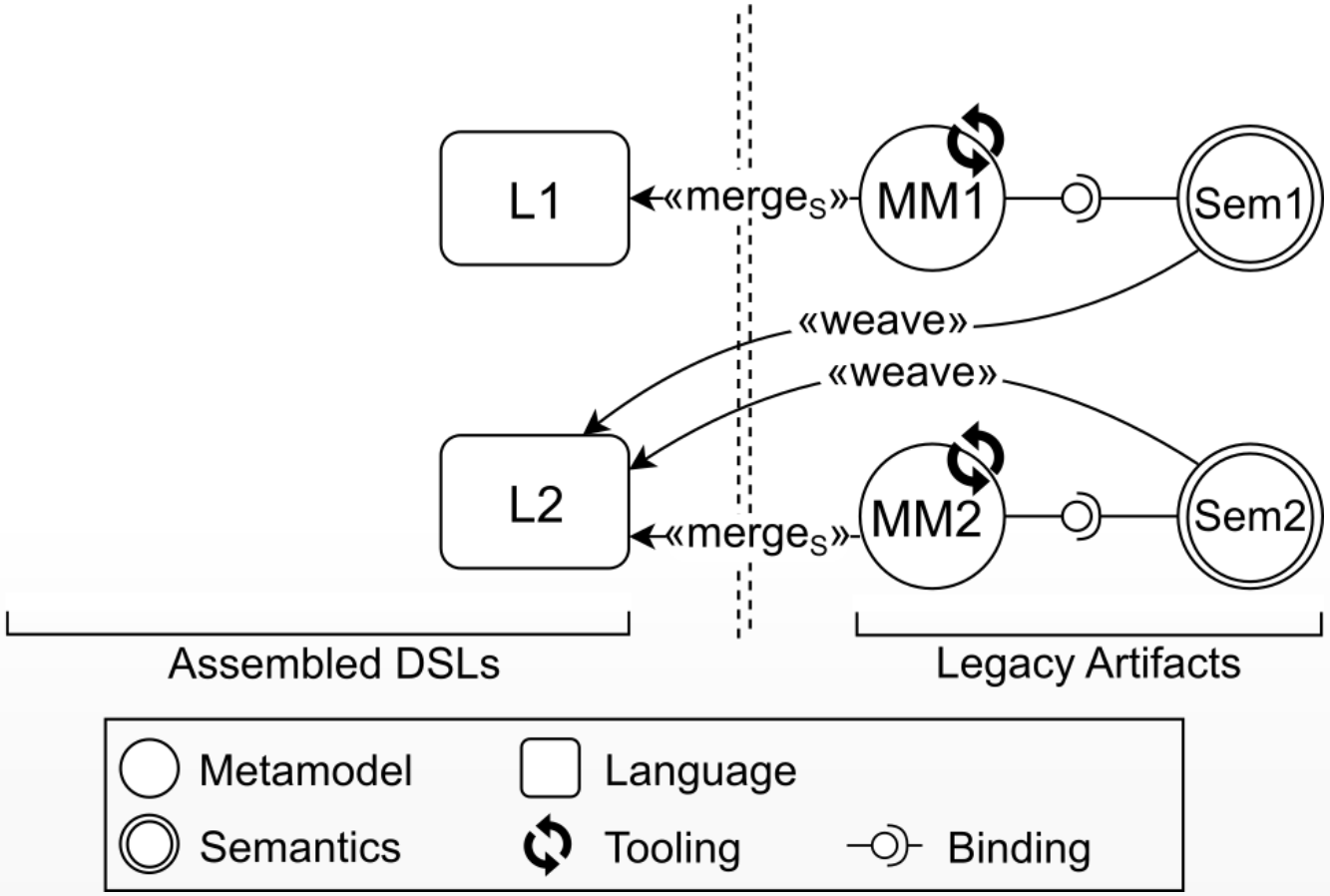
- A metamodel specifies the *AS*
- *Sem* consists of computation steps and runtime data
- Aspect-oriented modeling: *Sem* is *woven* as methods in the *AS*
  - Based on static introduction
- Interpreter pattern



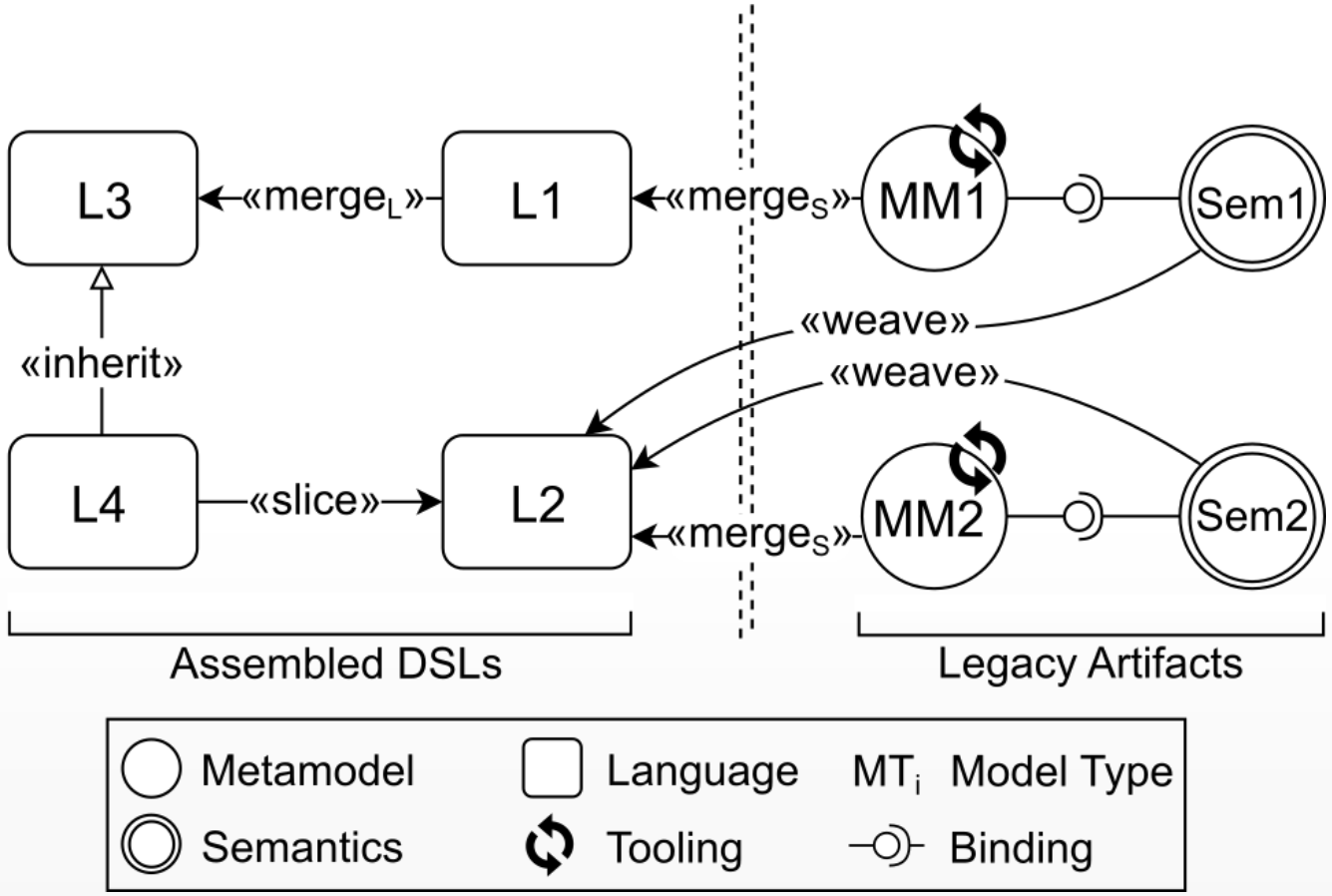
APPROACH OVERVIEW



APPROACH OVERVIEW

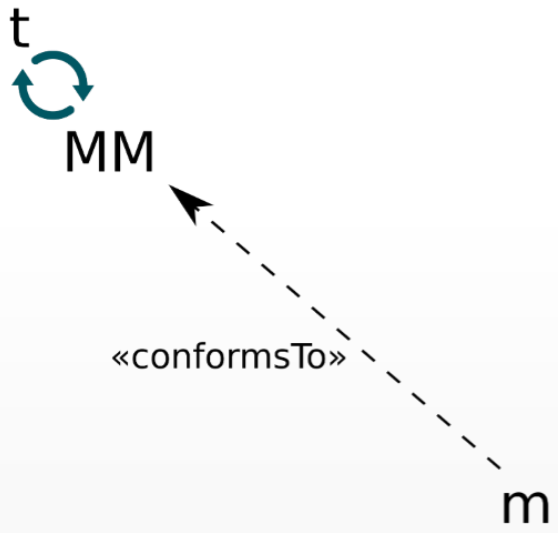


APPROACH OVERVIEW

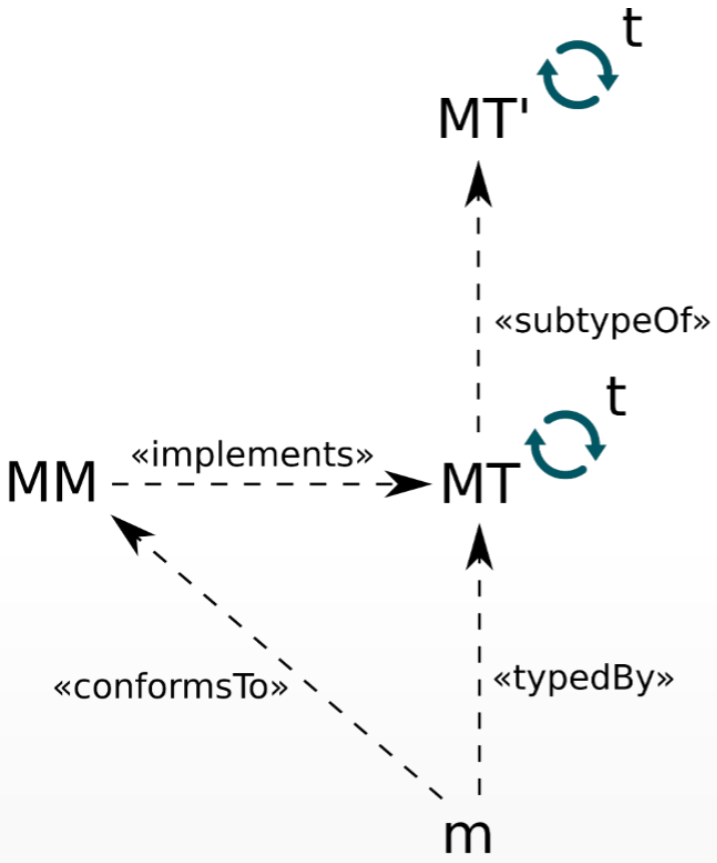


Inspired by *eg*. Erdweg *et al.*, *Language Composition Untangled*, LDTA, 2012

TOOL REUSE THROUGH MODEL TYPING



TOOL REUSE THROUGH MODEL TYPING



Steel *et al.*, *On Model Typing*, SoSyM, 2007
Guy *et al.*, *On Model Subtyping*, ECMFA, 2012

AN ALGEBRA FOR DSL ASSEMBLY AND CUSTOMIZATION

Running Example: a Simple State Machine Language in Melange

LANGUAGE DEFINITION

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$
$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$
$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \mathop{\bigcup}_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$
$$MT'' = MT \circ MT' \text{ and}$$
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$
$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$
$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$
$$MT_1 <: MT_2,$$

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

LANGUAGE DEFINITION

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

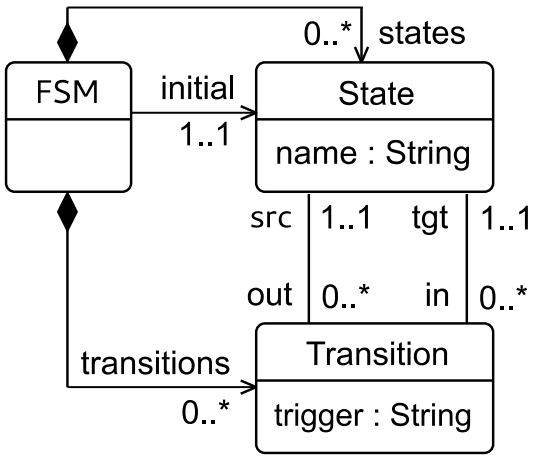
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language Fsm {
 → syntax 'FSM.ecore'



}
```

LANGUAGE DEFINITION

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$
$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$
$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

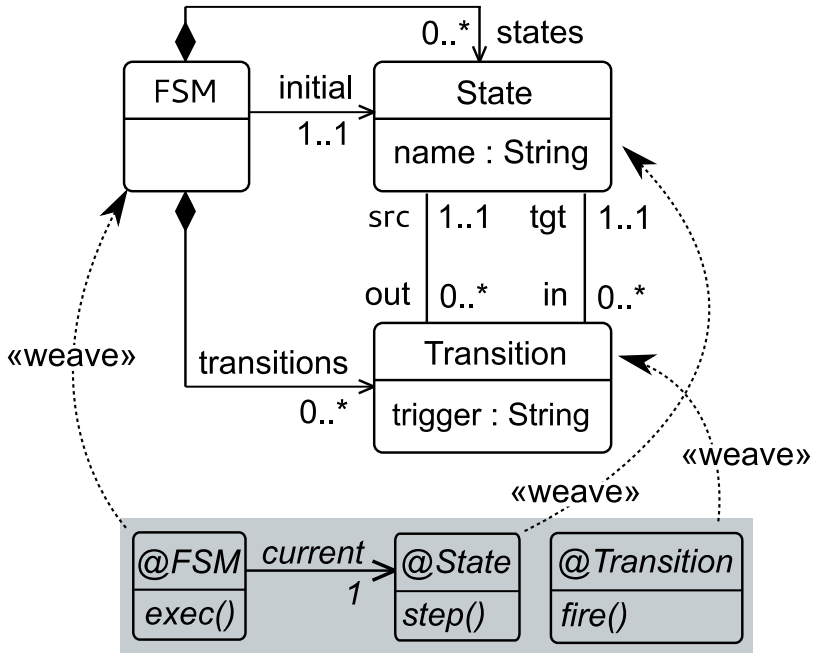
$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$
$$MT'' = MT \circ MT' \text{ and}$$
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$
$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$
$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$
$$MT_1 <: MT_2,$$




```
language Fsm {
    syntax 'FSM.ecore'
  → with ExecutableFsm
  → with ExecutableState
  → with ExecutableTransition

}
```

LANGUAGE DEFINITION

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

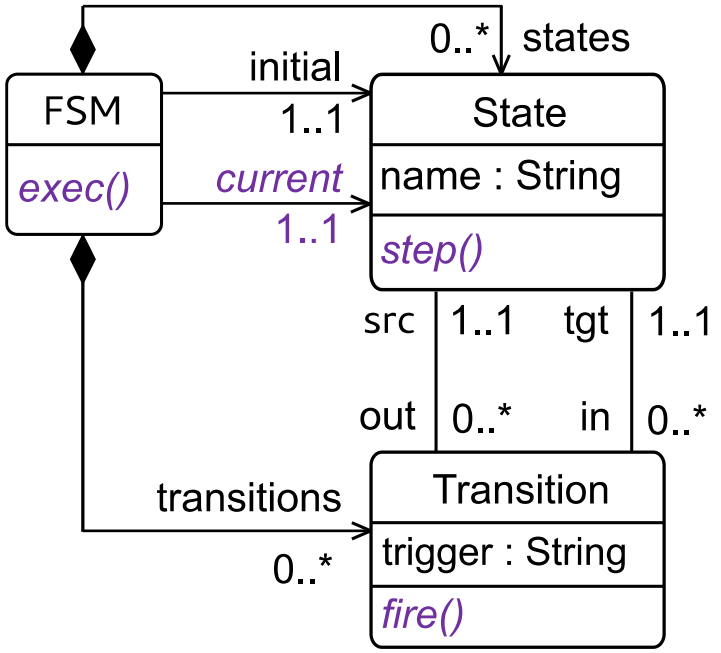
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language Fsm {
    syntax 'FSM.ecore'
    with ExecutableFsm
    with ExecutableState
    with ExecutableTransition
  → exactType FsmMT
}
```

SYNTAX MERGING

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

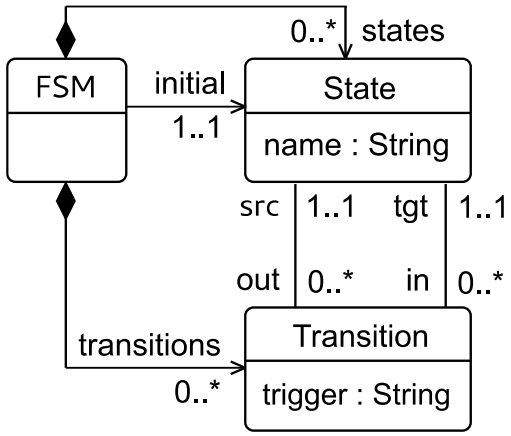
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language GuardedFsm {
    syntax 'FSM.ecore'

    with ExecutableFsm
    with ExecutableState
    with ExecutableTransition


    exactType GuardedFsmMT
}
```

SYNTAX MERGING

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

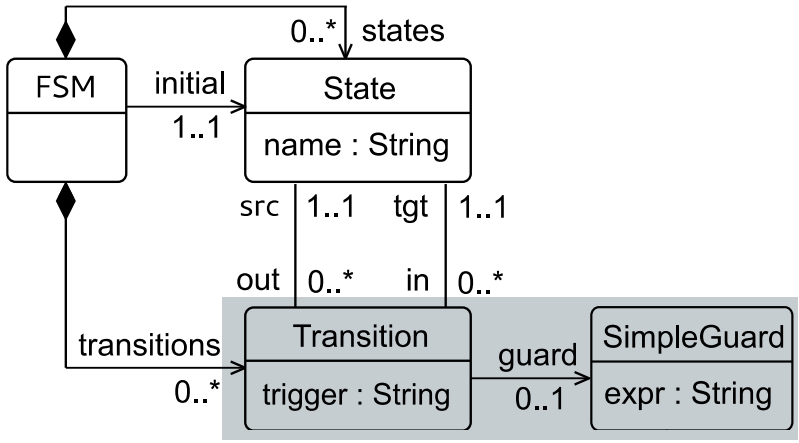
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language GuardedFsm {
    syntax 'FSM.ecore'
    syntax 'Guard.ecore'
    with ExecutableFsm
    with ExecutableState
    with ExecutableTransition


    exactType GuardedFsmMT
}
```

SEMANTICS WEAVING

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

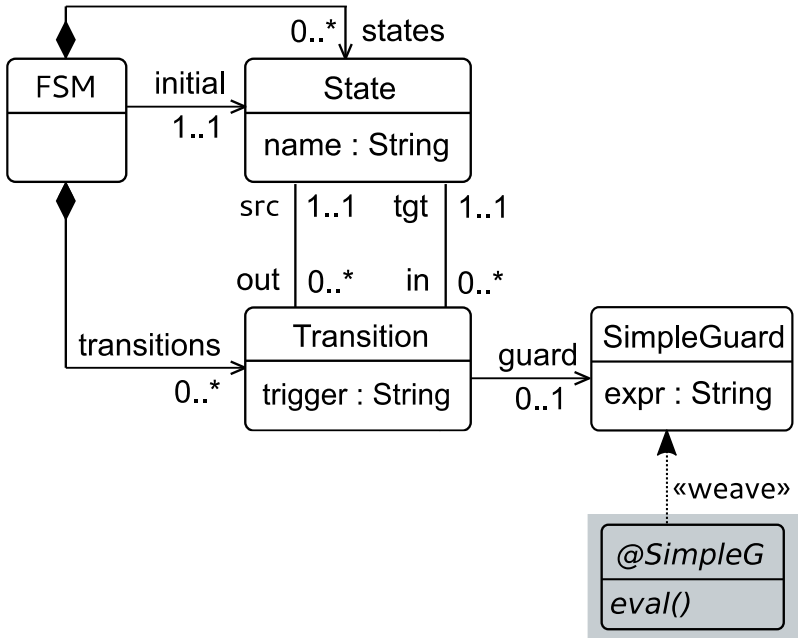
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language GuardedFsm {
    syntax 'FSM.ecore'
    syntax 'Guard.ecore'
    with ExecutableFsm
    with ExecutableState
    with ExecutableTransition
  → with EvaluateGuard

    exactType GuardedFsmMT
}
```

SEMANTICS WEAVING

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

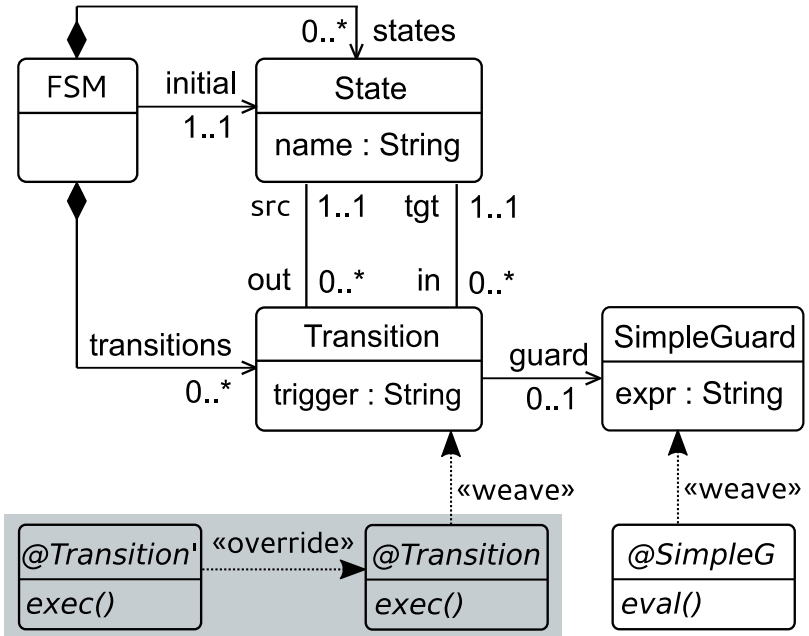
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language GuardedFsm {
    syntax 'FSM.ecore'
    syntax 'Guard.ecore'
    with ExecutableFsm
    with ExecutableState
    with ExecutableTransition
    with EvaluateGuard
→   with OverrideTransition
    exactType GuardedFsmMT
}
```

LANGUAGE MERGING

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \mathop{\bigcup}_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

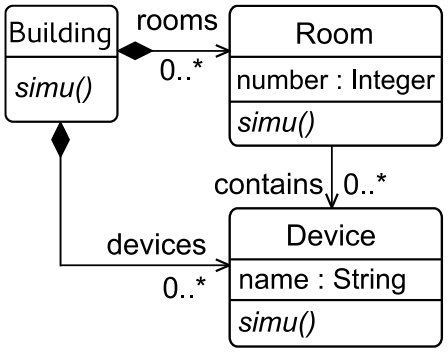
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language Building {
    syntax 'Building.ecore'
    with SimulatorAspect...


    exactType BuildingMT
}
```

LANGUAGE MERGING

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

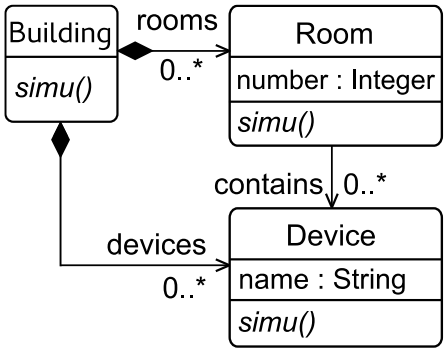
$$MT'' <: MT'$$

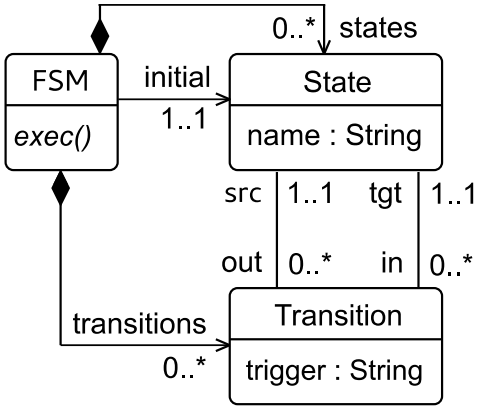
$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$






```
language Building {
    syntax 'Building.ecore'
    with SimulatorAspect...
  → merge Fsm

    exactType BuildingMT
}
```

LANGUAGE MERGING

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \mathop{\overset{\circ}{\bigcup}}_{A_i^t \in Sem} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

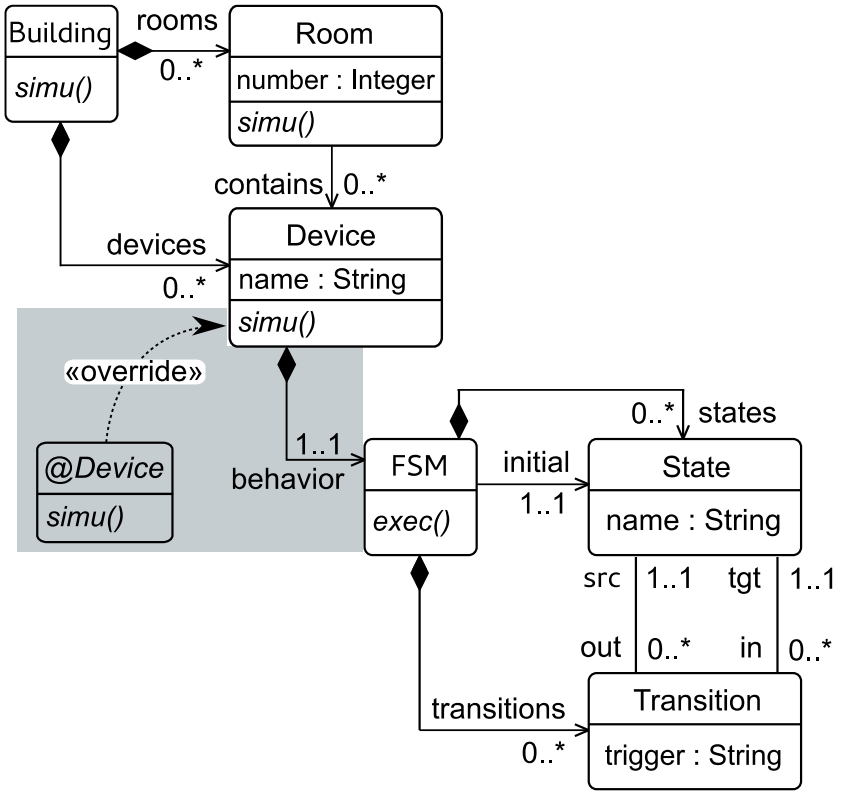
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language Building {
    syntax 'Building.ecore'
    with SimulatorAspect...
    merge Fsm
  → with GlueDeviceToFsm
    exactType BuildingMT
}
```

LANGUAGE INHERITANCE

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

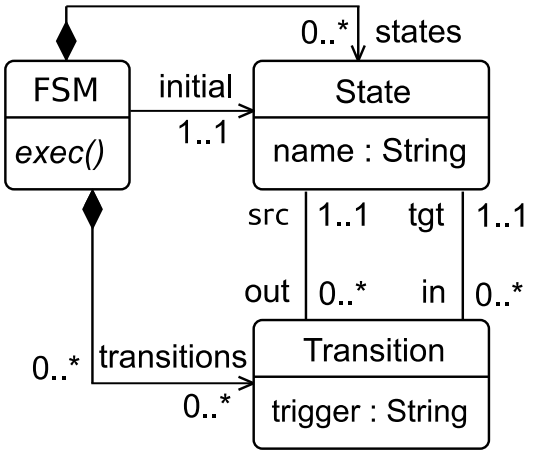
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language TimedFsm inherits Fsm {

    exactType TimedFsmMT
}
```

LANGUAGE INHERITANCE

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

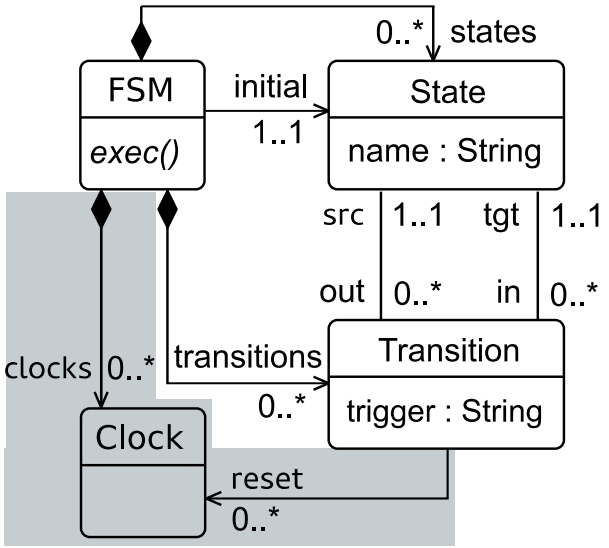
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language TimedFsm inherits Fsm {
 → syntax 'Clocks.ecore'


    exactType TimedFsmMT
}
```

LANGUAGE INHERITANCE

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

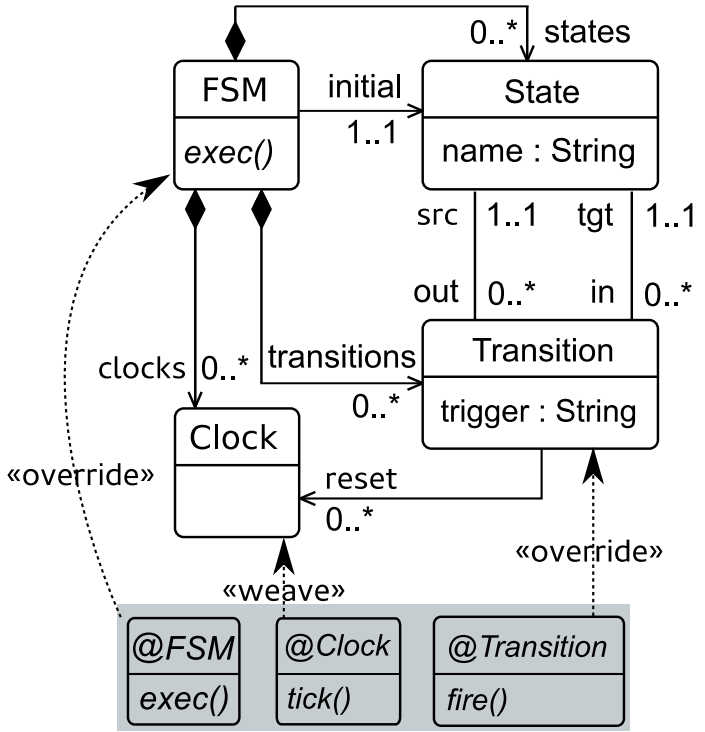
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language TimedFsm inherits Fsm {
    syntax 'Clocks.ecore'
  → with ClockTick
  → with OverrideFsm
  → with OverrideTransition
    exactType TimedFsmMT
}
```

LANGUAGE SLICING

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \mathop{\Cup}_{A_i^t \in Sem} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

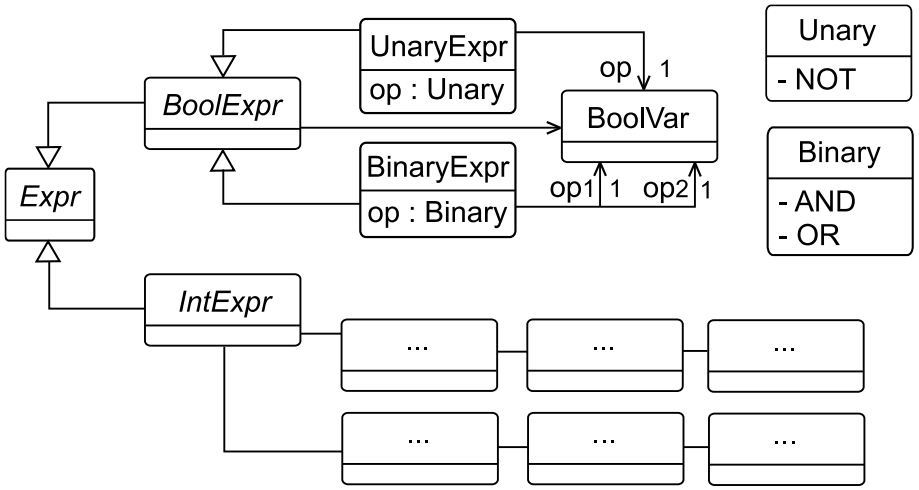
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language Expressions {
    syntax 'Expressions.ecore'
    with EvaluateBoolean
    with EvaluateInteger
    exactType ExpressionsMT
}
```

LANGUAGE SLICING

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \overset{m}{\leftarrow} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \overset{w}{\leftarrow} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

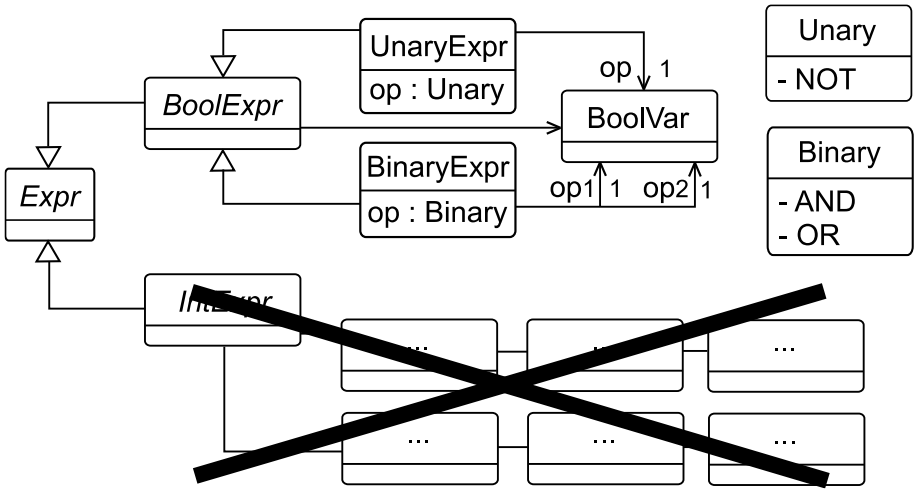
$$MT'' <: MT'$$

$$\Lambda_{-}^{+}(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

$$AS_2 \triangleq \lambda_{-}^{+}(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language Expressions {
    syntax 'Expressions.ecore'
    with EvaluateBoolean
    with EvaluateInteger
    exactType ExpressionsMT
}

language BooleanExpressions {
 → slice Expressions on ['BoolExpr']
    exactType BooleanExpressionsMT
}
```

LANGUAGE SLICING

$$\mathcal{L} \triangleq \langle AS, Sem, MT \rangle$$

$$Sem(\mathcal{L}) \triangleq (A_i^t \in Aspects) \text{ where}$$

$$\forall A_i^t \in Sem(\mathcal{L}), \exists c \in AS(\mathcal{L}) : c \text{ match } t$$

$$\forall A_i^t, A_j^t \in Sem(\mathcal{L}) : A_i^t \lhd A_j^t \implies i > j$$

$$Sem \bullet Sem' \equiv Sem \frown Sem'$$

$$sig(Sem) \triangleq \bigcup_{A_i^t \in Sem}^{\circ} sig(A_i^t)$$

$$MT(\mathcal{L}) \triangleq AS(\mathcal{L}) \circ sig(Sem(\mathcal{L}))$$

$$\mathcal{L} \xleftarrow{m} AS' = \langle AS \circ AS', Sem, MT \circ AS' \rangle$$

$$\mathcal{L} \xleftarrow{w} Sem' = \langle AS, Sem \bullet Sem', MT \circ sig(Sem') \rangle$$

$$\mathcal{L} \uplus \mathcal{L}' = \langle AS \circ AS', Sem \bullet Sem', MT \circ MT' \rangle$$

$$\mathcal{L} \oplus \mathcal{L}' = \langle AS \circ AS', Sem' \bullet Sem, MT'' \rangle \text{ where}$$

$$MT'' = MT \circ MT' \text{ and}$$

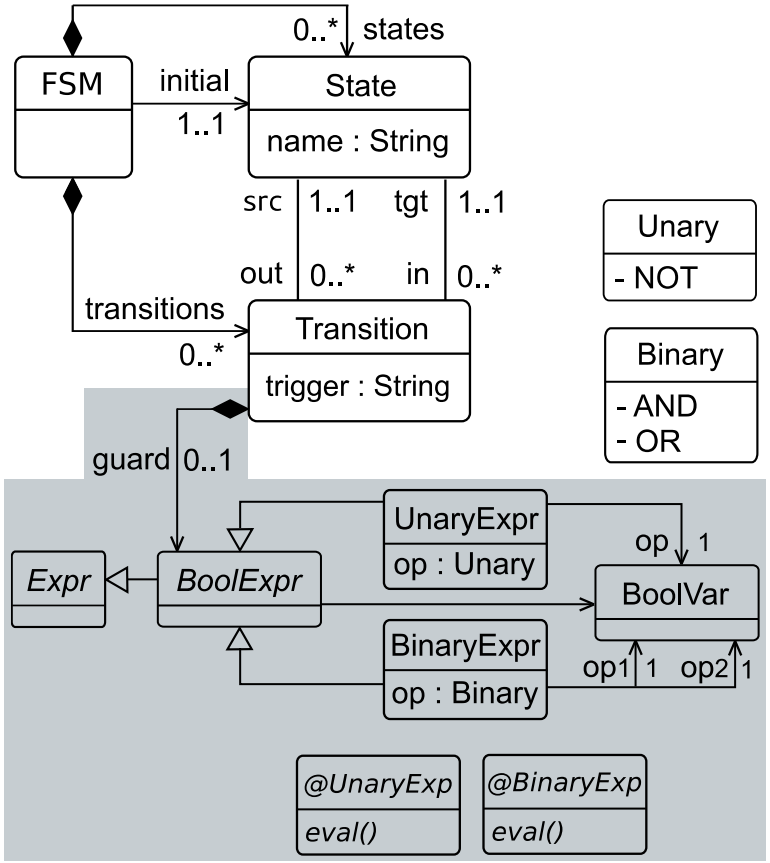
$$MT'' <: MT'$$

$$\Lambda_-^+(\mathcal{L}_1, c) = \langle AS_2, Sem_2, MT_2 \rangle, \text{ where:}$$

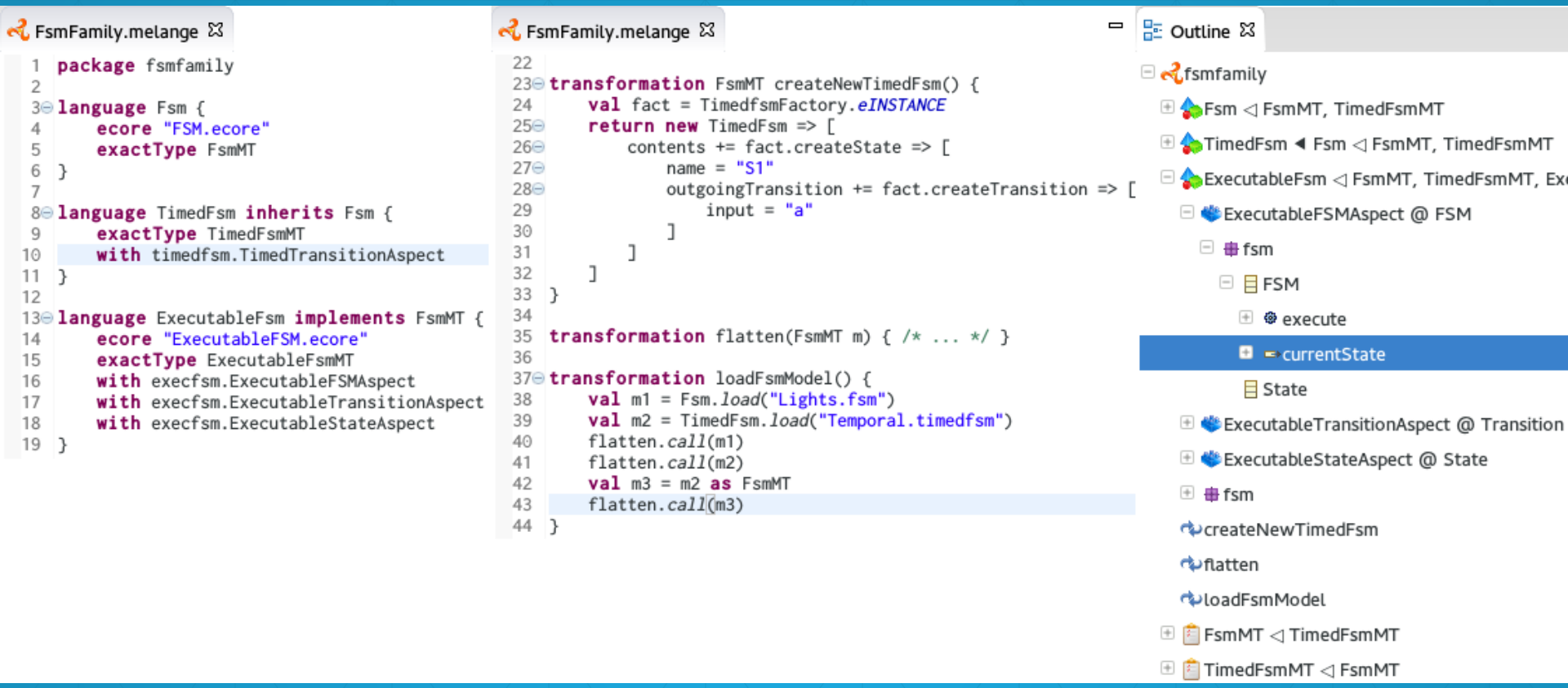
$$AS_2 \triangleq \lambda_-^+(AS_1, c),\ AS_2 \subseteq AS_1,$$

$$Sem_2 \triangleq \left\{ A_i^t \in Sem_1, fp(A_i^t, AS_1) \subseteq AS_2 \right\},$$

$$MT_1 <: MT_2,$$




```
language GuardedFsm inherits Fsm {
    with ...
    merge BooleanExpressions
    with AttachGuardToTransition
    exactType GuardedFsmMT
}
```



MELANGE

A Language Workbench

MELANGE

- An open-source (EPL) language workbench
- or... a language-based, model-oriented language for DSL engineering
- An implementation of the algebra
- Supported by a model-oriented type sytem
- Based on Xtext
- Seamlessly integrated with the EMF ecosystem
- Bundled as a set of Eclipse plug-ins

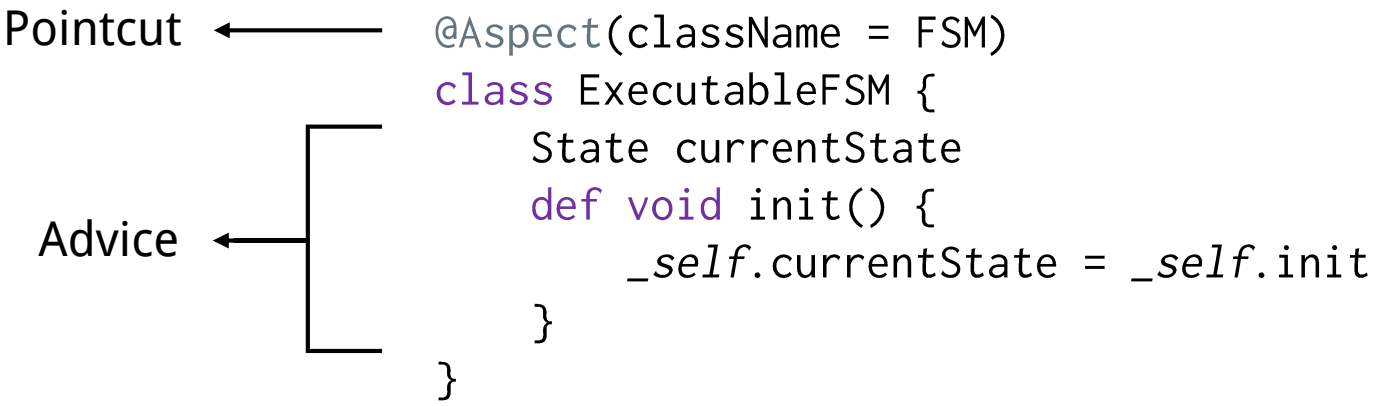
IMPLEMENTATION CHOICES

- **Abstract syntax**: Ecore (EMOF)
- **Merging**: Customized UML PackageMerge[1]
  - Trading UML specificities with EMOF specificities
  - Support for renaming
- **Slicing**: Kompren[2]
- **Operational semantics**: K3 (Xtend on steroids)

[1] Dingel *et al.*, *Understanding and Improving UML PackageMerge*, SoSyM, 2008
[2] Blouin *et al.*, *Kompren: Modeling and generating model slicers*, SoSyM, 2012

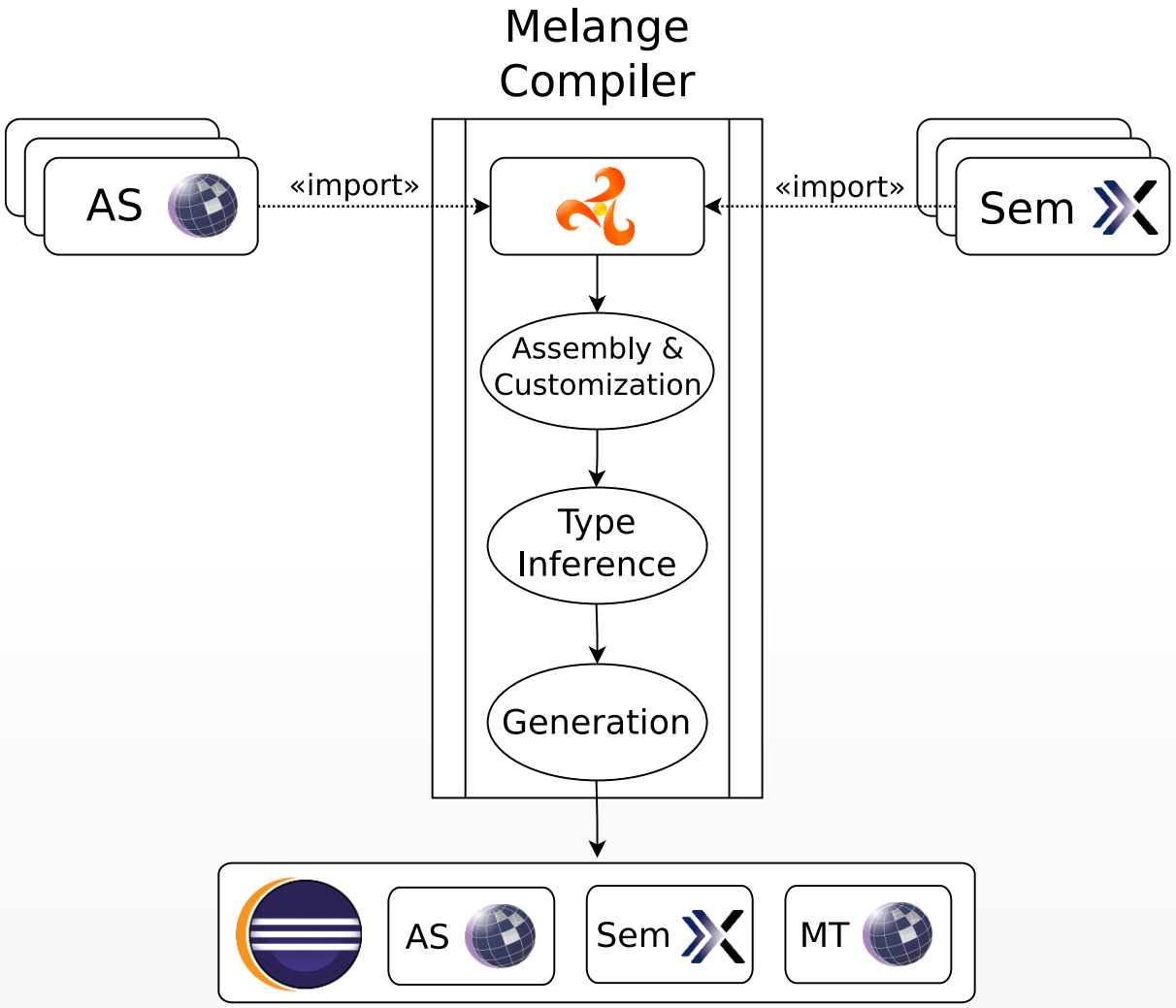
IMPLEMENTATION CHOICES

- **Abstract syntax**: Ecore (EMOF)
- **Merging**: Customized UML PackageMerge[1]
  - Trading UML specificities with EMOF specificities
  - Support for renaming
- **Slicing**: Kompren[2]
- **Operational semantics**: K3 (Xtend on steroids)

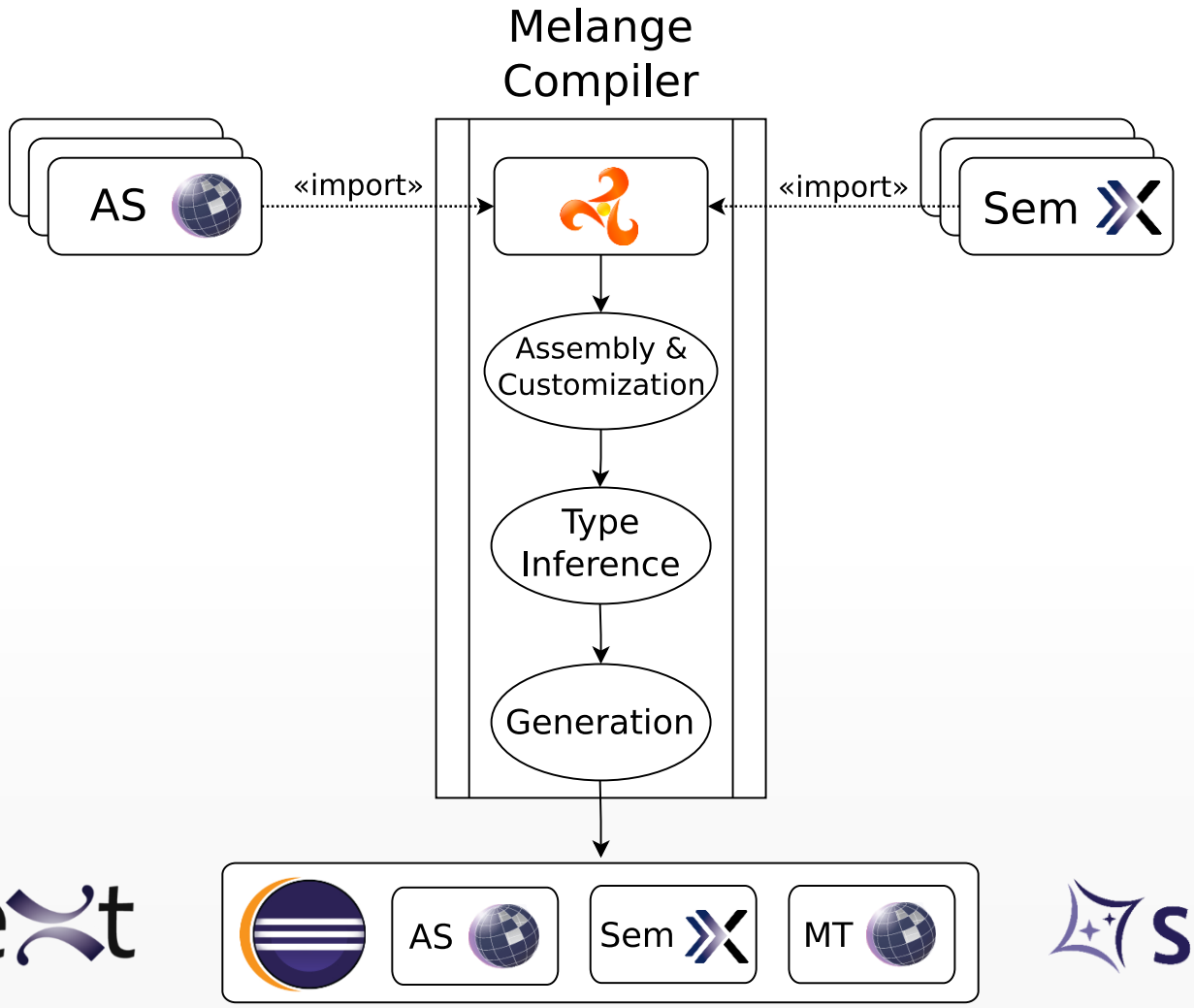


[1] Dingel *et al.*, *Understanding and Improving UML PackageMerge*, SoSyM, 2008
[2] Blouin *et al.*, *Kompren: Modeling and generating model slicers*, SoSyM, 2012

COMPILATION SCHEME



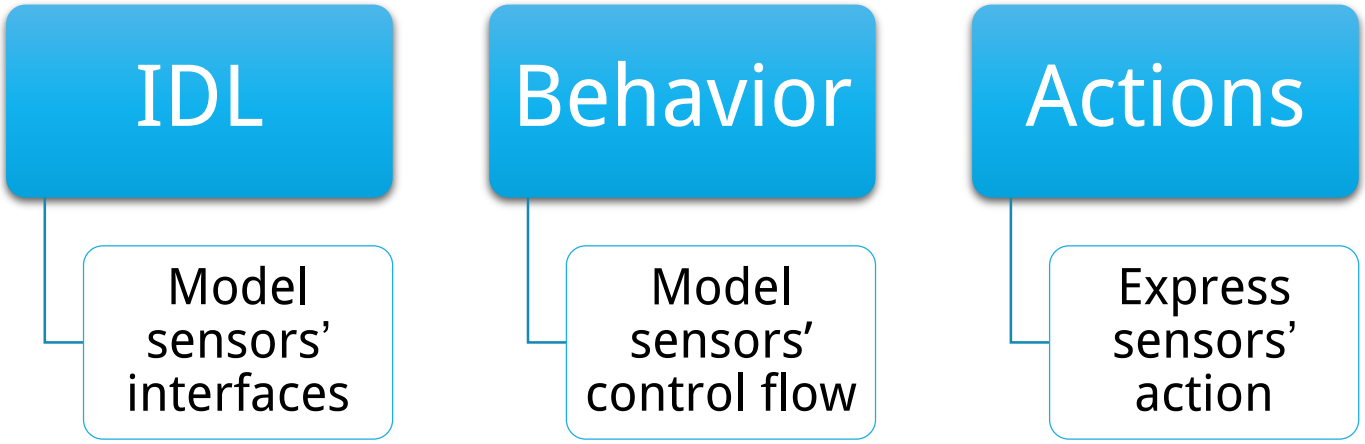
COMPILATION SCHEME



CASE STUDY

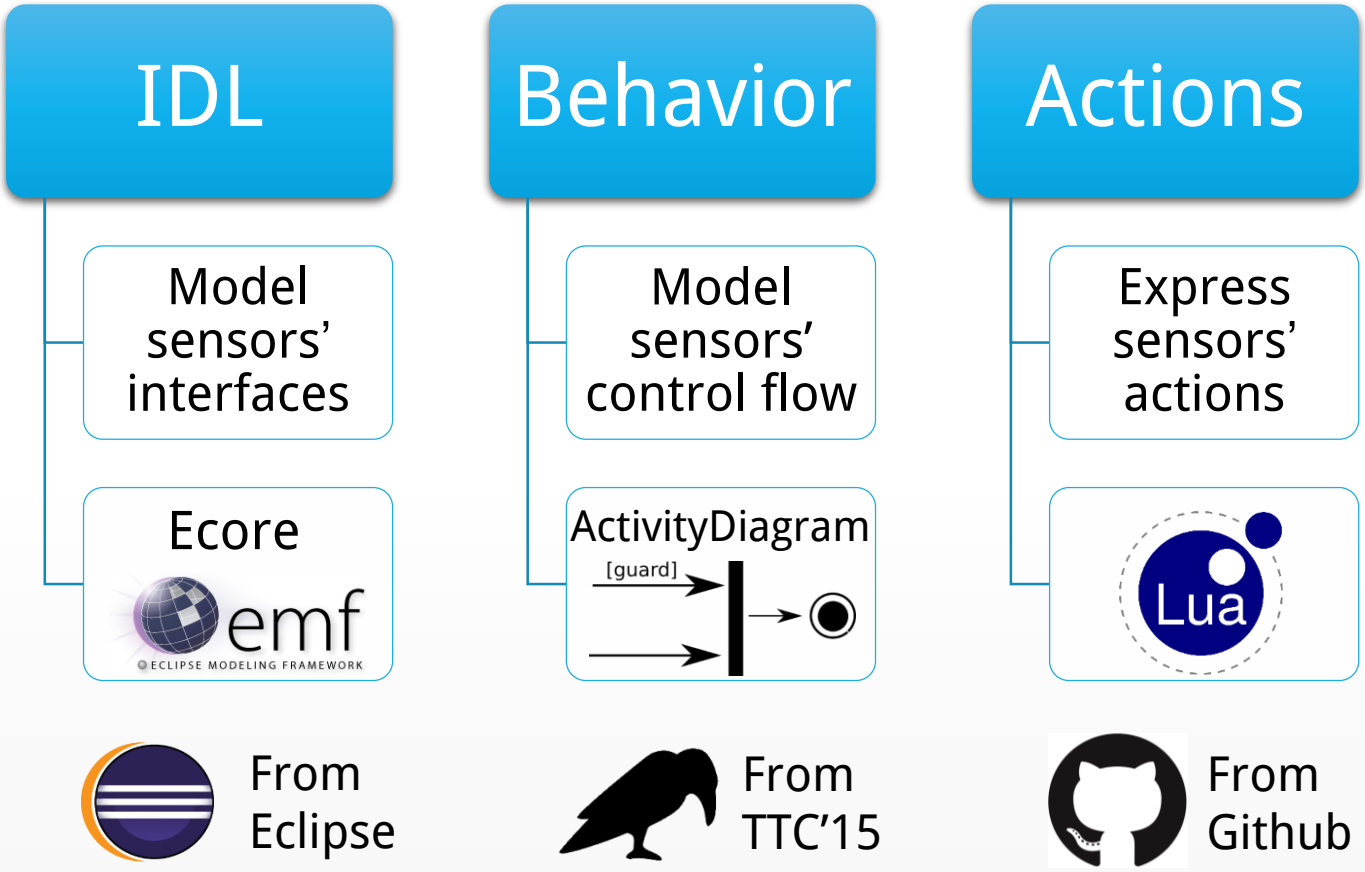
An Executable Modeling Language for the Internet of Things

REQUIREMENTS FOR THE IOT LANGUAGE

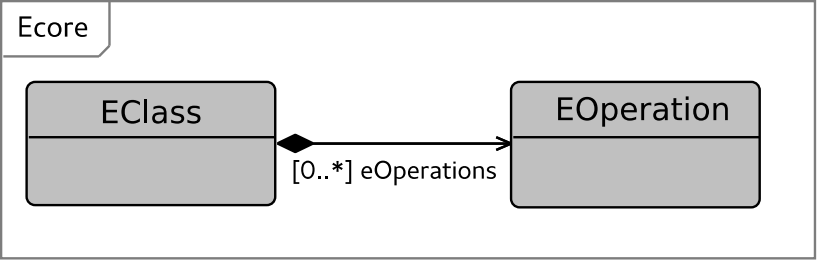


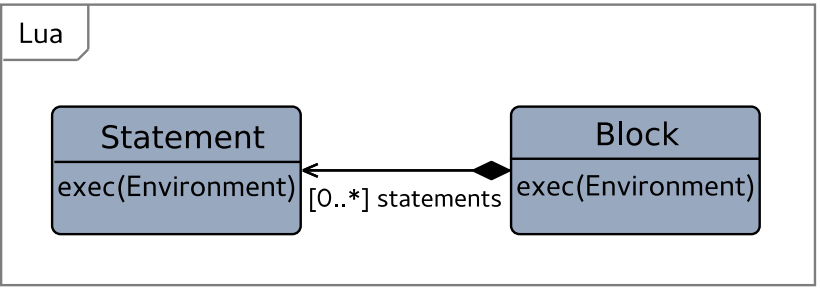
cf. fUML, ThingML, etc.

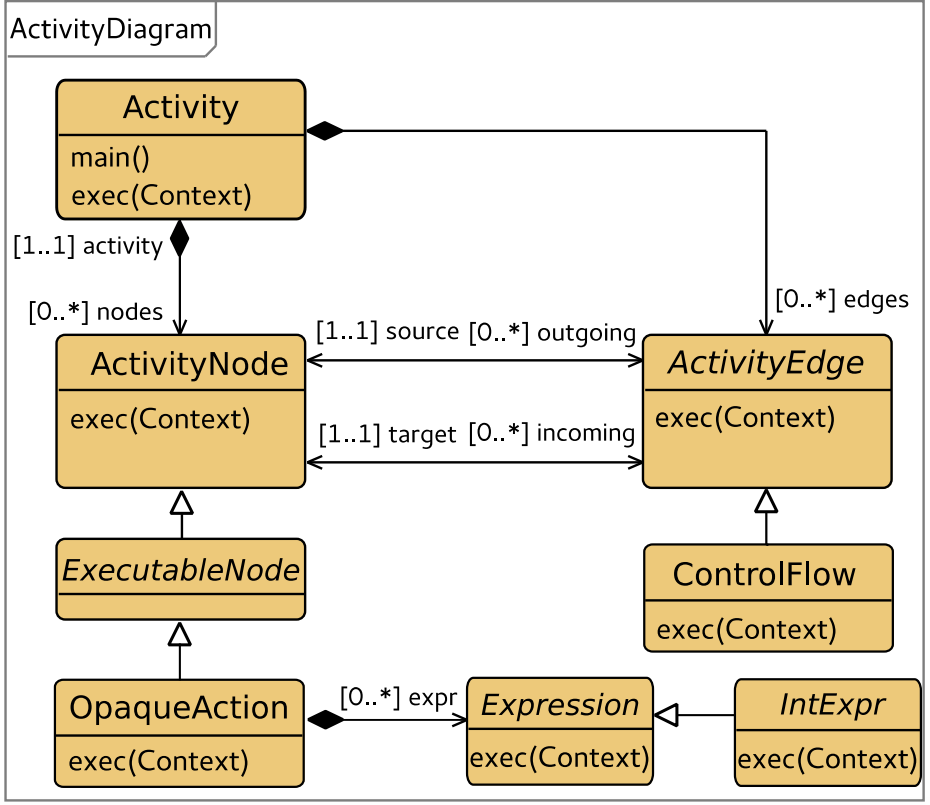
REQUIREMENTS FOR THE IOT LANGUAGE



cf. fUML, ThingML, etc.



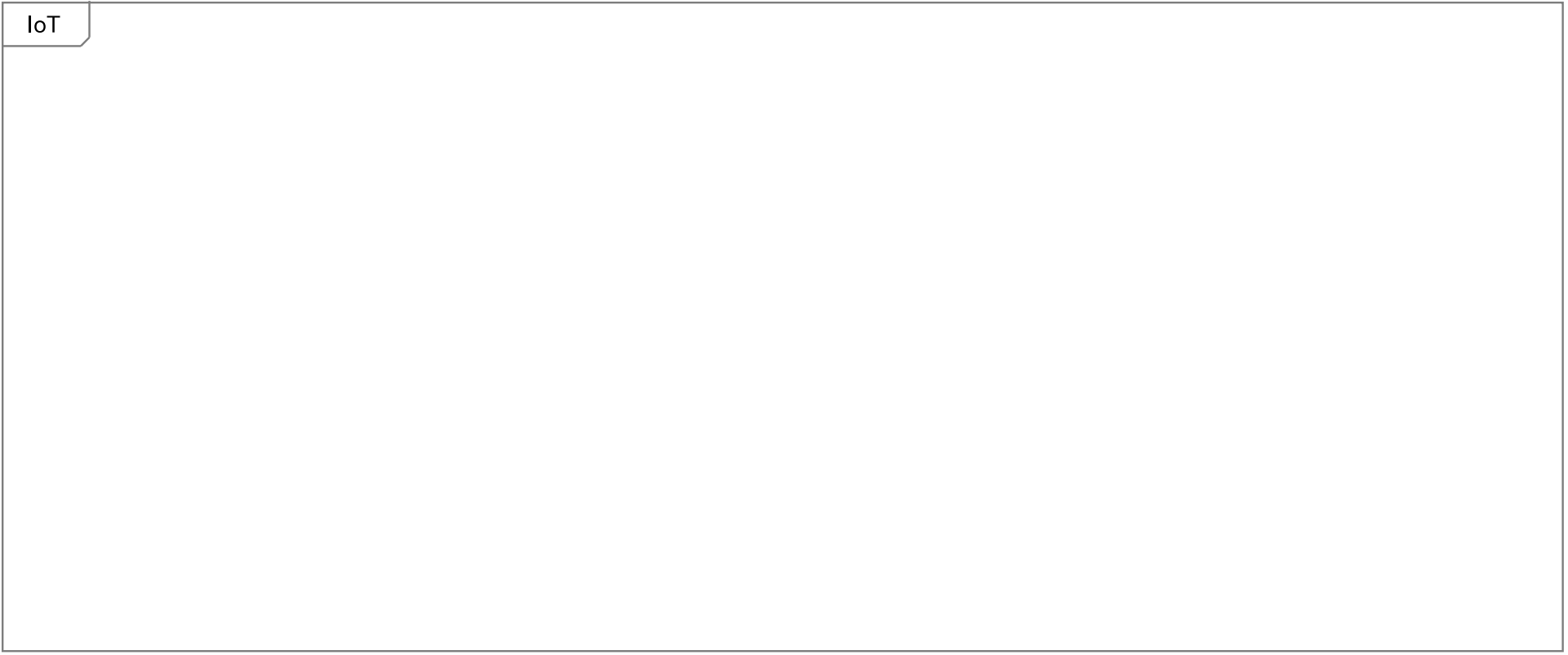




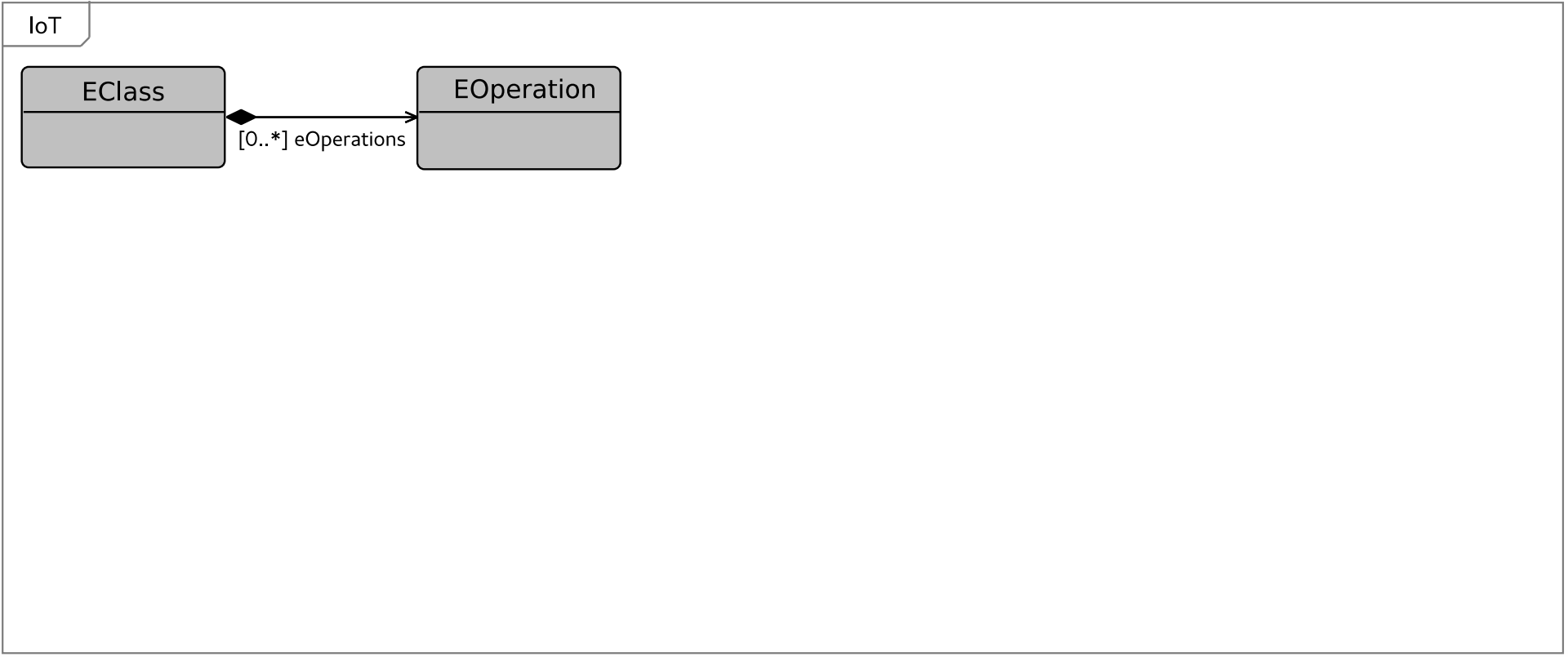

```
language Ecore {
    syntax 'Ecore.ecore'
    exactType EcoreMT
}
```

```
language Activity {
    syntax 'Activity.ecore'
    with activity.semantics.*
    exactType ActivityMT
}
```

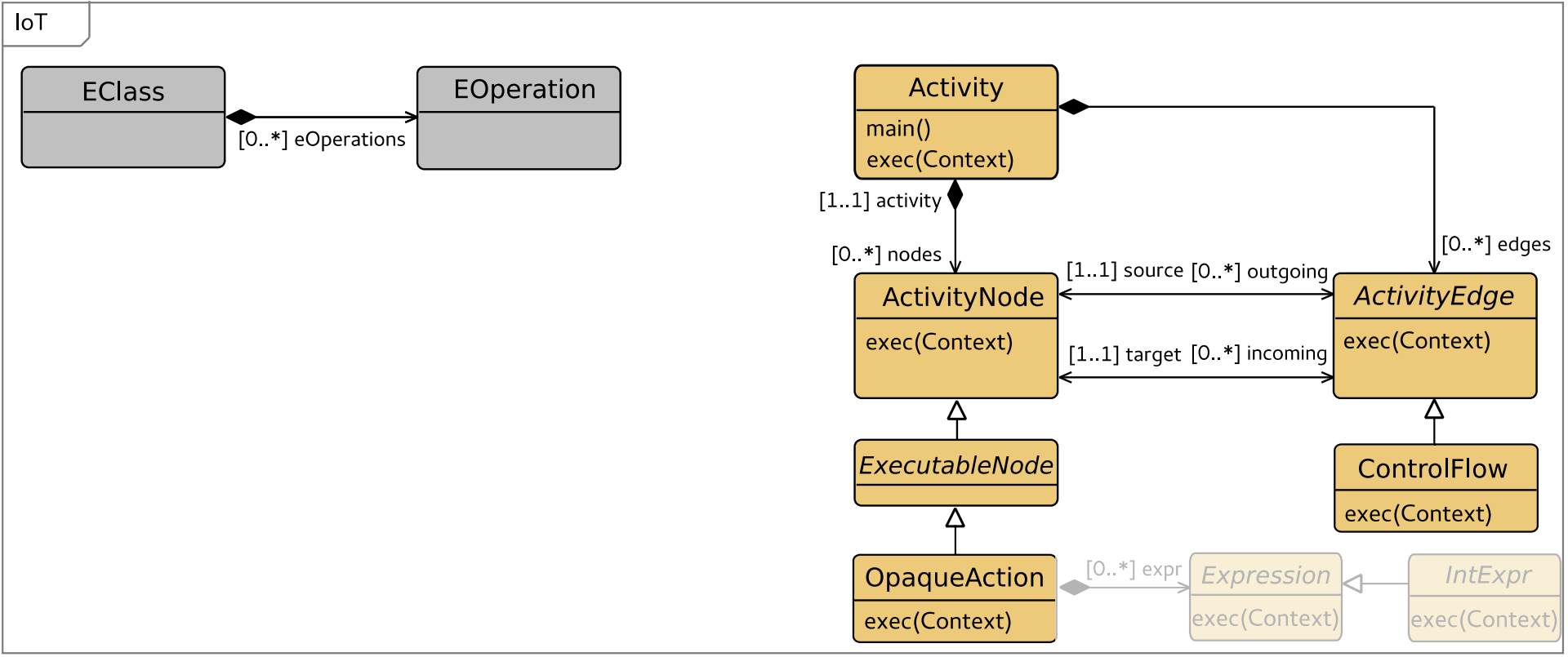
```
language Lua {
    syntax 'Lua.ecore'
    with lua.semantics.*
    exactType LuaMT
}
```



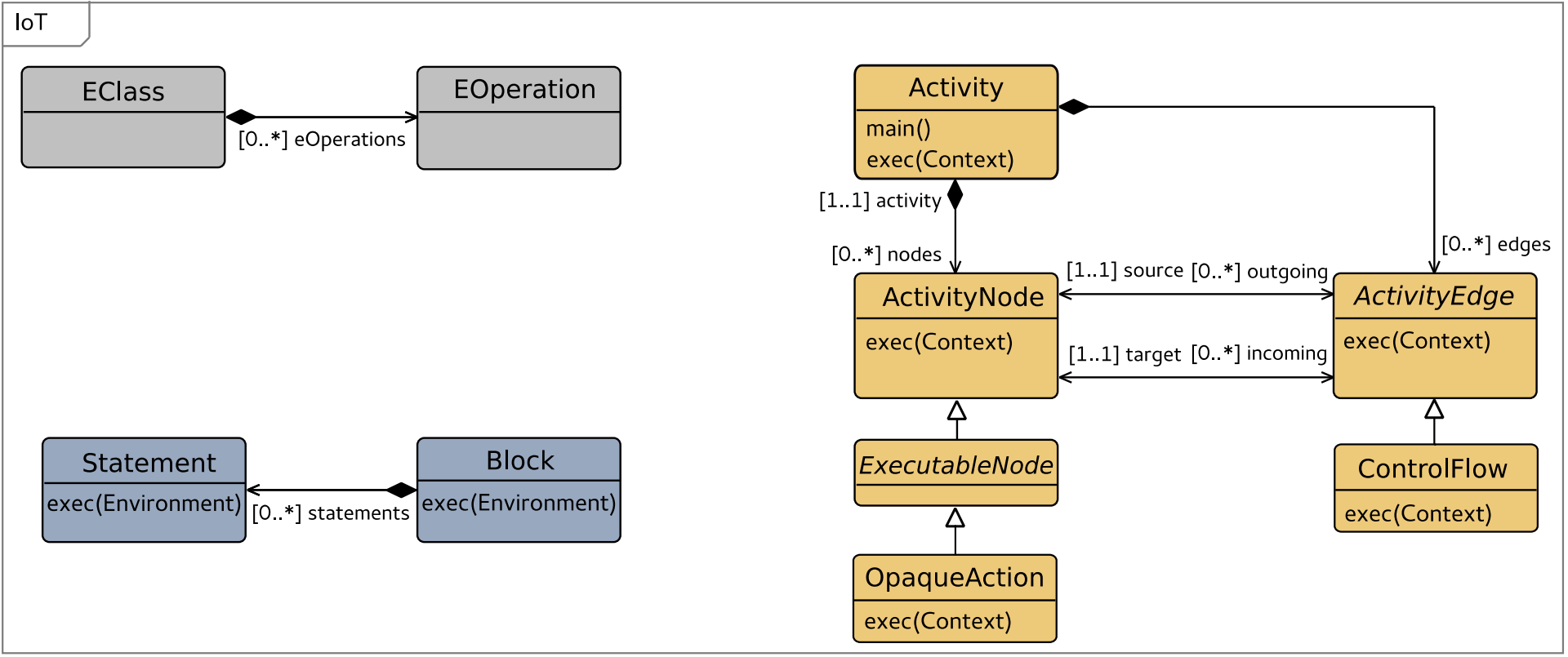

```
language IoT {

    exactType IoTMT
}
```



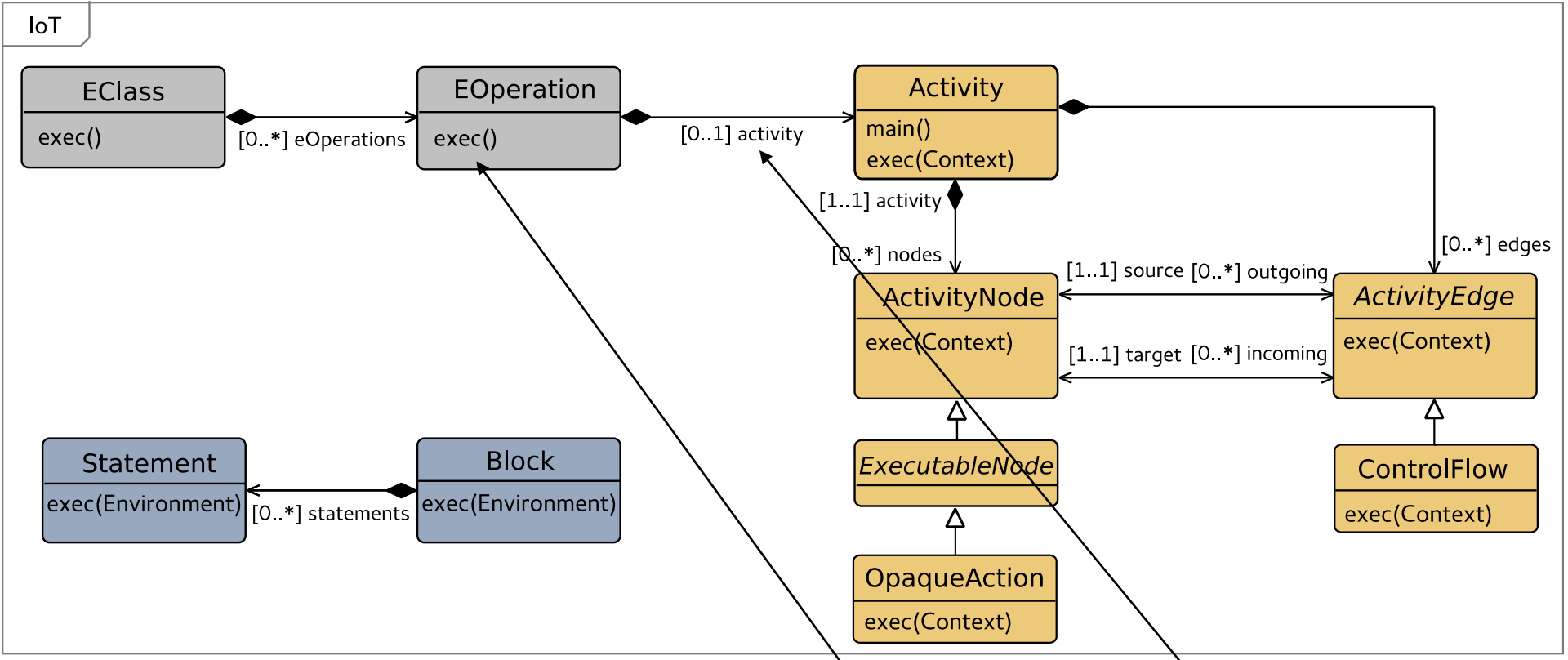

```
language IoT {
 → merge Ecore renaming { ‘ecore’ to ‘IoT’ }




    exactType IoTMT
}
```




```
language IoT {
    merge Ecore renaming { ‘ecore’ to ‘IoT’ }
 → slice Activity on [‘OpaqueAction’, ...]
         renaming { ‘activitydiagram’ to ‘IoT’ }



    exactType IoTMT
}
```

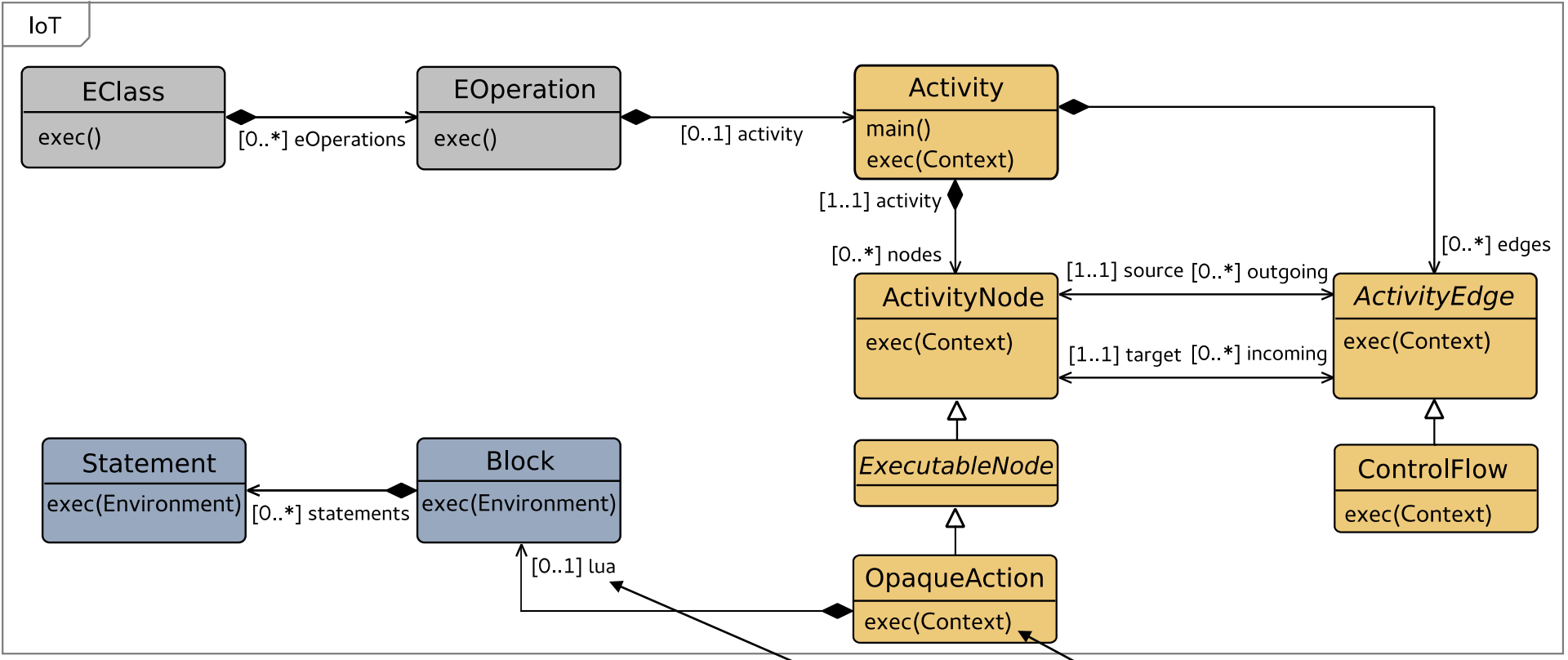



```
language IoT {
    merge Ecore renaming { 'ecore' to 'IoT' }
    slice Activity on ['InputValue', ...]
          renaming { 'activitydiagram' to 'IoT' }
 → merge Lua renaming { 'lua' to 'IoT' }


    exactType IoTMT
}
```




```
language IoT {
    merge Ecore renaming { 'ecore' to 'IoT' }
    slice Activity on ['InputValue', ...]
          renaming { 'activitydiagram' to 'IoT' }
    merge Lua renaming { 'lua' to 'IoT' }
 → with iot.EOperationActivity

    exactType IoTMT
}
```

```
@Aspect(className = EOperation)
class EOperationActivity {
    @Containment
    Activity activity
    def void exec() {
        _self.activity.main(#[])
    }
}
```

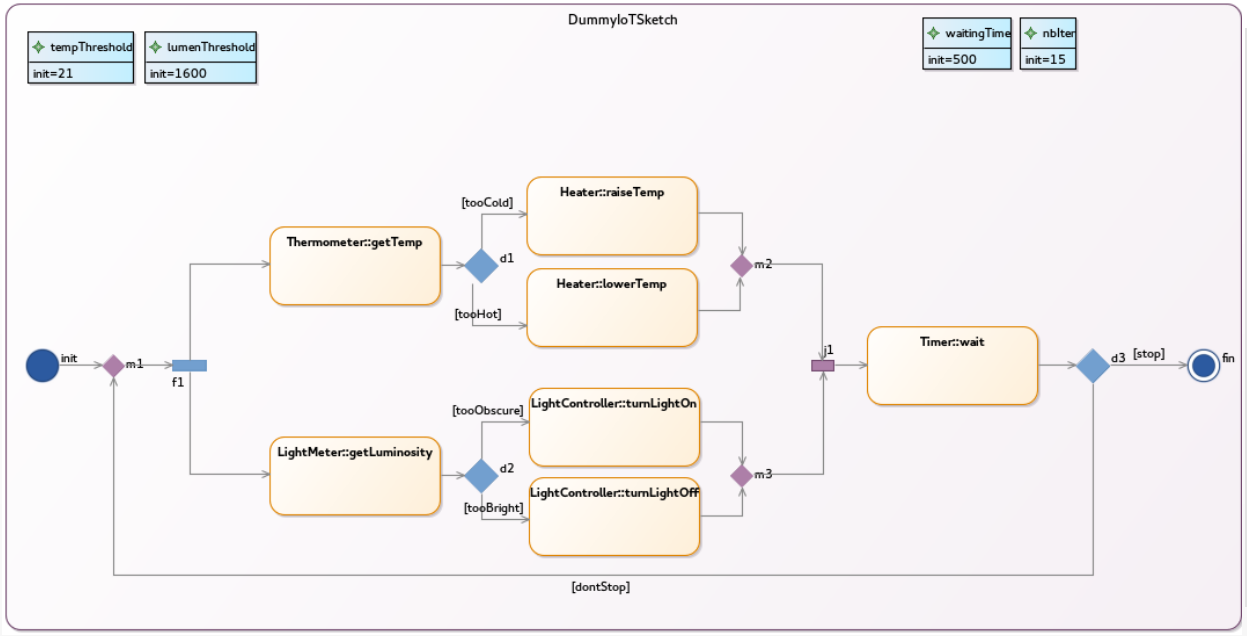


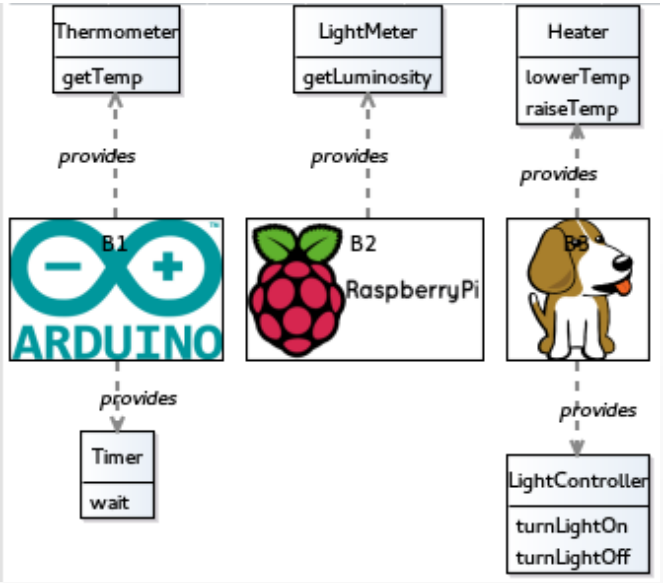

```
language IoT {
    merge Ecore renaming { 'ecore' to 'IoT' }
    slice Activity on ['InputValue', ...]
          renaming { 'activitydiagram' to 'IoT' }
    merge Lua renaming { 'lua' to 'IoT' }
    with iot.EOperationActivity
→   with iot.OpaqueActionBlock
    exactType IoTMT
}
```

```
@Aspect(className = OpaqueAction)
class OpaqueActionBlock {
    @Containment
    Block lua
    def void exec(Context c) {
        // context translation
    }
}
```

RESULTS

- Comparison with a top-down approach
- No runtime penalty
- Reuse and customization operators ease the development
- Glue: ~30 LoC (mainly Lua/AD context translation)





FUTURE WORK

- Viewpoints engineering

- Fine-grained modularity: language units / features / modules

THANK YOU

- Feel free to ask for a demonstration ☺

http://melange-lang.org