Formal Validation of Domain-Specific Languages with Derived Features and Well-Formedness Constraints

Presenter: Nick

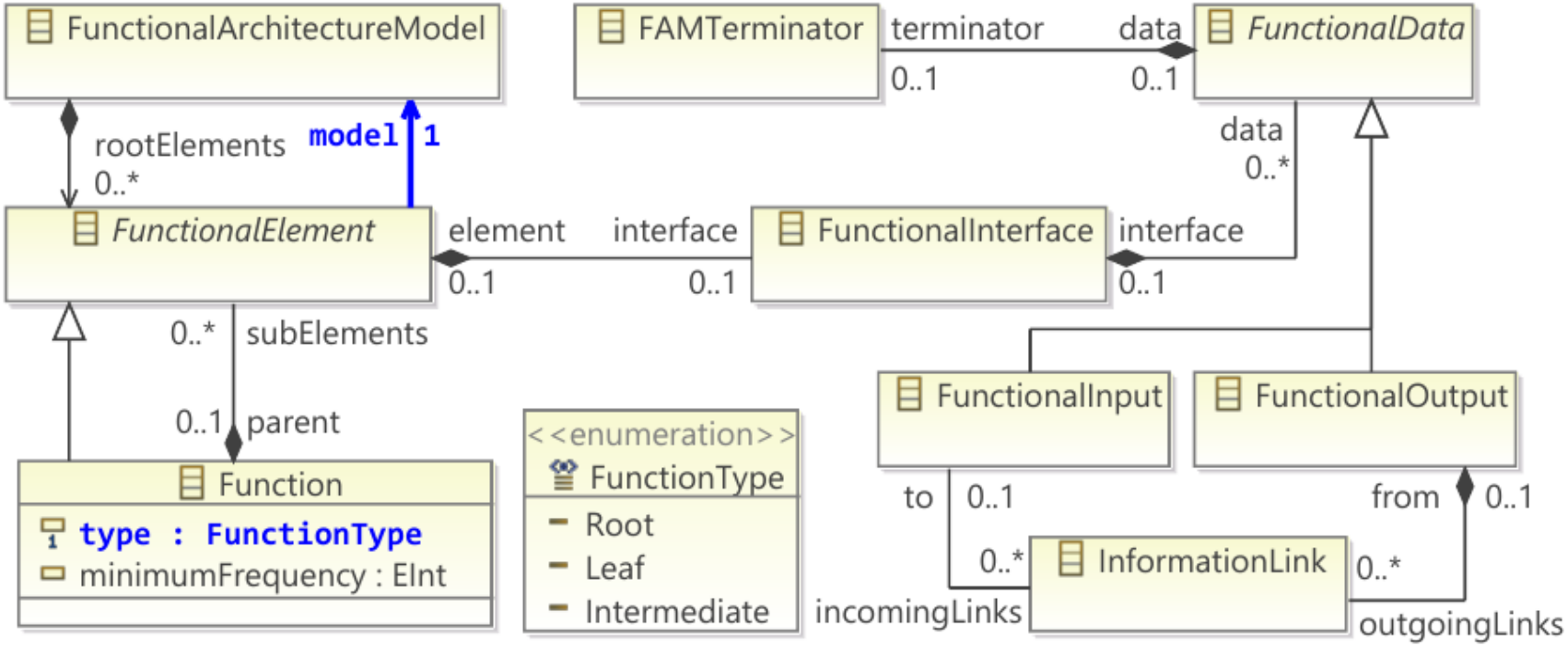
February 05, 2018

Overview

- Motivation
- DSL Validation Tool
- DSL Validation Workflow
- Experimental Results
- Conclusions

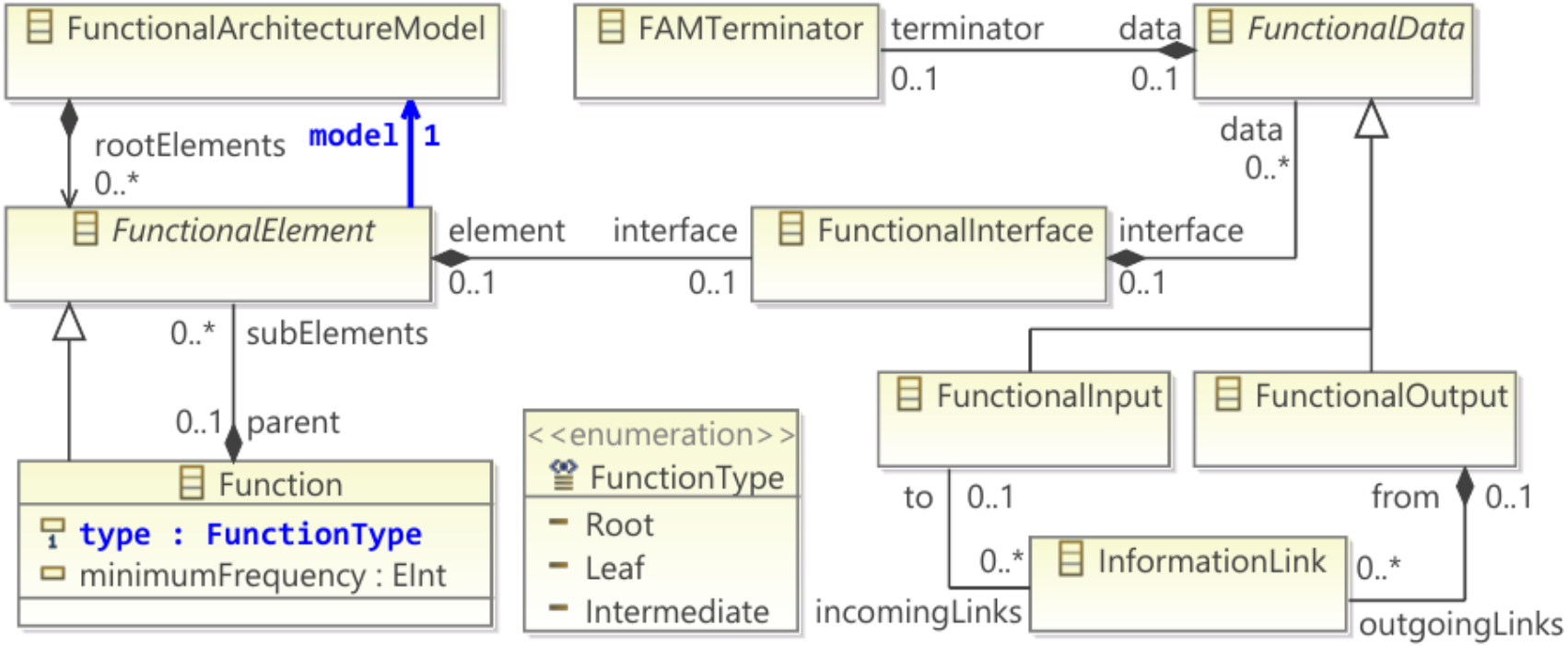
Domain Specific Languages (DSLs)

- **Components of a DSL**
  - Metamodel
  - Derived features
  - Well-formedness constraints



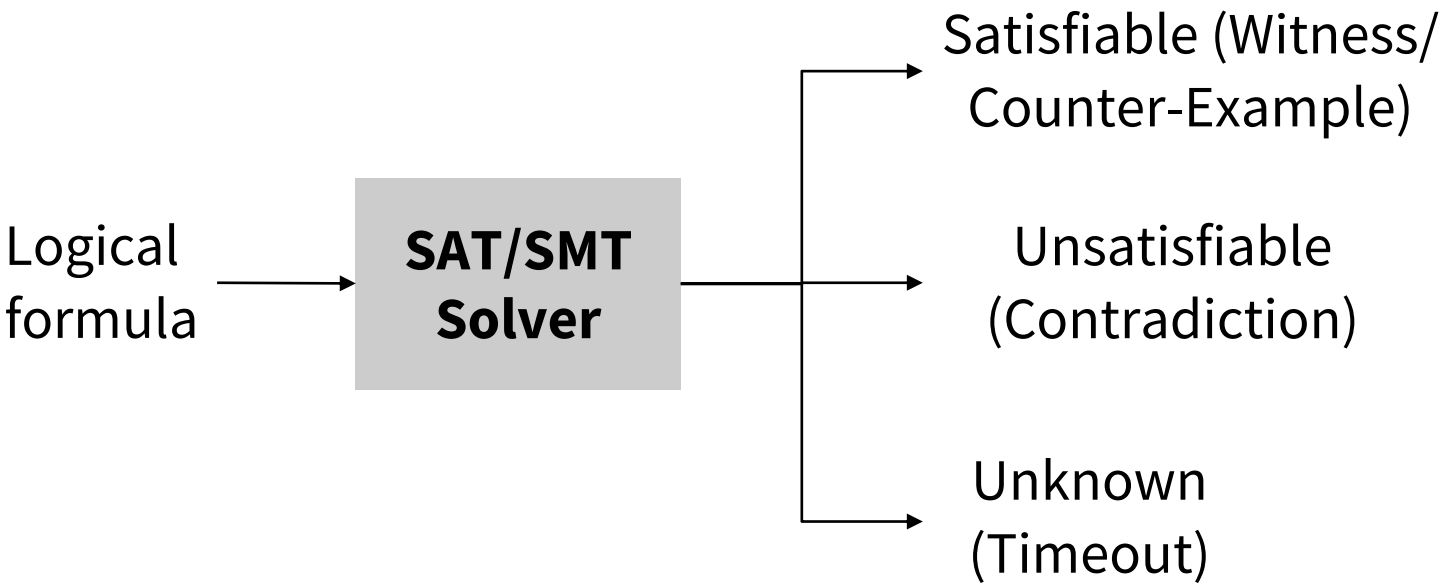
Domain Specific Languages (DSLs)

- **Validation Challenges**
  - Complex metamodel and constraints
  - Infinite range of models

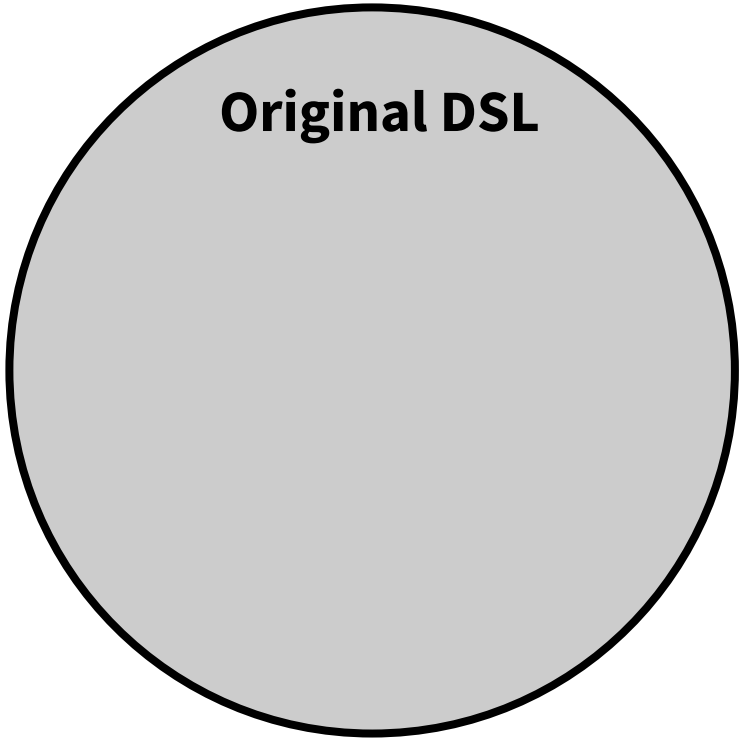


SAT/SMT Solvers

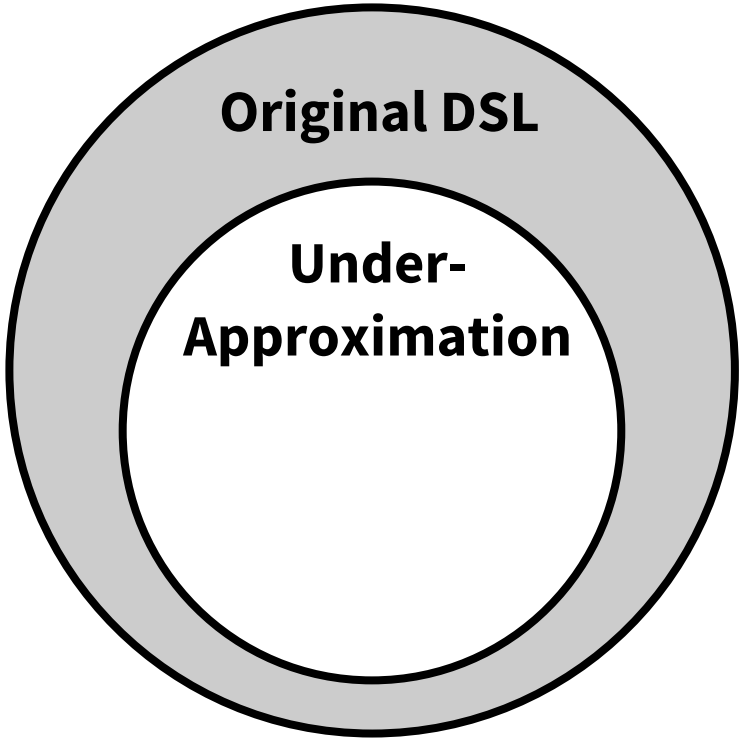
- **Properties**
  - Checks satisfiability of a logical claim.
  - SMT is more expressive than SAT



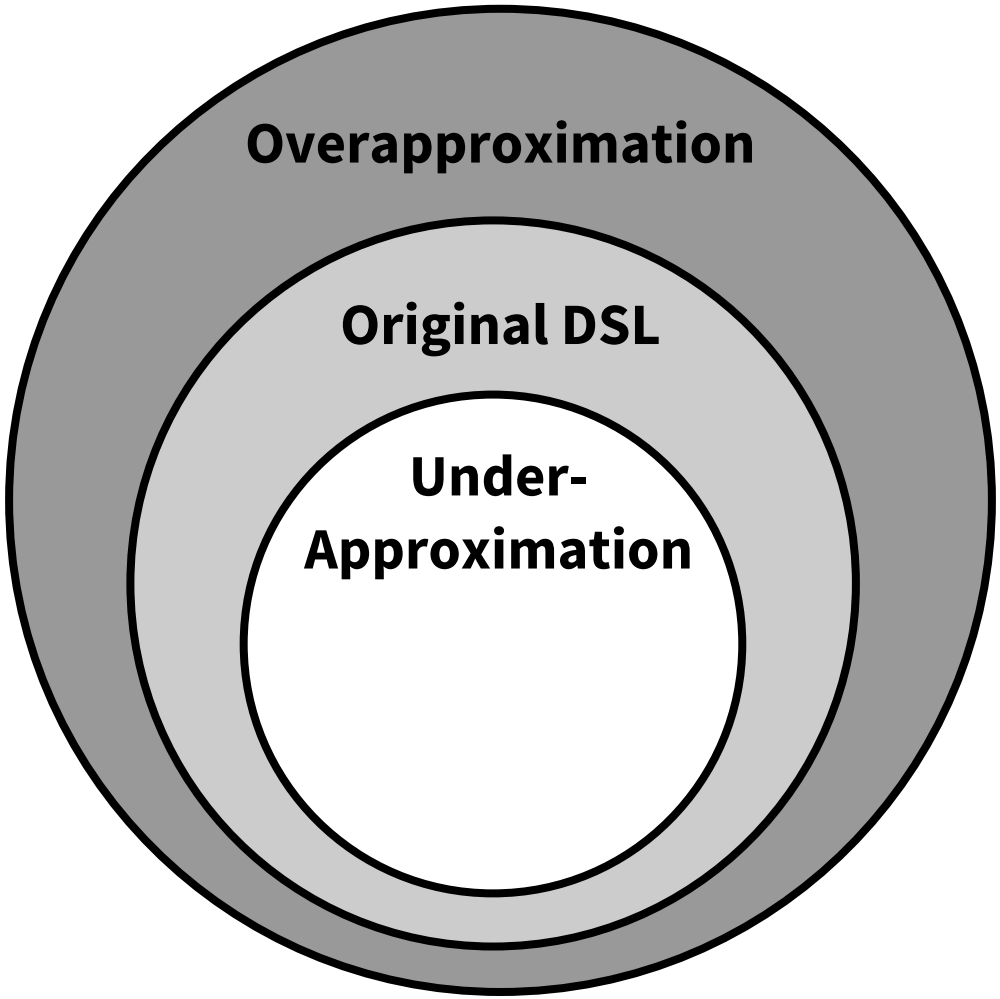
First-Order Logic (FOL) Approximation



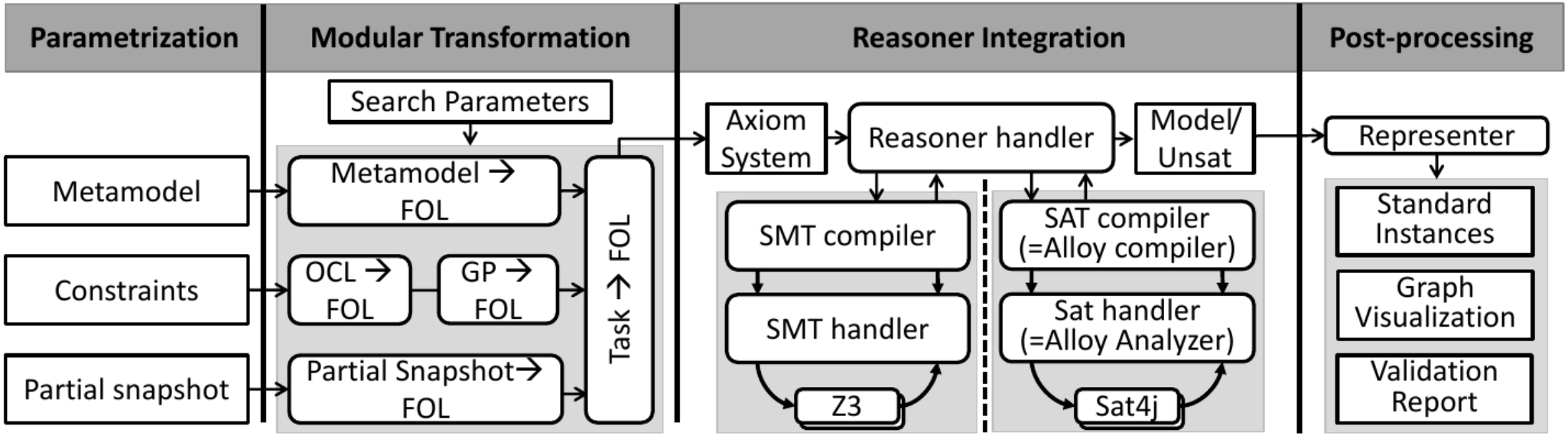
First-Order Logic (FOL) Approximation



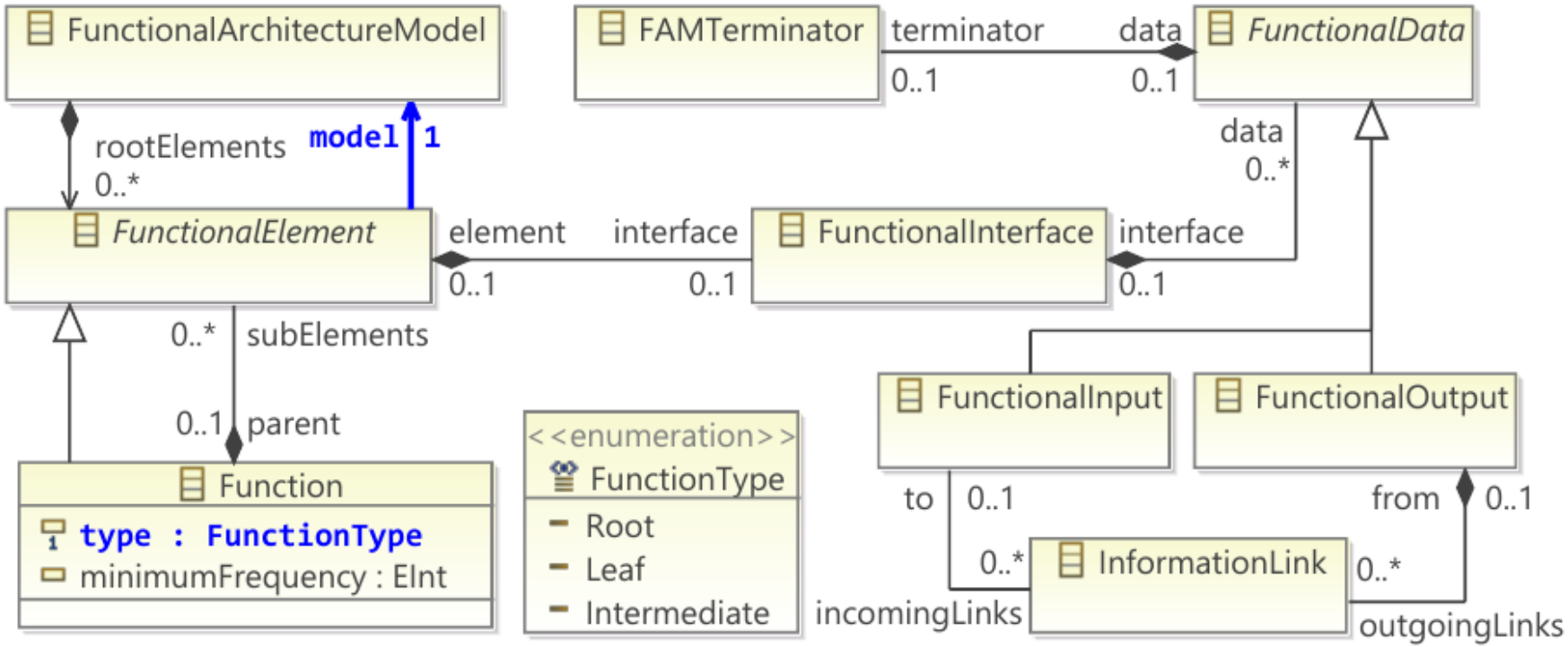
First-Order Logic (FOL) Approximation



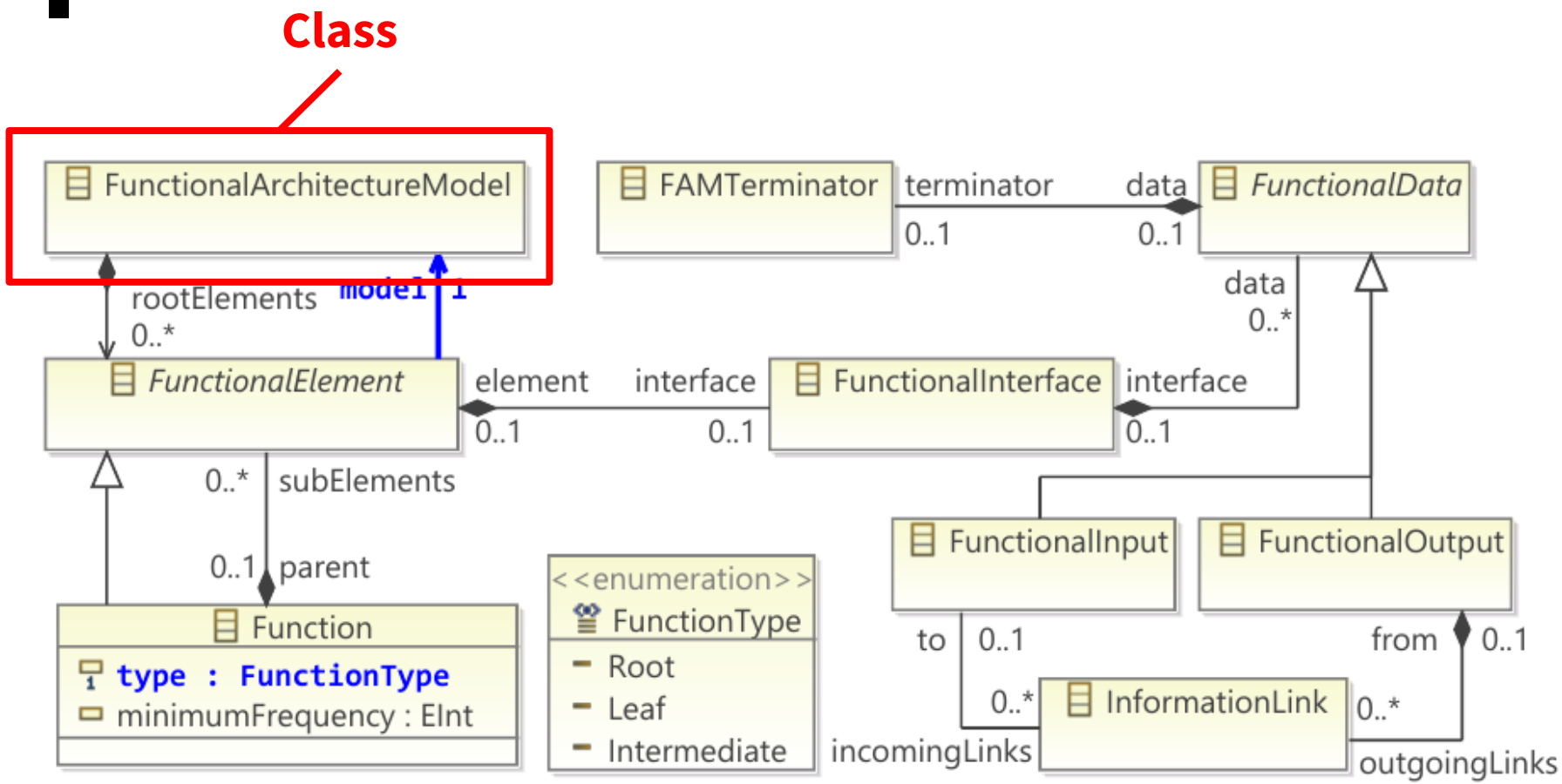
DSL Validation Tool



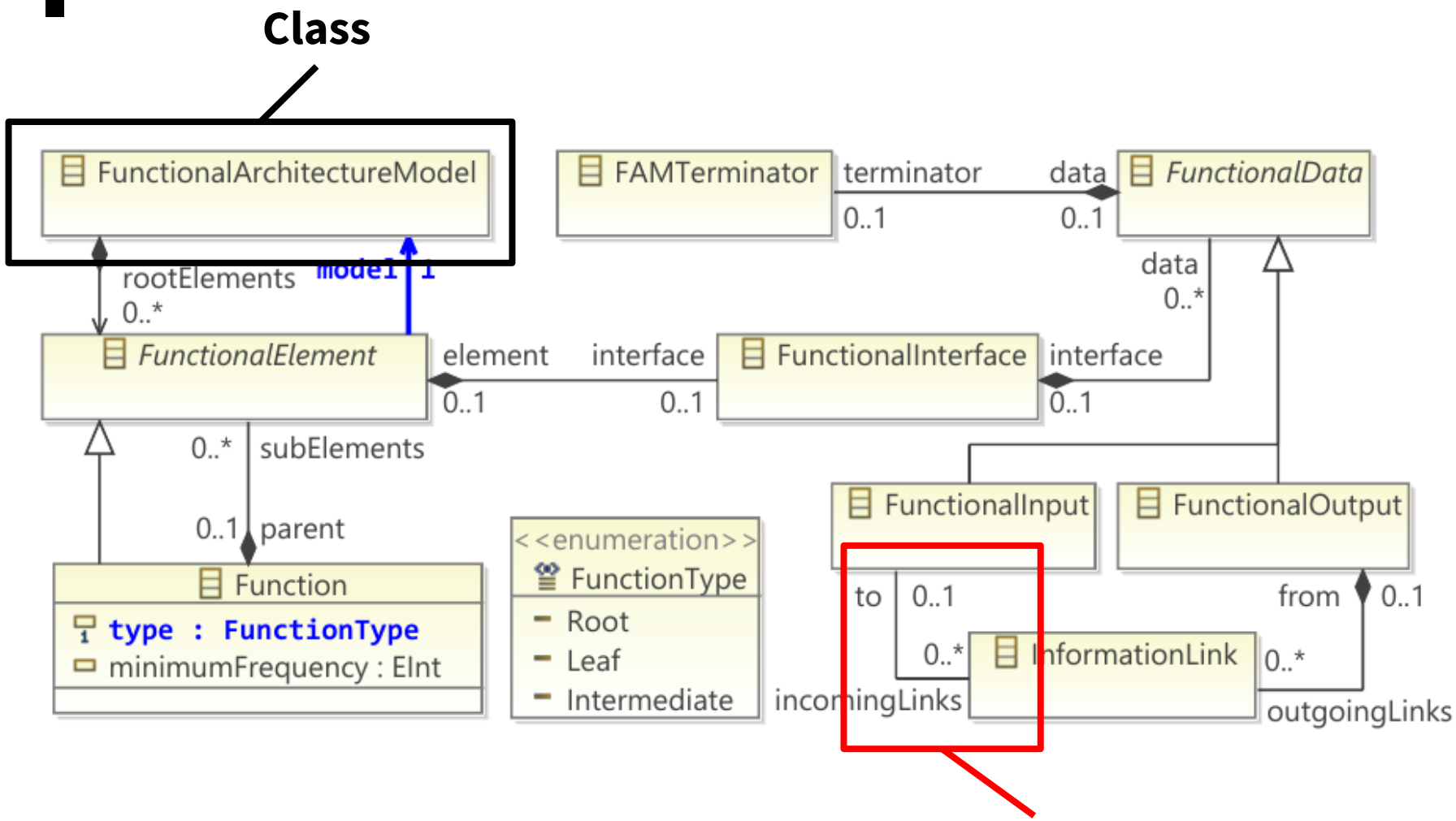
Formalisation of the Metamodel



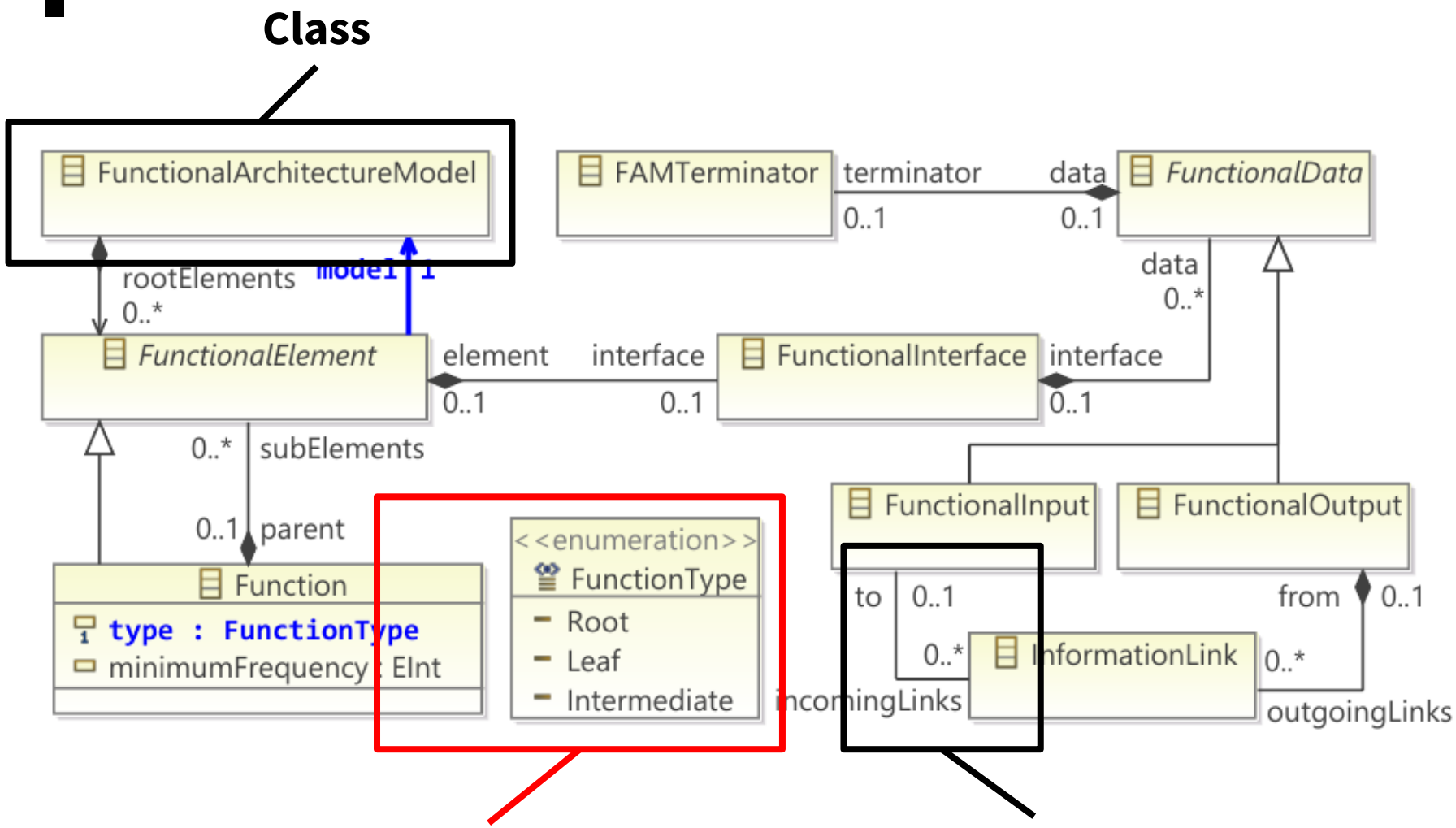
Formalisation of the Metamodel



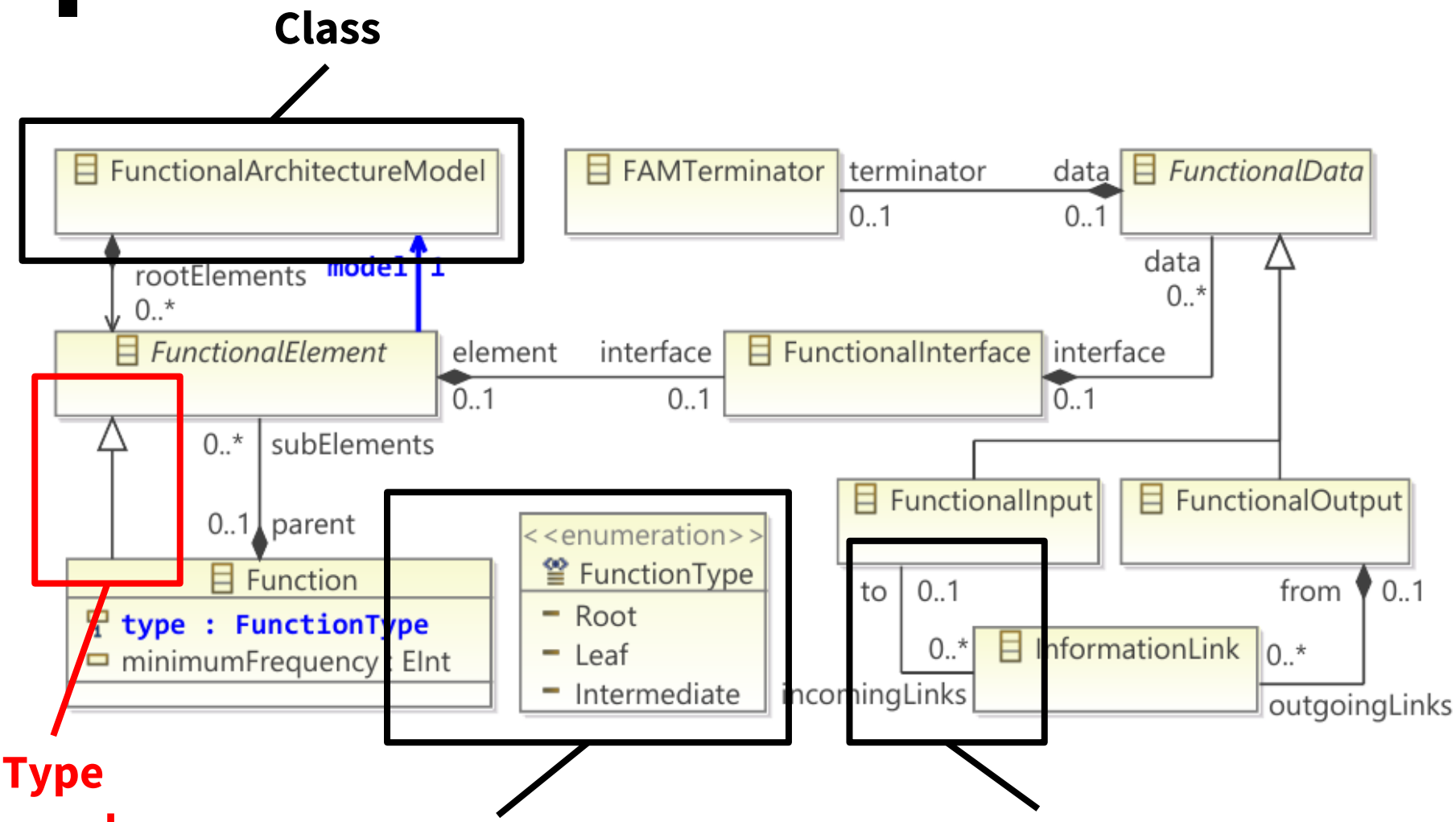
Formalisation of the Metamodel



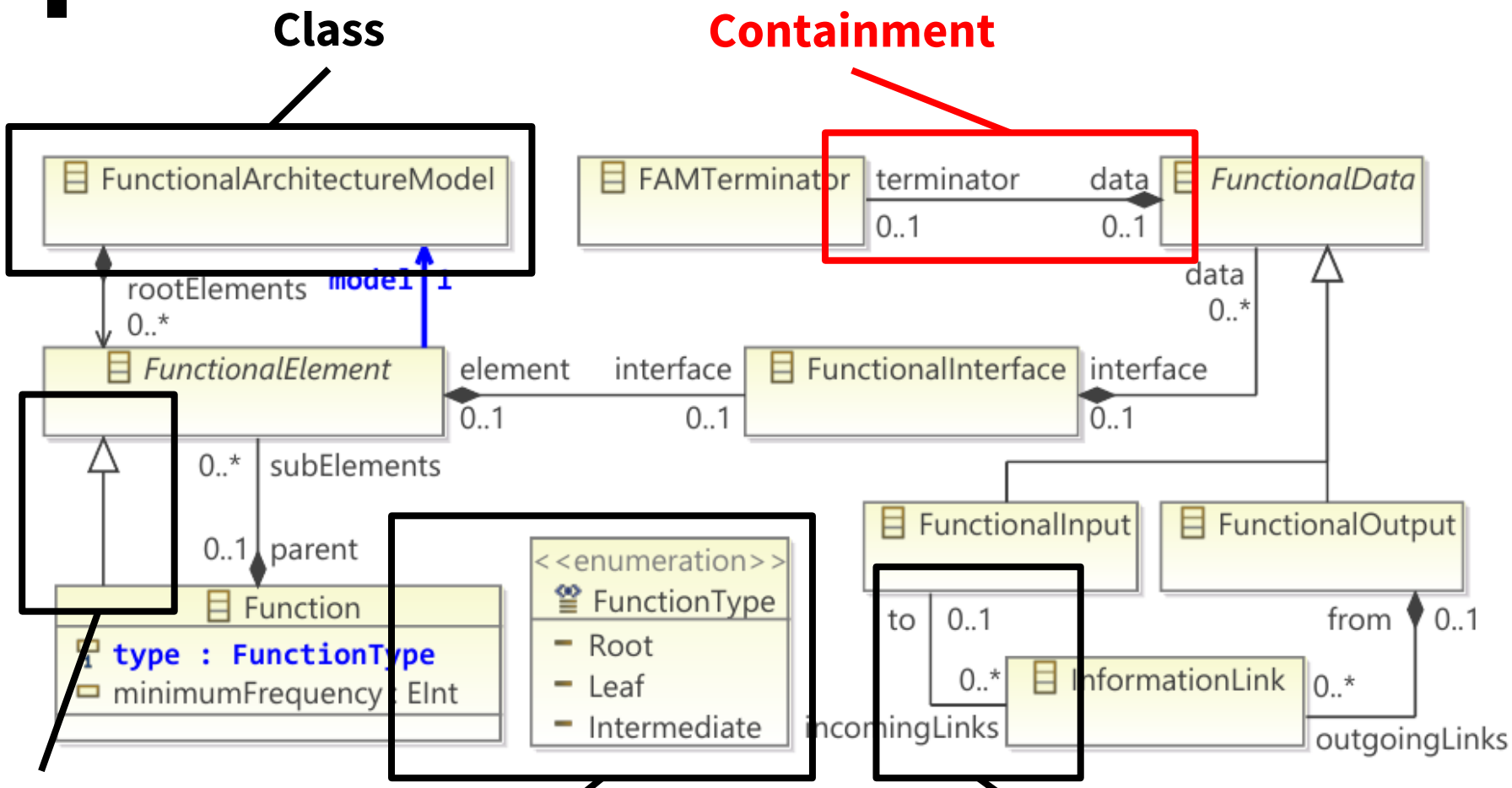
Formalisation of the Metamodel



Formalisation of the Metamodel

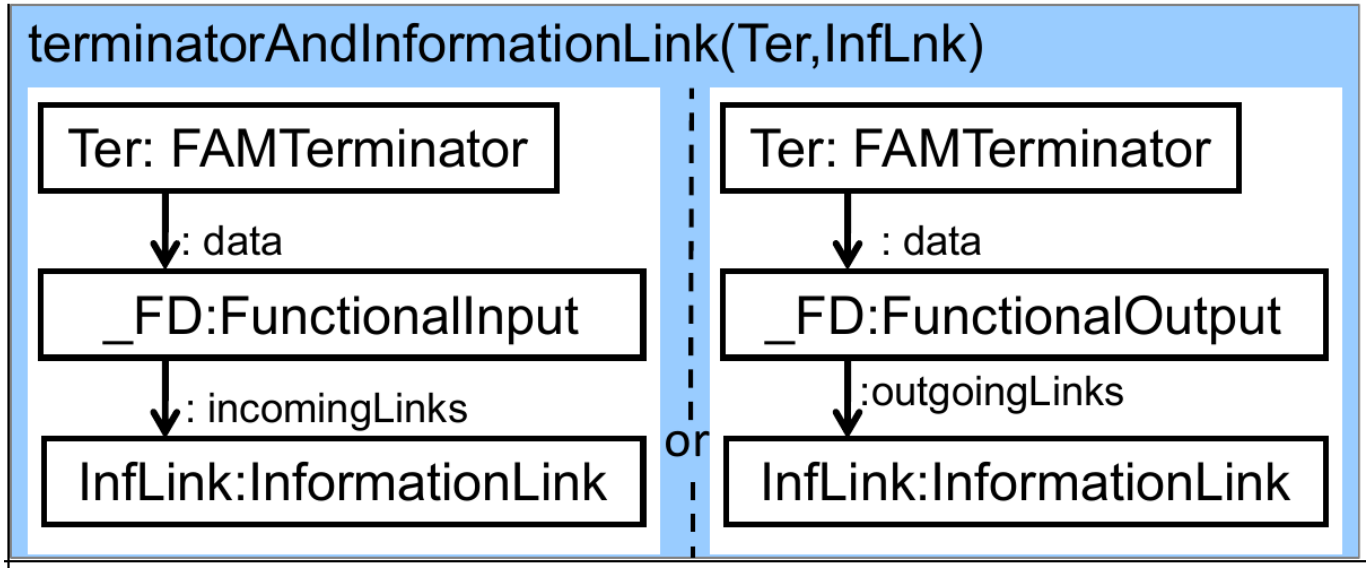


Formalisation of the Metamodel



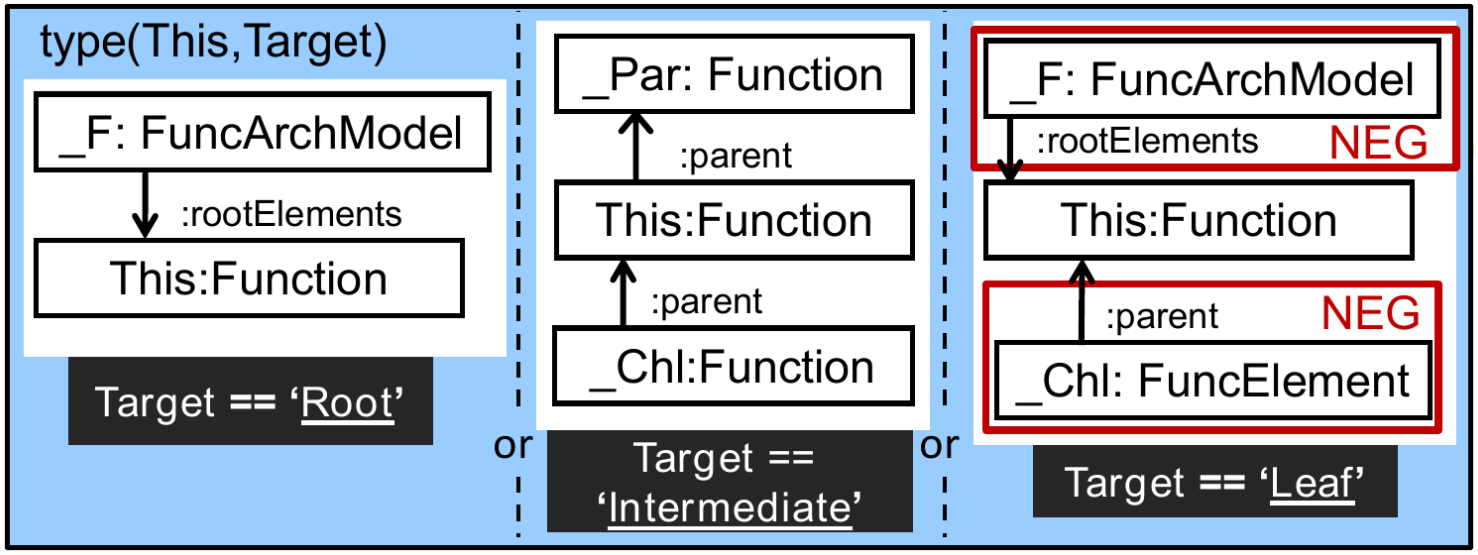
Formalisation of Constraints

- **Supported Constraints**
  - Classifier
  - Path
  - Equality
  - Pattern call
  - Check



Formalisation of Constraints

- **Supported Constraints**
  - Classifier
  - Path
  - Equality
  - Pattern call
  - Check
- **Usage**
  - Well-formed constraints
  - Derived patterns



Partial Snapshots

- **Relaxed Constraints**
  - Undefined attributes
  - Abstract objects
  - Unconnected partitions
  - Missing/extra edges
  - Removed objects

r1: Function
type = ::Root minimumFrequency = ?

r2: Function
type = ::Root minimumFrequency = ?

i1: Function
type = ::Intermediate minimumFrequency = ?

i2: Function
type = ::Intermediate minimumFrequency = ?

l1: Function
type = ::Leaf minimumFrequency = ?

l2: Function
type = ::Leaf minimumFrequency = ?

Partial Snapshots

- **Relaxed Constraints**
  - Undefined attributes
  - Abstract objects
  - Unconnected partitions
  - Missing/extra edges
  - Removed objects
- **Semantic Modifiers**
  - Positive/Negative
  - Injective/Shareable
  - Modifiable/Unmodifiable

r1: Function
type = ::Root minimumFrequency = ?

r2: Function
type = ::Root minimumFrequency = ?

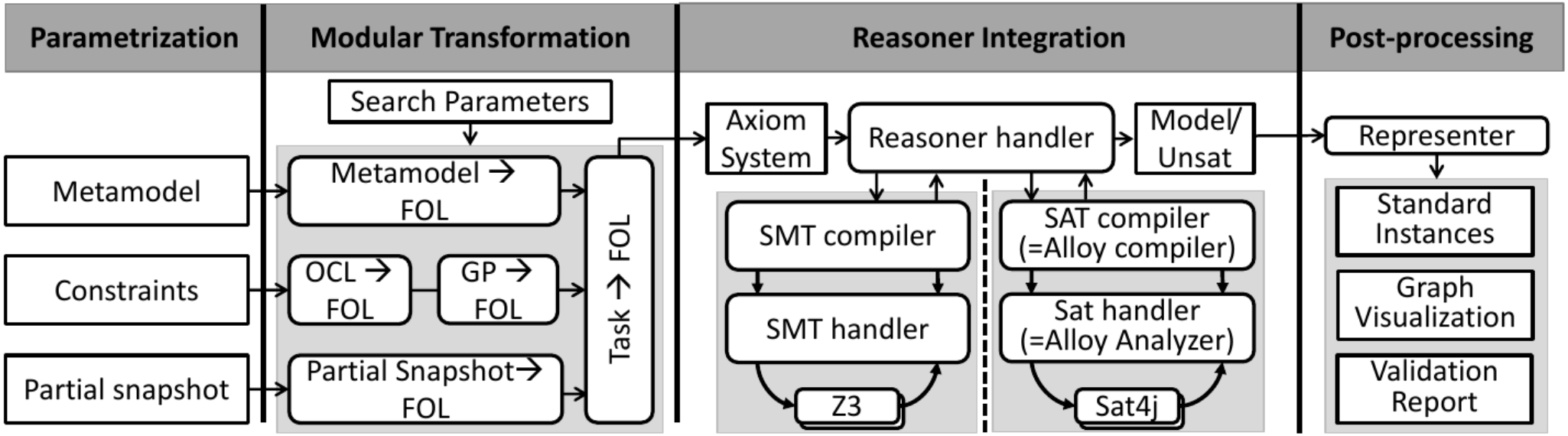
i1: Function
type = ::Intermediate minimumFrequency = ?

i2: Function
type = ::Intermediate minimumFrequency = ?

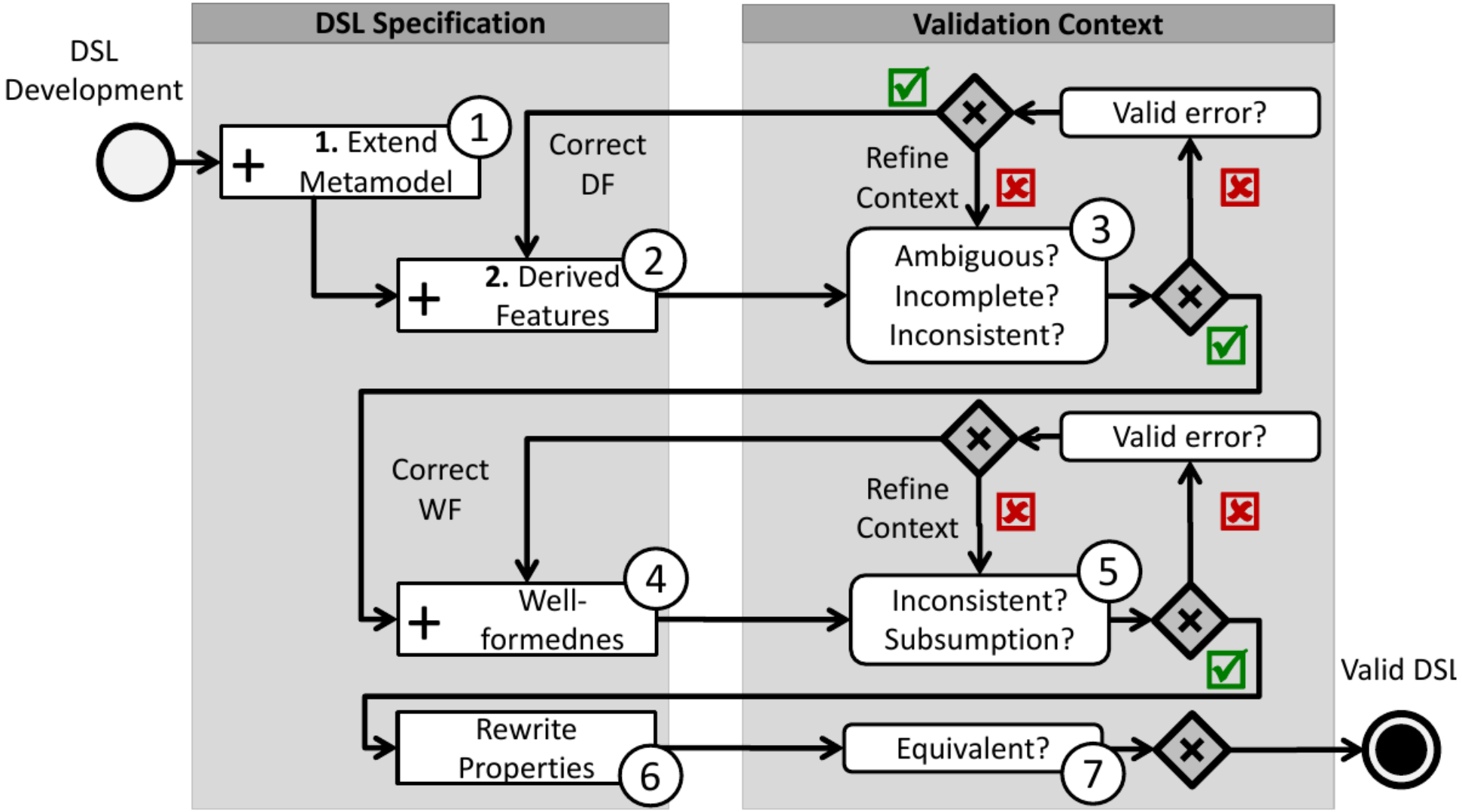
l1: Function
type = ::Leaf minimumFrequency = ?

l2: Function
type = ::Leaf minimumFrequency = ?

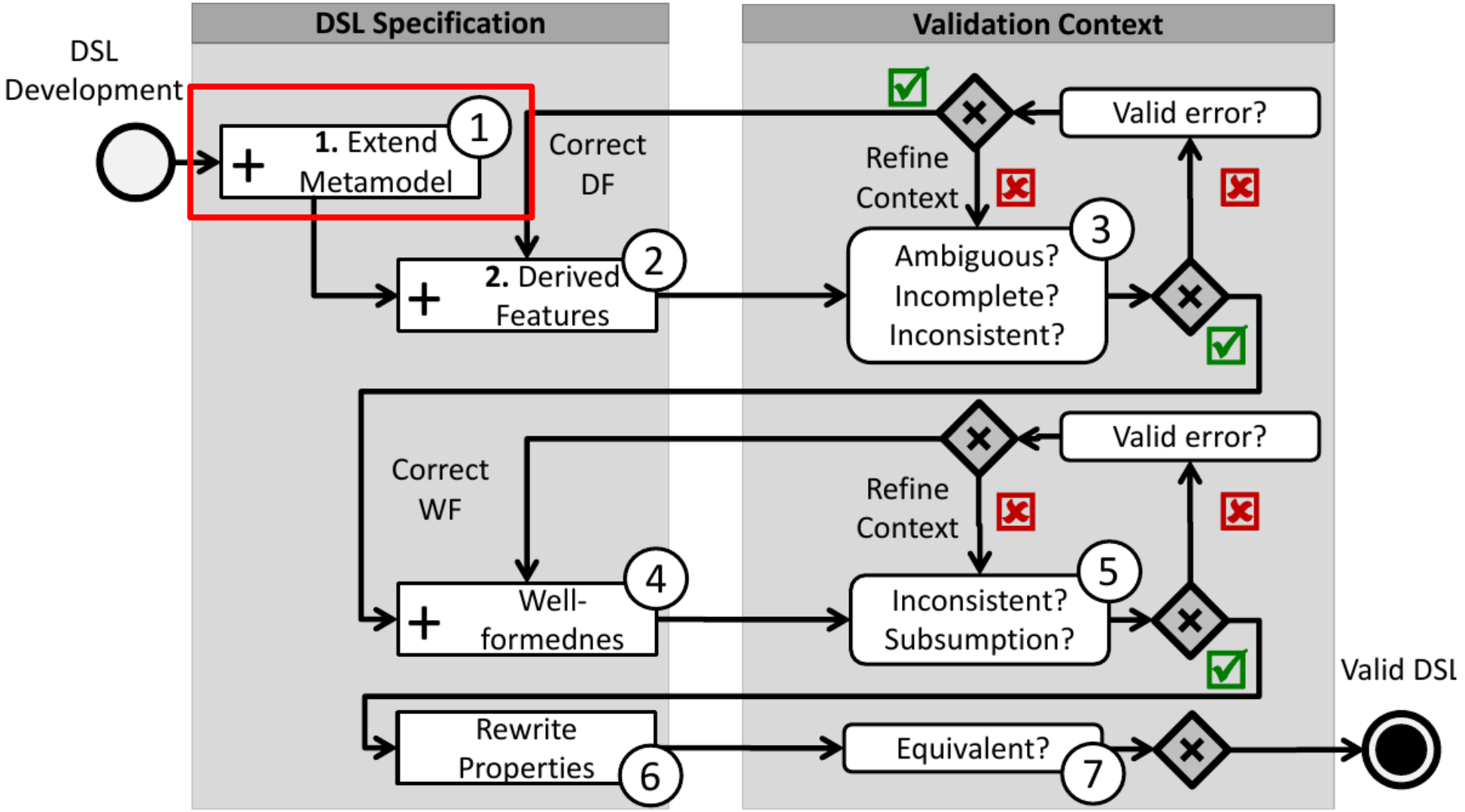
DSL Validation Tool



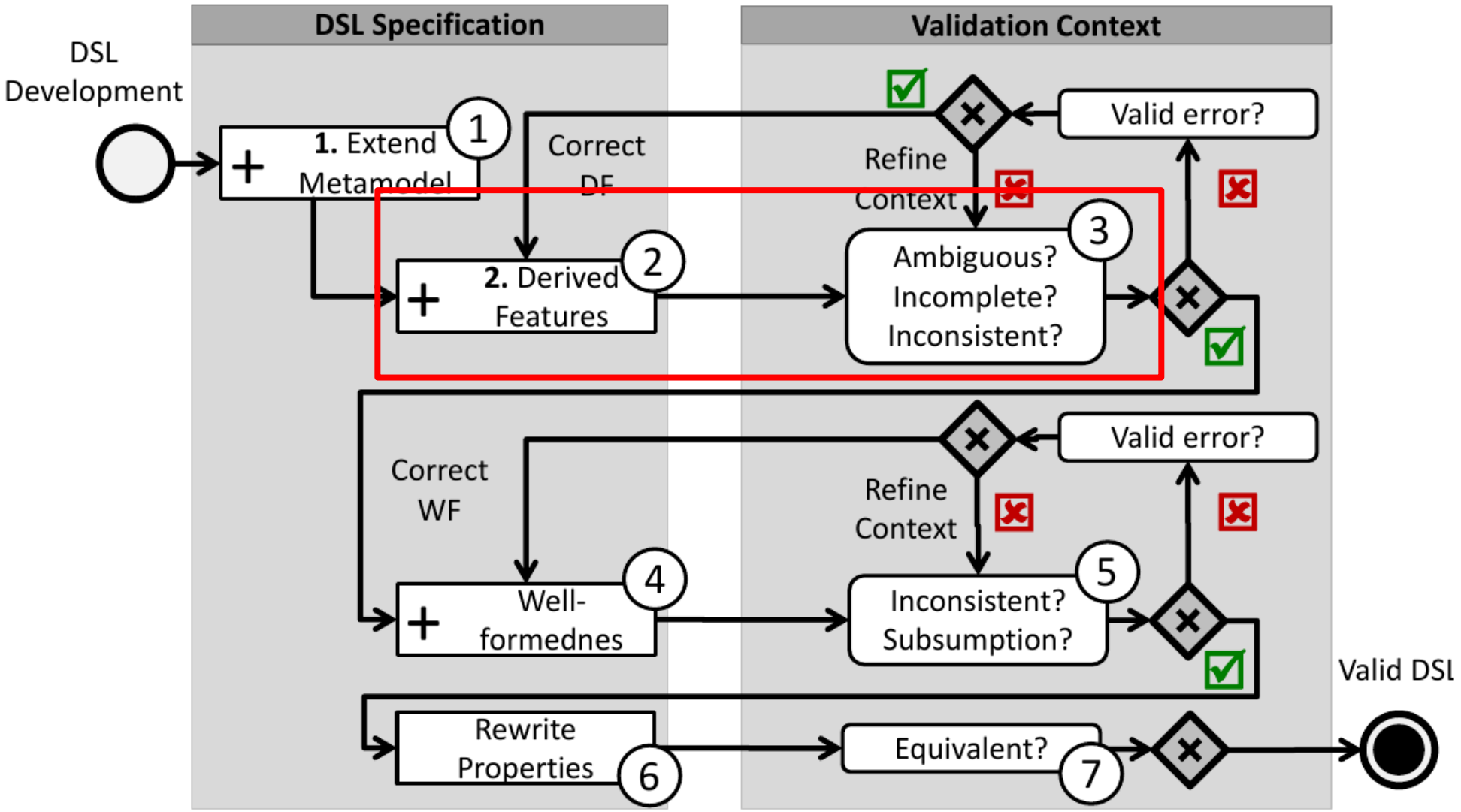
DSL Validation Workflow



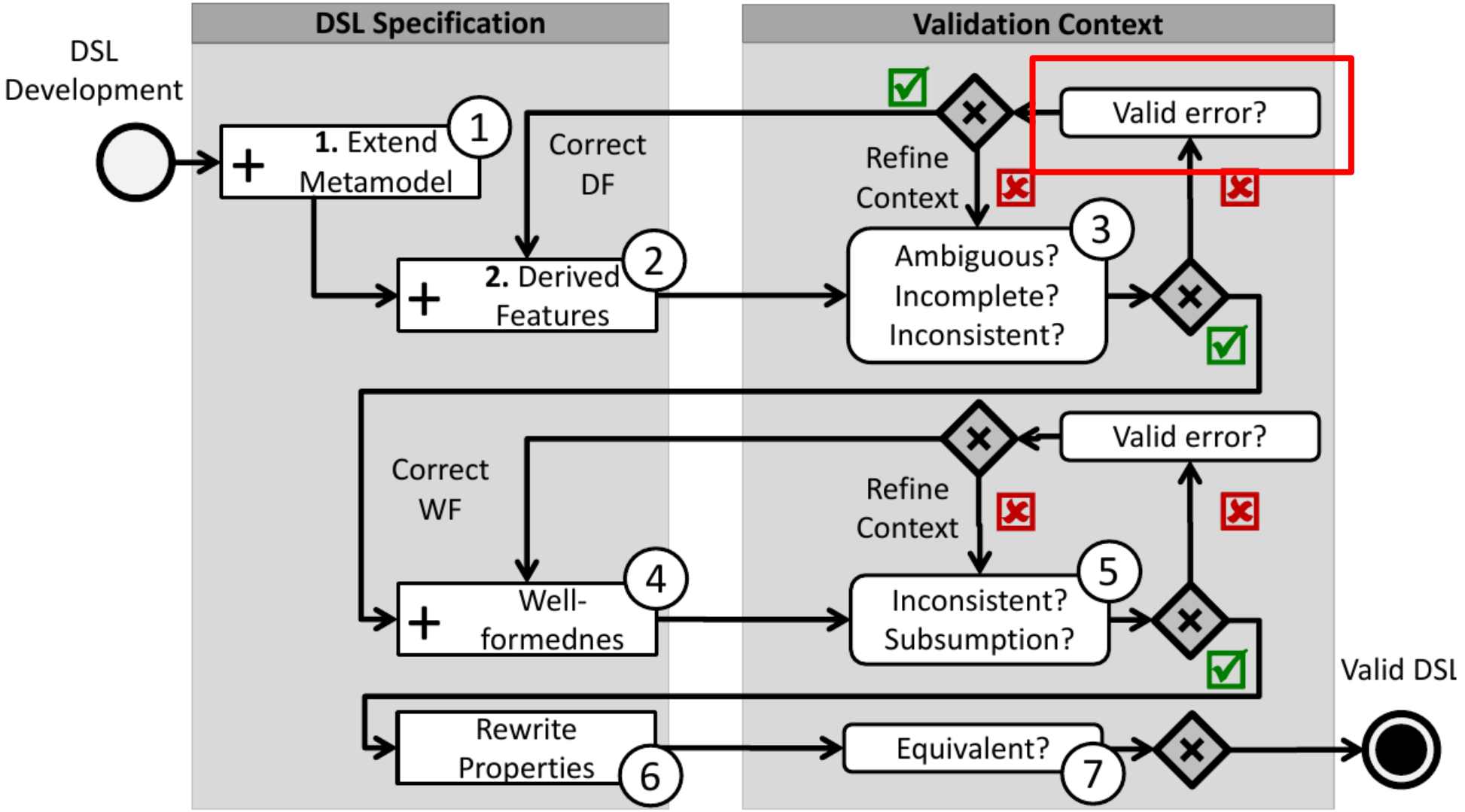
DSL Validation Workflow



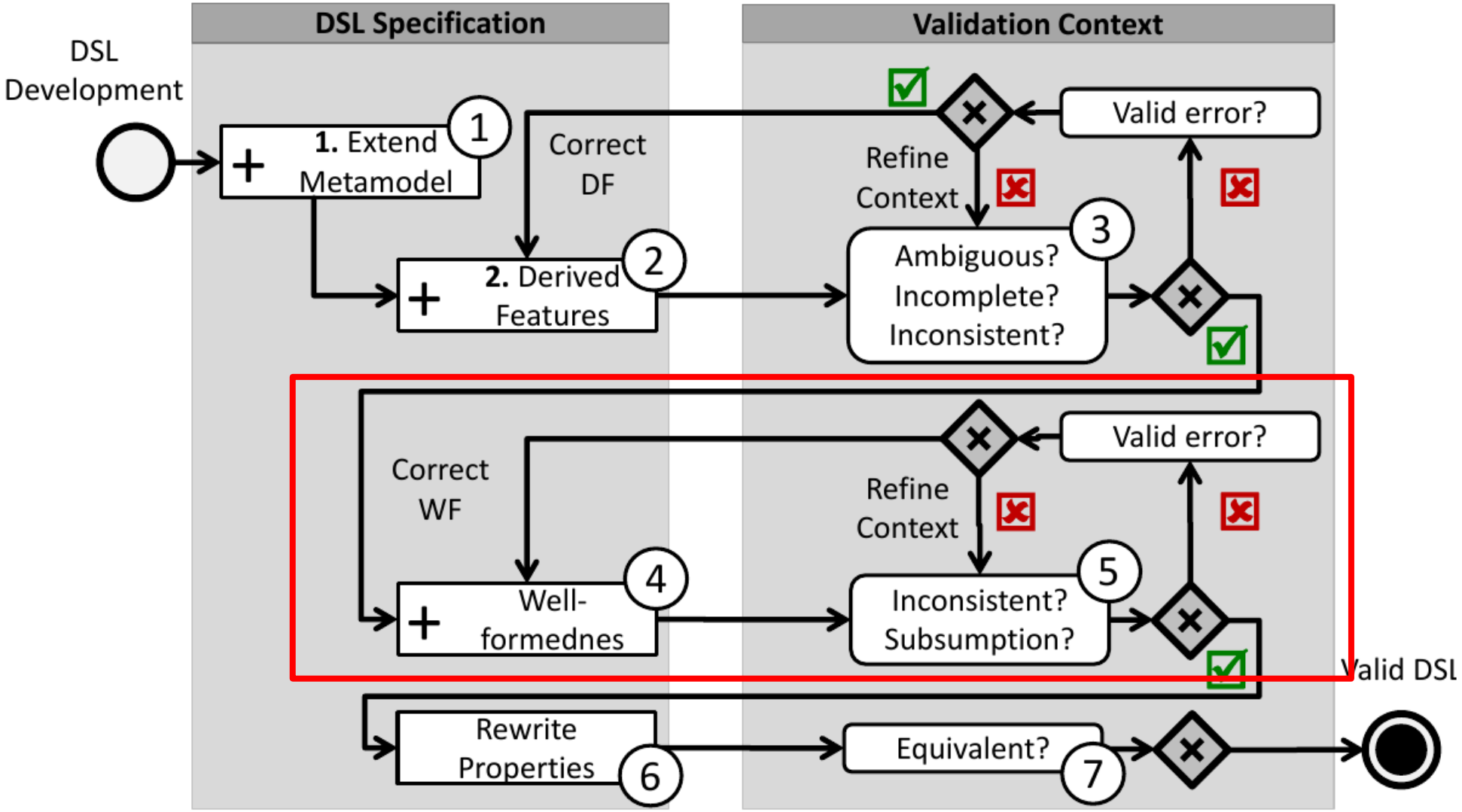
DSL Validation Workflow



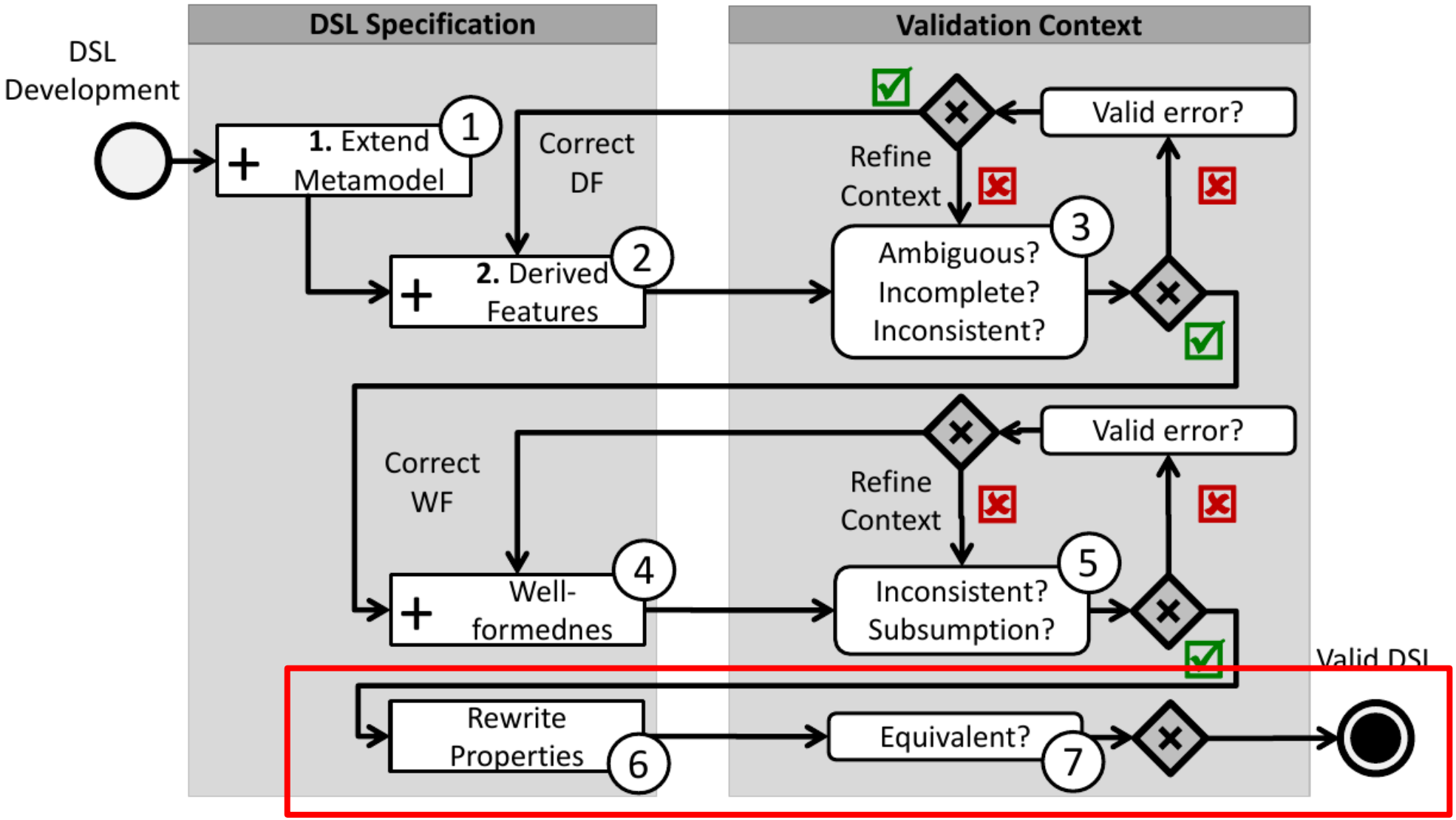
DSL Validation Workflow



DSL Validation Workflow

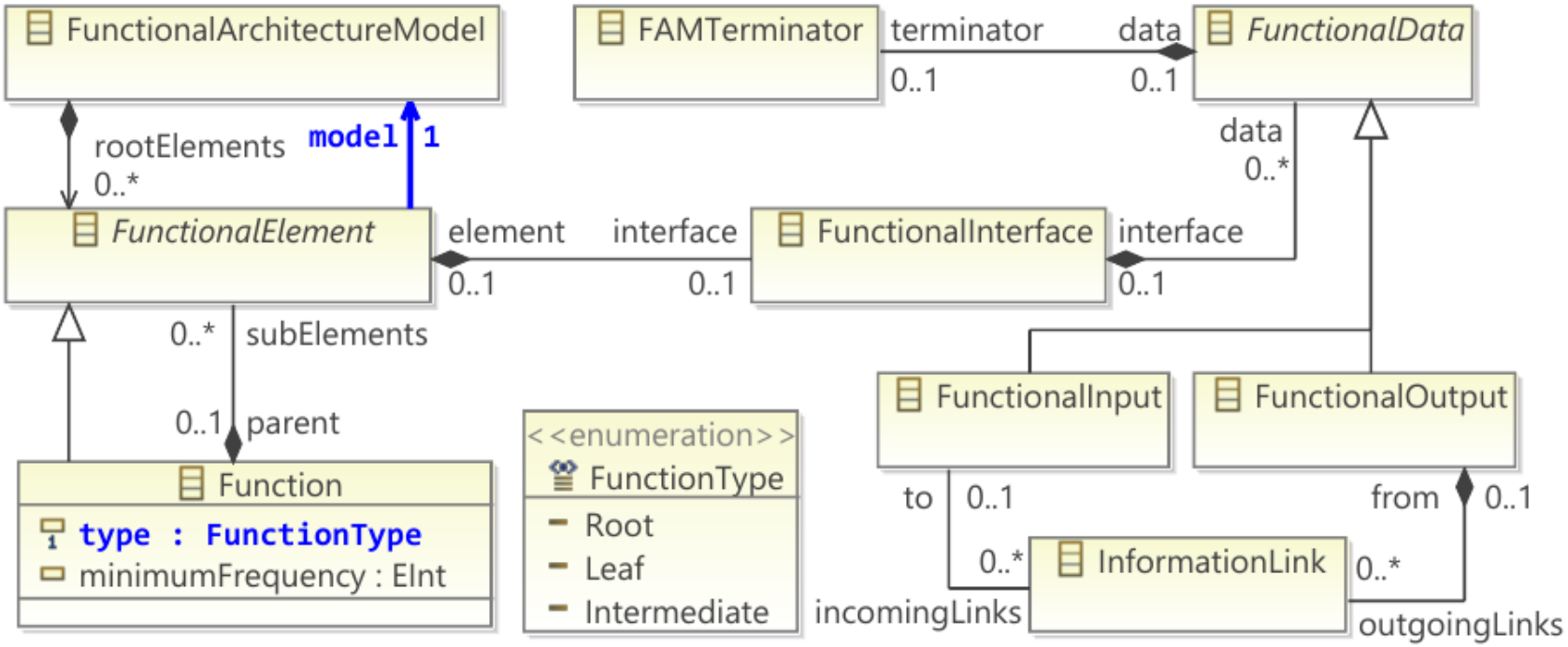


DSL Validation Workflow



Runtime Measurements

- **Preliminary Results**
  - Z3 (SMT) generally outperforms Alloy (SAT)
  - Alloy outperforms Z3 in model structure generation



The End

- **Problem**
  - Validation of complex DSLs
- **Approach**
  - Approximate DSL in first-order logic
  - Check for satisfiability of resulting formula
  - Convert witness/counterexample into model
- **Discussion Points**
  - How well does the approach scale?
  - Is it applicable to validating UML?
  - What are “useful” constraints for a DSL?
  - Is it applicable to synthesising constraints?