Ark: A Kernel For Multi-Paradigm Modeling

MSDL Summer Presentation 2009

Xiaoxi Dong,

MSDL, McGill University

Content

- Intention of A kernel for multi-paradigm modeling
- Ark Overview: hierarchical modeling environment
- Ark breakdown: framework, ArkM3, Himesis, functionality and examples
  - two dimension metamodeling framework
  - Physical implementation using Himesis
  - ArkM3: the meta metamodel
  - Function Module
- Conclusion and future work

Intention

- An system for

Multi-paradigm modeling

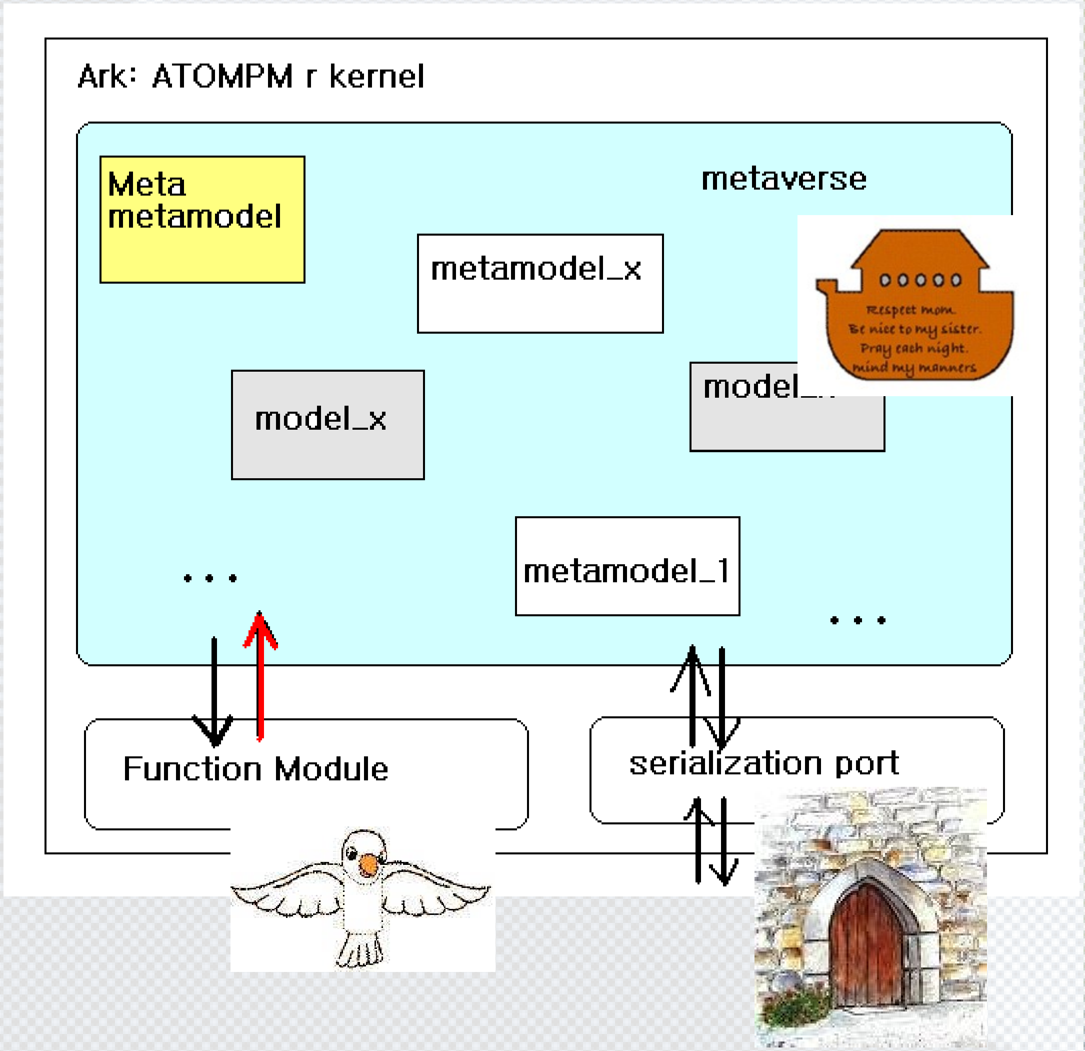
- Including executability into metamodel
  - The predominant metamodeling languages not designed to encode the behavioral information. Designers have to either refer to extensions or resort to programming languages to describe the actions.
  - prevents including complete model information in the model;
  - makes transferring models between tools and transforming models between formalisms non-trivial.

•Ark: AToMPM r... kernel

- Kernel structure overview
- Kernel breakdowns
  - two dimension metamodeling framework
  - Physical implementation using Himesis
  - ArkM3: the meta metamodel
  - Function Module

Kernel Structure Overview

- “Metaverse”: A universe of models and metamodels
- The ability to update the model
- Each model inside metaverse is accessed by unique path

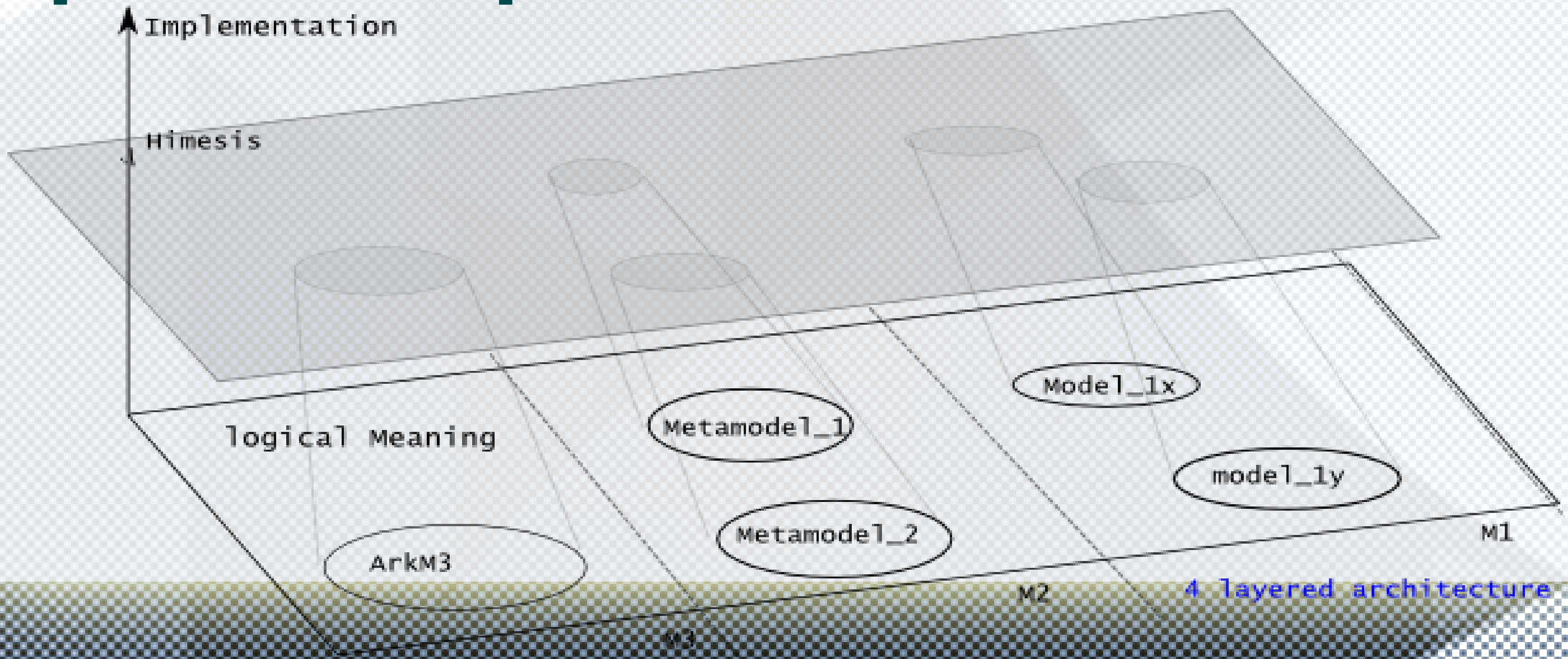


Kernel breakdowns

- Two dimension of the A self-sufficient, strict metamodeling framework
- Physical implementation using Himesis
- ArkM3: Meta metamodel
  - class diagram of ArkM3
  - mapping from ArkM3 to Himesis
- Function Module
  - create elements according to metamodel
  - automatic checking of model consistency and constraints conformance
  - interpreting the action model

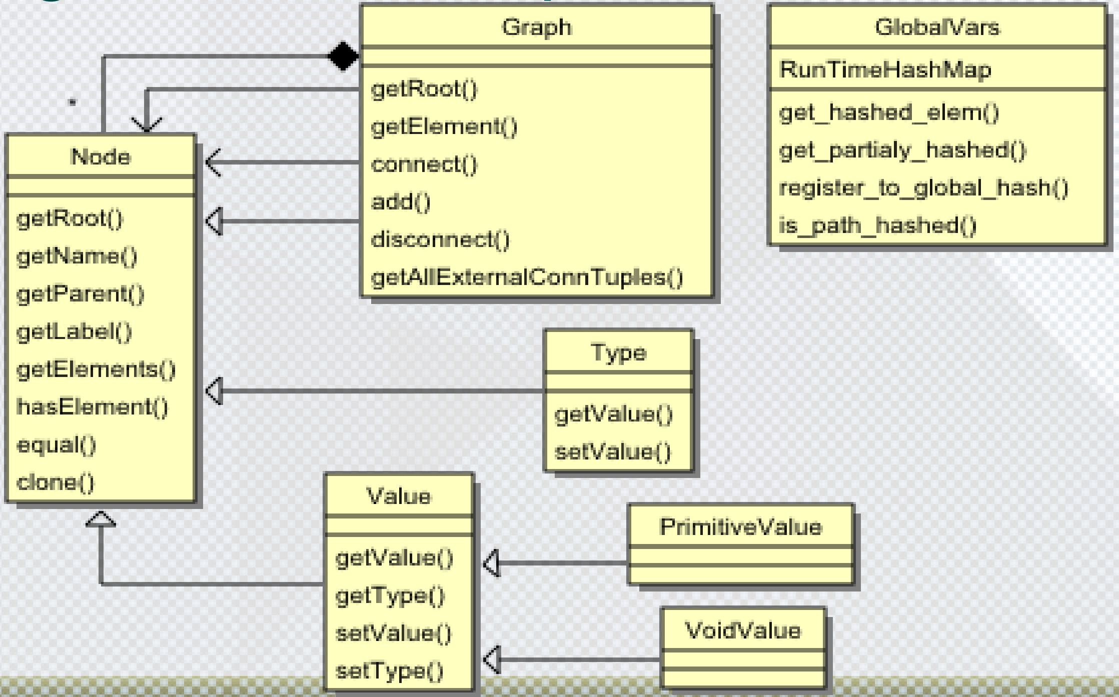
ArkM3 metamodeling framework

- Modified from MOF[OMG MOF2.0]
- Strict metamodeling
- self sufficient and closed system
- A two dimension metamodeling framework [Atkins&Kunhe]



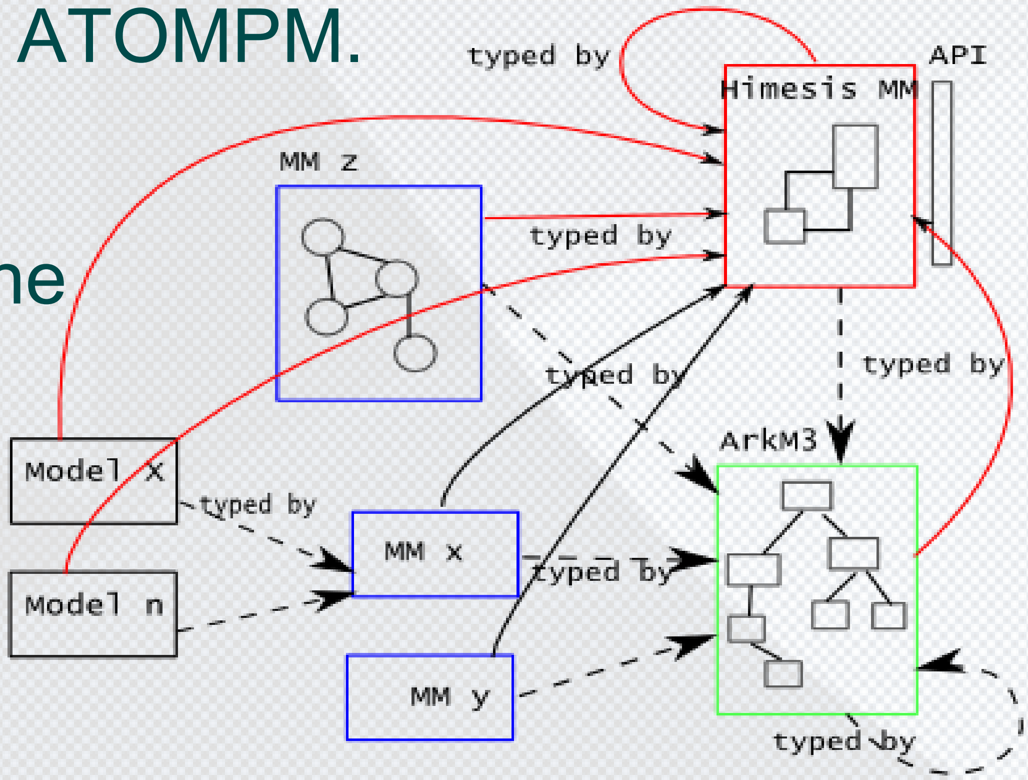
Implementation of models: Himesis

- Himesis serves as the basic structure of all the models defined in AToMPM. It is modified from Himesis by Marc Provost. [Hm]
- Modified classes as in the figure
- Added global hash map for faster traversing



Implementation of models: Himesis

- Himesis is also typed by ArkM3.
- It serves as the basic structure of all the models in the world of ATOMPM.
- In other word, it is the metamodel of all the models in ATOMPM.



ArkM3: AToMPM r kernel Meta metamodel

- It is modified from EMOF so that it has the definitions needed for OO design.
- It is a self-sufficient metamodel.
- It is an hierarchical model that support packages.
- Consider every object is an Element and that an element can have constraints and actions
- It has Action Language model.
- It is a constraint metamodel, representing an unambiguous metamodeling language.
- Some model elements are reusable, such as Action Language, DataType and DataValue

Mapping between Himesis and Instances of ArkM3

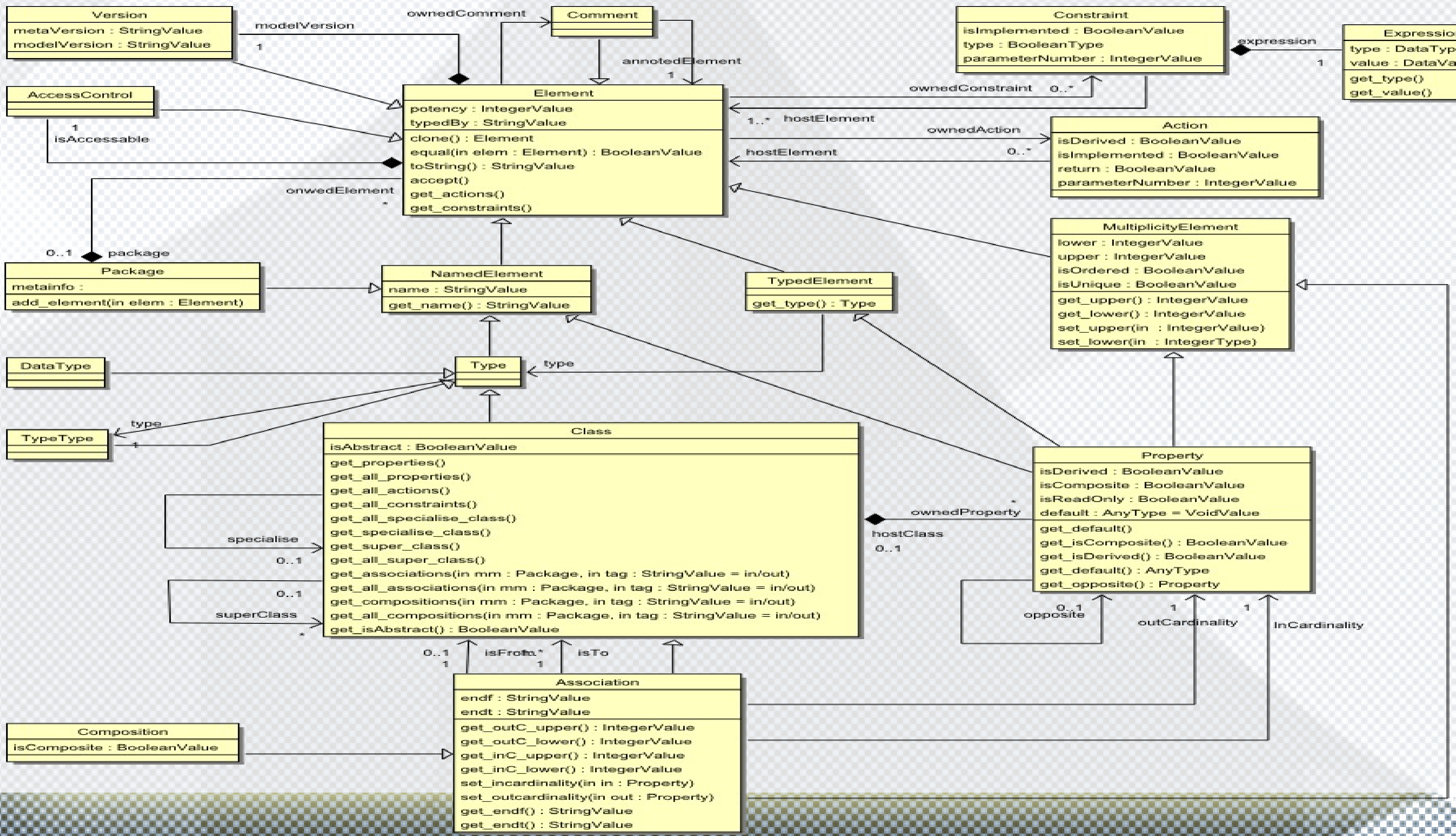
- We need clear define the mapping from ArkM3 models to Himesis in order to,
  - develop formalism specific function modules.
  - accurately transform and transfer models
  - details please refer to the Ark manual.
- To be mentioned later.

ArkM3 Pakcages

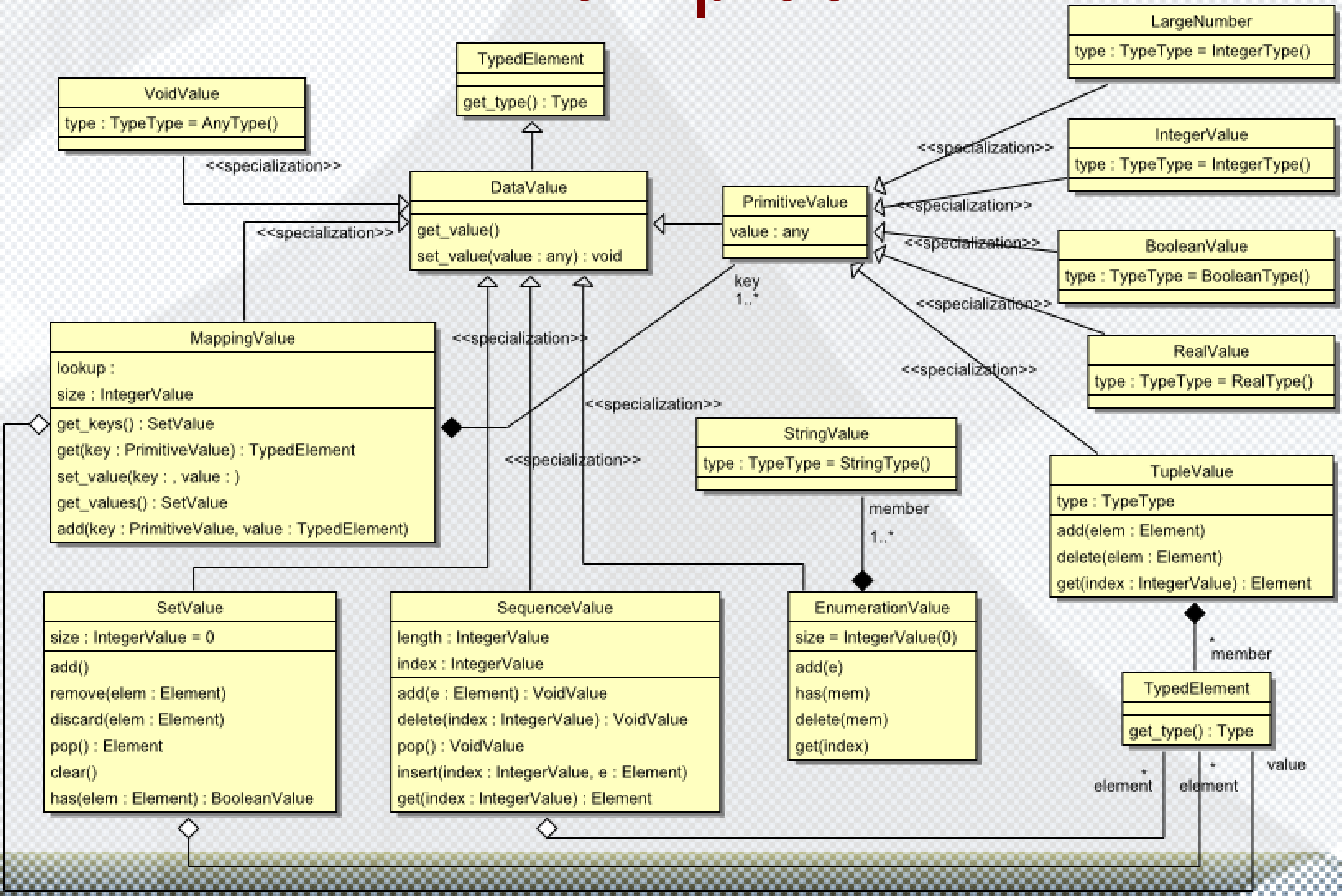
- Packages:
  - ArkM3
    - DataValue (reusable)
    - DataType (reusable)
    - ActionLanguage (reusable)
      - Literal
      - Operator
- Next: Class diagrams of ArkM3 and Examples

•ArkM3 Class Diagrams

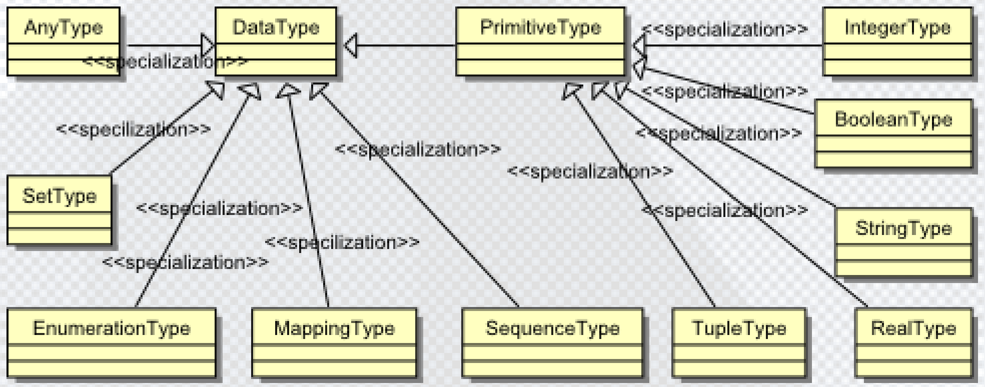
- class diagram



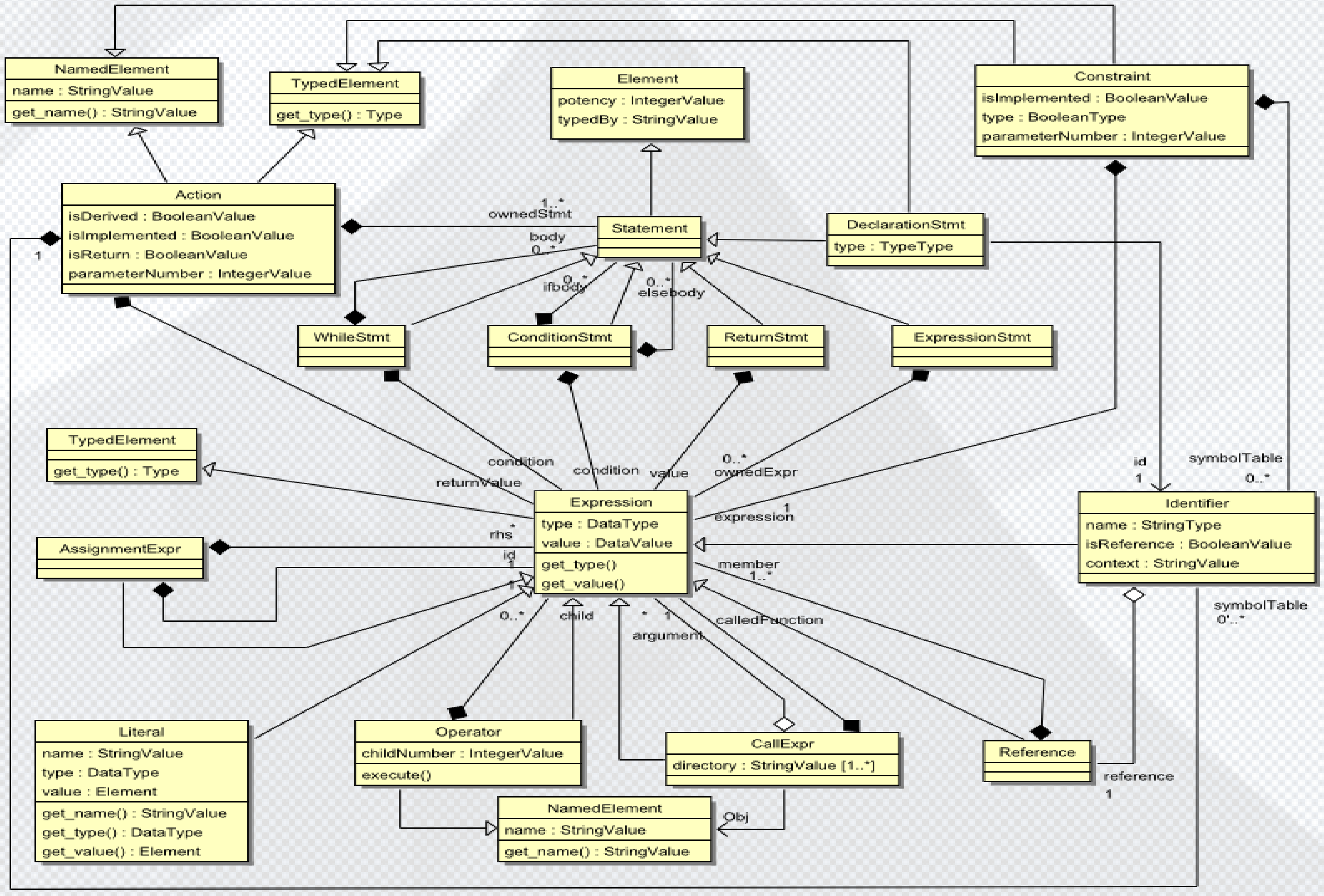
•ArkM3 Class Diagrams and Examples



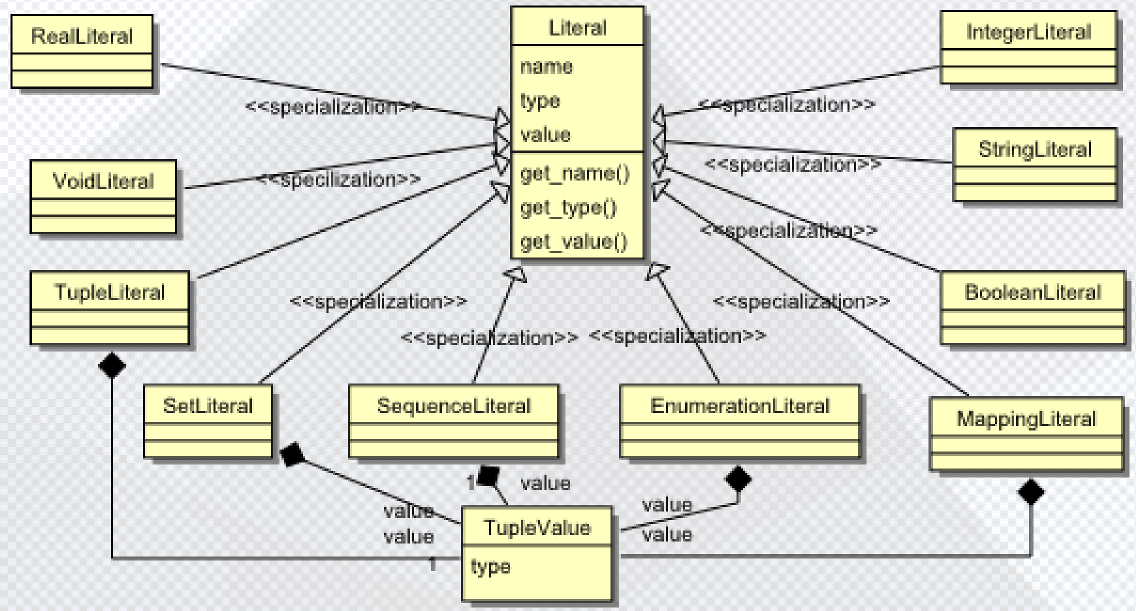
- ArkM3 Class Diagrams and Examples



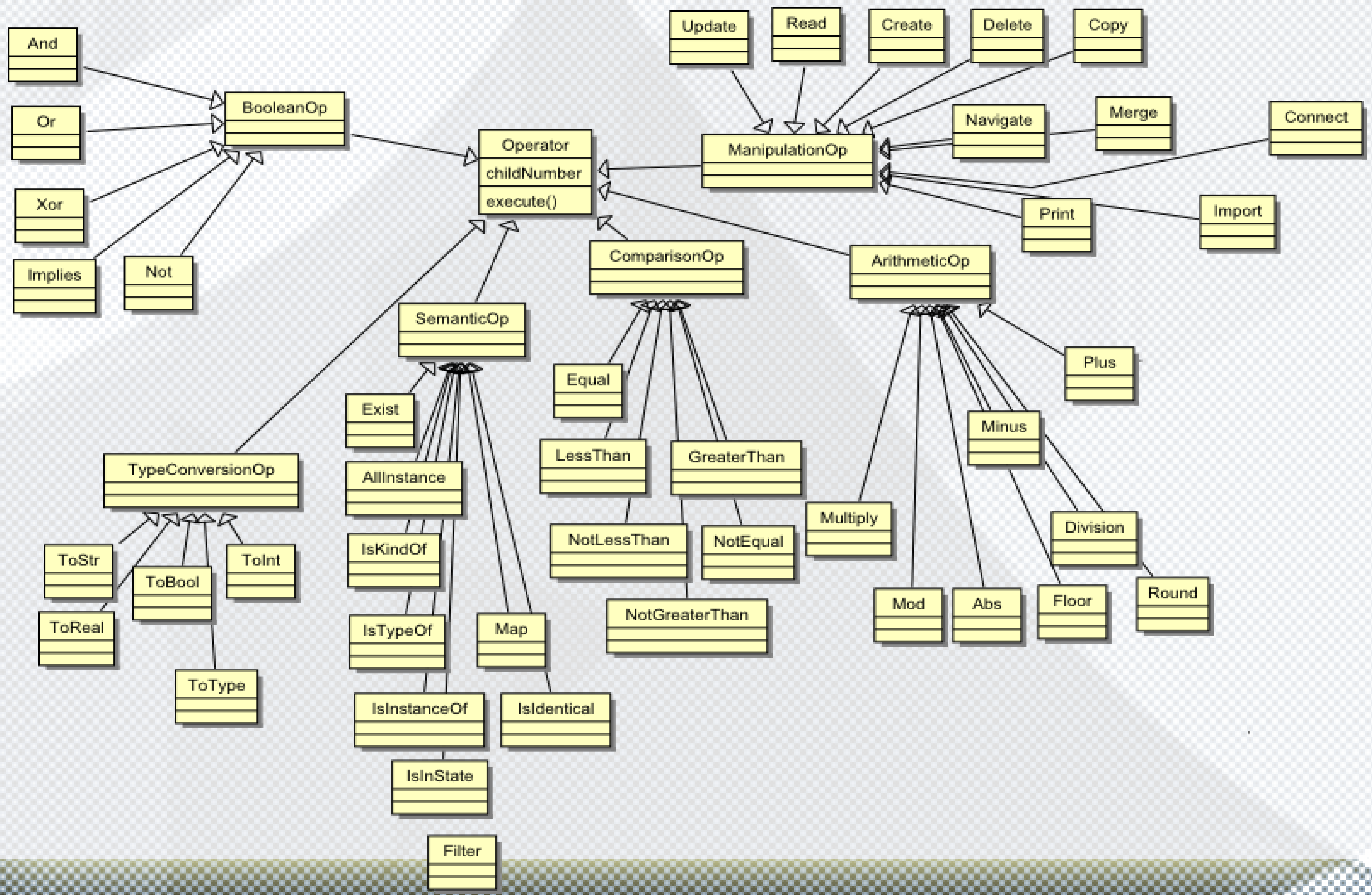
- ArkM3 Class Diagrams



•ArkM3 Class Diagrams



•ArkM3 Class Diagrams



Ark Function Module

- The kernel provides some basic functions for metamodeling and transformation
  - C of CRUD creating/instantiating: in detail
  - RUD of CRUD is primitives action language
  - conformance checking
  - constraint checking
  - Serialization
  - action model interpreting: in detail

How to: sketch and example

- creating
- checking
- action model interpreting

Use the example that metamodel is instance of M3 which has packages, classes, association, composition and etc.

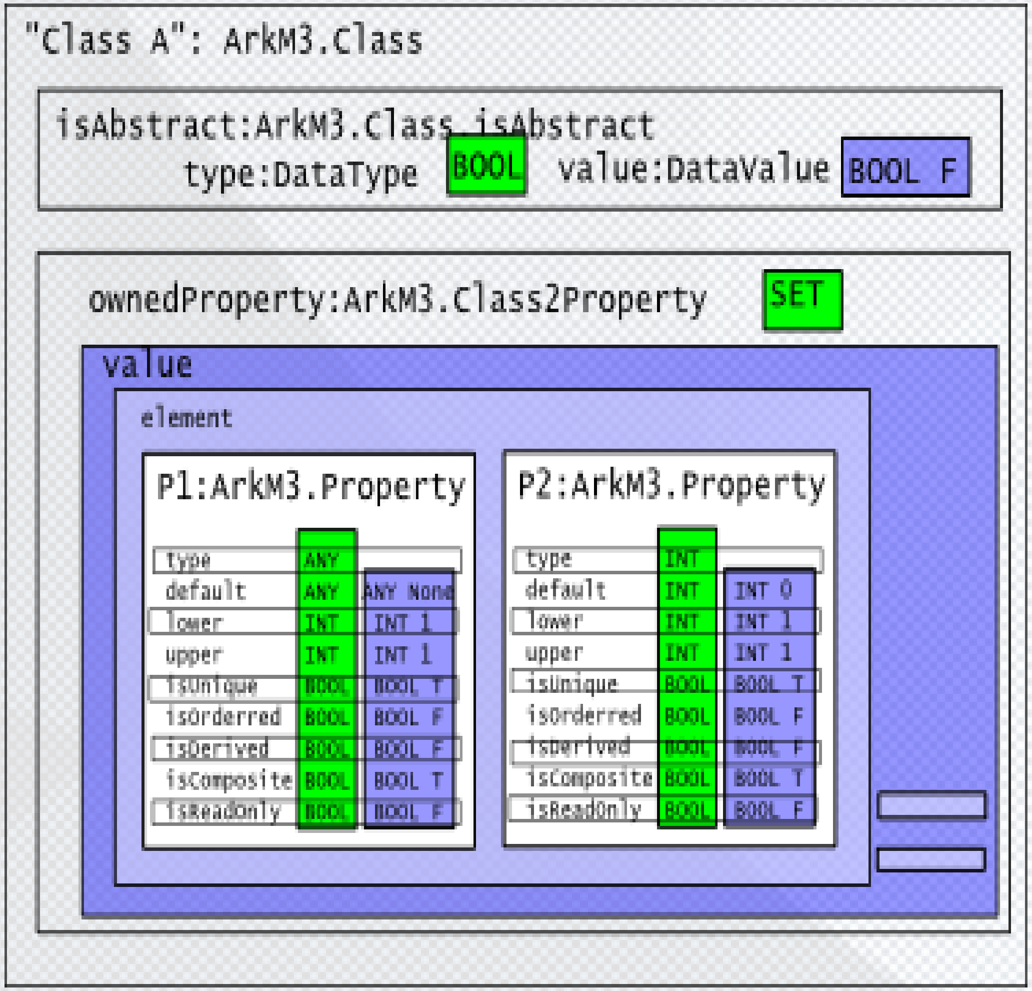
Instantiating

- Retrieve the class definition from the metamodel using unique path
- Create an object according to the definition: Flatten model
  - Traverse the metamodel elements and get the list that contains super classes of this class.
  - Traverse the properties defined in the listed classes and create objects accordingly.
  - Traverse the associations and compositions connecting the listed classes and create nodes accordingly.
- Overwrite default value of the attributes if customized value exist.

•ArkM3 Class Diagrams and Examples

•

```
C1 = factory.createClass(
  "Class A", isAbstract = False)

C1.addProperty(createProperty(
  "P1", lower=1, upper=1))

C1.addProperty(createProperty(
  "P2", lower=1, upper=1,
  type=Int, default=0))
```



Action Interpreting

- Retrieve the required action model
- Execute this model
  - load the value of the parameters into the symbol table
  - traverse the model and interpret the action
  - return the result of the execution if applicable.

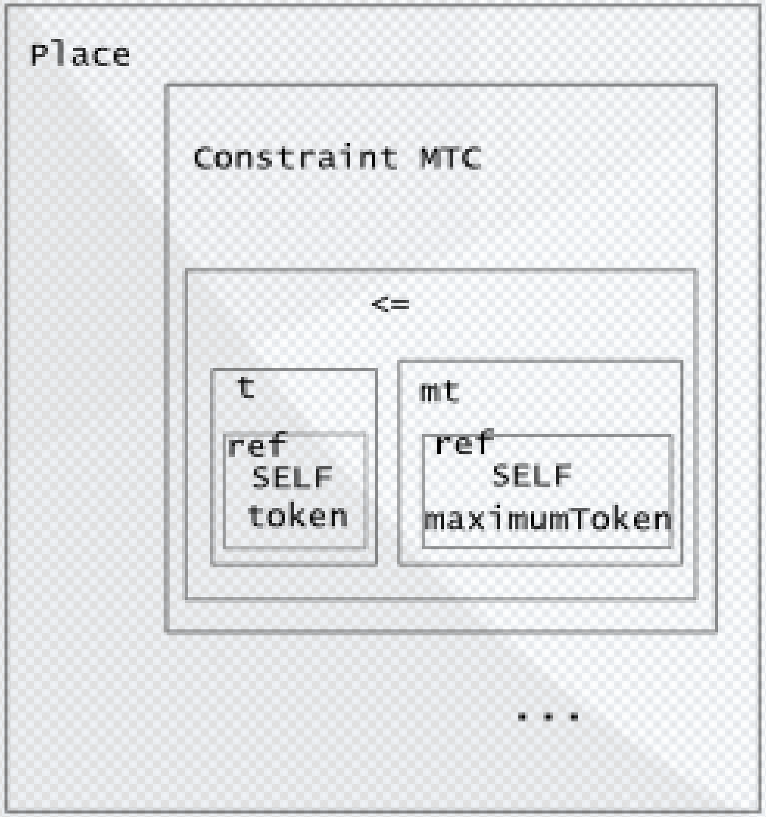
•ArkM3 Class Diagrams and Examples

•

```
    const1 =
factory.createConstraint("maximumToken
Constraint",
root=place["ownedConstraint.value.elem
ent"], condition="CONDITION",
isImplemented = True, hostElement =
[place])

    lhs =
factory.createIdentifier("t", isRef =
True, reference =
factory.createReference(id="ref",
ref=["SELF","token"],
meta="ArkM3.AL.IdentifierReference"))

    rhs =
factory.createIdentifier("mt", isRef =
True, reference =
factory.createReference(id="ref",
ref=["SELF","maximumToken"],
meta="ArkM3.AL.IdentifierReference"))

    factory.createNotGreaterThan(root
= const1["expression"], child =
[lhs,rhs])
```

Constraint `maximumTokenConstraint`

token<=maximumToken



Reference

- Please refer to http://msdl.cs.mcgill.ca/people/xiaoxi/14_literature

Summary

- Intention of a unified, self-sufficient and executable metamodeling and transfor-mation tool
- Ark: A kernel for multi-paradigm modeling
  - Overview: hierarchical modeling environment
  - Kernel breakdown: framework, ArkM3, Himesis, functionality and examples
- Demo
- Reference
- Conclusion and future work

Conclusion and Future Work

- Performance
- Serialization
- Primitives
- Save the world with Ark!