



Fraunhofer Institute for Open Communication Systems | Kaiserin-Augusta-Allee 31 | 10589 Berlin, Germany



Generation of Formal Model Metrics for MOF based Domain Specific Models

Marcus Engelhardt, Christian Hein, Tom Ritter, Michael Wagner

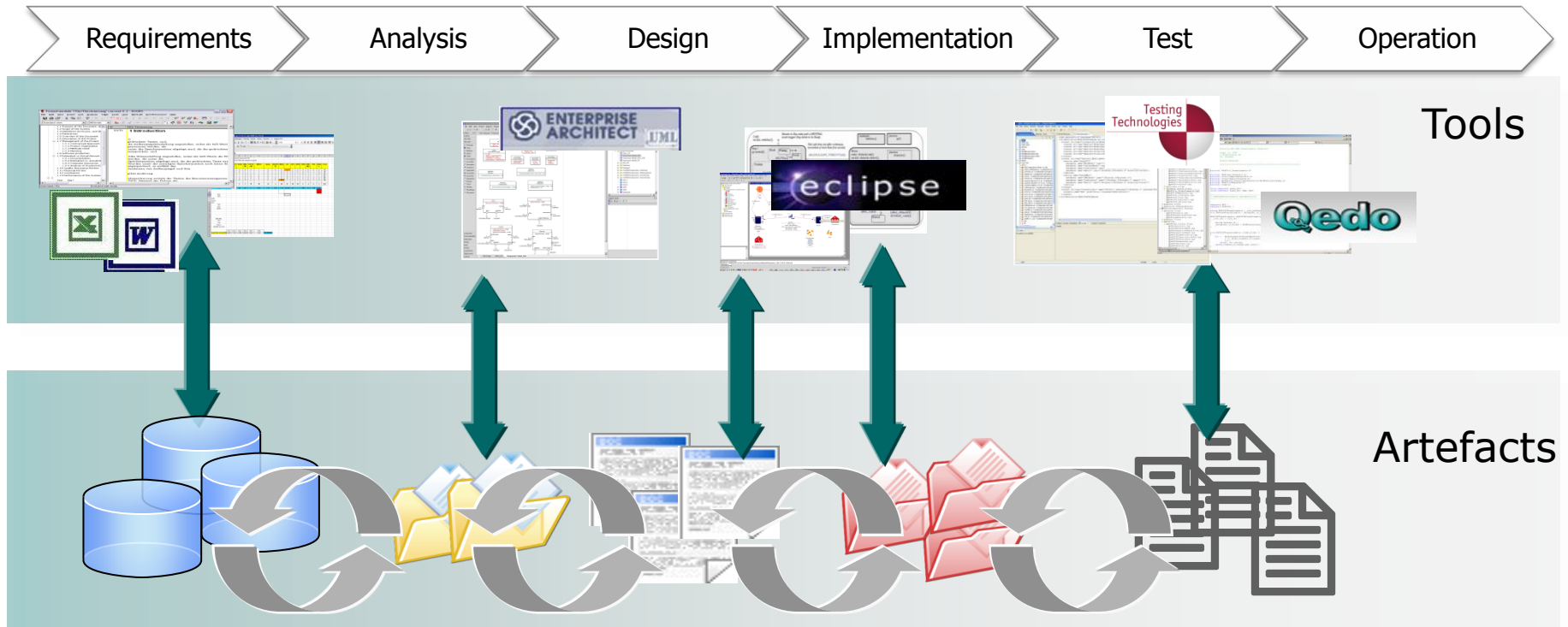
Michael Wagner | OCL Workshop | October 2009 | Denver |



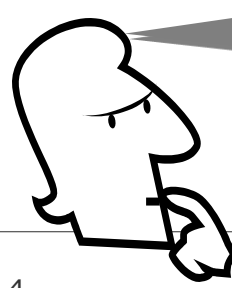
Outline

- Introduction
- Automatic Generation of Metrics
 - Meta Model for Metric Definition
 - description of rules by means of an example
- Tool
 - Architecture
 - Demo Video

Introduction



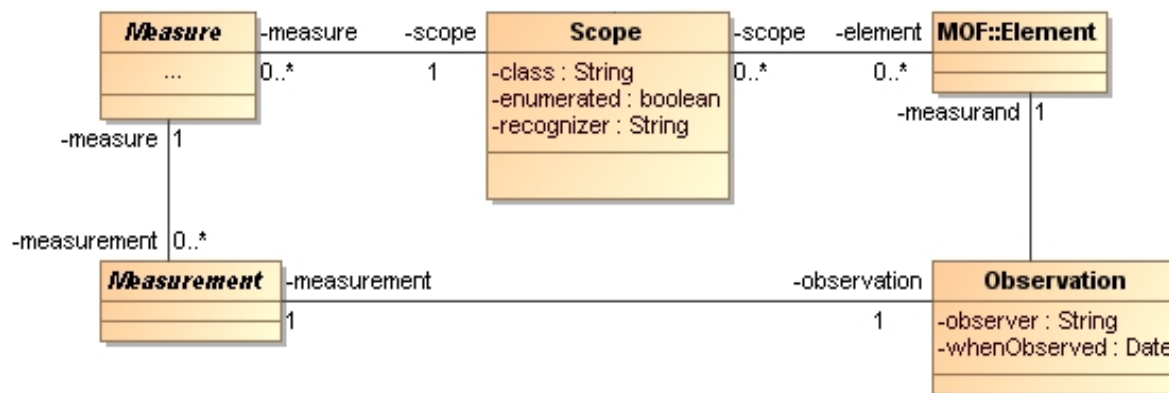
- Inconsistency, Low Degree of Automation, Insufficient Common Terminology
- Complexity, Costly
- Decoupled software tools
- Produced data remains proprietary and depends on specific tools



Automatic Generation of Metrics

Meta Model for Metric Definition

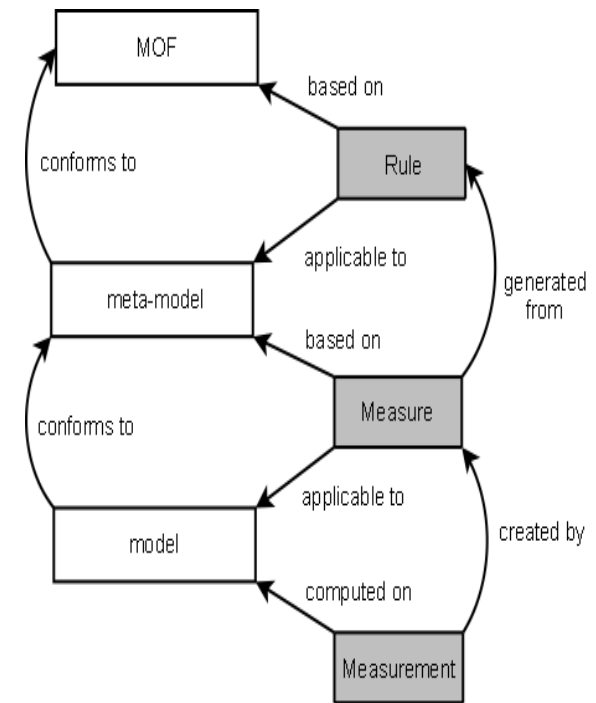
- Usage of Software Metric Meta-Model (SMM) for the definition of metrics and their computational results provided by OMG
 - SMM distinguishes between **measures** as the evaluation process of particular quality aspects of software artifacts and **measurements** which can be interpreted as the results of those processes.
 - specifies several types of measures and measurements for different outcome values
 - *DimensionalMeasure, DirectMeasure, BinaryMeasure, CollectiveMeasure*



Automatic Generation of Metrics

Metric Generation Rules

- MOF-based DSL models
 - basic patterns and common aims of particular groups of metrics
 - patterns lead to the definition of numerous generation rules which allow the automatic derivation of domain specific measures (metrics)
 - generation rules themselves are formulated as OCL expressions
 - evaluation of a rule's OCL expression returns a set of OCL tuples
 - A tuple provides measure specific data, e.g. its generated name, which is required to create the corresponding SMM Measure



Automatic Generation of Metrics

Metric Generation Rules – Rule 1

- Partitioning based on enumeration typed class property
 - rule is applicable to classes in the analyzed meta-model having one or more attributes with an enum type
 - an enumeration defines a data type with a finite domain
 - it is possible to partition the set of instances of such a class based on the enumeration's literals
 - rule generates a specific Counting measure for each enumeration literal
 - number of the generated measures is equal to the number of the literals of the examined enumeration type

```
context EClass inv: self.eAttributes ->select(a | a.eType.oclIsKindOf(EEnum))
->collect(a | a.eType.oclAsType(EEnum).eLiterals->collect(l | Tuple {
scopeClass = self.name, nameFragments = Sequence {
'NoOf', a.eType.oclAsType(EEnum).getEEnumLiteral(l.value).literal, 'Books'},
operationFragments = Sequence{
'self.', a.name, ' = ', a.eType.name, '::',
a.eType.oclAsType(EEnum).getEEnumLiteral(l.value).literal}
})
)
```

Automatic Generation of Metrics

Metric Generation Rules – Rule 1

- Partitioning based on enumeration typed class property (Library Example)
 - Book class of the library model has an attribute category of the enumerated type BookCategory
 - enumeration type contains three literals, namely *Mystery*, *ScienceFiction* and *Biography*
 - generation process on the library model using this rule would result in the following three OCL tuples

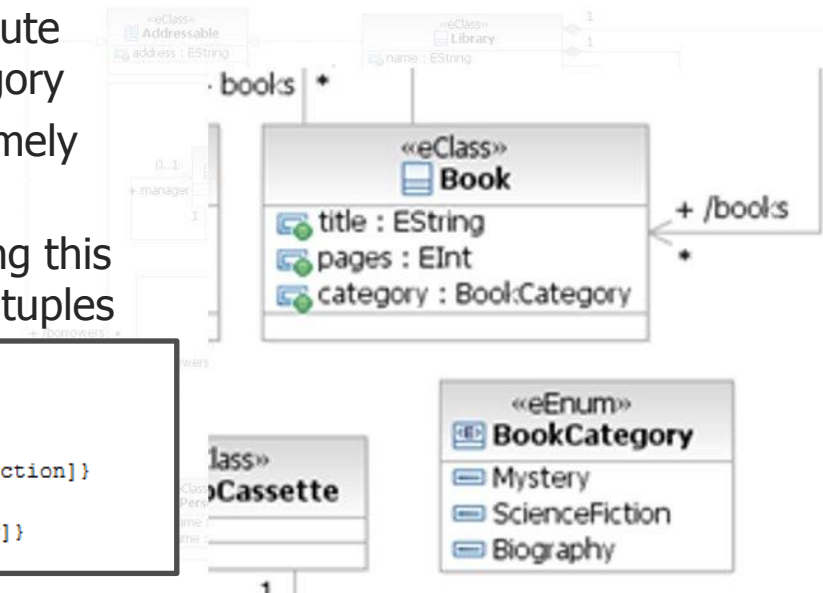
```

Tuple{scopeClass = 'Book', nameFragments = [NoOf, Mystery, Books],
      operationFragments = [self., category, =, BookCategory, ::, Mystery]}
Tuple{scopeClass = 'Book', nameFragments = [NoOf, ScienceFiction, Books],
      operationFragments = [self., category, =, BookCategory, ::, ScienceFiction]}
Tuple{scopeClass = 'Book', nameFragments = [NoOf, Biography, Books],
      operationFragments = [self., category, =, BookCategory, ::, Biography]}

```

- *NoOfMysteryBooks* generated for the enumeration literal *Mystery*
 - generated Counting measure

```
context Book inv: self.category = BookCategory::Mystery
```



Automatic Generation of Metrics

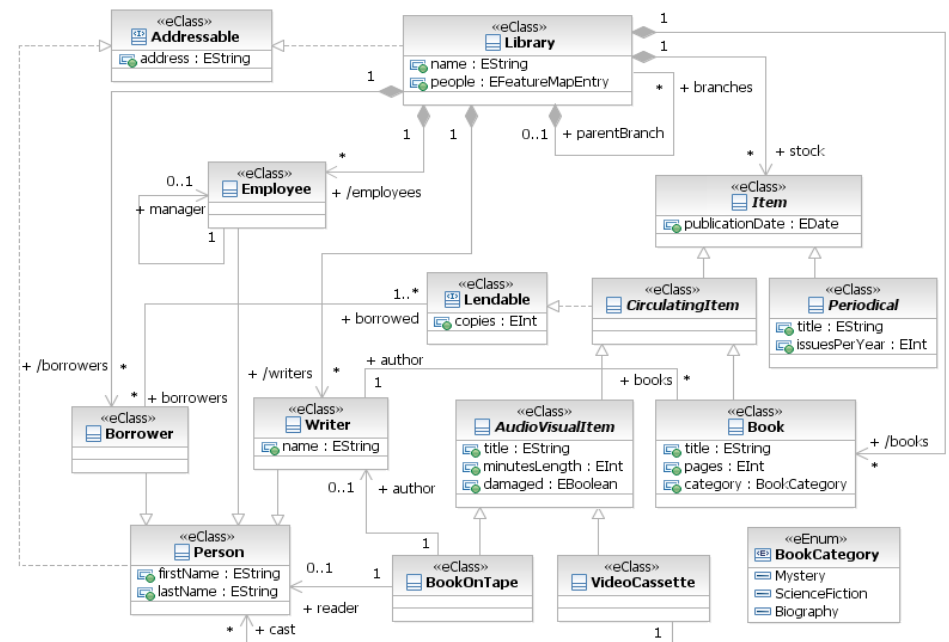
Metric Generation Rules – Rule 2

■ Number of contained Elements

- rule detects classes in a meta-model having containment references
- specific measure is derived which aims to count the referenced elements of a detected containing element in an instance model of the meta-model

■ Library example

- NoOfBorrowersLibrary
- NoOfEmployeesLibrary
- NoOfWritersLibrary
- NoOfBranchesLibrary
- NoOfStockLibrary
- NoOfBooksLibrary



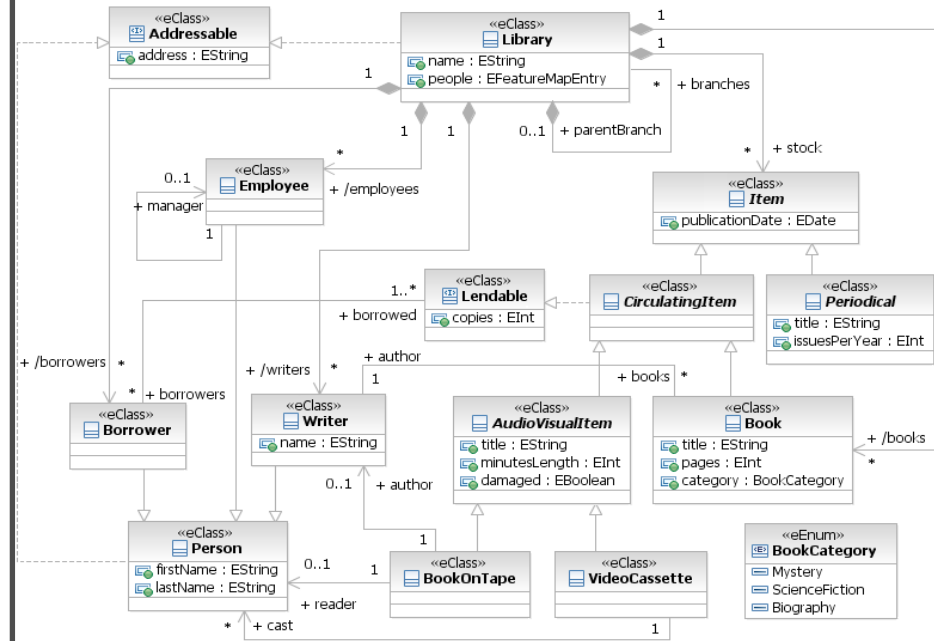
Automatic Generation of Metrics

Metric Generation Rules – Rule 3

- Partitioning based on Boolean class property
 - rule matches to meta-model classes which have at least one attribute of type Boolean
 - rule leads to the generation of a model specific measure which counts the instances of a matching class whose value for the examined attribute is “true”

■ Library example

- rule generates three measures for the attribute *damaged* of the model class *AudioVisualItem*
- measure *NoOfDamagedAudioVisualItem* will be generated
- classes *BookOnTape* and *VideoCassette* are subclasses of *AudioVisualItem*, the rule would additionally create the measures *NoOfDamagedBookOnTape* and *NoOfDamagedVideoCassette*



Automatic Generation of Metrics

Metric Generation Rules – Rule 4

■ Referential optionality

- rule matches to model classes that are the origin of an association with a lower bound value equal to zero
- each match the rule causes the generation of a measure which counts the instances of a matching model class where the reference's upper bound value is equal to zero

■ Library example

- NoOfBookOnTapeWithoutAuthor
- NoOfBookOnTapeWithoutReader
- NoOfEmployeeWithoutManager
- NoOfLibraryWithoutParentBranch
- NoOfLibraryWithoutBranches
- NoOfLibraryWithoutBorrowers
- NoOfLibraryWithoutBooks
- NoOfLibraryWithoutStock
- NoOfLibraryWithoutEmployees
- NoOfLibraryWithoutWriters
- NoOfWriterWithoutBooks
- NoOfLendableWithoutBorrowers
- NoOfVideoCassetteWithoutCast



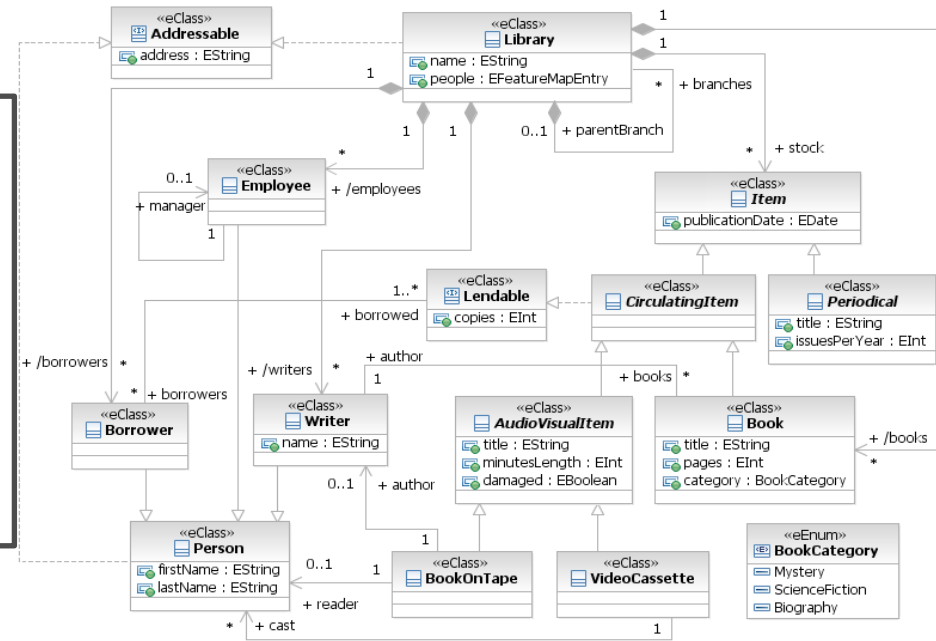
Automatic Generation of Metrics

Metric Generation Rules – Rule 5

- Depth of instance tree
 - rule matches to nested class structures in a meta-model
 - hierarchies are modeled through recursive class associations or associations between classes joining the same inheritance hierarchy (composite)
 - equivalent to the Chidamber & Kemerer metric Depth of inheritance tree (DIT)

■ Library example

- *Library* class has a containment reference to the *Library* class itself
- association is meant to model a branch hierarchy of a library
- rule would create a measure named *DepthInLibraryTree*



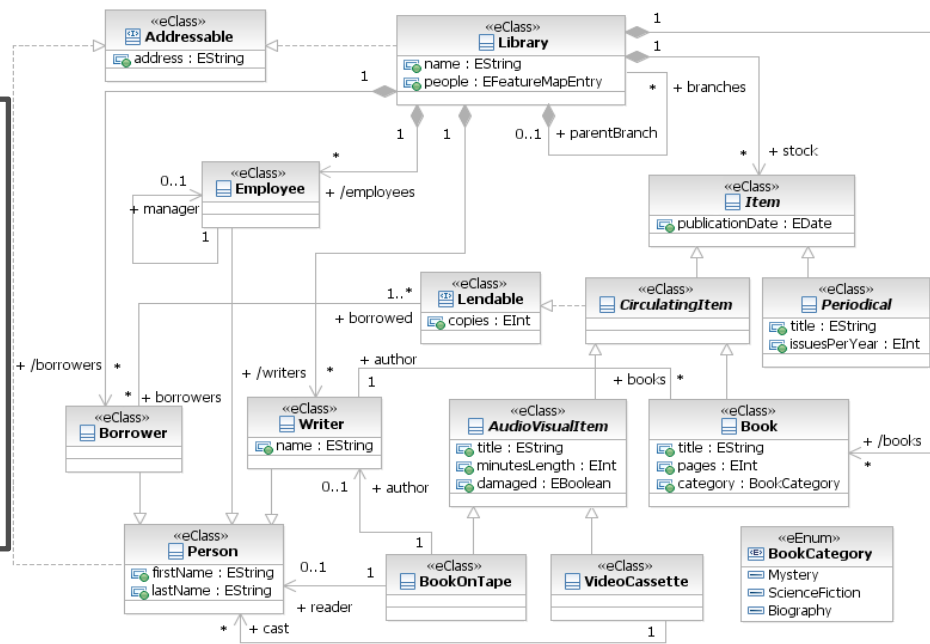
Automatic Generation of Metrics

Metric Generation Rules – Rule 6

- Number of children of an instance
 - rule is applicable to nested class structures in a meta-model
 - specific measure will be generated which calculates the number of children of each class instance in the hierarchy

■ Library example

- rule would generate a measure for the *Library* class
- Measure *NoOfChildrenLibrary* counts the number of branches of each particular Library class instance



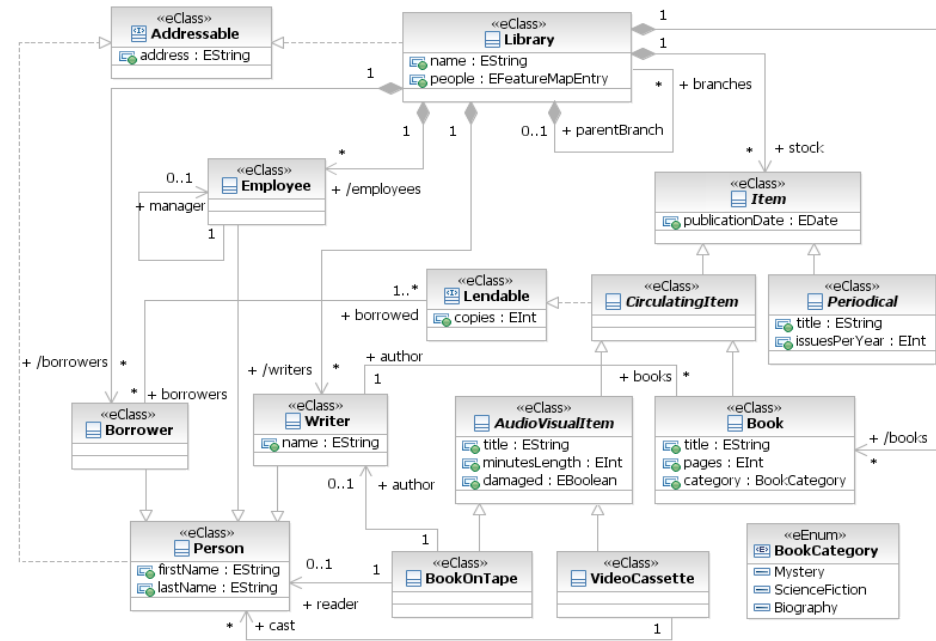
Automatic Generation of Metrics

Metric Generation Rules – Rule 7

- Referential existence dependencies between classes
 - rule is meant to generate measures which will detect referential existence dependencies between instances of model classes sharing an association
 - dependencies can be found regarding the lower bound value of the association

■ Library example

- rule would match to the association between the *Borrower* class and the interface *Lendable*
- borrower will only be captured in the model if he borrows at least one lendable item
- rule would create the measure *NoOfBorrowersMinimalBorrowed* which will count those instances of the *Borrower* class that are associated exactly with only one instance of a class implementing the *Lendable* interface



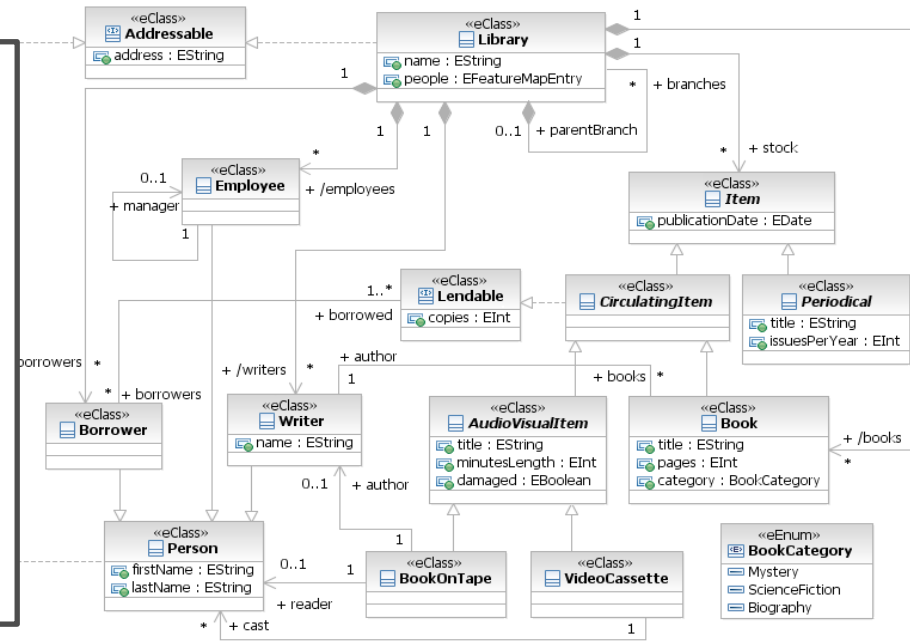
Automatic Generation of Metrics

Metric Generation Rules – Rule 9

- Aggregation of associated elements
 - rule matches to classes which have at least one multiplicity-many association
 - It creates measures which apply aggregate functions to the instances of such a class to assess the class-wide maximum, minimal, average, etc. value of referenced elements count

Library example

- rule would generate
- *MaxEmployeesAllLibrary*
- *MinEmployeesAllLibrary*
- *AvgEmployeesAllLibrary*



Automatic Generation of Metrics

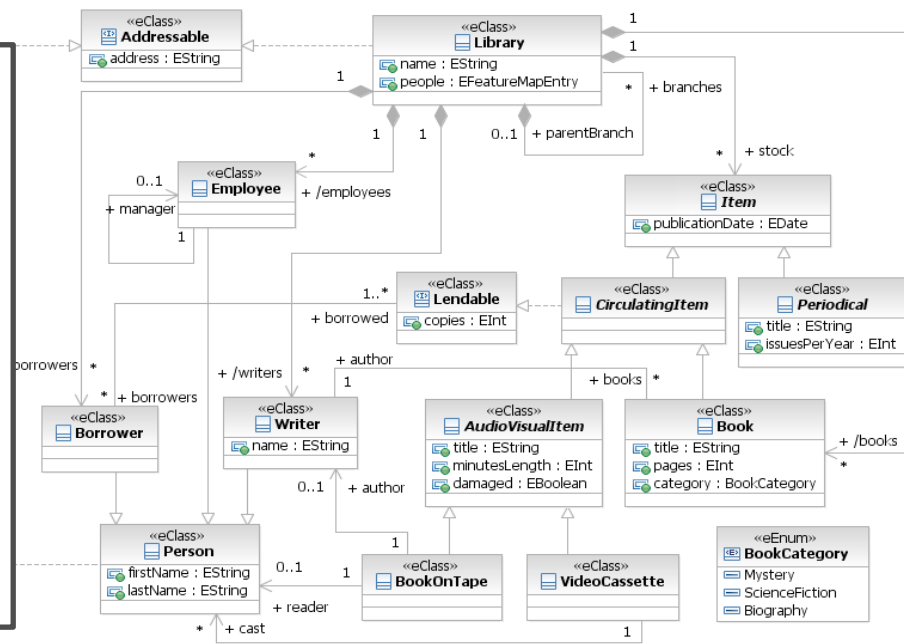
Metric Generation Rules – Rule 10

■ Aggregation based on numeric property

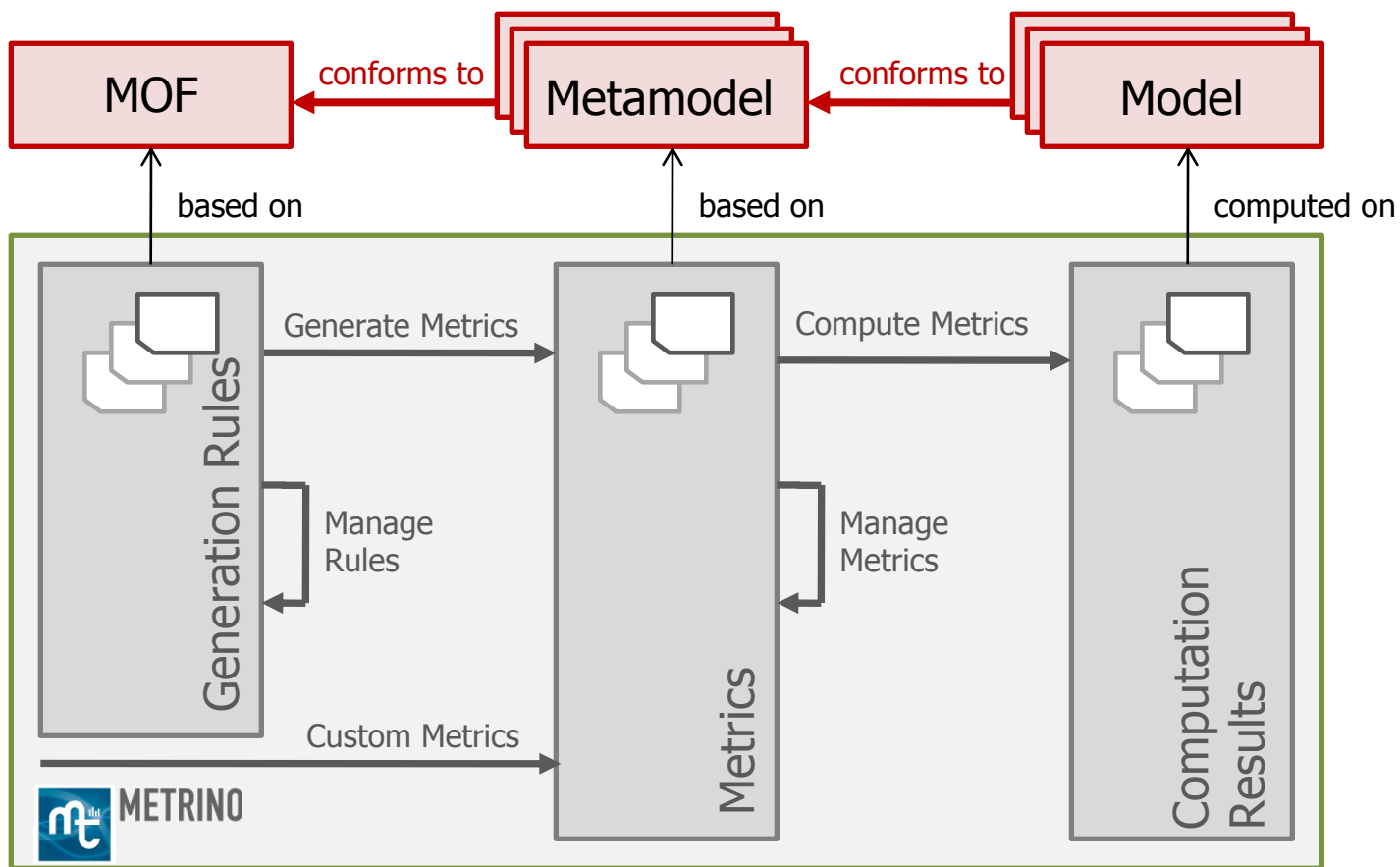
- rule matches to classes which have at least one property with a numeric type. It generates measures which apply aggregate functions to the instances of such a class to assess the class-wide maximum, minimal, average, etc. value for the examined property

■ Library example

- rule would generate
- *MaxPagesAllBook*
- *MinPagesAllBook*
- *AvgPagesAllBook*



METRINO Architecture



Tool Demo

- A screen cast showing the Metrino tool is available at:

<http://www.modelbus.org/modelbus/index.php/metrino>

