

# Experiences of using the Dagstuhl Middle Metamodel for defining software metrics

---

Jacqueline A. McQuillan



Principles of Programming Research Group  
Department of Computer Science  
National University of Ireland Maynooth, Co. Kildare, Ireland



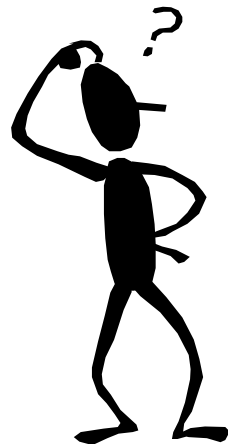
# Overview

---

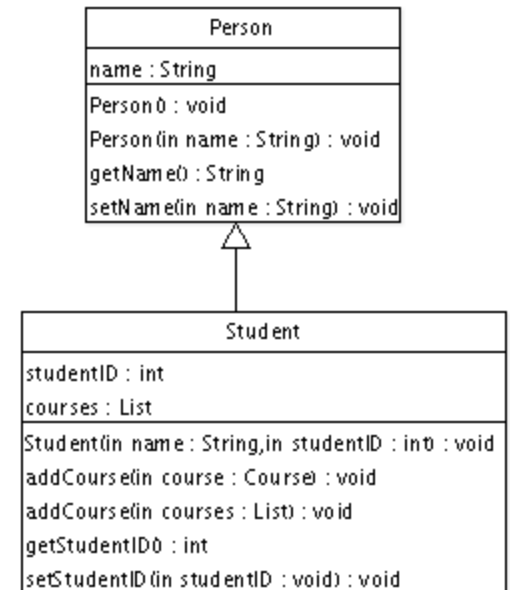
- Motivation
- Metamodels and the OCL
- Approach to defining software metrics
- Applying the approach to Java
- Implementation
- Exploratory data analysis
- Summary

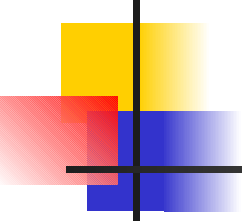
# 1 Motivation

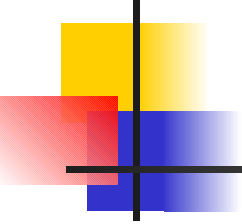
- Software metrics
- No standard terminology or formalism
- Different interpretations



9  
5  
3  
4  
Number of Methods?



- 
- 
- Cohesion metrics
  - Based on pairs of methods which use common attributes
    - exclude inherited methods and inherited attributes
    - include inherited methods and inherited attributes
    - exclude inherited methods but include inherited attributes
  - Range and variance of among metric definitions requires a flexible and reusable definition environment

- 
- 
- Measure metrics from both design (UML) and implementation (Java) level
  - Metric definitions should be re-usable
  - Ideally should be possible to define certain concepts once and adapt them to each level
  - Ensure that the same concepts are being measured at each level



## 2 Metamodels and the OCL

---

- Language meta-models and the Object Constraint Language (OCL) 2.0 as a mechanism for defining metrics

metamodel?

OCL?



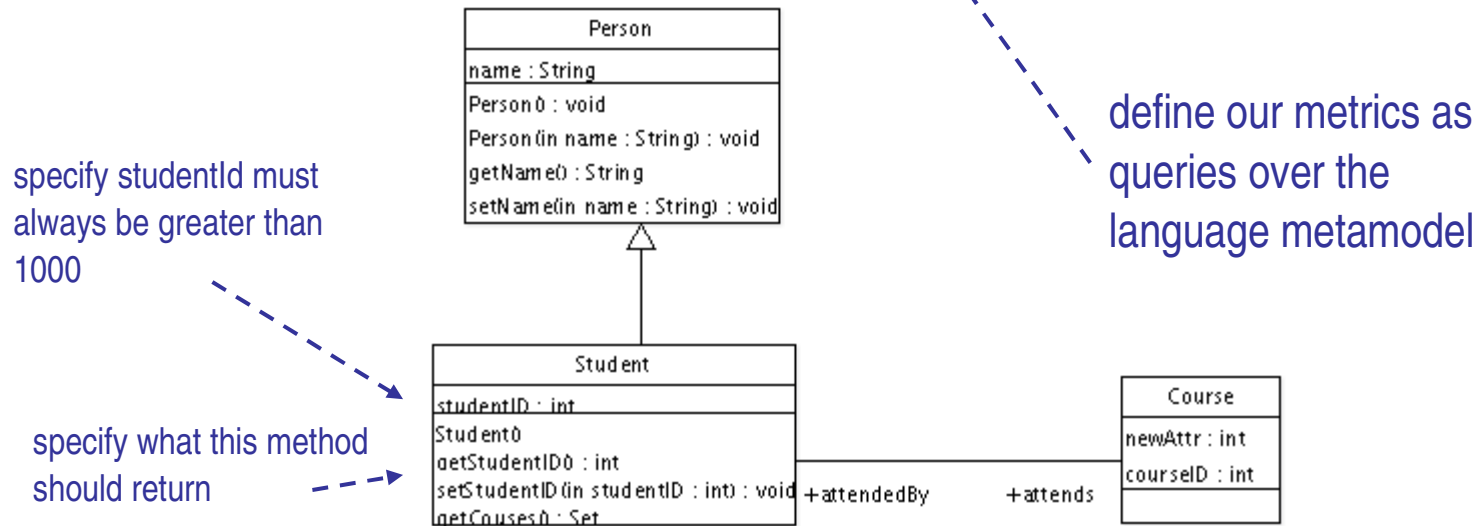
## 2.1 Metamodels

---

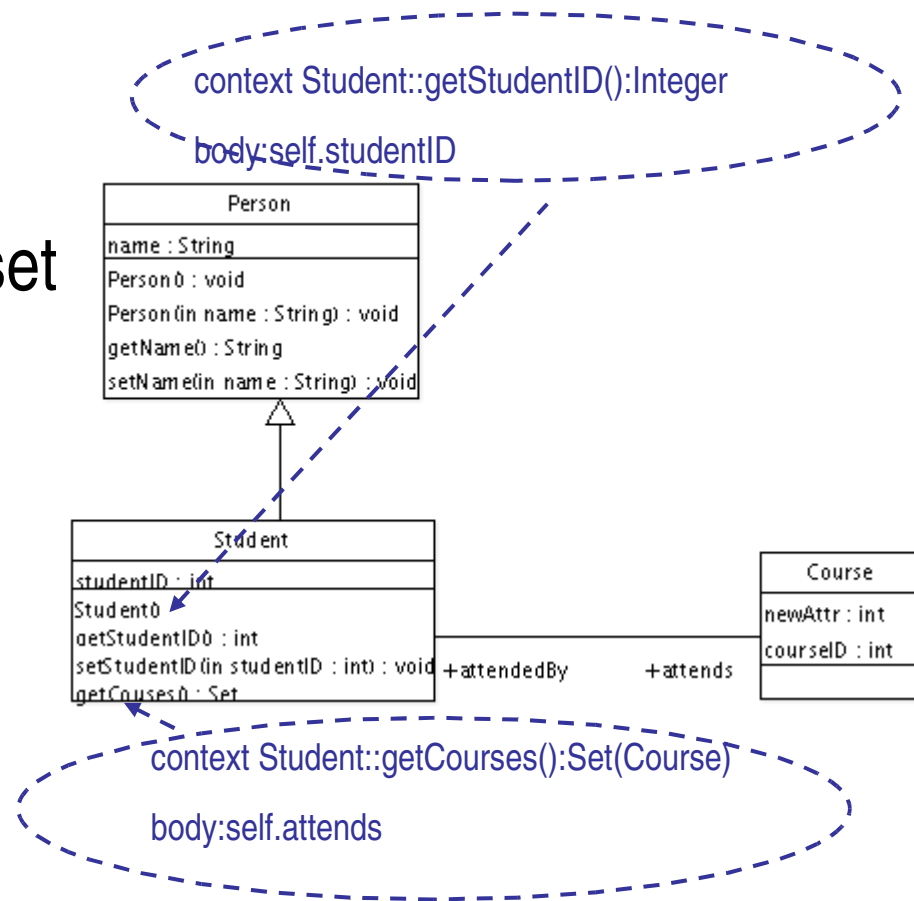
- What is a meta-model?
  - a model
  - used to define a language
  - specifies what constructs are allowed in the language and the relationships between the constructs eg. UML, Java metamodels

## 2.2 The Object Constraint Language

- Standard language that allows you to write constraints and queries over object oriented models in a clear and unambiguous manner



- This example states that the operation *getCourse* will always result in the set of Courses associated with the Student





## 3 Approach to defining metrics

---

- Express metrics as OCL queries
- Over a language metamodel
- Based on approach by Baroni *et al.*



# Approach by Baroni *et al.*

---

- Propose expressing design metrics as OCL expressions using the UML 1.3 metamodel
- Metrics defined as additional operations in the metamodel
- Involves modifying the metamodel



# Extensions to the approach

---

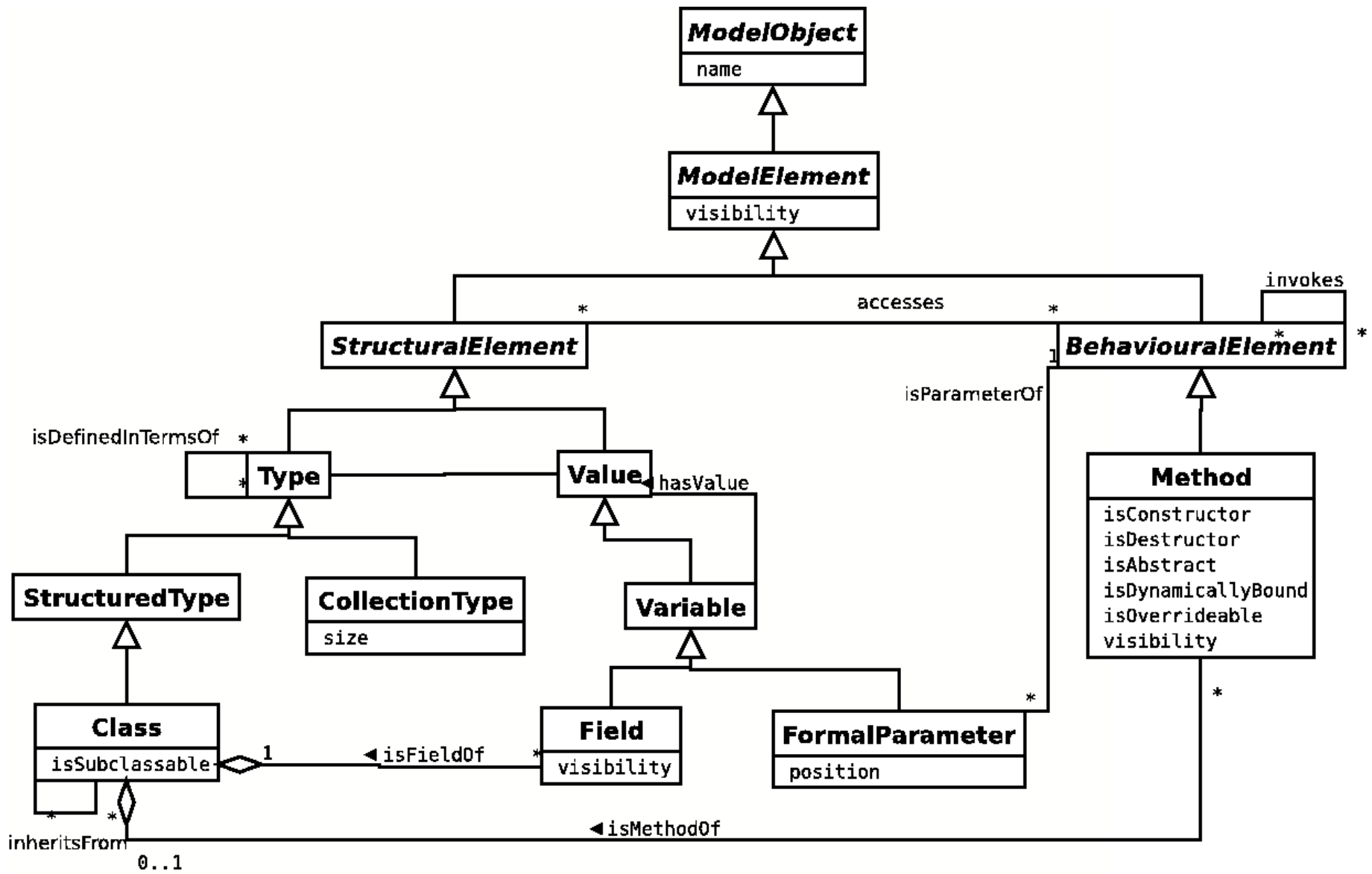
- Decoupling the metric definitions from the metamodel
- Express the definitions as OCL queries using the *body* expression
- Parameterised the definitions by the metamodel elements
- Approach is generalisable as it is parameterised by the both the metamodel and metric definitions



## 4 Applying the approach

---

- Express metrics as OCL queries
- Over a Java metamodel
- A number of metamodels have been proposed
  - Netbeans, Kollmann, Java Reflection API, jnome
- Choose Dagstuhl Middle Metamodel (DMM)
  - schema for reverse engineering
  - language independent, maximise the reusability of the metric definitions



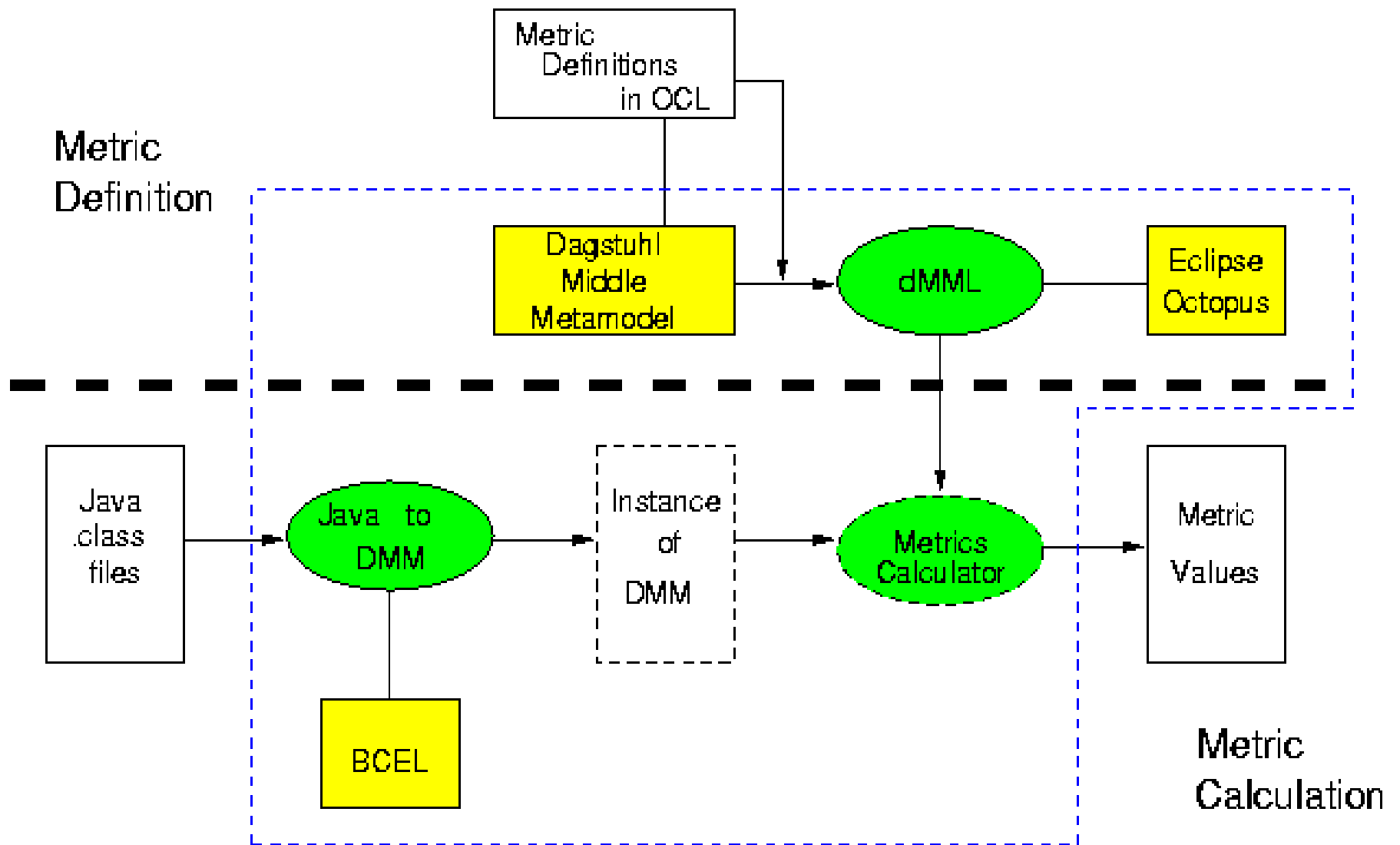
Part of the Dagstuhl Middle Metamodel

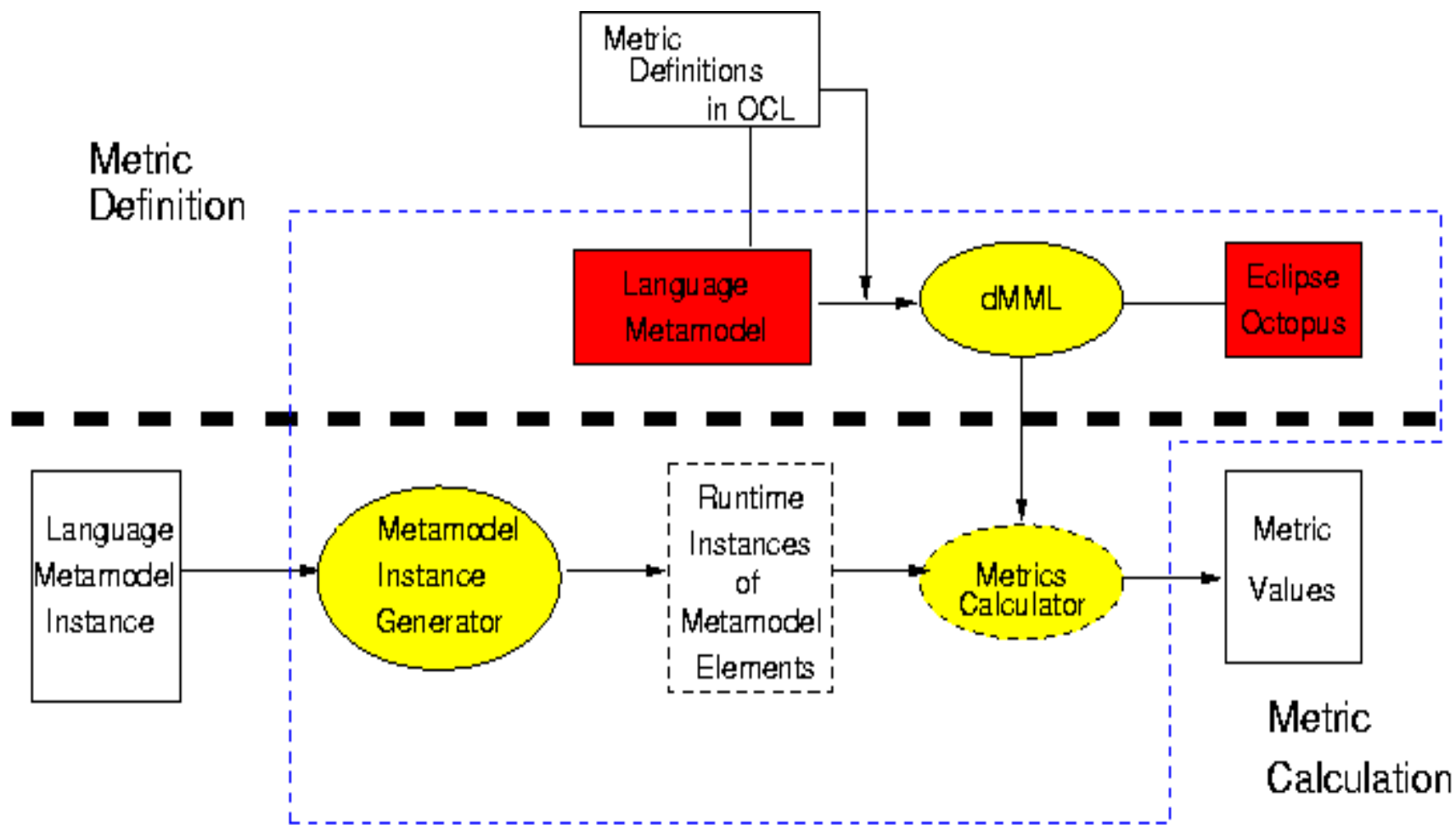


# 5 Implementation

---

- Implemented a system to calculate metrics for Java programs based on this approach
  - create a representation of the DMM in Java
  - convert Java programs to instances of DMM
  - apply metric definitions to the instances of the DMM







# 6 Exploratory data analysis

---

- Objectives
  - Validate our implementation
  - Investigate the relationship between the metric values
- Implemented several cohesion metrics
  - LCOM1, LCOM2, LCOM5, ICH
- Applied the metrics to the DaCapo benchmark suite

	<b><i>LCOM1</i></b>	<b><i>LCOM2</i></b>	<b><i>LCOM5</i></b>	<b><i>ICH</i></b>
<b>Min.</b>	0.0	0.0	0.0000	-3887.0
<b>Median</b>	11.0	6.0	0.8000	-2.0
<b>Mean</b>	185.5	135.5	0.7063	-20.5
<b>Max.</b>	110902.0	94039.0	2.0000	0.0

**Table 1:** Summary of the values for the metrics applied to 4836 of the classes from DaCapo suite

	<b><i>LCOM1</i></b>	<b><i>LCOM2</i></b>	<b><i>LCOM5</i></b>	<b><i>ICH</i></b>
<b>LCOM1</b>	1.0	0.8597	0.5305	-0.7439
<b>LCOM2</b>	0.8597	1.0	0.6453	-0.6463
<b>LCOM5</b>	0.5305	0.6453	1.0	-0.3739
<b>ICH</b>	-0.7439	-0.6463	-0.3739	1.0

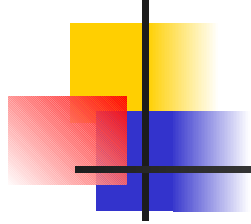
**Table 2:** Spearman's statistic for each pairing of the four metrics



# 7 Summary

---

- Presented an approach to define software metrics
- Applied approach to Java-based metrics using the DMM
- Developed a system to calculate metrics for Java programs
- Conducted an exploratory data analysis using cohesion metrics and DaCapo benchmark



---

Thank You!