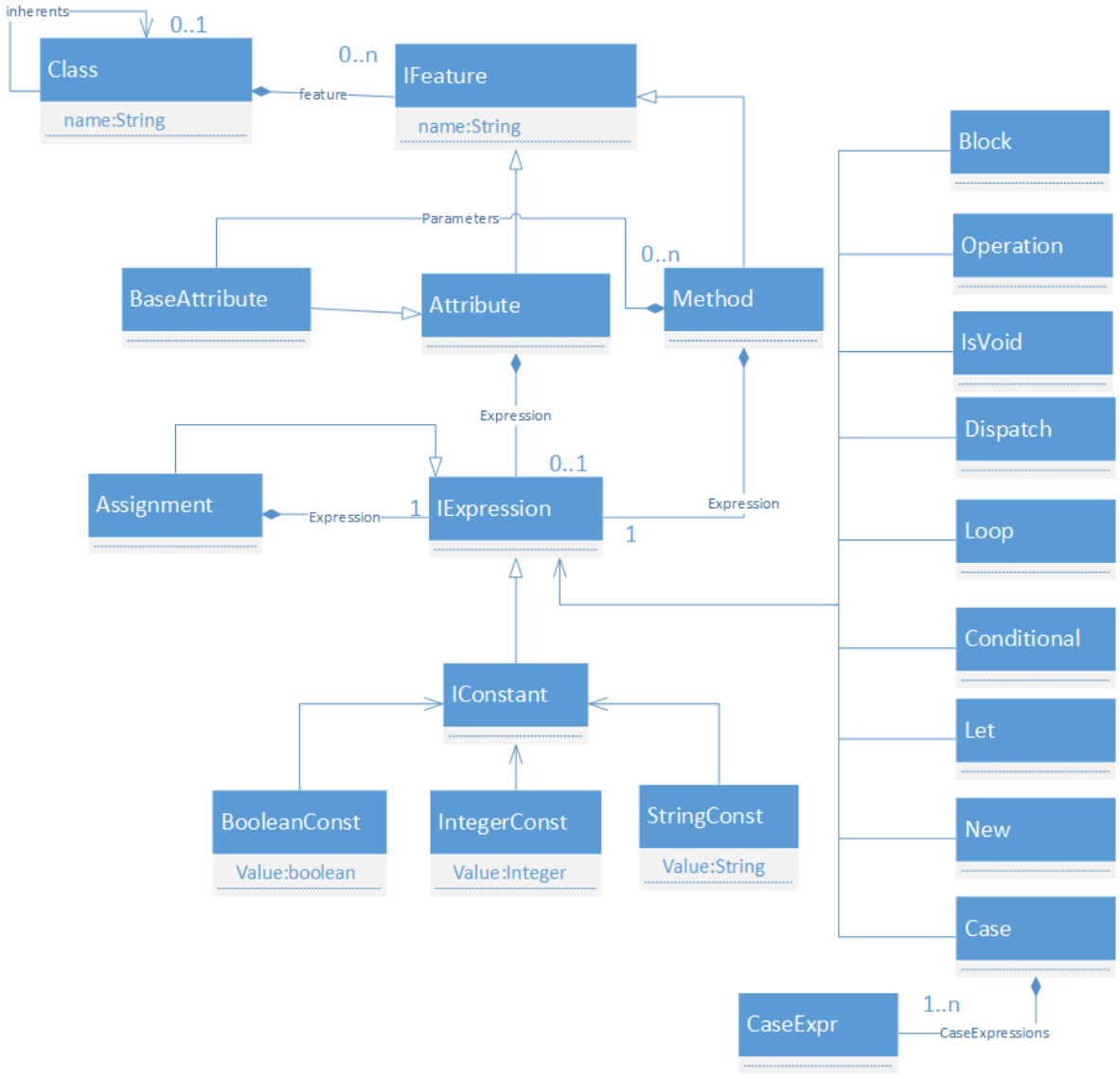
IKT-415 - Generative Programming

Exercise 2 - COOL Structure

Thomas Fauskanger Alexander Imenes Torry Tufteland
Guro Ødesneltvedt

Grimstad, September 2, 2015

1 MPS Structure of Cool



This structure is only an overview of the classes created in MPS. Not all details are fully captured here. For more details, browse the structure concepts from within MPS. For example, Dispatch has multiple compositions to IExpression, but this is not shown here as it would not be room to capture all relations completely.

2 Stash Structure

The structure in stash is only a single folder named languages which can be opened in MPS. All generated classes are ignored from the repository and has to be generated from within MPS.

3 Work Distribution

During this project we have mostly worked together. We started by drawing an AST individually, then we met and discussed our results. Based on our drawings we came up with one solution, and used this when creating the structure in MPS. While working in MPS we sat together on one computer. We have work closely together on all the steps of this project.

4 Cool Program described with Structure

4.1 Program

```
class Animal {
  name : String;
  get_name(): String {
    name <- "Animal"
  };
};

class Horse inherits Animal {
  get_name(): String {
    name <- "Horse"
  };
};

class Dog inherits Animal {
  get_name(): String {
    name <- "Dog"
  };
};

class Main inherits IO {
  dog: Dog;
  horse: Horse;
  main() : Object {{
    dog <- (new Dog);
    horse <- (new Horse);
    print_animal(horse);
    print_animal(dog);
  }};

  print_animal(animal: Animal): String {{
    out_string(animal.get_name());
  }};
};
```

4.2 Abstract syntax of Program

Animal

```
Concept: Class
    - name: Animal
    - feature: Method
        - name = get_name: String
        - Expression = Assignment [ implements IExpression, INamedConcept ]
            - Expression = StringConst inherits IConstant implements IExpression
                - value = “Animal”: String
```

Horse

```
Concept: Class
    - name: Horse
    - reference to class Animal
    - feature: Method
        - name = get_name: String
        - Expression = Assignment [ implements IExpression, INamedConcept ]
            - Expression = StringConst inherits IConstant implements IExpression
                - value = “Horse”: String
```

Main

```
Concept: Class
    - name = Main: String
    - feature_list:
        - Attribute:
            - name: dog
            - reference to class Dog
        - Attribute:
            - name: horse
            - reference to class Horse
        - Method:
            - name: main
            - reference to class Object
            - block with list of IExpressions
                - Assignment:
                    - Expression = New [ implements IExpression ]
                        - classref: Dog
                - Assignment
                    - Expression = New [ implements IExpression ]
                        - classref: Horse
                - Dispatch
                    - Expression = SELF : Type
                    - id = get_name
                    - className = “Dog”
                - Dispatch
                    - Expression = SELF : Type
                    - id = get_name
                    - className = “Horse”
        - Method:
            - name: print_animal
            - reference to class to String
            - parameters:
                - parameter:
                    - name: animal
                    - reference to concept Animal
            - Block with one IExpression
                - Dispatch:
                    - name: out_string
                    - parameter:
                        - Dispatch:
                            - name: get_name
                            - reference to className animal
```