Oblig 4 - Text Notation

Thomas Fauskanger
Alexander Imenes
Torry Tufteland
Guro Ødesneltvedt

Cool text notation

Attributes
<id> : <type> [<- <expr>]

Methods
<id>(<id> : <type>,...,<id> : <type>) : <type> { <expr> }

Assignment
<id> <- <expr>

Dispatch
<expr>.<id>(<expr>,...,<expr>)
<id>(<expr>,...,<expr>)
<expr>@<type>.<id>(<expr>,...,<expr>)

Conditionals
if <expr> then <expr> else <expr> fi

Loops
while <expr> loop <expr> pool

Blocks
{ <expr>; ... <expr>;}

Let
let <id1> : <type1> [<- <expr1>], ..., <idn> : <type> [<- <exprn>] in <expr>

Case
case <expr0> of
  <id1> : <type1> => <expr1>;
  ...
  <idn> : <type> => <exprn>;
esac

New
new <type>

isVoid
isVoid <expr>

Arithmetic and Comparison operations
<expr1> <op> <expr2>

where operations: +, -, *, / and <, <=, =

Description of text notation in stash

The notation are defined in the ‘editor’ folder within MPS. We have chosen a COOL program from previous hand-in to test our text notation in MPS (See left figure). Our MPS solution (figure 1, right side) has some differences from the original COOL languages. For example *identifiers* in COOL is should look like this: id <- <somevalue>, where id is an identifier that references to an attribute named id. To accomplish this we needed to add a constant (either a symbol or a string) in the front. Therefore our identifiers looks like this: @id <- <somevalue>.

We are also missing *dispatch*. We could not find a way to describe it using the MPS editor language. Therefore we replaced all original dispatches with an IntConst ‘5’.

```
class Main inherits IO {

  input_string : String;
  i : Int;

  main() : Object {
    {
      out_string("Write a sentence: ");
      input_string <- in_string();
      i <- 0;
      while (not i = input_string.length()) loop
        {
          if input_string.substr(i, 1) = " "
          then i <- input_string.length()
          else {
            out_string(input_string.substr(i, 1));
            i <- i+1;
          }
          fi;
        }
      pool;
      out_string("\n");
    }
  };
};
```

```
class Main inherits IO {
  input_string : String  ;
  i : Int  ;
  main (  ) : Object {
    {
      i <- 0 ;
      while not @i = 5 loop
        {
          if  @input_string = " "
          then i <- 5
          else {
                 i <- @i + 5
               }
          fi
        }
      pool
    }
  }
}
```

Figure 1: Original program(left) compared to our editor(right)

Work distribution

Firstly, we worked individually when finding the different text notations described in the COOL manual. Then we created a list of all the unique notations we had and prioritized them, in the order they had to be implemented for us to test properly. When the list was complete we divided the tasks among us and worked individually on implementing them in MPS.