CPIB Zwischenbericht

Daniel Gürber

November 15, 2012

Contents

1 Abstract

2 Erweiterung

2.1 Verkettete Vergleichsoperatoren

2.1.1 Ziel

Das Ziel ist es, IML so zu erweitern, dass Vergleichsoperatoren verkettet verwendet werden können. Ausdrücke sollen nicht mehrmals geschrieben werden müssen, um sie mit mehreren Werten zu vergleichen. Es soll nicht möglich sein, die Richtung zu wechseln; wenn ein < oder <= Operator verwendet wurde, darf also kein > oder >= Operator verwendet werden und umgekehrt.

2.2 Beispiele

- x < y < z anstatt x < y cand y < z
- x = y = z anstatt x = y cand y = z
- x > y = z anstatt x > y cand y = z
- a > b > c >= d anstatt a > b cand b > c cand c >= d

2.3 Grammatikänderungen

Um dies umzusetzen wird die Grammatik von IML angepasst:
Die Definition von term1 wird von

term1 ::= term2 [RELOPR term2]

zu

term1 ::= term2 {RELOPR term2}

geändert.

2.4 Aufrufe ohne call

2.4.1 Ziel

Das Ziel ist es, IML so zu erweitern, dass Prozeduraufrufe ohne das SchlÃijsselwort call mÃűglich sind.

2.5 Beispiele

- doSomething(); anstatt call doSomething;
- calculateIt(x init); anstatt call calculateIt(x init);

2.6 Grammatikänderungen

Um dies umzusetzen wird die Grammatik von IML angepasst:

Die Definition von cmd wird von

```
cmd ::= SKIP
      | expr BECOMES expr
      | IF LPAREN expr RPAREN blockCmd ELSE blockCmd
      | WHILE LPAREN expr RPAREN blockCmd
      | CALL IDENT exprList [INIT globInitList]
      | QUESTMARK expr
      | EXCLAMARK expr
```

zu

```
cmd ::= SKIP
      | expr [BECOMES expr]
      | IF LPAREN expr RPAREN blockCmd ELSE blockCmd
      | WHILE LPAREN expr RPAREN blockCmd
      | QUESTMARK expr
      | EXCLAMARK expr
```

geändert.

Die Definition von factor wird von

```
factor ::= LITERAL
         | IDENT [INIT | exprList]
         | monadicOpr factor
         | LPAREN expr RPAREN
```

zu

```
factor ::= LITERAL
         | IDENT [INIT | exprList [INIT globInitList]
         | monadicOpr factor
         | LPAREN expr RPAREN
```

geändert.