

Elements of Formal Specifications

Markus Roggenbach, Markus Bandt

October 18, 2001



Aim of Formal Specifications

Non Formal Specification



Formal Specification



Implementation

extract the “essentials” of

- a non formal specification
- all desirable implementations

Running Example: Database

Non Formal Specification:

Write a Java program that implements a database with

- “Name” and
- “Telephone Number”

as entries.

Implementation by Lists

Start Database

Show Source

Implementation by Sorted Binary Trees

Show Source

Compile Database

Essential for Programs

distinguish between functions of the

- **interface**
(functions that can be used “safely”)
- **implementation**
(functions that make sense only in a particular realization)

Formal Specification – First Element

Formal Specifications describe an **Interface** –
written down as **Signature**, i.e. a list consisting
of the

- Name and
- Profile

of all functions.

Interfaces and Programming Languages

PL supporting Interfaces:

C++, Modula, ML, Haskell, Java, Eiffel, ...

PL not supporting Interfaces:

Fortran, Pascal, C, Lisp, ...

Implementation by Lists – with Interface

Show Interface

Specifying the Interface in CASL

spec Database =

sorts *Database; String; Nat*

ops *initial* : *Database*;

look_up : *Database* \times *String* \rightarrow *Nat*;

update : *Database* \times *String* \times *Nat* \rightarrow *Database*

end

A Wrong Implementation

Start Database

Formal Specifications – Second Element

A formal specification includes beside the

- (i) signature
- (ii) a description of the functions' properties.

Programming Languages fail for (ii):

Expressing properties of a function involves implementation details.

Interface and Properties in CASL

spec Database =

sort *Database; String; Nat*

ops *initial : Database;*

0 : Nat;

look_up : Database \times String \rightarrow Nat;

update : Database \times String \times Nat \rightarrow Database

vars *s : Database; n : Nat; v, w : String*

• $\%[\text{initial}] \text{ look_up}(\text{initial}, v) = 0$

• $\%[\text{look_up_1}] v = w \Rightarrow$
 $\text{look_up}(\text{update}(s, v, n), w) = n$

- $\%[\text{look_up_2}] \neg v = w \Rightarrow$
 $\text{look_up}(\text{update}(s, v, n), w) = \text{look_up}($

end



Useless Database

Start Database

Formal Specifications = Abstract Datatypes

An Abstract Datatype consists of a

- (i) Signature,
- (ii) a description of the functions' properties,
- (iii) a description of the domains.

A Domain Description in CASL

spec Nat =

free types $Nat ::= 0 \mid$ **sort** $Pos;$
 $Pos ::= suc(pre : Nat)$

end