Foundations of System Development

Prof. Dr. Martin Wirsing

04.10.2002

Data-Oriented System Development

Prof. Dr. Martin Wirsing

History

1975 Guttag, Application of equational specifications in Programming; ADJ (Goguen, Thatcher, Wagner, Wright) first formal foundation, so called initial semantics.

1976 Gimona, Giarratana, Montanari, Wand: terminal semantics.

1978 CIP-Group (Bauer, Broy, Dosch, Partsch, Pepper, Wirsing) loose semantics of all term generated models.

Signatures for Interface Descriptions

Goals

- Learn to describe (the syntax of) interfaces with signatures
- Understand the construction of terms of a signature

Signatures

Required for the interface of a system:

- Names of externally visible data types ("sorts", "types")
- Names and type of externally visible functions (function symbols)
- Names and type of externally visible relations (predicate symbols)

This is called a signature.

Definition 1 (Signature):
An (algebraic) signature Σ is a triple (S, F, P) with

S : a set of sorts

F : a $S^* \times S$-sorted family of function symbols, whereby

- $\langle s_1, \ldots s_n \rangle \in S^*$: the domain of $F_{\langle \langle s_1, \ldots s_n \rangle, s \rangle}$
- $s \in S$: the range of $F_{\langle \langle s_1, \ldots s_n \rangle, s \rangle}$

P : a S^*-sorted family of predicate symbols.

Notice:
Total function: Instead of $f_{\langle \langle s_1, \ldots s_n \rangle, s \rangle}$ we write

$$f : s_1 \times \ldots \times s_n \longrightarrow s$$

CASL: Additional notation for (possibly) partial functions

$$f : s_1 \times \ldots s_n \longrightarrow ?s$$

Example 1 (Boolean Values):

sig BOOL0 $=$
    sorts Bool
    ops true: Bool;
        false : Bool;
        not _ : Bool $\rightarrow$ Bool;
        _ and _, _ or _, _ implies _ : Bool $\times$ Bool $\rightarrow$ Bool
end

Example 2 (Natural Numbers):

sig NAT0 $=$
    sorts Nat
    ops zero: Nat;
        succ _: Nat $\rightarrow$ Nat
end

Terms

So-called $\Sigma-$Terms can be build from the elements of a signature Σ. Such terms are constructed by free variables and the function symbols of Σ.
More exactly:

Definition 2 (Inductive Definition of Terms):

1. Let $\Sigma = (S, F, P)$, family $X = (X_s)_{s \in S}$ of identifiers be given. Then $T(\Sigma, X)_s$ (for all $s \in S$) is defined as the smallest set,
   s. t.:
   (a) For all $x \in X_s$ is $x \in T(\Sigma, X)_s$
   (b) If $f \in F_{\langle \epsilon, s \rangle}$ then $f \in T(\Sigma, X)_s$
   (c) If $f \in F_{\langle \langle s_1, \ldots, s_n \rangle, s \rangle}$ and $t_i \in T(\Sigma, X)_s (i = 1, \ldots, n)$ then $f(t_1, \ldots, t_n) \in T(\Sigma, X)_s$
2. Σ-terms without elements of X are called ground terms, i. e. every $t \in T(\Sigma, \emptyset)_s$ is ground term
3. A signature is called sensible, if for every $s \in S$ exists a ground term $t \in T(\Sigma, \emptyset)_s$.
4. A term $t \in T(\Sigma, X)_s$ is called of the sort **s**.
5. $T(\Sigma, X) \ =_{\text{def}} \ T(\Sigma, X)_s \in S$

Example 3 (Terms):

Let x be a variable of the sort Nat, s be a variable of the sort set.
So:

empty, $\{$succ(x), succ(succ(zero))$\}$, $s \cup$ empty	are terms of sort Set (whereby $Elem = Nat$)
zero, succ(succ(zero)), succ(x)	are terms of sort Nat

Summary

- A signature serves as syntactic description of interfaces.
- The set of syntactically correct terms is generated from a signature (and a family of variables)