Software specification in CASL - Datatypes

Till Mossakowski, Lutz Schröder

October 2006

SFB/TR 8 SPATIAL COGNITION

Free datatypes: Natural numbers

spec $\text{NAT} =$

free type $Nat ::= 0 \mid suc(Nat)$

preds $\_\_<=\_\_,\ \_\_>=\_\_ : Nat \times Nat$

ops $\_\_+\_\_ : Nat \times Nat \to Nat;$

$\forall\ m,\ n,\ r,\ s,\ t : Nat$

%% axioms concerning predicates

- $0 <= n$ **%(**leq_def1_Nat**)%**
- $\neg\ suc(n) <= 0$ **%(**leq_def2_Nat**)%**
- $suc(m) <= suc(n) \Leftrightarrow m <= n$ **%(**leq_def3_Nat**)%**
- $m >= n \Leftrightarrow n <= m$ **%(**geq_def_Nat**)%**

%% axioms concerning operations

- $0 + m = m$ **%(**add_0_Nat**)%**
- $suc(n) + m = suc(n + m)$ **%(**add_suc_Nat**)%**

end

Lists

spec LIST [**sort** $Elem$] = **%mono**

free type $List[Elem] ::= [] \mid \_\_::\_\_(Elem; List[Elem])$

op $\_\_++\_\_ : List[Elem] \times List[Elem] \to List[Elem]$

$\forall\ x : Elem;\ L,\ K : List[Elem]$

%% axioms concerning operations

- $[] ++ K = K$ **%(**concat_nil_List**)%**
- $(x :: L) ++ K = x :: (L ++ K)$ **%(**concat_NeList_List**)%**

end

Mutually Recursive Datatypes

spec UnboundedBranchingTree[**sort** $Elem$] =

free types

$Tree ::= Leaf(Elem) \mid Branch(Forest);$

$Forest ::= Nil \mid Cons(Tree; Forest)$

end