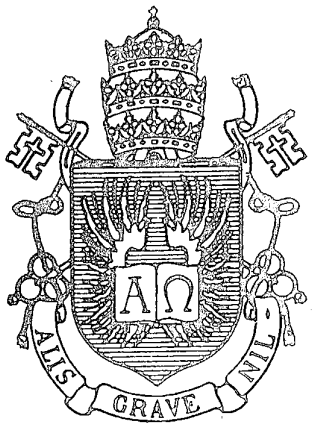

PUC

Series: Monografias em Ciência da Computação

Nº 7/82

SPECIFYING ABSTRACT DATA TYPES VIA SERIES OF REWRITING SYSTEMS

by

P. A. S. Veloso

and

T. S. E. Maibaum

Departamento de Informática

PONTIFÍCIA UNIVERSIDADE CATÓLICA DO RIO DE JANEIRO

RUA MARQUÊS DE SÃO VICENTE, 225 – CEP-22453

RIO DE JANEIRO – BRASIL

Series: Monografias em Ciência da Computação

Nº 7/82

Series Editor: Marco A. Casanova July 1982

SPECIFYING ABSTRACT DATA TYPES VIA SERIES OF REWRITING SYSTEMS *

by

P. A. S. Veloso

and

T. S. E. Maibaum +

*Research partly spondored by Finep and CNPq

+Dept. of Computing, Imperial College of Science and Technology, London, U.K.; partly sponsored by IBM do Brasil.

A B S T R A C T

Abstract data types have been widely recognized as a powerful programming tool. However, the full exploitation of their benefits requires formal specifications, which are generally not so easy to obtain or to understand. Helping alleviating both problems is an important motivation for proposing multi-level specifications. Each level is to consist of a system of rewrite rules transforming an approximation to a canonical form into a better one. This supports a stepwise refinement approach to the problem, leading to a better documentation for the specification.

The problem of formally specifying a model is decomposed into first obtaining canonical representatives, then designing a rewriting system . Each of these subproblems in turn is factorized into a finite sequence of approximations.

Some criteria are developed which guide the application of the methodology and justify it, in that they give sufficient conditions for the correcteness and completeness of the specification so obtained.

Key words: abstract data types, specification methodology, term rewriting rules, stepwise refinement , canonical forms.

R E S U M O

Tipos abstratos de dados têm se firmado como uma poderosa ferramenta de programação. Porém, para se explorar realmente seus benefícios são necessárias especificações formais, as quais geralmente não são fáceis de se obter nem de se compreender. Aliviar um pouco estes problemas é uma motivação importante para se propor a especificação multi-nível. Cada nível deve consistir de um sistema de regras de reescrita transformando uma aproximação de forma canônica em outra melhor. Com isso, o problema é atacado por meio de refinamento sucessivos, o que acaba levando a uma melhor documentação para a especificação.

O problema de se especificar formalmente um modelo é decomposto em dois: primeiro, obter representantes canônicos; depois, projetar um sistema de reescrita. Cada um desses problemas, por sua vez, é decomposto em uma sequência finita de aproximações.

São desenvolvidos alguns critérios que guiam a aplicação da metodologia e a justificam, pois dão condições suficientes para a correção e completeza da especificação obtida.

Palavras chaves: tipos abstratos de dados, metodologia de especificação, regras de reescrita de termos, refinamentos sucessivos, formas canônicas.

1. INTRODUCTION

This paper proposes multi-level specifications as a way to alleviate the problem of obtaining a (correct and complete) formal specification for an abstract data type. As a by-product, a better documentation suggests itself, which contributes to the understandability of the formal specification. Each level is to consist of a system of rewrite rules (not necessarily Church-Rosser or terminating) transforming an approximation to a canonical form into a better one.

Abstract data types (ADT's) constitute a powerful programming tool [LZ'74]. They are useful in connection with program construction, verfication, documentation, etc., due to the factorization of a programming task into an abstract program and an implementation module. However, a formal specification of the ADT, independent of any particular representation, is needed in order to fully exploit the benefits of abstraction [G'77, GHM'76]. Unfortunately formal specifica - tions are not generally praised for their legibility or ease in obtaining them [LZ'75, FM'81]. In fact, we are not exaggerating in saying that the art of designing formal specifications is a difficult one, which is aggravated by the fact that it is not yet sufficiently well understood.

A widely known specification formalism for ADT's is the initial algebra approach [GTW'78, TWW'76]. For this approach, the concept of canonical term algebra (cta, for short), employed as an aid in proving the correctness of specifications [GTW'78], seems to provide a useful tool for systematically obtaining a correct and complete specification for a given model [PV'78]. Together with the idea of tree transformations given by term rewriting rules [HO'80], one has a helpful methodology for the difficult task of constructing an algebraic specification [V'81]. Its main steps are basically:

1) elect a cta isomorphic to the given model;
2) obtain a set of rewrite rules to transform each ground term into its canonical representative.

This methodology gives some hints on how to proceed and its tests provide suggestions for corrective actions [PV'78, V'81]. However, it still leaves the designer facing some problems:

i) it may be difficult to see how to choose canonical representatives so as to have uniqueness;
ii) obtaining a consistent and complete set of rewrite rules amounts to designing a term-manipulating program (generally involving non-deterministic and parallel actions), which may very well turn out to be a non-trivial task.

In order to help overcoming the above difficulties we propose a "stepwise refinement" approach, as follows:

I) use a (finite) sequence $T_0, T_1, \ldots, T_M$ of sets of canonical terms as partial approximations to one having uniqueness of representatives;

III) use a (finite) sequence $\Gamma_1, \ldots, \Gamma_n$ of rewrite systems producing partial approximations to the desired overall transformations.

In the sequel we shall develop some results underlying this methodology to the extend that they indicate and certify the various decisions to be taken in applying it.

2. PRELIMINARES

The specification problem in the initial algebra framework may be stated as follows. We are given, say, set-theoretically,

(a) (the syntax of) a (may-sorted) language L, consisting of a set S of sorts and a set Σ of operations symbols together with their profiles $\pi : \Sigma \to S^* \times S$;

(b) a finitely generated L-algebra M.

Notice that this amounts to giving

(α) the free term algebra T of L;

(β) the unique (surjective) homomorphism $g: T \to M$

We are required to find, assuming its existence, a finite set E of (conditional) equations such that $M \cong T/\equiv [E]$, where $\equiv [E]$ is the congruence induced on T by E [GTW'78].

Now, consider the expansion $L[M]$ of L by one constant symbol i_m for each element m of M [S' 67]. Then α and β amount to [$\alpha+\beta$] a rewrite system G in $L[M]$ such that for all $t \in T$

$$t \xrightarrow[G]{*} i_{g(t)}$$

Then, G is a finitely terminating, Church-Rosser system with set of irreducible elements consisting exactly of the set of names added to L.

Of course, G involves extra names which ought to be discarded. In order to state more clearly our approach to the problem let us fix some terminology.

Consider a system Γ of rewrite rules and sets $P,Q \subseteq T$. Call Q Γ-reachable from P ($P \xrightarrow{\Gamma} Q$) iff for every $p \in P$ there exists $q \in Q$ such that $p \xrightarrow[\Gamma]{*} q$. Also, define the set of constructors in P as $Constr(P) = \{\sigma \in \Sigma \ / \sigma\ p_1 \ldots p_n \in P$ for some $p_1, \ldots, p_n \in P\}$ and, X being a family of sets of variables, let the stabilizer of Q be

$$Stab(Q) = \{t(x_1, \ldots, x_n) \in T(X) \ / \ t(q_1, \ldots, q_n) \in Q \text{ for all } q_1, \ldots, q_n \in Q\}$$

Finally, call Γ g-consistent on P iff whenever $p \xrightarrow[\Gamma]{*} t$ with $p \in P$ then $g(p) = g(t)$, and for $\sigma \in \Sigma$, let $\sigma Q = \{\sigma q_1 \ldots q_n / q_1, \ldots q_n \in Q\}$.

Now, we can get rid of the extra names, by reformulating the problem as

(ρ) obtain a set $P \subseteq T$ of unique representatives for the elements of H, i.e. the restriction $g|P$ is bijective;

(γ) obtain a rewriting system Γ on L, such that

Γ is complete: $T \xrightarrow{\Gamma} P$;

Γ is sound : Γ is g-consistent on T.

Such a system Γ must have the Church-Rosser property, but not necessarily that of finite termination (the role of the latter here is played by reachability).

3. CANONICAL REPRESENTATIVES

Our methodology hinges on the notion of a canonical form, i.e. a family R of sets $R_s \subseteq T_s$ for each $s \in S$, such that whenever $\sigma t_1 \ldots t_n \in R_s$ and $\sigma \in \Sigma$ has profile $(s_1 \ldots s_n, s)$ then $t_1 \in R_{s_1}, \ldots, t_n \in R_{s_n}$. Given M, we want a canonical form R where each element of M has a unique representative in R, i.e. the restriction of g to R is a bijection.

Now, notice that a canonical form Q with $g|Q$ surjective can be regarded as consisting of stepwise constructions for all the elements of M. Thus, if $g(q) = m$ then we regard q as a "trace" of a particular way of constructing m. As such q may contain redundant information, in that it pertains to this particular construction for m, rather than to the object m itself [FV'81]. Our strategy to obtain the desired R in one-to-one correspondence with M consists of identifying such redundant information in the following sense

- start with $T_0 = T$ the set of all ground terms;
- while $g|T_i$ is not injective, find a "smaller" canonical form T_{i+1} with $g(T_{i+1}) = g(T_i)$.

Clearly, one way to satisfy the above requirement of "smaller" is taking $T_{i+1} \subset T_i$. However, this is not necessary. Some helpful hints to guarantee $g(T_{i+1}) = g(T_i)$ are given in section 5.

As a simple example let us consider the case of finite sets of elements from some set. Here we have two sorts F and D and operations empty, ins, rem with profiles $\pi(\underline{empty}) = (\lambda,F)$, $\pi(\underline{ins}) = \pi(\underline{rem}) = (FD,F)$. For simplicity sake we assume sort D to be already specified, which supplies a set E of unique representatives for the elements of the domain of sort D of the model M. The domain of sort F is to consist of the finite sets of these elements, the operations being interpreted as $\emptyset$, insertion and removal.

In order to obtain a canonical form consisting of unique representatives we start with $T_0 = T_F$, all ground terms of the forms empty, ins(t,e), rem(t,e) with $t \in T_F$ and $e \in E$. Now we may obtain the following sequence of canonical forms

1) It is useless to insert an element more than once; so take
   $T_1 = \{t \in T_0 \ / \$ for each $e \in E$ at most one ins(.,e) occurs in t$\}$.
2) Rather than removing an element present in a set we can avoid inserting it in the first place; so take
   $T_2 = \{t \in T_0 \ / \ t$ has no occurrence of rem $\}$.
3) $T_3 = T_1 \cap T_2 = \{\underline{ins}(\ldots\underline{ins}(\underline{empty},e_1),\ldots e_k) / e_1 \neq \ldots \neq e_k \in E, k \in N\}$
4) Choose a total order $<$ on E (which is always possible) and put $T_4 = \{\underline{ins}(\ldots\underline{ins}(\underline{empty},e_1),\ldots,e_k) / e_1 < \ldots < e_k \in E,\ k \in N\}$

Now we have

$$T = T_0 \supset T_1 \supset T_3 \supseteq T_4 = R$$
$$\qquad\quad \supset T_2 \supset$$

with $g|R$ bijective.

4. SEQUENCE OF REWRITE SYSTEMS

Having obtained a canonical form R in one-to-one correspondence with M our task now is step (γ) of section 2. Now, obtaining such a system Γ may very well prove to be a difficult task. So, we decompose it into obtaining

[$\gamma\rho$] a sequence $T=R_0, R_1,\ldots,R_n = R$ of canonical forms (not necessarily the same used previously to approximate R);

[$\gamma\gamma$] a sequence of rewriting systems $\Gamma_1,\ldots,\Gamma_n$ such that for each $k = 1,\ldots,n, R_k$ is Γ_k-reachable from R_{k-1} and Γ_k is g-consistent on R_{k-1}.

Then $\Gamma = \Gamma_1 \cup \ldots \cup \Gamma_n$ will be a complete sound system in the sense of (γ) of section 2. Moreover, a better documentation for Γ suggests itself; namely $\Gamma_1\{R_1\}\,\Gamma_2\ldots\Gamma_{n-1}\{R_{n-1}\}\Gamma_n$, where the commentary $\{R_k\}$ gives a description of this intermediate canonical form. This presents the advantage of suggesting a prospective user of the specification a good and safe way to use it; first use the rules of Γ_1 to reduce into R_1, then apply rules of Γ_2, and so forth.

Two remarks on this method seem worthwhile. First, each individual Γ_k does not have to be Church-Rosser, only Γ. Second, we do not have to choose all the R_k's before the Γ_k's; it may be simpler to alternate: choose R_1 and find Γ_1, then choose R_2 and find Γ_2 and so forth.

For our example of finite sets, an apparently reasonable choice is $R_0 = T_0$, $R_1 = T_2$, $R_2 = T_3$, $R_3 = T_4$.

5. SOME CRITERIA

We shall now examine some criteria which may be helpfull in obtaining the rewriting systems Γ_k and ensuring that they exhibit the required properties (It will also be apparent that the same criteria correspond to analogous ones useful in obtaining $T_0, T_1, \ldots, T_M$ as in section 3). To this end, let P and Q be canonical forms and Δ a system of rewrite rules.

The first criterion exploits the recursive nature of canonical forms together with the notion of constructors to cut down the number of cases to be considered

(CQ) A sufficient condition for $P \xrightarrow{\Delta} Q$ is for all $\sigma \in \text{Constr}(P)$ $\sigma Q \xrightarrow{\Delta} Q$.

Also a frequently useful, though a bit strong, way to enforce the above condition is

(SQ) A sufficient condition for $\{\sigma t_1, \ldots t_m\} \xrightarrow{\Delta} Q$ is the existence of $t(x_1, \ldots, x_n) \in \text{Stab}(Q)$ and $q_1, \ldots, q_n \in Q$ with $\sigma t_1 \ldots t_m \xrightarrow[\Delta]{*} t(q_1, \ldots, q_n)$

Turning now to conditions for consistency, a natural one is the following

[gc] Δ is g-consistent on P if the system $\Delta+G$ is Church-Rosser.

However, consistency is a property of each individual rule and the following simple criterion offers some advantages

[gr] $\{t(x_1,\ldots,x_n) \longrightarrow t'(x_1,\ldots,x_n)\}$ is g-consistent on P iff for all $p_1,\ldots,p_n \in P$ $g[t(p_1,\ldots,p_n)] = g[t'(p_1,\ldots,p_n)]$.

The main advantage of using [gr] is the fact that whenever a rule fails to pass the test, this very test pinpoints the trouble spots and helps suggesting appropriate corrections [PV'78].

A typical example of this is the rule $\underline{rem}(\underline{ins}(x,e), e') \longrightarrow \underline{ins}(\underline{rem}(x,e'),e)$, which is g-consistent exactly when $e \neq e'$, thus suggesting a precondition for the corrected version.

Of course, when trying to obtain Δ one looks for simple transformations, hoping that simple rules will perform them. The crucial step here is, given $p \in P$, the choice of a $q \in Q$ to which p is to be transformed. A natural and useful heuristics is to choose $q \in Q$ with $g(p) = g(q)$ so that the term structure of q is similar to that of p. These observations together with the above criteria suggest the following:

(*) For each $\sigma \in Constr(P)$, obtain two (possibly infinite) sequences $P_0, P_1, \ldots$ and $Q_0, Q_1, \ldots$ of canonical forms and four rewriting systems such that

Ψ is g-consistent on σQ and $\sigma Q \xrightarrow{\Psi} \bigcup_i P_i$;

Φ is g-consistent on $\bigcup_i P_i$ and $P_{i+1} \xrightarrow{\Phi} P_i$;

Λ is g-consistent on P_0 and $P_0 \xrightarrow{\Lambda} Q_0$

Π is g-consistent on $\bigcup_j Q_j$ and $Q_j \xrightarrow{\Pi} Q \cup Q_{j+1}$

The idea here is that the sequences of P_i's and Q_j's reflect the syntactical structure of the terms in Q.

6. CONCLUSIONS

We have proposed multi-level specifications for ADT's as supporting a stepwise refinement approach to the problem of obtaining a formal specification, within the initial algebra framework, for a given model. We have also discussed some advantages of this methodology and presented some guidelines for its application. The basic underlying idea is that of factorizing $T \xrightarrow{\Gamma} R$ into $T \xrightarrow{\Gamma_1} R_1 \ldots \xrightarrow{\Gamma_n} R$.

Our experience suggests that the methodology is quite helpful and applicable. Besides, not being tied down to the initial algebra approach, it is adaptable to other approaches.

REFERENCES

[FM'81] A.L. Furtado, T.S.E. Maibaum - An informal approach to formal specifications;
Res. Rept.MCC 6/81, PUC - RJ

[FV'81] A.L. Furtado, P.A.S. Veloso - On multi-level specifications based on traces;
Res. Rept. MCC 8/81, PUC - RJ

[G' 77] J.V. Guttag - Abstract data types and the development of data structures;
CACM 20(1977), 397 - 404.

[GHM'76] J.V.Guttag, E. Horowitz, D.R.Musser - Abstract data types and software validation;
Res. Rept. 76 - 48 , ISI - USC.

[GTW'78] J.A.Goguen, J.W. Thatcher, E.G.Wagner - An initial algebra approach to the specification, correctness and implementation of abstract data types; in R.T. Yeh(ed.) Current trends in programming methodology, vol. IV, Prentice-Hall, 1978.

[HO'80] G. Huet, D.C. Oppen - Equations and rewrite rules: a survey; in R.V. Book (ed.) Formal language theory: perspectives and open problems; Academic, 1980.

[LZ'74] B.H. Liskov, S.N. Zilles - Programming with abstract data types ; SIGPLAN Notices 9 (1974), 50 - 59

[LZ'75] B.H. Liskov, S.N. Zilles - Specification techniques for data abstractions ; IEEE Trans.'Software Engin. SE - 1 (1975), 7 - 18 .

[PV'78] T.H.C. Pequeno, P.A.S. Veloso - Do not write more axioms than you have to; Proc. Internat. Computing Symp., vol 1; Academia Sinica, 1978, 487-498

[S'67] J.R. Shoenfield - Mathematical logic; Addison Wesley, 1967.

[TWW'76] J.W. Thatcher, E.G. Wagner, J.B. Wright - Specification of abstract data types using conditional axioms. Res. Rept. RC6214, IBM Yorktown Heights, 1976.

[V'81] P.A.S. Veloso - Methodical specification of abstract data types via rewriting systems; Res. Rept. MCC 7/81, PUC - RJ , 1981.