Some properties of implementations of abstract data types

by

Jacques Loeckx

A 80/14

December 1980

Fachbereich 10 - Informatik
Universität des Saarlandes
6600 Saarbrücken
West Germany

1. Introduction

Algorithmic specifications of abstract data types have been introduced in [Lo 80 b]. In [Lo 80 c] this specification method has been used for proving the correctness of an implementation of an abstract data type. The goal of the present paper is to formally prove the validity of the proof methodology used in [Lo 80 b]. Moreover the paper shows that the correctness of an implementation implies the validity of certain verification conditions; this allows the designer of a specification to omit certain proofs.

A large number of definitions and notations are borrowed from [Lo 80 b]; the reading of this report is therefore a prerequisite for understanding the present one.

2. The definition of implementation

Let A_σ be the algebra defined by an hierarchically structured set of algorithmic specifications containing a specification of the data type σ. Suppose that the set of specifications obtained by replacing the specification of σ by the specification of a data type τ is also hierarchically structured and call A_τ the corresponding algebra. The data types σ and τ are called *(strongly) equivalent* when the algebras A_σ and A_τ are isomorphic and when under this isomorphism the undefined and error values of type σ correspond respectively to the undefined and error values of type τ. Instead of saying that σ and τ are equivalent one may say that τ is a *(strong) implementation* of σ if τ is felt to be more "elementary" than σ, for instance because it is easier to write efficient programs for the external functions of τ.

When the specification sets defining A_σ and A_τ are both total, error-free and surjective it is also possible to introduce a weaker notion. Consider the subalgebras A'_σ and A'_τ of the algebras A_σ and A_τ obtained by deleting the undefined and error values from the carrier sets. The data types σ and τ are *weakly equivalent* if these A'_σ and A'_τ are isomorphic; *weak implementa-*

tions are defined similarly. Note that from a practical point of view weak implementations suffice to capture the intuitive notion of an implementation: if undefined and error values cannot occur as the result of a computation it is not necessary to specify how they are handled.

Section 3 introduces and proves the correctness conditions for weak equivalence. Section 4 is concerned with (strong) equivalence. The case of non-surjective specification sets is shortly discussed in Section 5. Section 6 indicates the verification conditions, the validity of which is implied by the validity of the correctness conditions.

3. Proving weak equivalence

3.1 The correctness conditions in the case of a representation function

Let A_σ and A_τ be the algebras defined by two hierarchically structured sets of specifications which are total, error-free and surjective as indicated above. Let moreover RP be a *representation function*, i. e. a function

$$RP : \underline{\tau} \rightarrow \underline{\sigma}$$

(cf. [SWL 77, GHM 78]). Consider now the following three *correctness conditions*:

(i) for all $d \in \underline{\tau}$:
if $Is.\tau(d) = \underline{true}$
then $Is.\sigma(RP(d)) = \underline{true}$

(ii) for all $d_1, d_2 \in \underline{\tau}$:
if $Is.\tau(d_1) = Is.\tau(d_2) = \underline{true}$
then $Eq.\tau(d_1, d_2) = Eq.\sigma(RP(d_1), RP(d_2))$

(iii) there exists a one-to-one correspondence between the external functions of σ and τ such that for each function

$$F : \underline{\rho}_1 \times \ldots \times \underline{\rho}_n \to \underline{\rho}_{n+1} \qquad (n \geq o)$$

of σ the corresponding function of τ is a function

$$Im.F: \underline{\rho}'_1 \times \ldots \times \underline{\rho}'_n \to \underline{\rho}'_{n+1}$$

with $\rho'_i = \begin{cases} \tau \text{ if } \rho_i = \sigma & , 1 \leq i \leq n+1 \\ \rho_i \text{ if } \rho_i \neq \sigma & , 1 \leq i \leq n+1 \end{cases}$;

moreover, for all $d_i \in \underline{\rho}_i$, $1 \leq i \leq n$:

if $Is.\rho_i(d_i) = \underline{true}$, $1 \leq i \leq n$

then $\begin{cases} Eq.\sigma\ (F\ (d'_1, \ldots , d'_n),\ RP\ (Im.F\ (d_1, \ldots d_n))) \\ = \underline{true} \\ \qquad \text{if } \rho_{n+1} = \sigma \\ Eq.\rho_{n+1}\ (F\ (d'_1, \ldots , d'_n),\ Im.F\ (d_1, \ldots, d_n)) \\ = \underline{true} \\ \qquad \text{if } \rho_{n+1} \neq \sigma \end{cases}$

where $d'_i = \begin{cases} RP\ (d_i) \text{ if } \rho_i = \sigma , & 1 \leq i \leq n \\ d_i \text{ if } \rho_i \neq \sigma , & 1 \leq i \leq n \end{cases}$

Note that the condition (ii) is compatible with the fact that $Eq.\tau$ has to be an equivalence relation: if $Eq.\sigma$ is an equivalence relation then it results from (ii) that $Eq.\tau$ is reflexive, symmetric and transitive.

It will now be shown that σ and τ are weakly equivalent if the correctness conditions are satisfied.

3.2. Theorem: Let A_σ , A_τ and RP be defined as in Section 3.1. If the correctness conditions of Section 3.1. are verified then σ and τ are weakly equivalent.

Proof

(a) The algebras A_σ and A_τ are defined by

- a carrier set $\underline{C}_\rho$ for each data type ρ; note that $\underline{C}_\rho$ contains in particular the elements $UNDEFINED_\rho$ and $ERROR_\rho$;

- a certain number of operations F_{op} on these carrier sets;

Consider now the subalgebras A'_σ and A'_τ of A_σ and A_τ defined by:

- the carrier set

$$\underline{C}'_\rho = \underline{C}_\rho - \{UNDEFINED_\rho, ERROR_\rho\}$$

  for each data type ρ;

- the restrictions F'_{op} to the sets $\underline{C}'_\rho$ of the operations F_{op}; note that each F'_{op} is a total function because the bases B_σ and B_τ are error-free, total and surjective.

According to the definition given in Section 2 the data types σ and τ are weakly equivalent if the algebras A'_σ and A'_τ are isomorphic.

Note that for each operation

$$F_{op} : \underline{C}_{\rho 1} \times \ldots \times \underline{C}_{\rho n} \rightarrow \underline{C}_{\rho n+1} \qquad , n \geq o$$

the corresponding operation

$$F'_{op} : \underline{C}'_{\rho 1} \times \ldots \times \underline{C}'_{\rho n} \rightarrow \underline{C}'_{\rho n+1}$$

is univocally defined by its values

$$F'_{op} ([t_1], \ldots, [t_n]) = [F (t_1, \ldots, t_n)]$$

for all terms t_i of type ρ with $Is._{\rho i}(t_i) = \underline{true}$, $1 \leq i \leq n$ (cf [Lo 80 b, Section 4.3]) (*)

(b) Associate with RP a function

$$RP_{op} : \underline{C}'_\tau \rightarrow \underline{C}'_\sigma$$

defined by its values

$$RP_{op} ([d]) = [RP (d)]$$

for all terms d of type τ with $Is.\tau(d) = \underline{true}$; note that this definition is consistent because:

- the notation [RP (d)] makes sense (*) because of the correctness conditions (i);

(*) Remember that for a term t of type ρ the notation [t] makes sense only if $Is.\rho(t) = \underline{true}$ (cf. [Lo 80 b])

- by the correctness condition (ii), $[RP\ (d_1)] = [RP\ (d_2)]$ if $[d_1] = [d_2]$ and $Is.\tau\ (d_1) = Is.\tau\ (d_2) = \underline{true}$.

We will now show that RP_{op} is a one-to-one function from $\underline{C}'_\tau$ onto $\underline{C}'_\sigma$.

(c) In order to show that R_{op} is one-to-one we show that it is injective and surjective.

R_{op} is injective by the correctness condition (ii): if $Is.\tau\ (d_1) = Is.\tau\ (d_2) = \underline{true}$ and $[RP\ (d_1)] \neq [RP\ (d_2)]$ then $[d_1] \neq [d_2]$.

To show that R_{op} is surjective we show that for each term c of type σ with $Is.\sigma\ (c) = \underline{true}$ there exists a term d of type τ with

- $Is.\tau\ (d) = \underline{true}$ (A 1)
- $[RP\ (d)] = [c]$
  or, equivalently (because of the correctness condition (i)):
  $Eq.\sigma\ (c,\ RP\ (d)) = \underline{true}$ (A 2)

Now, as A_σ is surjective there exists an expression built with external functions only, the value of which is c' with

$$Eq.\sigma\ (c',\ c) = \underline{true} \qquad (B)$$

(and with $Is.\sigma(c') = \underline{true}$ by the verification condition (iii) for the specification of σ : see [Lo 80 b, Section 5.2]). The proof of (A 1) and (A 2) is by induction on the (maximal) nesting depth of this expression.

If the nesting depth is zero, i. e. if there exists an external function F = c', there exists by the correctness condition (iii) an external function Im.F with

$$Eq.\sigma\ (F,\ RP\ (Im.F)) = \underline{true}$$

i. e.

$$Eq.\sigma\ (c',\ RP\ (Im.F)) = \underline{true}$$

or, by (B):

$$\text{Eq.}\sigma \quad (c,\ \text{RP}\ (\text{Im.F})) = \underline{\text{true}}$$

In order to satisfy (A_1) and (A_2) it is sufficient to choose d = Im.F; note in particular that

$$\text{Is.}\tau\ (\text{Im.F}) = \underline{\text{true}}$$

by the verification condition (iii) for the specification of τ.

If the nesting depth is n > o let

$$c' = F\ (c_1,\ \ldots,\ c_m)$$

where

$$F : \underline{\tau}_1 \times \ldots \times \underline{\tau}_m \rightarrow \underline{\sigma} \qquad ,\ m \geq 1$$

and $c_1, \ldots, c_m$ are values obtainable by expressions with nesting depth < n. Note that by the verification condition (iii) of the different types of the algebra A_σ one has for all i, $1 \leq i \leq m$:

$$\text{Is.}\tau_i\ (c_i) = \underline{\text{true}} \qquad \text{(C)}$$

Suppose now that $c_{j1}, \ldots, c_{jk}$, $1 \leq j_1 < j_2 < \ldots < j_k \leq m$,

$o \leq k$, are the values of type σ. By the correctness condition (iii) there exists an external function Im.F of τ such that for all d_i with $\text{Is.}\tau_i(d_i) = \underline{\text{true}}$, $1 \leq i \leq m$:

$$\left.\begin{array}{l} \text{Eq.}\sigma\ (F(d'_1,\ \ldots,\ d'_m),\ \text{RP}\ (\text{Im.F}(d_1,\ \ldots,\ d_m))) = \underline{\text{true}} \\ \text{with} \\ d'_i = \begin{cases} \text{RP}(d_i) & \text{if } i \in \{j_1,\ \ldots,\ j_k\} \\ d_i & \text{otherwise} \end{cases} \end{array}\right\} \qquad \text{(D)}$$

But by inductive assumption there exists for each i, $i \in \{j_1, \ldots j_k\}$, a term d_i^* of type τ with

$$\text{Is.}\tau\ (d_i^*) = \underline{\text{true}}$$

$$\text{and} \quad \text{Eq.}\sigma\ (c_i,\ \text{RP}\ (d_i^*)) = \underline{\text{true}} \qquad \text{(E)}$$

Putting in (D)

$$d_i = \begin{cases} d_i^* & \text{if } i \in \{j_1,\ \ldots,\ j_k\} \\ c_i & \text{otherwise} \end{cases} \qquad \text{(D1)}$$

one obtains

$$d'_i = \begin{cases} RP(d_i^*) & \text{if } i \in \{j_1, \ldots, j_k\} \\ c_i & \text{otherwise} \end{cases} \qquad (D2)$$

On the other hand the verification condition (ii) for the specification of type σ implies that equivalent arguments lead to equivalent values; hence, by (E), (D2) and the correctness condition (i):

$$Eq.\sigma\ (F\ (d'_1, \ldots, d'_m),\ F\ (c_1, \ldots, c_m)) = \underline{true}$$

Together with (D) this leads to

$$Eq.\sigma\ (F\ (c_1, \ldots, c_m),\ RP\ (Im.F\ (d_1, \ldots, d_m))) = \underline{true}$$

or

$$Eq.\sigma\ (c',\ RP\ (Im.F\ (d_1, \ldots, d_m))) = \underline{true}$$

or

$$Eq.\sigma\ (c,\ RP\ (Im.F\ (d_1, \ldots, d_m))) = \underline{true}$$

Hence, by taking

$$d = Im.F\ (d_1, \ldots, d_m)$$

we satisfy (A2). We also satisfy (A1) if we can prove

$$Is.\tau\ (Im.F\ (d_1, \ldots, d_m)) = \underline{true}$$

or, because of the verification condition (iii) of τ, if for all i, $1 \leq i \leq m$:

$$Is.\tau_i\ (d_i) = \underline{true} \qquad ;$$

this holds by (D2), (E) for the case $i \in \{j_1, \ldots j_k\}$; the other case holds by (C).

(d) We now show that RP_{op} commutes with the (restrictions of the) operations of σ and τ . More precisely, let

$$F : \underline{\rho}_i \times \ldots \times \underline{\rho}_n \rightarrow \underline{\rho}_{n+1} \qquad , n \geq o$$

be an external function of σ and Im.F the corresponding external function of τ; let F_{op} and $Im.F_{op}$ be the corresponding operations and F'_{op} and $Im.F'_{op}$ their restrictions; we have to show:

for all terms d_i of type τ_i with $Is.\tau_i(d_i) = \underline{true}$,

$1 \leq i \leq n$:

$$F'_{op}(s_1, \ldots, s_n) = \begin{cases} RP_{op}(Im.F'_{op}([d_1], \ldots, [d_n])) \\ \qquad \text{if } \rho_{n+1} = \sigma \\ Im.F'_{op}([d_1], \ldots, [d_n]) \\ \qquad \text{if } \rho_{n+1} \neq \sigma \end{cases}$$

where for all i, $1 \leq i \leq n$:

$$s_i = \begin{cases} RP_{op}([d_i]) & \text{if } \rho_i = \sigma \\ [d_i] & \text{if } \rho_i \neq \sigma \end{cases}$$

This relation is illustrated by:

$$\begin{tikzcd}
(s_1, \ldots, s_n) \arrow[d, "F'_{op}"'] & ([d_1], \ldots, [d_n]) \arrow[l, "(RP_{op})"'] \arrow[d, "Im.F'_{op}"] \\
\otimes & \otimes \arrow[l, "(RP_{op})"]
\end{tikzcd}$$

in this figure (RP_{op}) denotes the application of the function RP_{op} to the arguments of type σ only.

Consider first the case $\rho_{n+1} = \sigma$.

$RP_{op}(Im.F'_{op}([d_1], \ldots, [d_n]))$

$= RP_{op}(Im.F_{op}([d_1], \ldots, [d_n]))$

by the definition of $Im.F'_{op}$

$= RP_{op}([Im.F(d_1, \ldots, d_n)])$

by the definition of $Im.F_{op}$

$= [RP(Im.F(d_1, \ldots, d_n))]$

by the definition of RP_{op}

$= [F(d'_1, \ldots, d'_n)]$

with for each i, $1 \leq i \leq n$:

$$d'_i = \begin{cases} RP(d_i) & \text{if } \rho_i = \sigma \\ d_i & \text{otherwise} \end{cases}$$

by the correctness condition (iii)

$= F_{op}([d'_1], \ldots, [d'_n])$

$= F'_{op}([d'_1], \ldots [d'_n])$

$= F'_{op}(s_1, \ldots, s_n)$

with for each i, $1 \le i \le n$:

$$s_i = \begin{cases} [RP(d_i)] & \text{if } \rho_i = \sigma \\ [d_i] & \text{otherwise} \end{cases}$$

This concludes the proof for the case $\rho_{n+1} = \sigma$ because

$[RP(d_i)] = RP_{op}([d_i])$ if $\rho_i = \sigma$, $1 \le i \le n$

by the definition of RP_{op}.

The case $\rho_{n+1} \neq \sigma$ may be treated similarly.

(e) The theorem directly results from the sections (c) and (d)

☒

Examples of application of this theorem are in [Lo80c, Lo80a].

3.3. The case of an implementation function

Due to the symmetry of the notion of weak equivalence the use of an *implementation function*

$$IM : \underline{\tau} \rightarrow \underline{\sigma}$$

(cf [ADJ 78, Ga 79, Su 79]) instead of a representation function RP puts no problem: Section 3.1 and 3.2 remain valid, if RP is replaced by IM and if σ and τ are permuted.

4. Equivalence

Let the algebras A_σ and A_τ be defined as in Section 2. Again these algebras are assumed to be surjective but as a difference with Section 3 they are not assumed to be total and error-free.

The study of equivalence of σ and τ may then be carried out in a way similar to the one of Section 3.

More precisely the correctness conditions of Section 3 have to be modified as follows:

- each expression of the form

  $$Is.\rho\ (d) = \underline{true}$$

  has to be replaced by

  $(Is.\rho\ (d) = \underline{true})$ or $(d = \omega)$ or $(d = \Omega)$;

- each expression of the form

  $$Eq.\rho\ (d_1, d_2) = \underline{true}$$

  has to be replaced by

  $(Eq.\rho\ (d_1, d_2) = \underline{true})$ or $(d_1 = d_2 = \omega)$ or $(d_1 = d_2 = \Omega)$.

The theorem corresponding to the one of Section 3.2 is now a theorem on equivalence rather than on weak equivalence. The proof is similar to that of Section 3.2 but:

- the algebras A_σ and A_τ (rather than the subalgebras A'_σ and A'_τ) have to be shown isomorphic; hence it is necessary to consider the sets $\underline{C}_\rho$ and the operations F_{op} (rather than $\underline{C}'_\rho$ and F'_{op});

- $RP_{op} : \underline{C}_\tau \to \underline{C}_\sigma$ is defined by its values:

  RP_{op} ([d]) = [RP (d)] for all terms d of type τ with $Is.\tau(t) = \underline{true}$

  $RP_{op}\ (ERROR_\tau) = ERROR_\sigma$

  $RP_{op}\ (UNDEFINED_\tau) = UNDEFINED_\sigma$

5. The case of non-surjective specification sets

The case of algebras which are not necessarily surjective may be treated in a similar way; essentially it suffices to treat separately the case of non-accessible elements of the carrier set. Details are omitted here because of the lack of practical interest of this case: generally one will be interested in surjective algebras only or, alternatively, it is in general easy to transform a specification in such a way that the resulting algebra is surjective; moreover the proof that analgebra is surjective is in general relatively easy.

6. Reducing the number of proofs

The goal of the present Section is to show that the correctness conditions of Section 3.1 partly imply the validity of the verification conditions (i) and (ii) of the data type τ.

6.1. Theorem: Let A_σ, A_τ and RP be defined as in Section 3.1. From the validity of the verification conditions of σ and from the correctness conditions of Section 3.1 one may deduce the validity

(a) of the verification condition (i) of τ (expressing that $Eq.\tau$ is an equivalence relation);

(b) of the verification condition (ii) of τ (expressing that for each external function of τ equivalent arguments lead to equivalent values).

Proof

(a) One has to prove that for all terms t, t_1, t_2, t_3 of type τ:

if $Is.\tau\,(t) = Is.\tau(t_1) = Is.\tau(t_2) = Is.\tau(t_3) =$ <u>true</u>

then

(a) either $Eq.\tau(t_1, t_2)$ = true or $Eq.\tau(t_1, t_2)$ = false;

(b) $Eq.\tau(t, t)$ = true;

(c) $Eq.\tau(t_1, t_2) = Eq.\tau(t_2, t_1)$

(d) if $Eq.\tau(t_1, t_2) = Eq.\tau(t_2, t_3)$ = true
then $Eq.\tau(t_1, t_3)$ = true

These four properties directly result from the correctness condition (ii):

for all terms t_1, t_2 of type τ:
if $Is.\tau(t_1) = Is.\tau(t_2)$ = true
then $Eq.\tau(t_1, t_2) = Eq.\sigma(RP(d_1), RP(d_2))$

and from the fact that the verification condition (i) of σ hold (i. e. from the fact that $Eq.\sigma$ is an equivalence relation).

(b) One has to prove that for each external function of τ, say

$Im.F : \underline{\tau}_1 \times \ldots \times \underline{\tau}_n \rightarrow \underline{\tau}_{n+1}$, $n \geq o$

one has:

for all terms d_i, e_i of type τ_i, $1 \leq i \leq n$:
if $Is.\tau_i(d_i) = Is.\tau_i(e_i)$ = true for all i, $1 \leq i \leq n$ (A)
and if $Eq.\tau_i(d_i, e_i)$ = true for all i, $1 \leq i \leq n$ (B)
then $Eq.\tau_{n+1}(Im.F(d_1,\ldots,d_n), Im.F(e_1,\ldots e_n))$ = true (C)

Let for all i, $1 \leq i \leq n$, d_i, e_i be terms satisfying (A) and (B) and let us prove (C).

Let us first consider the case $\tau_{n+1} = \tau$. The correctness condition (iii) then leads to

$$\begin{aligned} & Eq.\sigma(F(d'_1,\ldots d'_n), RP\ (Im.F(d_1,\ldots,d_n))) \\ & = Eq.\sigma\ (F(e'_1,\ldots,e'_n), RP\ (Im.F(e_1,\ldots,e_n))) \\ & = \underline{true} \end{aligned} \tag{D}$$

where for each i, $1 \leq i \leq n$

$$d'_i = \begin{cases} RP(d_i) & \text{if } \tau_i = \tau \\ d_i & \text{otherwise} \end{cases}$$

and e'_i defined similarly. Hence, for all i, $1 \leq i \leq n$, with $\tau_i = \tau$:

$$Eq.\sigma(d'_i, e'_i)$$

$$= Eq.\sigma(RP(d_i), RP(e_i))$$

$$= Eq.\tau(d_i, e_i)$$

by correctness condition (ii)

$$= \underline{true}$$

As a result,

$$Eq.\sigma(d'_i, e'_i) = \underline{true}$$

for all i, $1 \leq i \leq n$. The verification condition (ii) of σ then implies:

$$Eq.\sigma(F(d'_1,...,d'_n), F(e'_1,...,e'_n)) = \underline{true}$$

Together with (D) this leads to

$$Eq.\sigma(RP\ (Im.F\ (d_1,...,d_n)), RP\ (Im.F\ (e_1,...,e_n)))$$

$$= \underline{true}$$

Together with the correctness condition (ii) this in turn leads to

$$Eq.\tau(Im.F\ (d_1,...,d_n), Im.F\ (e_1,...,e_n)) = \underline{true}$$

The case $\tau_{n+1} \neq \tau$ may be treated similarly. ☒

6.2. A practical consequence

Suppose one has a specification of a data type σ the verification conditions of which have been checked, and a specification of a data type τ. Suppose moreover that the algebras A_σ and A_τ are as in Section 3.1. In order to check the verification conditions of τ and to prove that τ is a weak implementation of σ it is sufficient

- to prove the verification condition (iii) of τ;
- to prove the three correctness conditions with the help of a representation function.

In other words the user is dispensed from a proof of the verification conditions (i) and (ii) of τ.

A similar remark holds when using an implementation instead of a representation function. From a practical point of view this may be less helpful: normally σ is a "known" data type the verification conditions of which have been already checked.

6.3. Theorem: as in Section 6.1 but with σ and τ permuted and with the correctness conditions of Section 3.3.

The proof of this theorem is left to the reader.

The theorem leads to a practical consequence similar to (i. e. symmetric with) the one of Section 6.2.

References

[ADJ 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 "Current Trends in Programming Methodology IV" (R. Yeh, ed.), pp. 80-149, Prentice-Hall, 1978

[Ga 79] M.-C. Gaudel, "Algebraic specification of abstract data types", Internal Rep. 360, IRIA, Le Chesnay (Aug. 1979)

[GHM 78] J. V. Guttag, E. Horowitz, D. R. Musser, "Abstract data types and software validation", Comm. ACM 21, 12, pp. 1048-1064 (1978)

[Lo 80a] J. Loeckx, "Proving properties of algorithmic specifications of abstract data types in AFFIRM", AFFIRM-Memo-29-JL, USC-ISI, Marina del Rey, 1980

[Lo 80b] J. Loeckx, "Algorithmic specifications of abstract data types", Internal Report A 80/12, Fachbereich 10, Universität des Saarlandes, Saarbrücken, 1980 (submitted for publication)

[Lo 80c] J. Loeckx, "Implementations of abstract data types and their correctness proofs", Internal Report A 80/13, Fachbereich 10, Universität des Saarlandes, Saarbrücken, 1980

[Su 79] P. A. Subrahmanyam, "On proving the correctness of data type implementations", Internal report, Dept. Comp. Sc., Univ. of Utah, Sept. 1979

[SWL 77] M. Shaw, W. A. Wulf, R. L. London, "Abstraction and verification in ALPHARD: Defining and specifying iteration and generators", CACM 20, 8 (1977)