Correctness of horizontal and vertical composition for implementation concepts based on constructors and abstractors.

Hartmut EHRIG, Hans-Jörg KREOWSKI and Fernando OREJAS

Abstract

A new implementation concept for parameterized specifications based on constructors and abstractors was recently introduced by Orejas, Navarro and Sánchez which includes most of the implementation concepts in the literature for initial as well as loose semantics. In this paper we redefine vertical and different kinds of horizontal compositions using the new concept of semi-pushout defined for a mixture of signature and specification morphisms. The main results concerning correctness of horizontal and vertical composition are based on new correctness requirements for constructors and abstractors.

1 Introduction

Inspired by various approaches in the literature a unifying implementation concept for unparameterized and parameterized specifications was recently presented by Orejas, Navarro and Sánchez [ONS 93, ONS 96]. On one hand it is based on the ideas of constructors and abstractors in the sense of Sannella and Tarlecki [ST88], on the other hand it includes explicitly extension and restriction steps as considered in various other approaches (e.g. [Ehrich 82], [EK 83], [EG 94]). Although the problems of horizontal and vertical composition of unparameterized and parameterized specifications have been discussed in [ONS 93, ONS 96],

Mathematics Subject Classification: 68Q65, 68Q60, 68N05.
only explicit conditions for correctness of vertical composition in the un-parameterized case have been given up to now. It remains to study correctness of horizontal and vertical composition for parameterized specifications in this framework.

The general case of horizontal composition of parameterized specifications allows to have a parameter passing morphism between the parameter of the second and the body of the first specification. Hence actualization is an essential part of horizontal composition, which has been studied in our paper [EKO 95]. In this paper we study the general case of horizontal and vertical composition. The main results of this paper show under which conditions for constructors and abstractors we have correctness of horizontal and vertical composition of implementations.

The constructions for horizontal and vertical composition of parameterized specifications are based on a new concept, called semi-pushouts defined for a mixture of signature and specification morphisms. The general case of horizontal composition can be obtained as a combination of direct horizontal composition, where the given parameterized specifications are directly composable, and actualized implementation, where a given implementation is actualized according to a given parameter passing morphism. The constructions for vertical and horizontal composition are given in section 2 of this paper, where we assume to have an implementation concept based on an institution in the sense of [GB 84], which associates to each specifications morphism a constructor and for each signature morphism an abstractor in the sense of [ONS 96].

In section 3 of this paper we define axiomatic properties for constructors and abstractors and show under which of these properties we have correctness of vertical composition, direct horizontal composition, special horizontal composition used in [ONS 96], actualized implementations and of general horizontal composition of implementations. Moreover we give some remarks how corresponding results in the literature can be obtained as special cases, especially those in [EK 83] and [EG 94] for initial and final semantics and a standard case for loose semantics.

Finally in section 4 we discuss some remaining conceptual problems with the framework presented in sections 2 and 3 and several proposals how to solve these problems. Especially it is open to analyse the implementation concepts and results known from the literature in a systematic way, to find out which are the corresponding constructors and
Abstractors and to analyse how far the axiomatic properties given in section 3 are satisfied in each of these cases.

For a more detailed motivation of the general implementations based on these concepts we refer to [ST 88, ONS 93, ONS 96].

2 Horizonal and vertical composition of implementations

In this section we introduce a general implementation concept based on abstractors and abstractors in the sense of [ONS 93, ONS 96], which are motivated by those in [ST 88]. With respect to this general notion we define the implementation of parameterized specification as well as different kinds of horizontal and vertical composition.

2.1 Implementations

In 1972 Hoare [Hoare 72] presented the first notion of data type implementation in the literature. This was the beginning of a long series of papers dealing with the same concept (see [ONS 96]). There are several reasons for such a number of approaches: in some papers the framework studied is different ("loose" vs "initial" specifications, parameterized vs non-parameterized specifications, partial vs total data types, etc.), in other papers the aim is different (some approaches would stress only "semantic" aspects of implementation while others would focus on "syntactic" or "proof-theoretic" aspects); nevertheless, the underlying intuition is often the same.

Given specifications $SP_1 = (\Sigma_1, E_1)$ and $SP_2 = (\Sigma_2, E_2)$, where $E_1$ and $E_2$ are sets of formulae over any suitable institution (i.e. not necessarily equations), we may consider that implementing the data type specified by $SP_1$ by the data type specified by $SP_2$ consists in defining the operations (and the data sorts) in $\Sigma_1$ in terms of the operations (and data sorts) from $\Sigma_2$, in such a way that the enriched $SP_2$-models "behave" like the $SP_1$-models. In this sense, syntactically, an implementation would be an enrichment together with a mapping (technically, a signature morphism) relating the sorts and operations from $\Sigma_1$ with the sorts and operations from the enriched $\Sigma_2$ signature and, semantically, it would be a construction (associated to the enrichment) together with
some kind of abstraction (associated to the given signature morphism) that relates the models of $SP_1$ with the enriched $SP_2$-models.

For example, suppose that we want to implement (finite) sets of integers by sequences of integers. The implemented (abstract) specification is:

$$SP_1 = \text{INTEGER + BOOLEAN +}$$

**sorts** set

**operations**

- $\emptyset : \rightarrow \text{set}$
- $\text{add} : \text{set} \times \text{int} \rightarrow \text{set}$
- $\text{is-in} : \text{int} \times \text{set} \rightarrow \text{bool}$

**var**

- $S : \text{set}$
- $n, n' : \text{int}$

**equations**

- $\text{add} (\text{add} (S, n), n) = \text{add} (S, n)$
- $\text{add} (\text{add} (S, n), n') = \text{add} (\text{add} (S, n'), n)$
- $n \text{ is-in } \emptyset = \text{false}$
- $n \text{ is-in } \text{add} (S, n') = (n \text{eq} n')$ or $n \text{ is-in } S$

and the specification of the “implementing” (concrete) data type is:

$$SP_2 = \text{INTEGER +}$$

**sorts** seq

**operations**

- $e - s : \rightarrow \text{seq}$
- $\text{app} : \text{seq} \times \text{int} \rightarrow \text{seq}$
- $\text{head} : \text{seq} \rightarrow \text{int}$
- $\text{tail} : \text{seq} \rightarrow \text{seq}$

**var**

- $S : \text{seq}$
- $n : \text{int}$

**equations**

- $\text{tail} (\text{app} (S, n)) = S$
- $\text{tail} (e - s) = e - s$
- $\text{head} (\text{app} (S, n)) = n$

Now, if we decide to represent the sets of integers by sequences without repetitions then, the first step would consist in an enrichment, where all the set operations are defined in terms of the sequence operations in an adequate manner. For example:
The second step would consist in relating the sorts and operations (to be implemented) defined in $SP1$ with the sorts and operations (in the implementation) defined in $IMPL$. As said above, this can be done by means of a signature morphism mapping the sort set into the sort $seq$ and the operations $\emptyset$, add and is-in from $SP1$ into the operations $\emptyset$, add and is-in from $IMPL$.

Semantically, the result of this enrichment would be an $IMPL$-algebra $A$ not satisfying some of the axioms in $SP1$. In particular, $A$ does not satisfy the equation

$$add(add(S, n), n') = add(add(S, n'), n)$$

The reason is that, in general, a set may be represented by several different sequences. For instance, the set $\{1, 2\}$ is represented in this algebra by two sequences: $<1, 2>$ and $<2, 1>$. Nevertheless, $A$ "behave" like the algebra of sets, in the sense that the evaluation of any $\Sigma(SP1)$-term $t$ of integer or boolean sort (and its corresponding translation through the given signature morphism) yields the same value in both algebras. $\Sigma(SP1)$ is assumed to denote the signature of the specification $SP1$.

These ideas can be easily generalized to deal with parameterized specifications. In particular, if $h : SP \to SP1$ and $h' : SP \to SP1'$ are parameterized specifications (for simplicity, let us assume that $h$ and $h'$ are inclusions, i.e. if $SP = (\Sigma, E)$, $SP1 = (\Sigma1, E1)$ and $SP2 = (\Sigma2, E2)$ then $\Sigma \subseteq \Sigma i$ and $\Sigma \subseteq Ei$, $i = 1, 2$), we may also consider that implementing the parameterized data type specified by $h$ by the parameterized data
type specified by \( h' \) consists in defining the operations and the data sorts in \( \Sigma_1 \) (or, rather, \( \Sigma_1 - \Sigma \)) in terms of the operations and data sorts from \( \Sigma_2 \), in such a way that for every \( SP \)-model \( A \), the enrichment applied to \( K_{h'}(A) \) "behaves" like \( K_h(A) \), where \( K_h \) and \( K_{h'} \) are, respectively, the meaning of the parameterizations \( h \) and \( h' \). Therefore, again, syntactically an implementation would be an enrichment together with a signature morphism relating the sorts and operations from \( \Sigma_1 \) with the sorts and operations from the enriched \( \Sigma_2 \) signature and, semantically, it would also be a construction (associated to the enrichment) together with some kind of abstraction (associated to the given signature morphism) that relates the models of \( K_h(SP) \) with the enriched \( K_{h'}(SP) \) models.

### 2.2 General assumptions

Given an institution in the sense of [GB 84] we have a category of signatures and signature morphisms as well as a category of specifications and specification morphisms and for each specifications \( SP \) a class \( \text{Mod}(SP) \) of models over \( SP \).

Moreover we assume to have an implementation concept \( IC \) defined by the property that we have for each specification morphism \( m : SP_1 \rightarrow SP_2 \) a constructor \( K_{IC}(m) : \text{Mod}(SP_1) \rightarrow \mathcal{P}(\text{Mod}(SP_2)), \) short \( K_m \), and for each signature morphism \( f : \Sigma(SP_1) \rightarrow \Sigma(SP_2) \) between specifications, short \( f : SP_1 \leftrightarrow SP_2 \), an abstractor \( \alpha_{IC}(f) : \text{Mod}(SP_1) \rightarrow \mathcal{P}(\text{Mod}(SP_1)), \) short \( \alpha_f \), in the sense of [ONS 93, ONS 95].

### 2.3 Definition (implementation of parameterized specifications)

Given parameterized specifications \( h : SP \rightarrow SP_1 \) and \( h' : SP \rightarrow SP_1' \), where \( h \) and \( h' \) are specification morphisms with same formal parameter specifications \( SP \), an implementation \( I = (m, f) \) of \( h \) by \( h' \) is given by a specification morphism \( m : SP_1' \rightarrow SP_2 \) and a signature morphism \( f : SP_1 \leftrightarrow SP_2 \) with same target specifications \( SP_2 \) s.t. \( f \circ h = m \circ h' \) (as signature morphisms).
The implementation \( I = (m, f) \) of \( h \) by \( h' \) is called correct, if for all \( A \in \text{Mod}(Sp) \) and all \( A2 \in K_{h' \circ m}(A) \) there is \( A1 \in K_h(A) \) s.t. \( U_f(A2) \in \alpha_f(A1) \).

Remark. \( U_f : \text{Mod}(\Sigma(SP2)) \rightarrow \text{Mod}(\Sigma(SP1)) \) is the forgetful functor corresponding to the signature morphism \( f : SP1 \leftrightarrow SP2 \) and \( \Sigma(SP_i) \) is the signature part of \( SP1 \) for \( i = 1,2 \).

2.4 Special cases of implementation concepts

For each choice of constructors and abstractors and by specialization of the components of an implementation we obtain a specific implementation concept. Reachability in the sense of [EK 83] or [EG 94] leads to an abstractor \( \alpha \) defined by

\[
B1' \in \alpha_r(B1) \Leftrightarrow \text{REACH}_{s1}(B1') = B1
\]

where \( \text{REACH}_{s1}(B1') \) is the intersection of all submodels \( B' \) of \( B1' \) with \( U_{s1}(B') = U_{s1}(B1') \). Standard abstraction \( \alpha_s \) is defined by

\[
B1' \in \alpha_s(B1) \Leftrightarrow \exists \text{ morphism } m : B1' \rightarrow B1.
\]

For other notions of abstractors we refer to [ONS 93, ONS 96], where especially behavioural abstraction is discussed as an important example.

If all constructors \( K_s \) for \( s : SP1 \rightarrow SP2 \) are free constructions \( F_s : \text{Mod}(SP1) \rightarrow \text{Mod}(SP2) \) and \( \alpha \) is the reachability abstractor we obtain implementation of parameterized specifications with \( IR \)-semantics in the sense of [EK 83]. If in addition the extension \( \epsilon \) is the identity we obtain \( R \)-implementations and without restriction refinements of parameterized specifications in the sense of [EG 94], where restriction is given by a specification morphism. If, on the other hand, we keep general monomorphic constructors and general abstractors, but have only identical restrictions, then we obtain constructor and abstractor
implementations in the sense of Sannella and Tarlecki [ST 88]. For other special cases we refer to [ONS 93, ONS 96].

In order to define vertical and horizontal composition of implementations in an elegant way we need the notion of semi-pushouts, defined for a mixture of signature and specification morphisms.

### 2.5 Definition (semi-pushout)

Given a signature morphism $f : SP \rightarrow SP1$ and a specification morphism $g : SP \rightarrow SP2$ a specification $SP3$ together with a signature morphism $f' : SP2 \rightarrow SP3$ and a specification morphism $g' : SP1 \rightarrow SP3$ is called semi-pushout of $f$ and $g$ if:

1. $g' \circ f = f' \circ g$ as signature morphisms.
2. For all $h1 : SP1 \rightarrow SP4$ and $h2 : SP2 \rightarrow SP4$, with $h1 \circ f = h2 \circ g$, there is a unique $h : SP3 \rightarrow SP4$ s.t. $h \circ g' = h1$ and $h \circ f' = h2$, where $h1$ and $h$ are specification morphisms and $h2$ is a signature morphism.

![Figure 1: Semi-Pushout](image)

### 2.6 Fact (semi-pushout)

1. The semi-pushout of $f : SP \rightarrow SP1$ and $g : SP \rightarrow SP2$ can be constructed as the pushout $\Sigma3$ of $f$ and $g$ in the category of signature morphisms, where $SP3 = (\Sigma3, E3)$ with $E3 = g'\#(E1)$, i.e. the translated axioms of $E1$ for $SP2 = (\Sigma1, E1)$. 

2. Semi-pushout objects are unique up to isomorphism in the category of specification morphisms.

3. Horizontal and vertical composition of semi-pushouts are semi-pushouts.

4. If \( f \) is a specification morphism the pushout of \( f \) and \( g \) is in general different from the semi-pushout of \( f \) and \( g \).

Proof.

1. By definition of \( SP3 \) the signature morphism \( g' \) in figure 1 becomes a specification morphism. Moreover, given \( h1 \) and \( h2 \), as in 2.5.2, there is a unique signature morphism \( h : SP3 \rightarrow SP4 \), where 
   \[ h\#(E3) = h\#(g'\#(E1)) = (h \circ g')\#(E1) = h1\#(E1) \]
   is derivable from \( E4 \) because \( h1 \) is a specification morphism. Hence, also \( h \) is a specification morphism.

2. Follows in the same way as uniqueness of pushouts using that \( h \) in figure 1 is a specification morphism.

3. Follows as usual from the universal properties of the construction in part 1 which is a characterization due to part 2.

4. If \( f \) is specification morphism the specification pushout is given by

   \[ SP3' = (\Sigma3, g'\#(E1) \cup f'\#(E2)) \]

   which is in general different from the semi-pushout
   \[ SP3 = (\Sigma3, g'\#(E1)) \].

2.7 Definition (vertical composition of implementations)

Given implementations \( I1 = (m1, f1) \) of \( h1 \) by \( h2 \) and \( I2 = (m2, f2) \) of \( h2 \) by \( h3 \) for \( h : SP \rightarrow SPi(i = 1, 2, 3) \) the vertical composition \( I3 = (m3, f3) \) of \( I1 \) and \( I2 \), written \( I3 = I2*I1 \), is given by \( m3 = m1' \circ m2 \) and \( f3 = f2' \circ f1 \) where (3) in figure 2 is the semi-pushout of \( m1 \) and \( f2 \). \( I3 \) is an implementation of \( h1 \) by \( h3 \).
Remark. In [ONS 96] the vertical composition is defined by a pushout \( \Sigma 6 \) of \( m1 \) and \( f2 \) considered as signature morphisms which causes the problem making \( m3 \) a specification morphism.

Before we define the general horizontal composition we consider the special cases of direct horizontal composition and actualized implementations which allow to obtain the general case by combination of both constructions.

2.8 Definition (direct horizontal composition of implementations)

Given implementations \( I1 = (m1, f1) \) of \( h1 \) by \( h1' \) and \( I2 = (m2, f2) \) of \( h2 \) by \( h2' \) as shown in figure 3, where \( h1 \) and \( h2 \) are directly composable. Then the direct horizontal composition \( I3 = (m3, f3) \) of \( I1 \) and \( I2 \), written \( I3 = f2 \circ I1 \), is an implementation of \( h3 \) by \( h3' \) given by
Correctness of horizontal and vertical... 375

\[ h_3 = h_2 \circ h_1, \quad h_3' = h_4 \circ m_1 \circ h_1', \quad m_3, \quad f_3 = f_5 \circ f_2 \] as defined in figure 3 where (3) and (4) are constructed as semi-pushouts.

![Figure 3: Direct horizontal composition of implementations](image)

2.9 Definition (actualized implementation)

Given an implementation \( I_1 = (m_1, f_1) \) of \( h_1 \) by \( h_1' \) and a specification morphism \( g \), called parameter passing morphism, the actualized implementation \( I_2 = (m_2, f_2) \) of \( I_1 \) via \( g \) is an implementation of \( h_2 \) by \( h_2' \), written \( I_2 = g \# (I_1) \), \( h_2 = g \# (h_1) \), \( h_2' = g \# (h_1') \), where \( h_2, h_2', \) and \( m_2 \) are constructed via the pushouts \( SF_3, SF_3' \) on the back, left and front square of figure 4 and \( f_2 \) is the induced signature morphism.

![Figure 4: Actualized implementation](image)

2.10 Definition (horizontal composition of implementations)

Given implementations \( I_1 = (m_1, f_1) \) of \( h_1 \) and \( h_1' \) and \( I_2^* = (m_2^*, f_2^*) \) of \( h_2^* \) by \( h_2'^* \) and a specification morphism \( g \) as shown in figure 5,
i.e. the actualization $g\#(h2^*) = h2$ of $h2^*$ is composable with $h1$. Then the horizontal composition $I3 = (m3, f3)$ of $I1$ and $I2^*$ via $g$, written $I3 = I2^* \circ_{g} I1$, is an implementation of $h3$ by $h3'$ given by $h3 = h2 \circ h1$, $h3' = h4 \circ m1 \circ h1'$, $f3 = f5 \circ f2$ as shown in figure 5 constructed as in figure 4 and then figure 3.

![Figure 5: Horizontal composition of implementations](image)

**Remark.** The horizontal composition considered in [ONS 96] is the special case of horizontal composition above where only an implementation $I1$ of $h1$ by $h1'$, $h2^*$ and $g$ but no implementation $I2^*$ of $h2^*$ by $h2'^*$ are given. This means that in figure 4 we have $h2^* = h2'^*$, $h2 = h2'$ and $m2^*$, $m2, m3$, and $f5$ are identities.

### 2.11 Fact (composability of horizontal composition)

The horizontal composition $I3 = I2^* \circ_{g} I1$ of $I1$ and $I2^*$ via $g$ is equal to the direct horizontal composition $I3 = I2 \circ I1$ of $I1$ with the actualized implementation $I2 = g\#(I2^*)$ of $I2^*$ via $g$, i.e.

$$I2^* \circ_{g} I1 = g\#(I2^*) \circ I1$$
3 Correctness results

In order to prove correctness of horizontal and vertical composition of implementations we have to formulate several axiomatic properties for the constructor and abstractor of the implementation concept which are defined below. In addition to the general assumptions in 2.2 we assume that the given institution has amalgamation in the sense of [EG 94], i.e. the model functor Mod transforms pushouts into pullbacks.

3.1 Definition (axiomatic properties of constructors)

1. The constructor $K$ is functorial, if we have

$$Kf_2 \circ f_1 = Kf_2 \circ Kf_1$$

for all specification morphisms $f_1 : SP \to SP2$ and $f_2 : SP2 \to SP3$, where $Kf_2 \circ Kf_1(A1) = \cup\{Kf_2(A2)/A2 \in Kf_1(A1)\} \subseteq \text{Mod}(SP3)$.

2. The constructor $K$ is persistent, if for each $f_1 : SP1 \to SP2$, $A1 \in \text{Mod}(SP1)$, $A2 \in Kf_1(A1)$ we have $U_{f1}(A2) = A1$.

3. The constructor $K$ is compatible with amalgamation if $K$ is persistent and for all pushout

$$\begin{array}{ccc}
SP & \xrightarrow{h_1} & SP1 \\
h_2 \downarrow & (PO) & \downarrow g_1 \\
SP2 & \xrightarrow{g_2} & SP3
\end{array}$$

and all $A \in \text{Mod}(SP)$, $A1 \in K_{h1}(A)$, $A2 \in \text{Mod}(SP2)$ with $U_{g2}(A2) = A$ we have

$$A1 +_A A2 \in K_{g2}(A2).$$

If moreover we have

$$K_{g2}(A2) = \{A1 +_A A2/A1 \in K_{h1}(A) \text{ and } A = U_{h2}(A2)\}$$

then $K$ is called strongly compatible with amalgamation.
4. The constructor $K$ has the extension property if for all pushouts as given above we have

$$K_{h_1} \circ U_{h_2} = U_{g_1} \circ K_{g_2}$$

3.2 Fact (strong compatibility of constructors)

A constructor is strongly compatible with amalgamation if and only if it has the extension property and is compatible with amalgamation.

**Proof.** If $K$ is strongly compatible then it is also compatible and $A_1 \in K_{h_1} \circ U_{h_2}(A_2) = K_{h_1}(A)$ implies $A_1 +_A A_2 \in K_{g_2}(A_2)$ and hence $A_1 \in U_{g_1} \circ K_{g_2}(A_2)$. Conversely, $A_1 \in U_{g_1} \circ K_{g_2}(A_2)$ implies $A_3 \in K_{g_2}(A_2)$ with $A_1 = U_{g_1}(A_3)$ and $U_{g_2}(A_3) = A_2$ by persistency of $K$. Hence we have $A_3 = A_1 +_A A_2 \in K_{g_2}(A_2)$ with $A = U_{h_2}(A_2)$ which implies $A_1 \in K_{h_1}(A) = K_{h_1} \circ U_{h_2}(A_2)$ by strong compatibility of $K$. Hence we have $K_{h_1} \circ U_{h_2} = U_{g_1} \circ K_{g_2}$. Conversely, let $K$ be a compatible with extension property. It remains to show that $A_3 = A_1 +_A A_2 \in K_{g_2}(A_2)$ implies $A_1 \in K_{h_1}(A)$. But $A_3 \in K_{g_2}(A_2)$ implies $A_1 \in U_{g_1} \circ K_{g_2}(A_2) = K_{h_1} \circ U_{h_2}(A_2) = K_{h_1}(A)$. 

3.3 Definition (axiomatic properties of abstractors)

1. An abstractor $\alpha$ is compatible with amalgamation if for all diagrams (1) and (2)

$$SP \xrightarrow{h_1} SP_1 \xrightarrow{f_1} SP_2'$$

$$SP_2 \xrightarrow{h_2} SP_3 \xrightarrow{f_2} SP_4$$

where (1) is a pushout and (2) a signature $PO$ (not necessarily semipushout as given in figure 4) then for all $A_1 \in \text{Mod}(SP_1)$, $A_2 \in \text{Mod}(SP_2)$ with $U_{g}(A_2) = A$ and all $A_1' \in \alpha_{f_1}(A_1)$ with $U_{h_1}(A_1') = A = U_{h_1}(A_1)$ we have

$$(A_1' +_A A_2) \in \alpha_{f_2}(A_1' +_A A_2)$$
Remark. Reachability abstractors and standard abstractors are compatible with amalgamation.

2. An abstractor $\alpha$ is compositional if for all signature morphisms $f_1 : SP_1 \rightarrow SP_2$ and $f_2 : SP_2 \rightarrow SP_3$ and all $A_i \in \text{Mod}(SP_i)$ $i = 1, 2, 3$ with

$$U_{f_1}(A_2) \in \alpha_{f_1}(A_1), \ U_{f_2}(A_3) \in \alpha_{f_2}(A_2)$$

we also have $U_{f_2 \circ f_1}(A_3) \in \alpha_{f_2 \circ f_1}(A_1)$.

Remark. This condition is similar to C1 in [ONS 96], especially satisfied for $\alpha$ defined by reachability and free constructions $K$ provided that $A_1$ and $A_3$ are freely generated.

3. An abstractor $\alpha$ has amalgamation complements w.r.t. a constructor $K$ if for all semi-pushouts

$$SP \xrightarrow{g} SP_2$$

$$\alpha \notin (S - PO) \xrightarrow{\psi' \psi'} SP_3$$

and all $A \in \text{Mod}(SP), A_i \in \text{Mod}(SP_i) i = 1, 3$ with $A_3 \in K_{\psi'}(A_1)$ and $U_f(A_1) \in \alpha_f(A)$ there is $A_2 \in \text{Mod}(SP_2)$ with $A_2 \in K_\psi(A)$ and $U_{f'}(A_3) \in \alpha_{f'}(A_2)$.

Remark. If $f, f'$ are specification morphisms and $\alpha = \text{equality}$ then the condition is satisfied for translation constructors.

3.4 Theorem (correctness of vertical composition)

Given correct implementations $I_1$ and $I_2$ the vertical composition $I_3 = I_2 * I_1$ is correct provided that

1. The constructor $K$ is functorial.
2. The abstractor $\alpha$ is compositional.
3. The abstractor $\alpha$ has amalgamation complements w.r.t. $K$.

Proof. Given correct $I_1$ and $I_2$, and $I_3$ as defined in figure 2 we have to show for all $A \in \text{Mod}(SP)$ and $A_6 \in K_{\alpha_3 \circ \alpha_3}(A)$ the existence of
$A_1 \in K_{h_1}(A)$ with $U_{f_3}(A6) \in \alpha_{f_3}(A1)$. For $A6 \in K_{m_3 \circ h_2}(A)$ we have by functoriality of $K$ some $A5 \in K_{m_2 \circ h_2}(A)$ with $A6 \in K_{m_1^1}(A5)$. Correctness of $I2$ implies that there is $A2 \in K_{h_2}(A)$ with $U_{f_2}(A5) \in \alpha_{f_2}(A2)$. Since $\alpha$ has amalgamation complements and (3) in figure 2 is semi-pushout we have some $A4 \in \text{Mod}(SP4)$ with $A4 \in K_{m_1}(A2)$ and $U_{f_2}(A6) \in \alpha_{f_2}(A4)$. Now correctness of $I1$ implies that there is $A1 \in K_{h_1}(A)$ with $U_{f_1}(A4) \in \alpha_{f_1}(A4) \in \alpha_{f_1}(A1)$. Finally compositionality of $\alpha$ applied to $f^3 = f_2' \circ f_1$ implies $U_{f_3}(A6) \in \alpha_{f_3}(A1)$. Since we have already $A1 \in K_{h_1}(A)$ this implies correctness of $I3$.

**Remark.** The correctness of vertical composition of refinements and $R-$implementations in [EG 94] restricted to identical parameter specifications is a special case of this result.

### 3.5 Theorem (correctness of direct horizontal composition)

Given correct implementations $I1$ and $I2$ the direct horizontal composition $I3 = I2 \circ I1$ is correct provided that

1. The constructor $K$ is functorial.

2. The abstractor $\alpha$ is compositional.

3. The abstractor $\alpha$ has amalgamation complements w.r.t. $K$.

**Proof.** Given correct $I1$ and $I2$, and $I3$ as defined in figure 3 we have to show for all $A1 \in \text{Mod}(SP1)$ and $A7 \in K_{m_3 \circ h_3}(A1)$ the existence of $A3 \in K_{h_3}(A1)$ with $U_{f_3}(A7) \in \alpha_{f_3}(A3)$. For $A1 \in \text{Mod}(SP1)$ and $A7 \in K_{m_3 \circ h_3}(A1)$ we have by functoriality of $K$ some $A4 \in K_{m_1 \circ h_2}(A1)$ with $A7 \in K_{m_3 \circ n(A4)}$. Correctness of $I1$ implies the existence of $A2 \in K_{h_2}(A1)$ with $U_{f_1}(A4) \in \alpha_{f_1}(A2)$. Since $\alpha$ has amalgamation complements and (3) $\cup$ (4) in figure 3 is a semi-pushout we have some $A5 \in \text{Mod}(SP5)$ with $A5 \in K_{m_2 \circ h_2}(A2)$ and $U_{f_5}(A7) \in \alpha_{f_5}(A5)$. Now correctness of $I2$ implies the existence of $A3 \in K_{h_2}(A2)$ with $U_{f_2}(A5) \in \alpha_{f_2}(A3)$, and compositionality of $\alpha$ applied to $f^3 = f_5 \circ f_2$ implies $U_{f_3}(A7) \in \alpha_{f_3}(A3)$. Finally $A2 \in K_{h_2}(A1)$ and $A3 \in K_{h_2}(A2)$ imply $A3 \in K_{h_3}(A1)$ by functoriality of $K$ and $h_3 = h_2 \circ h_1$, which implies correctness of $I3$.

■
3.6 Corollary (correctness of special horizontal composition)

The correctness of special horizontal composition considered as property P2 in [ONS 96] (see Remark below 2.10) holds provided that the constructor K is functorial and the abstracter a has amalgamation complements w.r.t. K.

Proof. Special case of the proof of theorem 3.5 where \( h_2 = h_2' \) and \( m_2, m_3, f_2 \) and \( f_5 \) are identities. Hence we only have to use the existence of amalgamation complements applied to semi-pushout (3) in figure 2 but do not need compositionality of \( a \).

3.7 Theorem (correctness of actualized implementations)

Given a correct implementation \( I1 \) of \( h_1 \) by \( h_1' \) and a parameter passing morphism \( g \) the actualized implementation \( I2 = g\#(I1) \) of \( h_2 = g\#(h_1) \) by \( h_2' = g\#(h_1') \) is correct provided that

1. The constructor \( K \) is functorial and is strongly compatible with amalgamation, i.e. persistent, compatible with amalgamation and has the extension property.

2. The abstracter \( a \) is compatible with amalgamation.

Proof. Given correct \( I1 \) of \( h_1 \) by \( h_1' \) and the actualized implementation \( I2 \) of \( I1 \) via \( g \) as shown in figure 4 correctness of \( I2 \) requires to show for each \( A2 \in \text{Mod}(SP2) \) and \( A4 \in K_{m_2 \circ h_2'}(A2) \) the existence of \( A3 \in K_{h_2}(A2) \) with \( Uf_2(A4) \in \alpha f_2(A3) \). For \( A2 \) and \( A4 \) as above we define \( A = Ug(A2) \) and \( A2' = Ug_1(A4) \). The extension property of \( K \) applied to the composed pushout of left and front square in figure 4 means \( K_{m_1 \circ h_1'} \cdot Ug = Ug_1' \cdot K_{m_2 \circ h_2'} \). Hence we have

\[
A2' = Ug_1(A4) \in Ug_1' \cdot Ug(A2) = K_{m_1 \circ h_1'} \cdot Ug(A2) = K_{m_1 \circ h_1'}(A).
\]

Now correctness of \( I1 \), implies the existence of \( A1 \in K_{h_1}(A) \) with \( Uf_1(A2') \in \alpha f_1(A1) \). Let \( A3 = A1 + A2 \in \text{Mod}(SP3) \) which is well-defined because \( A1 \in K_{h_1}(A) \) implies \( U_{h_1}(A1) = A \) by persistency of \( K \) and we have already \( Ug(A2) = A \). It remains to show \( A3 \in K_{h_2}(A2) \).
and $U_{f2}(A4) \in \alpha_{f2}(A3)$. The first property follows from compatibility of $K$ with amalgamation using $A3 = A1 +_A A2$ and $A1 \in K_{h1}(A)$. In order to show the second property we apply compatibility of $\alpha$ with amalgamation applied to diagrams (1) and (2) in 3.3.1, where (2) is a signature pushout as a consequence of the construction in figure 4. Let $A1' = U_{f1}(A2')$, then $A1' = U_{f1}(A2') \in \alpha_{f1}(A1)$ and we have $U_{h1}(A1') = U_{h1} \cdot U_{f1} \cdot (A2') = U_{f1 \circ h1}(A2') = U_{h1 \circ h1'}(A2') = A$ using $A2' \in K_{m1 \circ h1'}(A)$ as shown above and persistency of $K$. Hence we have $A1' \in \alpha_{f1}(A1), U_{h1}(A1') = A = U_{h1}(A1)$, and $Ug(A2) = A$ s.t. compatibility of $\alpha$ with amalgamation implies $(A1' +_A A2) \in \alpha_{f2}(A1 +_A A2) = \alpha_{f2}(A3)$ (see 3.3.1). In order to show $U_{f2}(A4) \in \alpha_{f2}(A3)$ it remains to show $U_{f2}(A4) = A1' +_A A2$ and hence by uniqueness of amalgamation we only have to show equality after application of $Ug_1$ and $Ug_2$. In fact we have:

$$Ug_1 \cdot U_{f2}(A4) = Uf_1 \cdot Ug_1(A4) = Uf_1(A2') = A1' = Ug_1(A1' +_A A2)$$

and using persistency of $K$ in the case $A4 \in K_{m2 \circ h2'}(A2)$ we have

$$Ug_2 \cdot U_{f2}(A4) = U_{m2 \circ h2'}(A4) = A2 = Ug_2(A1' +_A A2).$$

**Remarks.**

1. According to 2.4 we obtain implementation of parameterized specifications with $IR$-semantics in the sense of [EK 83] if all constructors are persistent free constructions and $\alpha$ is the reachability abstractor. The assumptions of theorem 3.7 are satisfied, because persistent free constructions which are closed under extension (see e.g. [EM 85]) and reachability is compatible with amalgamation as shown in [EKO 95]. The reason is that if $\text{REACH}_S(A1 +_A A2) = \text{REACH}_S(A1' +_A A2)$ then $\text{REACH}_S(A1) = \text{REACH}_S(A1')$ since REACH commutes with the forgetful functor. Hence theorem 3.7 can be applied leading to $IR-$ correctness of actualized implementations, which is shown explicitly in [EK 83].

2. If in addition in Remark 1 the specification morphism $m$ is the identity we obtain correctness of actualized $R$-implementations
and —without restriction— refinements of parameterized specifications in a suitable institution, which is explicitly shown in [EG 94]. The approach in [EG 94] also includes final semantics.

3. If all constructors are translations and abstraction is standard abstractions (see 2.4) then again the assumptions of theorem 3.7 are satisfied. In fact, it is easy to show that translations are (persistent) constructors which are closed under extension, i.e.

\[ \text{translates}^1(P') = \{ P' +_p B1/B1 \in \text{translates}^1(P) \}. \]

Moreover standard abstraction is compatible with amalgamation, because \( B1^* \in \alpha_a(B1) \) according to 2.4 means existence of \( m : B1^* \rightarrow B1 \) which implies

\[ m' = P' +_p m : P' +_p B1^* \rightarrow P' +_p B1 \]

and hence \( B1'' \in \alpha'(B1') \). This means that theorem 3.7 can be applied in this case leading to correct actualized implementations in the framework of loose semantics.

3.8 Theorem (correctness of horizontal composition of implementations)

Horizontal composition of correct implementations is correct provided that the constructor \( K \) and the abstractor \( \alpha \) satisfy all the properties stated in 3.1 and 3.2.

\[ \blacksquare \]

Proof. Direct consequence of fact 2.11, theorem 3.5 and 3.7.

Remark. It remains to check how far the properties stated in 3.1 and 3.2 are satisfied for various cases of implementations of parameterized specifications studied in the literature. Some examples have been discussed already in the remarks of 3.3, 3.4 and 3.7.
4 Open problems

Although we have shown, for some of the correctness results in section 3, how to obtain several results in the literature as special cases, it remains open to analyse all the results in the literature in a systematic way to see how far they can be obtained as special cases of the general approach in this paper. Moreover, there are still some conceptual problems which are discussed in 4.1-4.3 below.

4.1 Conceptual problem with semi-PO for composition

According to Def. 2.5 (semi-pushout) the specifications SP6 in the semi-PO of (3) in figure 2 includes only translated axioms from SP5 (relevant for I2) but not from SP4 (relevant for I1). Similarly SP7 in figure 3 lacks translated axioms from SP5 (relevant for I2). This problem can be solved by redefinition of semi-PO’s in Def. 2.5 requiring f’ and h2 in figure 1 to be specification morphisms. This can be achieved by defining E3 = g’#(E1) ∪ f’#(E2). The effect of this new definition is that f2’ in figure 2 (vertical composition), and f4, f5 in figure 3 and 5 (direct and general horizontal composition) become specification morphisms. Moreover the translated axioms of SP2 in figure 2 are derivable from those of SP6, because f2’ o m1 is specification morphism, although f2 is only signature-morphism.

4.2 Syntactical representation of composite implementations

Even with a redefinition of Semi-PO as above it might be too restrictive to require that the vertical composition I2 o I1 and the (direct) horizontal composition I2 o l1 have an explicit syntactical representation. This could be avoided by the following semantical redefinition of Def. 2.3 (in the spirit of Sannella-Tarlecki [ST 88]) which is partly included in our paper [EKO 95].

The parameterized specifications h, h’ are given by the constructors K_h : Mod(SP) → P(Mod(Σ1)) and K_h’ : Mod(SP) → P(Mod(Σ1’)), based on h : Σ(SP) → Σ1 and h’ : Σ(SP) → Σ’ and the implementation uses a constructor K_m : Mod(Σ’) → P(Mod(Σ2)) based on m : Σ1’ → Σ2.
Correctness: \( \forall A \in \text{Mod}(SP) \forall A2 \in K_m \circ K_h(A) \)
\[ \exists A1 \in K_h(A) \text{ s.t. } U_f(A2) \in \alpha_f(A1) \quad (\text{or } \alpha_h(A1)) \]

In this case all constructions and results are essentially the same. But it might be too restrictive to require that \( K_m \) is defined on \( \text{Mod}(\Sigma 1') \) and not only on a subclass \( \text{Def}(K_m) \subseteq \text{Mod}(\Sigma 1') \) with image \( (K_h') \subseteq \text{Def}(K_m) \). This additional correctness condition would have to be shown for all constructions of composite implementations and may lead to additional requirements for \( K \) (e.g. closure properties w.r.t. abstractors). In both cases we have to require an extension property for constructors w.r.t. pushouts and semi-PO.

4.3 Abstractors are based on \( f \) in \((m, f)\)

For reachability and behavioural abstractors it seems to be more suitable to consider \( \alpha_h \) instead of \( \alpha_f \) in Def. 2.3 of implementations. This, however, means that we have to redefine the axiomatic properties of abstractors (Def. 3.3) s.t. theorem 3.4-3.8 are still valid. Hence also reformulation of the proofs of these theorems are necessary. Moreover, it has to be checked for which kind of abstractors and constructors these conditions are satisfied. In [EKO 95] we have assumed already that abstractors can depend on morphisms \( f \) and \( f \) or \( h \).

Acknowledgements

The author would like to thank the anonymous referee for his/her helpful suggestions. This work has been supported by ESPRIT W. G COMPASS II (ref. 6112) and by the Spanish CICYT project COSMOS (ref. TIC95-1016-C02-01)

References


Technical University of Berlin,
Franklinstraße 28/29,
10587 Berlin
e-mail: ehrig@cs.tu-berlin.de
Correctness of horizontal and vertical...