Observer Complete Definitions are Behaviourally Coherent

Michel Bidoit*, Rolf Hennicker**

*Laboratoire Spécification et Vérification, CNRS & Ecole Normale Supérieure de Cachan
61, Avenue du Président Wilson, 94235 Cachan Cedex, FRANCE
**Institut für Informatik, Ludwig-Maximilians-Universität München
Oettingenstr. 67, D-80538 München, GERMANY

Abstract. We consider observational specifications of state-based systems which incorporate the declaration of a distinguished set of observer operations. These observers determine an indistinguishability relation for states which is called "observational equality". An important requirement for the non-observer operations is the compatibility with the observational equality. In the CafeOBJ language (and in extended hidden algebra) this property is called "behavioural coherence".
In this presentation we introduce the notion of an "observer complete definition" and we show that any (non-observer) operation which is defined using this pattern is behaviourally coherent. We also discuss some consequences of this result for relating observational logic and extended hidden algebra semantics and for proving the correctness of observational implementations.

1 Introduction

Observational approaches provide an important means to specify the observable behaviour of software systems in an abstract, implementation independent way. In the last years it has been recognized that observational approaches are particularly suited to describe observational properties of state-based systems since they allow us to abstract from concrete state representations and to consider any two states which cannot be distinguished by observable experiments as "observationally equal". A flexible approach to formalize observable experiments is suggested (in a similar way) in [DF 98], [D 98], [Pad 96] and [HB 98] by splitting the operations of an algebraic signature into a set of "observer" operations (called "behavioural" operations in [DF 98] and [D 98] and "actions" in [Pad 96]) which determine the observational equality of states and into the "other" operations which can be used to manipulate states (but do not contribute to the definition of the observational equality).
Having declared a set of observers one can require that the non-observer operations are compatible with the observational equality, an important property which is called "behavioural coherence" in the CafeOBJ language and in extended hidden algebra (cf. [DF 98] and [D 98]). In observational logic this property is required as a semantic constraint for any model of a specification while in CafeOBJ the behavioural coherence can be explicitly stated as an attribute of an operation. Then, according to the "persistent methodology" of [D 98], one has to prove that indeed the coherence of the respective operations is a consequence of the specification (see [DF 98] and [D 98] for examples of behavioural coherence proofs).
As a main result of the present paper we show that the behavioural coherence of a (non-observer) operation op is already guaranteed if the axioms for op have the form of an "observer complete definition". This kind of definition provides a general pattern for the specification of non-observer operations which is derived as a methodological consequence from the declaration of the observers. It formalizes the idea that a non-observer operation is specified by describing its observable effects by a complete case distinction with respect to the given observers. Thus the concept of an observer complete definition is analogous to the notion of a constructor complete definition for functions over inductive data types.
A useful consequence of our main result is that observational implementation proofs can be simplified whenever the specification to be implemented is observer complete. In this case it is sufficient to show that the implementation satisfies observationally the axioms of the requirements specification because then it is guaranteed that all operations are implemented in a behaviourally coherent way (which is an essential requirement for any reasonable realization).

The paper is organized as follows: After a brief overview of the algebraic notions that are used in this paper we introduce, in Section 3, the basic concepts of the observational approach like the notions of an observational signature, observational equality, behavioural coherence, observational satisfaction and observational specification. In Section 4, the pattern of an observer complete definition is introduced and the main theorem which shows that observer complete definitions are behaviourally coherent is proved. Then, in Section 5, we apply the results to observational implementation proofs. Finally, we end with some concluding remarks in Section 6.

2 Algebraic Preliminaries

We assume the reader to be familiar with the basic notions of algebraic specifications (cf. e.g. [LEW 96]), like the notions of (many-sorted) *signature* $\Sigma = (S, OP)$ (with a set S of *sorts* and a set OP of *operation symbols* op: $s_1,\ldots,s_n \to s$), *total* Σ-*algebra* $A = ((A_s)_{s\in S}, (op^A)_{op\in OP})$, reduct $A|_{\Sigma 0}$ of A to a subsignature $\Sigma 0 \subseteq \Sigma$, Σ-*congruence*, Σ-*term algebra* $T(\Sigma, X)$, *valuation* $\alpha : X \to A$ and *interpretation* $I_\alpha: T(\Sigma, X) \to A$. Throughout this paper we assume that the carrier sets A_s of a Σ-algebra are not empty and that $X = (X_s)_{s\in S}$ is a family of countably infinite sets X_s of variables of sort $s \in S$. The class of all Σ-algebras is denoted by $Alg(\Sigma)$.

3 Observational Concepts

In this section we summarize the underlying observational concepts which are the basis of the study of observer complete definitions and their consequences. The notions that we introduce in the following are based on the ideas of observational logic (cf. [HB 98]) and of extended hidden algebra (cf. [D 98]) which is a generalization of hidden algebra (cf. [GM 97]).
First we introduce the notion of an observational signature which is a standard algebraic signature together with a distinguished set of observable sorts (determining the carrier sets of the observable values) and with a distinguished set of observer operations (determining the experiments that can be used to distinguish non-observable elements, also called "states"). The following definition is a simplified version of the notion of an observational signature in observational logic since we assume here that an observer has only one non-observable argument sort.[1] Thus our notion of observer corresponds exactly to the notion of a "behavioural operation" in extended hidden algebra.

Definition 3.1 *(Observational signature)* Let $\Sigma = (S, OP)$ be a signature, let $S_{Obs} \subseteq S$ be a set of *observable sorts* and let $OP_{Obs} \subseteq OP$ be a set of *observer operations* (briefly called *observers*) such that any $op \in OP_{Obs}$ has a functionality $s_1,\ldots, s_n \to s$ with exactly one non-observable argument sort $s_i \in S\backslash S_{Obs}$. The sort s_i is called *observed sort*. An observer $op: s_1,\ldots, s_n \to s$ is called a *direct observer* of s_i if $s \in S_{Obs}$; otherwise op is an *indirect observer*. The triple $\Sigma_{Obs} = (\Sigma, S_{Obs}, OP_{Obs})$ is called an *observational signature*. ♦

Convention We implicitly assume in the following that whenever we consider an observational signature Σ_{Obs}, then $\Sigma_{Obs} = (\Sigma, S_{Obs}, OP_{Obs})$ with $\Sigma = (S, OP)$.

The above notion of observational signature correponds to the notion of a hidden signature in extended hidden algebra (cf. [D 98]). The only difference is that we do not explicitly introduce here a distinguished visible subsignature (for denoting a predefined data part).

Example 3.2 (1) The following is a simple observational signature for bank accounts with a direct observer "bal" determining the balance of an account and an indirect observer "undo" which is intended to reconstruct the previous state of an account after having performed an action. Besides the constant "new" which denotes the initial state of an account there is an operation "update" subsuming the usual credit and debit operations and an operation "paycharge" which is intended to reduce the balance of an

[1] This presentation can, however, easily be generalized to take into account observers with arbitrary many non-observable argument sorts as considered in observational logic.

account by a constant monthly fee. In this example we use postfix notation for unary operations and infix notation for binary operations.

sorts {account, int}
observable sorts {int}
observers { _.bal: account → int, _.undo: account → account}
operations {new: → account, _.update_ : account, int → account,
_.paycharge: account → account}

(2) The next example is an observational signature for streams. It contains obvious observers "head" and "tail" and a (binary) "merge" operation.

sorts {stream, int}
observable sorts {int}
observers {head: stream → int, tail: stream → stream}
operations {merge: stream, stream → stream} ♦

Any observational signature determines a set of observable contexts which represent those experiments that allow us to distinguish states by the given observers.

Definition 3.3 *(Σ_{Obs}-context)* Let Σ_{Obs} be an observational signature, let $X = (X_s)_{s\in S}$ be the generally assumed family of variable sets and let $Z = (\{z_s\})_{s\in S}$ be a disjoint S-sorted family of singleton sets. For all s, s' ∈ S the set $C(\Sigma_{Obs})_{s\to s'}$ of Σ_{Obs}-*contexts* with "application sort" s and "result sort" s' is inductively defined as follows:
(1) For each s ∈ S, $z_s \in C(\Sigma_{Obs})_{s\to s}$.
(2) For each (op, i) ∈ OP_{Obs} with op: $s_1,\ldots, s_n \to s'$, for each c ∈ $C(\Sigma_{Obs})_{s\to s_i}$ and pairwise disjoint variables $x_1,\ldots, x_n$ (not occurring in c) of sort $s_1,\ldots, s_n$,

$$op(x_1,\ldots, x_{i-1}, c, x_{i+1},\ldots, x_n) \in C(\Sigma_{Obs})_{s\to s'}.$$

The set of all Σ_{Obs}-*contexts* is denoted by $C(\Sigma_{Obs})$. Each context c ∈ $C(\Sigma_{Obs})$ contains, besides variables in X, exactly one occurrence of a "context variable" z_s. The application of such a context to a term t of sort s, denoted by c[t], is the term obtained by substituting the term t for z_s.
An *observable* Σ_{Obs}-*context* is a Σ_{Obs}-context with observable result sort s' ∈ S_{Obs}. We denote by $C(\Sigma_{Obs})_{S_{Obs}}$ the set of observable Σ_{Obs}-contexts. A context c0 is a *subcontext* of a Σ_{Obs}-context c, if c0 is a subterm of c. ♦

Elements which cannot be distinguished by the observable contexts of an observational signature are considered to be observationally equal, formally defined as follows.

Definition 3.4 *(Observational equality)* Let Σ_{Obs} be an observational signature. For any Σ-algebra A ∈ Alg(Σ) the *observational* Σ_{Obs}-*equality* on A is denoted by $\approx_{\Sigma_{Obs},A}$ and defined by:
For all s ∈ S, two elements a, b ∈ A_s are observationally equal w.r.t. Σ_{Obs}, i.e. $a \approx_{\Sigma_{Obs},A} b$, if and only if for all observable contexts c ∈ $C(\Sigma_{Obs})_{S_{Obs}}$ with context variable z_s and for all valuations α, β: $X\cup\{z_s\} \to A$ with α(x) = β(x) if x ∈ X, $\alpha(z_s) = a$, $\beta(z_s) = b$, we have $I_\alpha(c) = I_\beta(c)$. Obviously, if s is an observable sort, then for all a, b ∈ A, $a \approx_{\Sigma_{Obs},A} b$ is equivalent to a = b. ♦

For any Σ-algebra A, $\approx_{\Sigma_{Obs},A}$ is an equivalence relation on A. But it is important to note that for an arbitrary Σ-algebra A there may exist (non-observer) operations which are not compatible with the observational equality $\approx_{\Sigma_{Obs},A}$, i.e. $\approx_{\Sigma_{Obs},A}$ is in general not a Σ-congruence on A. Following the terminology of extended hidden algebra we call an operation "behaviourally coherent" if it is compatible with the observational equality.

Definition 3.5 *(Behavioural coherence)* Let Σ_{Obs} be an observational signature and let A be a Σ-algebra. An operation (op: $s_1,\ldots, s_n \to s$) ∈ OP is called *behaviourally coherent for* A if for all $a_1, b_1 \in A_{s_1},\ldots, a_n, b_n \in A_{s_n}$,

$a_1 \approx_{\Sigma_{Obs},A} b_1 ,\ldots , a_n \approx_{\Sigma_{Obs},A} b_n$ implies $op^A(a_1,\ldots, a_n) \approx_{\Sigma_{Obs},A} op^A(b_1,\ldots, b_n)$. ♦

It is easy to prove that observer operations and operations which have only observable argument sorts are behaviourally coherent for any Σ-algebra A. Hence the interesting cases are those operations which are not observers and have at least one non-observable argument sort (like the operations "update", "paycharge" and "merge" in Example 3.2).

Having given an observational signature Σ_{Obs}, according to our intuition only those algebras should be accepted as realizations for which all operations are behaviourally coherent. This assumption leads to the notion of an observational algebra. In observational logic observational algebras are used to build the category of admissible models of an observational signature.

Definition 3.6 *(Observational algebra)* Let Σ_{Obs} be an observational signature. An *observational* Σ_{Obs}*-algebra* is a Σ-algebra A such that $\approx_{\Sigma_{Obs},A}$ is a Σ-congruence on A (i.e. all operations of Σ are behaviourally coherent for A). The class of all observational Σ_{Obs}-algebras is denoted by $Alg_{Obs}(\Sigma_{Obs})$. ♦

In the next step we define the observational satisfaction of a formula by an algebra whereby the underlying idea is to interpret the equality symbol "=" not by the set-theoretic equality but by the observational equality of elements.
The observational satisfaction relation is defined for arbitrary first-order formulas which are built, as usual, from equations t = r (with terms $t, r \in T(\Sigma, X)$ of the same sort), the logical connectives $\neg$, $\wedge$ and the quantifier $\forall$. All other logical operators such as disjunction $\vee$, implication $\Rightarrow$, and the existential quantifier $\exists$ are defined as usual.

Definition 3.7 *(Observational satisfaction)* Let Σ_{Obs} be an observational signature. The *observational satisfaction relation* between a Σ-algebra A and a first-order Σ-formula is denoted by $|=_{\Sigma_{Obs}}$ and defined as follows:

(1) For any two terms $t, r \in T(\Sigma, X)_s$ of the same sort s and for any valuation $\alpha: X \to A$, $A, \alpha |=_{\Sigma_{Obs}} t = r$ holds if $I_\alpha(t) \approx_{\Sigma_{Obs},A} I_\alpha(r)$.
(2) For any arbitrary Σ-formula ϕ and for any valuation $\alpha: X \to A$, $A, \alpha |=_{\Sigma_{Obs}} \phi$ is defined by induction over the structure of the formula ϕ in the usual way.
(3) For any arbitrary Σ-formula ϕ, $A |=_{\Sigma_{Obs}} \phi$ holds if for all valuations $\alpha: X \to A$, $A, \alpha |=_{\Sigma_{Obs}} \phi$ holds.

The notation $A |=_{\Sigma_{Obs}} \phi$ is extended in the usual way to classes of algebras and sets of formulas. ♦

Remark 3.8 (1) In [HB 98] the observational satisfaction relation is only defined for observational algebras. We have chosen here the more general definition from above since this is useful for stating correctness conditions for implementations (see Section 5). However, it is important to note that this has by no means an impact on the definition of the institution of observational logic in [HB 98] where the underlying satisfaction relation is just observational satisfaction for observational algebras.
(2) In the framework of extended hidden algebra (conditional) equations are considered which may be turned into "behavioural sentences" if the equality symbol is replaced by a special predicate symbol "~". Then the satisfaction relation interprets "~" automatically by the observational equality of elements. Thus the observational satisfaction relation from above coincides exactly with the satisfaction of behavioural sentences in extended hidden algebra.

Let us now define the syntax of (flat) observational specifications.

Definition 3.9 An *observational specification* $SP = \langle \Sigma_{Obs}, Ax \rangle$ consists of an observational signature Σ_{Obs} and a set Ax of first-order Σ-formulas, called the *axioms* of SP. ♦

The semantics of an observational specification can be defined in two different ways. One possibility is to adopt the approach of observational logic where only observational algebras are admissible models. Another possibility is to consider all algebras as models which observationally satisfy the axioms of the specification (even if the algebra has operations which are not behaviourally coherent). In the following the former approach will be called the "observational logic" appproach and, in this case, the model class

of a specification will be denoted by $Mod_{OL}(SP)$. The latter approach will be called the "non-restricted" approach and then the model class of a specification will be denoted by $Mod_{NR}(SP)$.

Definition 3.10 Let $SP = \langle \Sigma_{Obs}, Ax \rangle$ be an observational specification.

$$Mod_{OL}(SP) =_{def} \{A \in Alg_{Obs}(\Sigma_{Obs}) \mid A \models_{\Sigma_{Obs}} Ax\},$$
$$Mod_{NR}(SP) =_{def} \{A \in Alg(\Sigma) \mid A \models_{\Sigma_{Obs}} Ax\}. \quad \blacklozenge$$

Example 3.11 (1) The following specification extends the observational signature of bank accounts (cf. Example 3.2(1)) by appropriate axioms which define the observable properties of "new", "update" and "paycharge".

spec ACCOUNT =
sorts {account, int}
observable sorts {int}
observers { _.bal: account → int, _.undo: account → account}
operations "operations for the integers" ∪
{new: → account, _.update_ : account, int → account,
_.paycharge: account → account}
axioms "axioms for the integers" ∪
{∀x: int, s: account.
new.bal = 0, new.undo = new,
s.update(x).bal = s.bal+x, s.update(x).undo = s,
s.paycharge.bal = s.bal-10, s.paycharge.undo = s}

A possible model of the specification ACCOUNT which satisfies the axioms even literally can be defined in terms of lists of integers. Another model which satisfies the axioms observationally (but not literally) can be constructed by using the well-known array with pointer realization of lists.
In the above specification the behaviour of the operations is uniquely specified w.r.t. the given observers. A proper loose specification can be obtained, for instance, by removing the equations for the "paycharge" operation. Then the observational logic semantics of the specification is still restricted to those models where the interpretation of "paycharge" is compatible with the given observational equality while the non-restricted semantics allows arbitrary interpretations of "paycharge".

(2) The next specification defines a "merge" operation on streams (of integers) such that the elements of two streams are mixed together in an alternating order. The axioms specify the effect of the "merge" operation w.r.t. the given observers "head" and "tail" (see also the coinductive definition of merge in [JR 97]). To be able to apply "merge" to concrete streams we also define in an observational style particular streams like "from(n)" (which consists, for each integer n, of the values n, n+1, n+2, etc.) and "swing(n)" (which consists, for each integer n, of the alternating values n, -n, n, -n, etc.).

spec STREAM =
sorts {stream, int}
observable sorts {int}
observers {head: stream → int, tail: stream → stream}
operations "operations for the integers" ∪
{merge: stream, stream → stream,
from: int → stream,
swing: int → stream}
axioms "axioms for the integers" ∪
{∀s1, s2: stream, n: int.
head(merge(s1, s2)) = head(s1),
tail(merge(s1, s2)) = merge(s2, tail(s1)),
head(from(n)) = n,
tail(from(n)) = from(n+1),
head(swing(n)) = n,
head(tail(swing(n))) = -n,
tail(tail(swing(n))) = swing(n)} ♦

Obviously, for any observational specification SP, the observational logic semantics of SP is included in the non-restricted semantics of SP, i.e. $Mod_{OL}(SP) \subseteq Mod_{NR}(SP)$. However, if all algebras which observationally satisfy the axioms of SP are already observational algebras, then both semantics coincide. This is the idea of the so-called "persistent methodology" discussed in [D 98]. Indeed it turns out that in many cases the behavioural coherence of the operations is already a consequence of the axioms of the specification.[2] In the following we will study general conditions on the form of the axioms of a specification which guarantee behavioural coherence of the operations. Hence, if this conditions are satisfied, it is no more necessary to perform explicit proofs of behavioural coherence as carried out, for instance, in [DF 98] and [D 98].

4 Observer Complete Definitions

The concept of an observer complete definition describes a general pattern for the specification of non-observer operations. Thereby the basic idea is to define the behaviour of a non-observer operation op by a complete case distinction with respect to the given observers. For instance, in the ACCOUNT specification of Example 3.11 the "update" operation is specified by describing its effect w.r.t. the observers "bal" and "undo" using the equations s.update(x).bal = s.bal+x and s.update(x).undo = s. In general this idea can be formalized by using the following notion of a "complete set of contexts". Intuitively such a set of contexts represents a complete case distinction for all observable experiments that can be performed by using the observers of a given observational signature. It may be interesting to note that this idea is quite analogous to the notion of a complete set of constructors for inductively defined data types (cf. e.g. [B 82], [HH 82]).

Definition 4.1 *(Complete set of Σ_{Obs}-contexts)* Let Σ_{Obs} be an observational signature. A set $C0 \subseteq C(\Sigma_{Obs})$ is called a *complete set of Σ_{Obs}-contexts* if for each observable Σ_{Obs}-context $c \in C(\Sigma_{Obs})_{S_{Obs}}$ there exists a context $c0 \in C0$ which is a subcontext of c. ♦

Example 4.2 Considering the observational signature of accounts of Example 3.2(1) the set $C0 = \{z_{account}.bal, z_{account}.undo\}$ forms a complete set of Σ_{Obs}-contexts (since in this case any observable Σ_{Obs}-context has the form $z_{account}.undo^i.bal$ with $i \geq 0$). Similarly, the set $\{head(z_{stream}), tail(z_{stream})\}$ is a complete set of Σ_{Obs}-contexts for streams; cf. Example 3.2(2). ♦

We are now able to describe the general form of an observer complete definition of a non-observer operation op: $s_1,\ldots, s_n \rightarrow s$. Such a definition consists of a set of equations of the form $c_i[op(x_1,\ldots, x_n)] = r_i$ where the contexts c_i represent a complete case distinction for the observable experiments. Again let us point out the analogy to constructor complete definitions of functions on inductive data types where a function f is defined by case analysis (or pattern matching) w.r.t. the given constructors.

Definition 4.3 *(Observer complete definition)* Let Σ_{Obs} be an observational signature and let $(op: s_1,\ldots, s_n \rightarrow s) \in OP\backslash OP_{Obs}$ be a non-observer operation. An *observer complete definition* of op is a set E_{op} of Σ-equations of the form

$$c_1[op(x_1,\ldots, x_n)] = r_1$$
$$c_2[op(x_1,\ldots, x_n)] = r_2$$
$$\vdots$$
$$c_k[op(x_1,\ldots, x_n)] = r_k$$

such that the following conditions are satisfied:

1. $\{c_1,\ldots, c_k\}$ is a complete set of Σ_{Obs}-contexts.
2. $x_1,\ldots, x_n$ are variables (not occurring in $c_1,\ldots, c_k$) of sort $s_1,\ldots, s_n$.

---

[2] Recently, in [DFI 99], non-observer operations (on hidden sorts) which are "automatically" behavioural coherent are called "conservative hidden constructors".

3. Each r_i (i = 1, …, k) is a Σ-term which contains at most one occurrence of op and all other operation symbols occurring in r_i with non-observable argument sorts are observers.
4. There exists a monotonic[3], well-founded ordering "$\prec$" on $C(\Sigma_{Obs})$ such that each r_i with an occurrence of op has the form $c[op(t_1,\ldots, t_n)]$[4] with $c \in C(\Sigma_{Obs})$ and $c \prec c_i$. ♦

Example 4.4 Obviously, the equations s.update(x).bal = s.bal+x and s.update(x).undo = s provide an observer complete definition of "update" because these equations have the form

$(z_{account}.bal)[s.update(x)] = s.bal + x,$
$(z_{account}.undo)[s.update(x)] = s$

where, according to Example 4.2, $\{z_{account}.bal, z_{account}.undo\}$ is a complete set of Σ_{Obs}-contexts. Similarly the equations defining "paycharge" in Example 3.11 are observer complete.
Considering the specification STREAM, the equations head(merge(s1, s2)) = head(s1) and tail(merge(s1, s2)) = merge(s2, tail(s1)) provide an observer complete definition of "merge" because these equations have the form

$head(z_{stream})[merge(s1, s2)] = head(s1),$
$tail(z_{stream})[merge(s1, s2)] = z_{stream}[merge(s2, tail(s1))].$

Note that in the last equation "merge" occurs also on the righthand side which is not a problem because, according to condition 4 of Definition 4.3, $z_{stream} \prec tail(z_{stream})$ if we use for "$\prec$" the relation which compares the depth of two contexts. ♦

Remark 4.5 (1) The concept of an observer complete definition and the following results can easily be extended to conditional equations if, instead of an equation $c_i[op(x_1,\ldots, x_n)] = r_i$, a set of conditional equations $\phi_{i_1} \Rightarrow c_i[op(x_1,\ldots, x_n)] = r_i$, ..., $\phi_{i_m} \Rightarrow c_i[op(x_1,\ldots, x_n)] = r_i$ is used such that the premisses ϕ_{i_1} , ..., ϕ_{i_m} provide a complete case distinction.
(2) To be even more powerful one may consider (non-observer) operations op: $s_1,\ldots, s_n \rightarrow s$ where some of the argument sorts are constrained by a reachability requirement w.r.t. a set of constructor symbols. It is then possible to integrate a case analysis w.r.t. a complete set of constructor terms into the notion of an observer complete definition. ♦

We can now state our main theorem which shows that whenever an algebra A observationally satisfies an observer complete definition of an operation op then op is behaviourally coherent for A.[5]

Theorem 4.6 Let Σ_{Obs} be an observational signature, let E_{op} be an observer complete definition of (op: $s_1,\ldots, s_n \rightarrow s) \in OP \backslash OP_{Obs}$ and let $A \in Alg(\Sigma)$ such that $A \models_{\Sigma_{Obs}} E_{op}$. Then the following holds:
(1) op is behaviourally coherent for A.
(2) op is uniquely determined for A up to observational equality
(i.e. if $A' \in Alg(\Sigma)$, $A' \models_{\Sigma_{Obs}} E_{op}$ and $A'|_{\Sigma\backslash\{op\}} = A|_{\Sigma\backslash\{op\}}$ then
$op^{A'}(a_1,\ldots, a_n) \approx_{\Sigma_{Obs},A} op^{A}(a_1,\ldots, a_n)$ for all $a_1 \in A_{s_1},\ldots, a_n \in A_{s_n}$).

Proof: For the proof we use the following lemma:

Lemma Under the assumptions of the theorem the following holds:
For all observable contexts $c \in C(\Sigma_{Obs})_{S_{Obs}}$ (with application sort s) and variables $x_1,\ldots, x_n$ of sorts $s_1,\ldots, s_n$ not occurring in c there exists a Σ-term r such that

---

[3] "$\prec$" is called "monotonic", if for any contexts c, c_1, $c_2 \in C(\Sigma_{Obs})$, $c_1 \prec c_2$ implies $c[c_1] \prec c[c_2]$ (provided that the application sort of c coincides with the result sort of c_1 and of c_2).

[4] Note that, according to condition 3, all operation symbols occurring in $t_1,\ldots, t_n$ with non-observable argument sorts must be observers.

[5] Some related results which also provide (syntactic) criteria for behavioural coherence are given in [Pad 97] and [RG 98].

$$A \models c[op(x_1,\ldots, x_n)] = r$$

and the only operation symbols occurring in r with non-observable argument sorts are observers.[6]

The lemma will be proved below. Let us first show that (1) and (2) are consequences of the lemma. Thereby we assume w.l.o.g. that op: $s_1 \to s$ is unary.

Proof of (1):
Assume that $a, b \in A_{s_1}$ and $a \approx_{\Sigma_{Obs},A} b$. We have to show $op^A(a) \approx_{\Sigma_{Obs},A} op^A(b)$, i.e. for all observable contexts $c \in C(\Sigma_{Obs})_{S_{Obs}}$ with context variable z_s and for all valuations $\alpha, \beta: X \cup \{z_s\} \to A$ with $\alpha(x) = \beta(x)$ if $x \in X$, $\alpha(z_s) = op^A(a)$, $\beta(z_s) = op^A(b)$, we have $I_\alpha(c) = I_\beta(c)$.
Now let c be an arbitrary observable context with context variable z_s and α, β be arbitrary valuations with the above properties. Moreover, let x_1 be a variable of sort s_1 not occurring in c and let $\alpha 1, \beta 1: X \to A$ such that $\alpha 1(x_1) = a$, $\beta 1(x_1) = b$ and $\alpha 1(x) = \alpha(x) = \beta(x) = \beta 1(x)$ if $x \in X \backslash \{x_1\}$. Obviously, $I_\alpha(c) = I_{\alpha 1}(c[op(x_1)])$ and $I_\beta(c) = I_{\beta 1}(c[op(x_1)])$. According to the lemma there exists a Σ-term r such that $I_{\alpha 1}(c[op(x_1)]) = I_{\alpha 1}(r)$ and $I_{\beta 1}(c[op(x_1)]) = I_{\beta 1}(r)$. Since $a \approx_{\Sigma_{Obs},A} b$ and since the only operation symbols occurring in r with non-observable argument sorts are observers, we have $I_{\alpha 1}(r) = I_{\beta 1}(r)$. Hence $I_{\alpha 1}(c[op(x_1)]) = I_{\beta 1}(c[op(x_1)])$ and therefore $I_\alpha(c) = I_\beta(c)$.

Proof of (2):
Let A' be as in (2) and let $a_1 \in A'_{s_1} = A_{s_1}$. We have to show $op^{A'}(a_1) \approx_{\Sigma_{Obs},A} op^A(a_1)$. Let $c \in C(\Sigma_{Obs})_{S_{Obs}}$ be an arbitrary observable context with context variable z_s and let $\alpha': X \cup \{z_s\} \to A'$ and $\alpha: X \cup \{z_s\} \to A$ be arbitrary valuations with $\alpha'(x) = \alpha(x)$ if $x \in X$, $\alpha'(z_s) = op^{A'}(a_1)$, $\alpha(z_s) = op^A(a_1)$. We have to show $I_{\alpha'}(c) = I_\alpha(c)$.
Let x_1 be a variable of sort s_1 not occurring in c and let $\alpha 1': X \to A'$ and $\alpha 1: X \to A$ such that $\alpha 1'(x_1) = \alpha 1(x_1) = a$ and $\alpha 1'(x) = \alpha 1(x) = \alpha'(x) = \alpha(x)$ if $x \in X \backslash \{x_1\}$. Obviously, $I_{\alpha'}(c) = I_{\alpha 1'}(c[op(x_1)])$ and $I_\alpha(c) = I_{\alpha 1}(c[op(x_1)])$. By the above lemma there exists a Σ-term r such that $I_{\alpha 1'}(c[op(x_1)]) = I_{\alpha 1'}(r)$ and $I_{\alpha 1}(c[op(x_1)]) = I_{\alpha 1}(r)$. Since r has no occurrence of op, $I_{\alpha 1'}(r) = I_{\alpha 1}(r)$. Hence $I_{\alpha 1'}(c[op(x_1)]) = I_{\alpha 1}(c[op(x_1)])$ and therefore $I_{\alpha'}(c) = I_\alpha(c)$.

Proof of the lemma:
We prove the lemma by Noetherian induction w.r.t. the well-founded ordering "$\prec$" on $C(\Sigma_{Obs})$ assumed in condition 4 of Definition 4.3.
Let $c \in C(\Sigma_{Obs})_{S_{Obs}}$ be an arbitrary observable context (with application sort s) and let $x_1,\ldots, x_n$ be variables of sorts $s_1,\ldots, s_n$ not occurring in c. Since E_{op} is an observer complete definition of op it has the form shown in Definition 4.3 with an observer complete set $C0 = \{c_1,\ldots, c_k\}$ of Σ_{Obs}-contexts. Hence, there exists a context $c_i \in C0$ such that c_i is a subcontext of c and, moreover, $c_i[op(x_1,\ldots, x_n)] = r_i$ is an equation of E_{op}. In the following of the proof let c* be the Σ_{Obs}-context such that $c^*[c_i]$ coincides with c.
Case 1: op does not occur in r_i.
By assumption, $A \models_{\Sigma_{Obs}} c_i[op(x_1,\ldots, x_n)] = r_i$. Hence it is obvious, since c is an observable context, that

$$A \models c[op(x_1,\ldots, x_n)] = c^*[r_i].$$

Then we can choose r to be $c^*[r_i]$ which has the required form.
Case 2: op occurs in r_i.
Then, according to condition 4 of Definition 4.3, r_i has the form $c'[op(t_1,\ldots, t_n)]$ with $c' \in C(\Sigma_{Obs})$ and $c' \prec c_i$. Since, by assumption, $A \models_{\Sigma_{Obs}} c_i[op(x_1,\ldots, x_n)] = r_i$ we know that $A \models_{\Sigma_{Obs}} c_i[op(x_1,\ldots, x_n)] = c'[op(t_1,\ldots, t_n)]$ and hence, since c is an observable context

(*) $A \models c[op(x_1,\ldots, x_n)] = c^*[c'[op(t_1,\ldots, t_n)]]$.

Since $c' \prec c_i$ and since "$\prec$" is monotonic, we have $c^*[c'] \prec c^*[c_i]$, i.e. $c^*[c'] \prec c$. We can now apply the induction hypothesis and obtain

$$A \models c^*[c'[op(x_1,\ldots, x_n)]] = r$$

---

[6] "$\models$" denotes the standard satisfaction relation of the first-order predicate calculus.

for some Σ-term r of the required form whereby we can assume that $x_1,\ldots, x_n$ do not occur in c*[c']. Hence $A \models c^*[c'[op(t_1,\ldots, t_n)]] = r[t_1/x_1,\ldots, t_n/x_n]$. Thus, by (*),

$$A \models c[op(x_1,\ldots, x_n)] = r[t_1/x_1,\ldots, t_n/x_n]$$

and $r[t_1/x_1,\ldots, t_n/x_n]$ has the required form. ♦

Definition 4.7 *(Observer complete specification)*
An observational specification $SP = \langle \Sigma_{Obs}, Ax \rangle$ is called *observer complete* if for any non-observer operation $op \in OP \backslash OP_{Obs}$ with at least one non-observable argument sort there is an observer complete definition E_{op} of op such that $E_{op} \subseteq Ax$. ♦

Example 4.8 According to Example 4.4 the specifications ACCOUNT and STREAM are observer complete. ♦

As an obvious consequence of Theorem 4.6 we obtain that for observer complete specifications there is no difference between the observational logic semantics and the non-restricted semantics.

Corollary 4.9 Let $SP = \langle \Sigma_{Obs}, Ax \rangle$ be an observer complete specification. Then the following holds:

(1) For any $A \in Mod_{NR}(SP)$ and any $op \in OP$, op is behaviourally coherent for A and uniquely determined up to observational equality.
(2) $Mod_{OL}(SP) = Mod_{NR}(SP)$. ♦

This result explains also the relationship between observational logic semantics and extended hidden algebra semantics.

5 Observational Implementations

Corollary 4.9 shows that for observer complete specifications the behavioural coherence of *all* operations is already a consequence of the axioms of the specification. In the sequel we will point out that this result is particularly useful for proving implementation correctness. Thereby, having given an observational specification SP, a class C of Σ-algebras is considered as a correct implementation of SP if all algebras of C observationally satisfy the axioms of SP and, moreover, have only behaviourally coherent operations. Formally this is expressed by the following definition.

Definition 5.1 *(Observational implementation)* Let $SP = \langle \Sigma_{Obs}, Ax \rangle$ be an observational specification and $C \subseteq Alg(\Sigma)$ be a class of Σ-algebras. C is called an *observational implementation* of SP if $C \subseteq Mod_{OL}(SP)$. ♦

Note that in the above definition we have used the observational logic semantics of a specification since this guarantees that implementations indeed have only behaviourally coherent operations. Hence, for proving that a class C provides an observational implementation of SP one has to show, first, that all algebras of C observationally satisfy the axioms of SP and, secondly, that all algebras of C are observational algebras. It is an important consequence of Corollary 4.9 that the second step can be omitted whenever the specification SP to be implemented is observer complete.

Corollary 5.2 Let $SP = \langle \Sigma_{Obs}, Ax \rangle$ be an observer complete specification and $C \subseteq Alg(\Sigma)$. If $C \models_{\Sigma_{Obs}} Ax$ then C is an observational implementation of SP. ♦

Proof: Since $C \models_{\Sigma_{Obs}} Ax$ we have $C \subseteq Mod_{NR}(SP)$ and since SP is observer complete we have, by Corollary 4.9, $Mod_{OL}(SP) = Mod_{NR}(SP)$. Therefore $C \subseteq Mod_{OL}(SP)$. ♦

Corollary 5.2 shows that if one has checked once for all the observational completeness of the specification SP to be implemented then for proving an observational implementation of SP it is always sufficient to prove the observational satisfaction of the axioms of SP. For this purpose we can apply the

proof techniques proposed in [BH 98] which allow us to reduce proofs of behavioural theorems to proofs of standard theorems by using the following axiomatization of the observational equality.

Definition 5.3 Let $\Sigma_{Obs} = (\Sigma, S_{Obs}, OP_{Obs})$ be an observational signature with $\Sigma = (S, F)$ and let $\mathcal{L}(\Sigma)$ be the signature Σ enriched by binary predicate symbols $\sim_s$ for each non-observable sort $s \in S \backslash S_{Obs}$.[7] The set $Ax[\sim]$ of $\mathcal{L}(\Sigma)$-formulas is defined by

$$Ax[\sim] =_{def} \{Ax[\sim_s] \mid s \in S \backslash S_{Obs}\} \text{ where for any } s \in S \backslash S_{Obs},$$

$$Ax[\sim_s] =_{def} \forall\, x_L, x_R{:}s.\Big(\bigwedge_{c \in C(\Sigma_{Obs})_{s \to S_{Obs}}} \forall\, Var(c).\ c[x_L] = c[x_R]\Big) \Leftrightarrow x_L \sim_s x_R.$$

Thereby $C(\Sigma_{Obs})_{s \to S_{Obs}}$ denotes the set of observable Σ_{Obs}-contexts with application sort s and Var(c) denotes the set of all variables occurring in c besides the context variable z_s. ♦

Note that the formula $Ax[\sim_s]$ may be infinitary if Σ_{Obs} contains indirect observers (because in this case there may be infinitely many observable contexts). Nevertheless the infinitary formulas $Ax[\sim_s]$ can still be very useful in practice because the validity of (an instantiation of) the infinitary conjunction occurring in $Ax[\sim_s]$ can often be verified by an induction proof. If, however, one still wants to replace the infinitary conjunction by a finitary one then it is shown in [BH 98] that this can always be done by introducing appropriate auxiliary symbols. The same idea is also used in hidden and extended hidden algebra where auxiliary symbols are introduced for denoting a hidden congruence (cf. [GM 97] and [D 98]).

The following proposition shows that the observational satisfaction of a formula ϕ can be characterized by the standard satisfaction of a corresponding formula $\mathcal{L}(\phi)$ (called the "lifted version" of ϕ).

Proposition 5.4 Let Σ_{Obs} be an observational signature, $C \subseteq Alg(\Sigma)$ be a class of Σ-algebras and ϕ be a first-order Σ-formula. Then

$$C+Ax[\sim] \models \mathcal{L}(\phi) \text{ if and only if } C \models_{\Sigma_{Obs}} \phi$$

where $C+Ax[\sim] =_{def} \{A \in Alg(\mathcal{L}(\Sigma)) \mid A|_\Sigma \in C \text{ and } A \models Ax[\sim]\}$ and $\mathcal{L}(\phi)$ is a syntactic translation of ϕ where each occurence of an equality symbol "=" that relates terms of a non-observable sort s is replaced by the predicate symbol $\sim_s$.

Proof: The proof relies on the fact that for any algebra A which satisfies $Ax[\sim]$ the interpretation of $\sim$ in A is the observational equality $\approx_{\Sigma_{Obs},A}$. ♦

As a consequence of Corollary 5.2 and Proposition 5.4 we obtain the following result which shows that for proving an observational implementation of an observer complete specification SP it is sufficient to verify the lifted versions of the axioms of SP over an approriate axiomatic extension of the implementation. Since in usual cases the implementing class C will be the model class of a "concrete" specification, say SP-I, it is then sufficient to apply standard reasoning w.r.t. the logic of the specification language used for the specification SP-I.

Corollary 5.5 Let $SP = \langle \Sigma_{Obs}, Ax \rangle$ be an observer complete specification and $C \subseteq Alg(\Sigma)$. If $C+Ax[\sim] \models \mathcal{L}(Ax)$ then C is an observational implementation of SP. ♦

6 Conclusion

We have introduced the notion of an observer complete definition which is based on the idea to specify the effect of an operation by a complete case distinction with respect to a given set of observers. Thus observer complete definitions can be considered as an observational counterpart to constructor complete definitions, a well-known method for definig functions on inductive data types.

---

[7] Instead of predicate symbols $\sim_s$ one could also use binary boolean-valued operation symbols.

As a main result we have shown that operations which are specified by an observer complete definition are behaviourally coherent which is an important property of state manipulating operations. Moreover, we have seen that observer complete definitions provide an efficient condition to simplify implementation proofs. This condition can of course be directly integrated into the proof rules for behavioural implementations presented in [BH 98] and [Hen 97].

References

[B 82] M. Bidoit. Proofs by induction in "fairly" specified equational theories. In Proc. of the 6th German Workshop on Artificial Intelligence, Springer-Verlag I.F.B. 58, 154-166, 1982.

[BH 98] M. Bidoit, R. Hennicker: Modular correctness proofs of behavioural implementations. Acta Informatica 35, 951-1005, 1998.

[D 98] R. Diaconescu: Behavioural coherence in object-oriented algebraic specification. Japan Advanced Institute for Science and Technology, IS-RR-98-0017F, 1998.

[DF 98] R. Diaconescu, K. Futatsugi: CafeOBJ Report: The Language, Proof Techniques, and Methodologies for Object-Oriented Algebraic Specification, AMAST Series in Computing, Vol. 6, World Scientific, 1998.

[DFI 99] R. Diaconescu, K. Futatsugi, S. Iida: Component-based algebraic specification and verification in CafeOBJ. Japan Advanced Institute for Science and Technology, IS-RR-99-0020S, 1999.

[GM 97] J. Goguen, G. Malcolm: A hidden agenda. Report CS97-538, Univ. of Calif. at San Diego, 1997.

[HB 98] M. Bidoit, R. Hennicker: Observational logic. In: A. M. Haeberer (ed.): Proc. AMAST '98, Algebraic Methodology and Software Technology, LNCS 1548, 263-277, 1999.

[Hen 97] R. Hennicker: Structured specifications with behavioural operators: semantics, proof methods and applications. Habilitation thesis, Institut für Informatik, Ludwig-Maximilians-Universität München, 1997.

[HH 82] G. Huet, J. M. Hullot: Proofs by induction in equational theories with constructors. Journal of Computer and System Sciences 25, 239-266, 1982.

[JR 97] B. Jacobs, J. Rutten: A Tutorial on (Co)Algebras and (Co)Induction. EATCS Bulletin 62, 222-259, 1997.

[LEW 96] J. Loeckx, H.-D. Ehrich, M. Wolf: Specification of Abstract Data Types. Wiley and Teubner, 1996.

[Pad 96] P. Padawitz: Swinging data types: syntax, semantics, and theory. In: M. Haveraaen, O. Owe, O.-J. Dahl (eds.): Recent Trends in Data Type Specification, LNCS 1130, 409-435, 1996.

[Pad 97] P. Padawitz: Towards the one-tiered design of data types and transition systems. In: F. Parisi Presicce (ed.): Recent Trends in Algebraic Development Techniques, LNCS 1376, 365-380, 1997.

[RG 98] G. Rosu, J. Goguen: Hidden congruent deduction. Proc. Workshop on First-Order Theorem Proving, 213-223, Technische Universität Wien, 1998.