Journal of Universal Computer Science, vol. 12, no. 10 (2006), 1413-1425
submitted: 8/11/05, accepted: 20/9/06, appeared: 28/10/06 © J.UCS

Proving Properties for Behavioural Specifications with Term Observation

Narjes Berregeb
Laboratoire d'Informatique de Productique et de Parallélisme.
Institut National des Sciences Appliquées et de Technologie, Tunisia
narjes.benrajeb@topnet.tn

Abstract: Behavioural specifications allow to focus only on the"observable" behaviour of objects. These observations are made through "observable contexts" which are particular terms with a hole to be filled in with an object. We consider behavioural specifications based on the observation of a specified set of linear terms. The set of observable contexts is often infinite; therefore, we give an algorithm for computing some special contexts that we call "covering contexts", and show that they are sufficient for proving that two terms are behaviourally equal.

Key Words: behavioural specifications, term observation, observable contexts, covering contexts
Category: F.3.1, F.4.1

1 Introduction

Observability concepts play an important role in software specification and development, since they allow to focus on the external (or observable) behaviour of software system, and to abstract away non relevant details [Giarratana et al. 1976, Bidoit et al. 2002, Bidoit et al. 2004]. The work devoted to defining an adequate observational (or behavioural) semantics can be classified into two main approaches: the first is based on an observational equivalence relation between algebras [Sanella and Tarlecki 1988, Hennicker 1989], while the second is based on a relaxing of the classical satisfaction relation [Nivela and Orejas 1987, Reichel 1995, Bernot and Bidoit 1991, Hennicker and Bidoit 1999]. In the latter approach, observations are made through experiments called contexts. Hence, two objects are considered equal if they are equal in all observable contexts. In the framework of algebraic specifications, various techniques of observations were defined: observations based on sorts [Nivela and Orejas 1987, Hennicker 1991], operators [Bernot and Bidoit 1991], terms [Sanella and Tarlecki 1988, Hennicker 1989, Bernot et al. 1992], or formulae [Sanella and Tarlecki 1988, Knapick 1991]. Consider the specification List of figure 1. Two lists are considered equal (in a classical sense) if they have the same elements in the same order. However, one can relax this condition by considering the lists in some observable contexts, for example w.r.t the operation *in*; it means that two lists are equal if they have the same elements, regardless of their order.

specification: spec List
sorts:$list$, $element$, $boolean$
operations:
$\emptyset \ \rightarrow list$
$add: \ element \times list \rightarrow list$
$in: \ element \times list \rightarrow boolean$
axioms
$in(x, \emptyset) = false$
$in(x, add(x, l)) = true$
$x \neq y \Rightarrow in(x, add(y, l)) = in(x, l)$

Figure 1: Specification List

It is shown in [Bernot et al. 1994], that considering observations based on terms is finer than observations based on operators and sorts, in the sense that it allows to observe more specific objets. There exist some approaches and proof methods of behavioural properties based on sort observation [Bidoit and Hennicker 1996, Berregeb et al. 1998, Goguen and Malcolm 2000, Goguen et al 2002, Goguen and Lin 2003, Berregeb et al. 2004], or on a combination of sorts and operators [Hennicker and Bidoit 1999, Bidoit and Hennicker 2005]. However, there are no proof methods for behavioural properties with term observation. The aim of this paper is to propose such a proof technique. The main problem we meet when dealing with behavioural proofs is that the number of observable contexts is often infinite. Under some reasonable hypothesis, we give an algorithm for constructing some special contexts called covering contexts and we show that they are sufficient for proving that two objects are observationally equal. Note that these covering contexts are constructed for specifications where a set of terms defines the observations, which are finer than observations made on sorts as in [Berregeb et al. 1998, Berregeb et al. 2004].

The structure of the paper is as follows: In section 2, we give some basic definitions and notations. In section 3, we define the observational semantics we use. In section 4, we present our construction of covering contexts and show that they are sufficient for proving observational equality under some assumption on the specification. In section 5, we conclude and discuss further possible work.

2 Basic notions

We assume that the reader is familiar with the basic concepts of algebraic specifications [Wirsing 1990], term rewriting and equational reasoning. A many sorted signature Σ is a pair (S, F) where S is a set of sorts and F is a set of function symbols $f : s_1, \ldots s_n \rightarrow s$. Let X be a family of sorted variables, and let $T(\Sigma, X)$ be the term algebra. Let $var(t)$ denote the set of variables appearing in t. A term is linear if all its variables occur only once. If $var(t)$ is empty then t is a ground term. A substitution η assigns terms of appropriate sorts to variables. The domain of η is denoted by $dom(\eta)$. If t is a term, then $t\eta$ denotes the application of η to t. The identity substitution is denoted by Id.

Let N^* be the set of finite sequences of positive integers. For any term t, $Pos(t) \subseteq N^*$ denotes the set of positions of t, and the expression t/u denotes the subterm of t at position u. The root position is denoted by ϵ. The depth of a position u is the length of the corresponding sequence.

The depth of a term t, denoted by $|t|$, is defined as follows: $|t| = 0$ if t is a constant or a variable, otherwise, $|f(t_1, \ldots, t_n)| = 1 + max_i|t_i|$. An equation is a formula of the form $l = r$, where l and r are two terms of equal sort. Let E be the set of equations of a specification. If $t_1 = t_2$ is an equation following from E, we write $E \models t_1 = t_2$ or $t_1 =_E t_2$.

3 Observational Semantics

The notion of observations has been introduced as a means for describing what is observed in a given algebra. Various techniques have been proposed: observations based on sorts, operators, terms or formulae (see [Bernot et al. 1994] for a survey). The semantics we choose is based on a relaxing of the satisfaction relation. The notion of context is fundamental in all approaches based on such observational semantics. A behavioural property is obtained by taking into account only observable information. To show that it is valid, one has to show its validity in all observable contexts.

Definition 1 (Context). Let $T(\Sigma, X)$ be a term algebra and (S, F) be its signature.

- a context over F with *argument sort* s and *result sort* s', is a non ground term $c \in T(\Sigma, X)$ of sort s', with a single occurence of a distinguished variable of sort s called the contextual variable of c. To indicate the contextual variable z_s occurring in c, we often write $c[z_s]$ instead of c, where s is the sort of z_s.
- a contextual variable z_s of sort s is called an empty context with argument sort s and result sort s.

specification: spec EX
sorts: s, s_1
operations:
$a : s_1$
$b : s_1$
$f : s_1 \rightarrow s_1$
$g : s_1 \rightarrow s$
observations:
$g(x)$
axioms:
$g(a) = g(b)$
$f(a) = f(b)$

Figure 2: Observational specification EX

- the application of a context $c[z_s]$ to a term $t \in T(\Sigma, X)$ of sort s, denoted by $c[t]$, is defined by the substitution of z_s by t in $c[z_s]$. In this case, the context c is said to be applicable to t.
- by assumption, $var_c(c)$ will denote the set of variables occurring in c except the contextual variable of c. A context c is ground if $var_c(c) = \emptyset$. We denote by $|c|$ the depth of c, and by $d(c)$ the depth of the position of the contextual variable of c.
- a subcontext (resp. strict subcontext) of c, is a context which is a subterm (resp. strict subterm) of c, having the same contextual variable as c.

Example 1. Let g be a function symbol. Suppose $g : s \times s \rightarrow s$. Let $c[z_s]$ be the context $g(x, g(z_s, g(a, x)))$. We have: $|c| = 3$, $d(c) = 2$, $g(z_s, g(a, x))$ and z_s are strict subcontexts of c, $c[g(a, a)] = g(x, g(g(a, a), g(a, x)))$

Definition 2 (Specification, Observable specification). A specification (or equational specification) SP is a triple (S, F, E) where (S, F) is a signature and E is a finite set of equations. An observational specification SP_{obs} is a couple (SP, W) such that $SP = (S, F, E)$ is a specification and $W \subseteq T(\Sigma, X)$ is a finite set of observable terms.

In the following we denote by $SP_{obs} = (SP, W)$ an observational specification, where $SP = (S, F, E)$.

Example 2. The specification of figure 2 is an observational specification with $W = \{g(x)\}$.

The example of figure 2 is simple and not meaningful. However, we will use it just to illustrate the main notions and ideas.

Definition 3 (Observable context). Let $W \subseteq T(\Sigma, X)$ be a set of observable terms. We say that a context c is an observable context for a term $t \in T(\Sigma, X)$ if there exists $w \in W$ and a substitution θ such that $c[t] = w\theta$.

Example 3. Consider the specification of figure 2. Let $t = a$. Among the observable contexts for t, we have: $g(z_{s_1}), g(f(z_{s_1})), g(f(\ldots f(z_{s_1})\ldots)), \ldots$.

We have the same observable contexts for b.

The notion of observational validity is based on the idea that two objects are equal if they cannot be distinguished by observable contexts.

Definition 4 (Observational validity). Let t_1, t_2 be two terms of the same sort. We say that $t_1 = t_2$ is observationally valid, and denote it by $E \models_{obs} t_1 = t_2$ or $t_1 =_{obs} t_2$, iff for all observable context c for t_1 and t_2, $E \models c[t_1] = c[t_2]$.

Note that if $E \models t_1 = t_2$ then $E \models_{obs} t_1 = t_2$, but the converse may not be true. Observational theories generalize first-order theories: if $W = T(\Sigma, X)$ then the satisfaction relation $\models_{obs}$ is equal to $\models$.

Example 4. Consider the specification of figure 2. The equality $a = b$ is not valid in the classical sense since $E \not\models a = b$. However, $a =_{obs} b$, since for all observable context c for a and b, $E \models c[a] = c[b]$: $g(a) = g(b), g(f(a)) = g(f(b)), g(f(f(a))) = g(f(f(b))), \cdots$.

4 Proving an observational equality

The main problem of proving observational properties is that the number of observable contexts is often infinite, as shown in example 3. It is necessary to find a way to schematise these observable contexts finitely if we want to automatise the proofs of observational equalities.

We introduce in this section the notion of *covering contexts* which allow to describe finitely the possibly infinite set of observable ground contexts.

Definition 5 (set of covering contexts). Let SP be an observational specification equipped with a set of observable terms W. Let t be a term. A set of covering contexts for t, denoted by $CC(t, W)$, is a set of contexts such that:
For each observable context c_{obs} for t, there exists $c \in CC(t, W)$, a context c' and a substitution θ such that $c_{obs} = c'[c]\theta$

Let us illustrate the idea of the computation of covering contexts on the specification of figure 2 but equipped with the set of observable terms $W = \{f(x)\}$. Let $t = f(f(a))$ be of sort s_1. Let us try to enumerate the observable contexts for t. We have:

i/ z_{s_1} is an observable context for t since $z_{s_1}[f(f(a))] = f(f(a)) = f(x)\theta$, where $x\theta = f(a)$. In this case the variable x captures a subterm of t.

ii/ $f(z_{s_1})$ is an observable context since $f(z_{s_1})[f(f(a))] = f(f(f(a))) = f(x)\theta$, where $x\theta = f(f(a))$. In this case the variable x captures exactly the term t.

iii/ $f(f(z_{s_1}))$ is an observable context since $f(f(z_{s_1}))[f(f(a))] = f(f(f(f(a)))) = f(x)\theta$, where $x\theta = f(f(f(a)))$. In this case the variable x captures more than the term t, and we can continue this process infinitely.

The idea of our method is to construct at first, all the observable contexts corresponding to points i/ and ii/, i.e, those observable contexts such that if x is a variable occuring in an observable term w, then x captures a subterm of t. In other words, these observable contexts are such that there exists a subterm in w having t as instance. We call them contexts of type 1. For the other remaining observable contexts, corresponding to point iii/ in the example, we construct some particular contexts, called contexts of type 2, that can be embedded in a context to form an observable term, and allowing to "cover" the remaining observable contexts.

Definition 6 (context of type 1). A context c is a context of type 1 for t if there exists a substitution θ, an observable term $w \in W$, a subterm u in w such that c is applicable to t, $c[t] = w\theta$ and $u\theta = t$.

Definition 7 (context of type 2). A context c is a context of type 2 for t if

- c is applicable to t
- there exists a context c' with result sort s, and a variable of sort s occurring in $w \in W$ such that c' is applicable to $c[t]$

Definition 8 (most general context). A context c with result sort s is a most general context if:

- either c is a contextual variable z_s
- or c is of the form $f(x_1, \ldots, c', \ldots, x_n)$ where $x_1 \ldots x_n$ are variables not in $var(c')$, $f : s_1, \ldots, s_n \to s$ is a function symbol and c' is a most general context.

```
input:
        $t$ /* a term*/
        $d$ /* a depth*/
        $W$ /* a set of observable terms*/


/*construction of contexts of type 1*/
$C_1 := \emptyset$
For all terms $w$ in $W$ do
  For all subterms $u$ in $w$ such that there exist $c$, $\theta$, verifying
  $u\theta = t$ and $c[t] = w\theta$ do
    $C_1 := C_1 \cup \{c\}$
  endfor
endfor


/* construction of contexts of type 2 of depth $d$ at most*/
Let $S_W = \{s \mid s$ is a sort of a variable appearing in $w \in W\}$
Let $s_t$ be the sort of $t$
$C_2 := \{z_{s_t}\}$
For all $c \in C_2$ such that $|c| < d$ and $c$ has a result sort $s$ not in $S_W$ do
  Construct the set $C'$ of all the contexts $c'$ of the form
  $f(x_1, \ldots, x_{i-1}, c, x_{i+1}, \ldots, x_n)$ such that:
     - the contextual variable of $c'$ is $z_{s_t}$
     - $x_1, \ldots, x_{i-1}, x_{i+1}, \ldots x_n$ are fresh variables
       resp. of sort $s_1, \ldots, s_{i-1}, s_{i+1}, \ldots, s_n$
     - $f$ is a function symbol such that $f : s_1, \ldots, s_{i-1}, s, s_{i+1}, \ldots, s_n \to s'$
     - $c'$ can be subcontext of some context with result sort in $S_W$.
  $C_2 := (C_2 \setminus \{c\}) \cup C'$
endfor

$CC(t, W) := C_1 \cup C_2$

output: $CC(t, W)$
```

Figure 3: Computation of a set of covering contexts

The idea of the construction of contexts of type 2 is as follows: We construct all most general contexts c' applicable to t having (at most) a chosen depth d such that c' can be subcontext of some observable context. Testing this condition comes to test whether c' can be a subcontext of some context with a result sort corresponding to a sort of a variable appearing in $w \in W$.
Now, to test whether a context c' can be a subcontext of a context with result sort s, one can rely on the specification signature. Moreover, the signature can be seen as a graph where the nodes are the sorts. If $f : s_1, \ldots, s_n \rightarrow s_{n+1}$ is a function symbol, then each $s_i (i \in \{1, \ldots, n\})$ is connected to the node s_{n+1}. The problem comes then to find a path from the result sort of c' to s.

The algorithm computes contexts of type 2 of any given depth d (for example, one can consider a depth $d = 1$). The computation of a set of covering contexts is given in figure 3.

Lemma 9. *The computation of a set of covering contexts given in figure 3 terminates.*

Proof. – Construction of contexts of type 1 terminates since the number of subterms u in an observable term w is finite and the number of observable terms in W is finite.

– Construction of contexts of type 2 terminates since the depth of these contexts is smaller or equal than d.

In the following, we denote by $CC(t, W)$, the set of covering contexts for a term t computed according algorithm of figure 3.

Lemma 10. *Let SP be an observational specification equipped with a set of observable terms W. We suppose that all the terms in W are linear. Let t be a term. The set $CC(t, W)$ computed according to the algorithm of figure 3 is a set of covering contexts for t.*

Proof. Let c_{obs} be an observable context for t.

case 1: c_{obs} is an instance of a context c of type 1 for t. Let θ be the substitution such that $c_{obs} = c\theta$, in this case $c' = z$ (empty context).

case 2: c_{obs} is not an instance of a context of type 1 for t. Then, there exists an observable term w and a variable x occurring in w such that $c_{obs}[t] = w\theta$ and t is a strict subterm of $x\theta$.

Let $x\theta = c_1[t]$. Since w is linear, we can denote by $w[x]$ the context with x as its contextual variable.

– If $|c_1| \leq d$ then there exists a context $c \in CC(t, W)$ of type 2 for t such that c_1 is an instance of c since c is most general. Let θ' be the substitution such that $c\theta' = c_1$. Thus, $c_{obs} = w[c]\theta\theta'$.

– if $|c_1| > d$, then let $c_1 = c''c'$ where $|c'| = d$. Then, there exists a context $c \in CC(t, W)$ of type 2 for t such that c' is an instance of c since c is most general.
Let θ' be the substitution such that $c\theta' = c'$. Thus, $c_{obs} = w[c''[c]]\theta\theta'$

The next theorem shows that the covering contexts obtained by applying the algorithm of figure 3 are sufficient for proving an observational equality.

Theorem 11. *Let SP be an observational specification equipped with a set of observable terms W. We suppose that all the terms in W are linear. Let t_1, t_2 two terms.*

$$\forall c \in (CC(t_1, W) \cap CC(t_2, W)),$$

$$E \models c[t_1] = c[t_2] \quad \Rightarrow \quad t_1 =_{obs} t_2$$

Proof. Let c_{obs} be an observable context for t_1 and t_2. We have to show that $E \models c_{obs}[t_1] = c_{obs}[t_2]$. Since c_{obs} is an observable context for t_1, then by lemma 10, there exists $c \in CC(t_1, W)$, a context c' and a substitution θ such that $c_{obs} = c'[c]\theta$. By hypothesis, $E \models c[t_1] = c[t_2]$, then $E \models (c'[c]\theta)[t_1] = (c'[c]\theta)[t_2]$.Thus $E \models c_{obs}[t_1] = c_{obs}[t_2]$.

Example 5. Let us consider the specification of streams of natural numbers in figure 4, inspired by [Goguen and Lin 2003].

The observable term $odd(x)$ indicates that we do not distinguish natural numbers that behave similarly under contexts of the form $odd(x)$. Moreover, we distinguish only natural numbers that do not have the same parity. For example, we have $1 =_{obs} 3$ but not $1 =_{obs} 2$ (for short we use the notation 1 for $s(0)$, 2 for $s(s(0))$ and so on).

Let us consider the equality $1 = 3$. Computing covering contexts of depth $d = 1$ for 1 and 3 gives the set $C = \{odd(z), s(z), cons(z, s)\}$. Note that we do not consider the context $null(z)$ in C, since it is with result sort bool and cannot be a subcontext of an observable context.

To prove $1 = 3$ is an observational equality, we have to apply all covering contexts of C on 1 and 3, we obtain:

specification: spec Stream
sorts: $stream$, nat, $bool$
operations:
$hd : stream \to nat$
$tl : stream \to stream$
$cons : nat\ stream \to stream$
$0 : nat$
$s : nat \to nat$
$false :\to bool$
$true :\to bool$
$odd : nat \to bool$
$null : nat \to bool$

observations:
$hd(x), tl(x), odd(x)$

axioms
$hd(cons(x, s)) = x$
$tl(cons(x, s)) = s$
$odd(0) = false$
$odd(s(0)) = true$
$odd(s(s(x))) = odd(x)$
$null(0) = true$
$null(s(x)) = false$

Figure 4: Specification stream

$$odd(1) = odd(3) \tag{1}$$
$$s(1) = s(3) \tag{2}$$
$$cons(1, s) = cons(3, s) \tag{3}$$

Equation 1 is easily simplified using the specification axioms to false=false. Equation 2 holds because if two natural numbers have the same parity then their successors also have the same parity. To show that equation 2 is an observational property, one still has to use a context induction reasoning (thus apply covering contexts and use the induction hypothesis $1 = 3$).
Equation 3 holds too. Computing covering contexts of depth $d = 1$ for $cons(1, s)$

and $cons(3, s)$ gives the set $C = \{hd(z), tl(z), cons(x, z)\}$. Applying these covering contexts to equation (3), we get:

$$hd(cons(1, s)) = hd(cons(3, s)) \tag{4}$$
$$tl(cons(1, s)) = tl(cons(3, s)) \tag{5}$$
$$cons(x, (cons(1, s)) = cons(x, cons(3, s)) \tag{6}$$

Equation 5 is simplified, using the axioms, to $s = s$. Equations 4 and 6 hold and require the use of the induction hypothesis $1 = 3$.

Now, let us prove some property on streams. According to the specification observations, two streams are observationally equal if their elements have the same parity. Consider stream *one* defined as an operator

$$one : \; \rightarrow stream$$

with the axioms

$$hd(one) = 1$$
$$tl(one) = one$$

In the same manner, consider stream *three* defined as an operator

$$three : \; \rightarrow stream$$

with the axioms

$$hd(three) = 3$$
$$tl(three) = three$$

To prove that $one = three$ is an observational property, we compute covering contexts of depth $d = 1$ for $cons(1, s)$ and $cons(3, s)$ gives: $C = \{hd(z), tl(z), cons(x, z)\}$. Applying these covering contexts to equation 3, we get:

$$hd(one) = hd(three) \tag{7}$$
$$tl(one) = tl(three) \tag{8}$$
$$cons(x, one) = cons(x, three) \tag{9}$$

Equation 7 is simplified to $1 = 3$ which has already been proved. Equation 8 and 9 still require an induction reasoning on contexts and can be proved by applying covering contexts and using induction hypothesis $one = three$.

5 Conclusion

In this paper, we presented first an observational semantics based on the notion of context for algebraic specifications with linear term observation. The problem met when one tries to automatise behavioural proofs is that the number of observable contexts is often infinite. We proposed an algorithm for constructing special contexts called "covering contexts" and showed that they are sufficient to prove behavioural equalities. A possible improvement of this work is to consider a behavioural semantics based on multicontexts, which are contexts with multiple occurences of the contextual variable, as defined in [Bernot et al. 1992]. Further work is to extend our techniques to non linear observable terms and to reduce the number of covering contexts for a term. Finally, it is interesting to build a theorem prover on top of this approach.

Acknowledgment

I would like to thank the anonymous referees for their valuable comments.

References

[Bernot and Bidoit 1991] Bernot, G., Bidoit, M.: "Proving the correctness of algebraically specified software: Modularity and observability issues"; 2nd International Conference on Albegraic Methodology on Software technology, Lect. Notes Comp. Sci. 1991, Springer, 216-239.

[Bernot et al. 1992] Bernot, G.,Bidoit, M.,Knapik, T.: "Toward an adequate notion of observation"; European Symposium on Programming; Lect. Notes Comp. Sci. 589, 1992, Springer, 39-55.

[Bernot et al. 1994] Bernot, G.,Bidoit, M.,Knapik, T.: "Behavioural approaches to algebraic specifications: a comparative study"; Acta Informatica, 31, 7, 1994, 651-671.

[Berregeb et al. 1998] Berregeb, N., Bouhoula, A., Rusinowitch, M.: "Observational proofs with critical contexts"; Fundamental approaches in Software, Lect. Notes Comp. Sci., 1389, 1998, Springer.

[Bidoit and Hennicker 1996] Bidoit, M., Hennicker, R.: "Behavioural theories and the proof of behavioural properties"; Theoretical Computer Science, 165, 1, 1996, 3-55.

[Bidoit and Hennicker 2005] Bidoit, M., Hennicker, R.: "Constructor-Based Observational Logic": Journal of Logic and Algebraic Programming, 2005.

[Bidoit et al. 2002] Bidoit, M., Sannella, D., Tarlecki, A.: "Global Development via Local Observational Construction Steps": Proc. 27th Intl. Symp. on Mathematical Foundations of Computer Science, 2420, 2002, Springer, 1-24.

[Bidoit et al. 2004] Bidoit, M., Sannella, D., Tarlecki, A.: "Toward component-oriented formal software development: An algebraic approach". Proc. 9th Monterey Software Engineering Workshop on Radical Innovations of Software and Systems Engineering in the Future, Lecture Notes in Computer Science, 2941, 2004, 75-90.

[Berregeb et al. 2004] Berregeb, N., Robbana, A., Tiwari, A.: "Towards automated proofs of observational properties"; Discrete Mathematics in Theoretical Computer Science, 6, 1, 2004, 143-162.

[Giarratana et al. 1976] Giarratana, V., Gimona, F., Montanari, U.: "Observability Concepts in Abstract Data Type Specifications"; MFCS, Lect. Notes Comp. Sci., 1976, Springer, 576-587.

[Goguen and Lin 2003] Goguen, J., Lin, K.: "Behavioural verification of distributed concurrent systems with BOBJ"; Conference on Quality Softawre, IEEE Press, 165, 1, 2003, 216-235.

[Goguen et al 2002] Goguen, J., Lin, K., Rosu, G., "Conditional Circular Coinductive Rewriting with Case Analysis.", 16th International Workshop WADT, Lect. Notes Comp. Sci., 2002, Springer, 216-232.

[Goguen and Malcolm 2000] Goguen, J., Malcolm, G.: "A hidden agenda"; Theoretical Computer Science, 245, 1, 2000, 55-101.

[Hennicker and Bidoit 1999] Hennicker, R., Bidoit, M. : "Observational logic"; Algebraic Methodology of Softawre Technology, Lect. Notes Comp. Sci., 1548, 1999, Springer, 263-277.

[Hennicker 1989] Hennicker, R.: "Implementation of parametrized observational specifications"; TAPSOFT'89, Lect. Notes Comp. Sci., 1989, Springer, 290-305.

[Hennicker 1991] Hennicker, R.: "Context induction: a proof principle for behavioural abstractions and algebraic implementations"; Formal Aspects of Computing, 3, 4, 1991, 3-55.

[Knapick 1991] Knapik, T.: "Specifications with observable formulae and observable satisfaction relation";Recent Trends in Data Type Specifications, 1991.

[Nivela and Orejas 1987] Nivela, P., Orejas, F.: "Initial behavioural semantics for algebraic specifications"; Recent Trends in Data Type Specifications, 332, 1987, 184-207.

[Reichel 1995] Reichel, H.: "An approach to object semantics based on terminal coalgebras";Mathematical structures in Computer Science, 5, 1995, 129-152.

[Sanella and Tarlecki 1988] Sanella, D., Tarlecki, A.: "Toward formal development of programs from algebraic specifications reviseted"; Acta Informatica, 25, 1988, 233-281.

[Wirsing 1990] Wirsing, M.: "Algebraic specifications", MIT Press, 1990, Chapter 13.