An Institution of Object Behaviour

Amílcar Sernadas[1,3], José Félix Costa[2,3] and Cristina Sernadas[1,3]

[1] Departamento de Matemática, IST
[2] Departamento de Informática, FCUL
[3] INESC, Apartado 10105, 1017 Lisboa Codex, PORTUGAL
E-mail: acs@inesc.pt

Abstract. An institution for a simple logic of behaviour is built using a cofibration from a category of transition systems into the envisaged category of signatures. The chosen propositional, linear temporal logic distinguishes between event occurrence and event enabling. The satisfaction condition is proved using a fibered adjunction between transition systems and their computations. The operational semantics of behaviour specifications is briefly discussed.

1 Introduction

Consolidating the conjectures in [SernadasA *et al* 92a], we study the problem of setting-up a suitable institution (see [Goguen and Burstall 84]) of linear temporal logic from a given reasonable semantic domain of object behaviour. For the sake of simplicity we work towards a propositional fragment. No major problem is expected when extending the results towards a first-order logic (for instance, like the logic outlined in [SernadasA *et al* 92b]).

The basic problem is to identify the appropriate cofibration from the given category of behaviours into the category of signatures. It should constitute an indexing mechanism (see [Tarlecki *et al* 91]) from which the semantic components of the envisaged institution should follow without much further ado (like a Grothendieck "deconstruction"). Namely, the semantic functor of the institution is defined using the functor induced by the cofibration at hand.

Many behaviour models have been proposed in the past and several of them have been adopted when studying the semantics of objects. Herein, we consider a notion of behaviour built around the notion of transition system, adapting from [Manna and Pnueli 92], instead of the naive notion of process as a set of life-cycles considered for instance in [SernadasA *et al* 92a]. Some care is needed when establishing the category of transition systems in order to ensure an adequate treatment of aggregation and hiding. Our categorial treatment of transition systems is an extension of the work in [Winskel 84].

Once the semantic domain is fixed with a suitable indexing mechanism, the development of the envisaged institution goes like this: (i) the syntax functor is defined towards the chosen language (in this case, propositional, linear temporal logic distinguishing between enabling and occurrence of events); (ii) the semantic functor is canonically established using the properties of the indexing mechanism as a cofibration; (iii) the satisfaction relation is defined using the standard Kripke technique on the set of computations of each transition system; (iv) the satisfaction condition is verified exploiting a fibered adjunction between transition systems and their computations. The use of fibered adjunctions was already advocated in [Winskel 85, 87] for relating different semantic domains.

This direct approach should be compared with the proposal in [Fiadeiro *et al* 93] where the bridge between behaviours (processes as sets of life-cycles) and theories goes through the standard interpretation structures for the chosen logic (a similar one but without distinguishing between occurrence and enabling of events).

Herein, beside working with a more general semantic domain (transition systems) and a more powerful logic, we identify a general approach to the problem: set-up a cofibration from the semantic domain into the category of signatures and establish a fibered adjunction between the elements of the semantic domain and their computations.

We assume that the reader is conversant with the field of temporal logic specification (for instance at the level of [Goldblatt 87]). We also use a little bit of category theory (the reader may find all relevant concepts in the introductory chapters of any textbook on the matter, eg [Adámek *et al* 90]). We borrow from [Winskel 87] the result on the preservation of cocartesian morphisms by left adjoints.

The underlying approach to system specification (following [SernadasA *et al* 92b]) is briefly described in Section 2. Section 3 is dedicated to the underlying semantic domain: transition systems and their computations. The envisaged institution is presented in Section 4. The concluding remarks in Section 5 include a brief discussion of the "operational" semantics of temporal specifications.

2 System specification

At first sight (see [SernadasA et al 89a,89b, SernadasA and Ehrich 91]), a system (or object) is an entity with an internal state (reflected in the values of its slots or attributes) that may interact with other systems. While so interacting a system displays some behaviour. That is, depending on its internal state, it is not always ready to provide the same services to the others and/or to ask the same services of others. The most basic unit of

interaction is the event: for instance, the action event open-door may be shared by a person and a car. Therefore, the behaviour of a system may be seen as corresponding to a state-dependent menu of enabled events (at each state the set of events in which the object is "willing" to involve itself).

At this naive level of analysis, the specification of a system should include the list of its attributes (with the indication of their codomain sorts), the list of its events (with the indication of their parameter sorts if any) and the constraints on its behaviour (including the observation events that allow the examination of values of the attributes). Furthermore, the "life" of a system should correspond to a trajectory of transitions that happened since its birth to its death. Therefore, if we ignore the birth and the death issues, a computation of the system can be presented as a map from $\mathbb{Z}$ (the set of integers) into the set of its transitions.

As an illustration consider a simple *flip-flop* as a system. The *flip-flop* has a unique attribute status with two possible values: on and off. It allows only two actions: *flip* and *flop*. Therefore, it allows four events: two observation events (*status=on* and *status=off*) plus two action events (*flip* and *flop*). Observation events do not change the state of the *flip-flop*. Actions do change it. Both kinds of events may be constrained by enabling conditions. The intended behaviour of the flip-flop is easily described using a simple propositional temporal logic, as follows:

(1) $(\Diamond \text{flip} \Leftrightarrow \Diamond \text{status=on})$
(2) $(\Diamond \text{flop} \Leftrightarrow \Diamond \text{status=off})$
(3) $(\nabla \text{flip} \Rightarrow (\mathbf{X} \Diamond \text{status=off}))$
(4) $(\nabla \text{flop} \Rightarrow (\mathbf{X} \Diamond \text{status=on}))$
(5) $(\Diamond \text{status=off} \mid \Diamond \text{status=on})$
(6) $((\nabla \text{status=on} \wedge (\neg \nabla \text{flip})) \Rightarrow (\mathbf{X} \Diamond \text{status=on}))$
(7) $((\nabla \text{status=off} \wedge (\neg \nabla \text{flop})) \Rightarrow (\mathbf{X} \Diamond \text{status=off}))$

The propositional symbols are of the form $\Diamond g$ or ∇g where g is an event (action or observation). A propositional symbol of the form $\Diamond g$ (read: g is possible) indicates that the event g is enabled. A propositional symbol of the form ∇g (read: g happens) indicates the occurrence of event g.

For the the purpose of this paper, there is no need to distinguish between the two kinds of events (actions and observations). Therefore, in the sequel, we consider a single alphabet of events G generating the following alphabet of propositional symbols $\{\Diamond g : g \in G\} \cup \{\nabla g : g \in G\}$. But first-order versions of this behaviour language will require that distinction, namely for imposing the functionality of attributes (see [SernadasA *et al* 92b]).

Returning to the behaviour specification above, axiom (1) states that *flip* may happen only in states where the observation *status=on* may also

happen: that is, *flip* may happen only in states where the attribute status has value on. Axiom (2) is similar for *flop*. Axiom (3) states that if *flip* happens then in the next state the observation *status=off* is possible: that is, if *flip* happens then in the next state status has value off. Axiom (4) is similar for *flop*. Axiom (5) states that status is either off or on (exclusive or): deterministic functionality of the attribute status. Axioms (6) and (7) reflect the fact that the observations of status do not affect the state of the *flip-flop*.

Such a logic of behaviour with happening (∇) and enabled ($\diamond$) symbols is rich enough for describing in practice useful interacting systems. For further examples see [SernadasA *et al* 92b] where a proof system is provided for dealing with object specialization and aggregation.

Note that fairness assumptions are easily specified in this logic. For instance, if so specified, $((\mathbf{F}(\mathbf{G}\diamond g))\Rightarrow(\mathbf{G}(\mathbf{F}\nabla g)))$ states that if g is continually enabled from a certain position on then it happens infinitely many times (**F** stands for "sometime" and **G** for "always"). Clearly, we expect the formula $(\nabla g\Rightarrow\diamond g)$ to be universally true, since an event g may happen only when enabled. As we shall see, the proposed semantics does ensure the universal validity of this assertion.

3 A semantic domain

3.1 Transition systems

Definition 3.1.1. A *transition system* $trs=<G,S,\delta>$ is a triple consisting of a set G, a set S, and a family $\delta=\{\delta_e\}_{e\in E}$, where $E=\mathbf{2}^G$, of total maps $\delta_e:S\rightarrow\mathbf{2}^S$ such that $s\in\delta_\varnothing(s)$ for each $s\in S$. □

The elements of G are called *prime events*, those of E *events* and those of S *states*. Also, sets E and S are said to be the *event space* and the *state space*, respectively, of the transition system. Each function $\delta_e:S\rightarrow\mathbf{2}^S$ is called the *transition map for e*: it returns the set of states reachable from a given state by the occurrence of event *e*.

Comparing with [Manna and Pnueli 92], this definition of transition system is different in two accounts: (i) it allows for the occurrence of "simultaneous" prime events; and (ii) it does not impose that $\delta_\varnothing(s)=\{s\}$. The former is essential to modelling "aggregation" (parallel composition) where several prime events may occur simultaneously. And the latter is needed for coping with "abstraction" (hiding of prime events).

Definition 3.1.2. A *transition system morphism* $h:<G,S,\delta>\rightarrow<G',S',\delta'>$ is a pair $<h_G,h_S>$ where $h_G:G'\rightarrow G$ and $h_S:S\rightarrow S'$ are total maps such that, for each $e\in E$, $h_S(\delta_e(s))\subseteq\delta'_{h_G^{-1}(e)}(h_S(s))$. □

The contravariance of the first component is to be expected if we anticipate

that the prime events constitute the signature of a transition system: signature morphisms and semantic morphisms go in opposite directions. Furthermore, the morphism condition just states that to each "transition" $s—e\rightarrow r$ of $\langle G,S,\delta\rangle$ corresponds a transition of $\langle G',S',\delta'\rangle$.

Fact/Definition 3.1.3. Transition systems and their morphisms constitute a concrete category TrS over $Set^{op}\times Set$ with respect to the functor $GS:TrS\rightarrow Set^{op}\times Set$ mapping each $trs=\langle G,S,\delta\rangle$ onto the pair (G,S), and each $h:\langle G,S,\delta\rangle\rightarrow\langle G',S',\delta'\rangle$ onto the pair $(h_G^{op}:G\rightarrow G', h_S:S\rightarrow S')$. □

Fact 3.1.4. The category TrS is small complete and cocomplete. □

Since this result is not essential to the development of this paper we omit its proof herein. The product of transition systems $trs'=\langle G',S',\delta'\rangle$ and $trs''=\langle G'',S'',\delta''\rangle$ reflects their parallel composition and it is, up to isomorphism, given by a 2-source with vertex $trs'\|trs''=\langle G'+G'',S'\times S'',\delta\rangle$ where, for $e\in \mathbf{2}^{G'+G''}$, $\delta_e(s)=\delta'_{i'^{-1}(e)}(\pi'(s))\times\delta''_{i''^{-1}(e)}(\pi''(s))$. The terminal transition system is the triple $\langle\emptyset,\{\bullet\},\delta\rangle$ where $\delta_\emptyset(\bullet)=\{\bullet\}$. Coproducts are only interesting for a given G and a given S: they reflect (external and internal) choice and are given by $trs'+trs''=\langle G,S,\delta'\cup\delta''\rangle$. And the initial transition system is the triple $\langle\emptyset,\emptyset,\delta_\emptyset\rangle$ where $\delta_\emptyset$ is the empty map.

Definition 3.1.5. The *indexing functor* $Gen:TrS\rightarrow Set^{op}$ is $\pi_1\circ GS$. □

Therefore, the functor Gen maps each $trs=\langle G,S,\delta\rangle$ onto G, and each $h:\langle G,S,\delta\rangle\rightarrow\langle G',S',\delta'\rangle$ onto $h_G^{op}:G\rightarrow G'$.

Fact 3.1.6. The indexing functor $Gen:TrS\rightarrow Set^{op}$ is a cofibration. Namely, given $h:G\rightarrow G'$ of Set^{op} and a transition system $trs=\langle G,S,\delta\rangle$, the cocartesian morphism for h and trs is $\underline{h}_{trs}:trs\rightarrow trs'$ defined as follows: (i) $trs'=\langle G',S,\delta'\rangle$ with $\delta'_{e'}=\bigcup_{x:(h^{op})^{-1}(x)=e'}\delta_x$ for each event $e'\in E'$; and (ii) $\underline{h}_{trs}=\langle(\underline{h}_{trs})_G,(\underline{h}_{trs})_S\rangle$ with $(\underline{h}_{trs})_G:G'\rightarrow G$ given by $(\underline{h}_{trs})_G=h^{op}$ and $(\underline{h}_{trs})_S:S\rightarrow S$ given by $(\underline{h}_{trs})_S=id_S$.

Proof: We verify first that $\underline{h}_{trs}$ is a morphism in TrS: indeed, $\delta_e(s)\subseteq\delta'_{(h^{op})^{-1}(e)}(s)$ because the latter is $\bigcup_{x:(h^{op})^{-1}(x)=(h^{op})^{-1}(e)}\delta_x(s)$ that contains $\delta_e(s)$. Moreover, $\underline{h}_{trs}$ is cocartesian: given a transition system $trs''=\langle G'',S'',\delta''\rangle$, $f:\langle G,S,\delta\rangle\rightarrow\langle G'',S'',\delta''\rangle$ of TrS and $k:G'\rightarrow G''$ of Set^{op} such that $f_G=h^{op}\circ k^{op}$, there is a unique $\underline{k}:\langle G',S',\delta'\rangle\rightarrow\langle G'',S',\delta''\rangle$ such that $f=\underline{k}\circ\underline{h}_{trs}$ with $\underline{k}_G=k^{op}$. Indeed, for the composition to hold $\underline{k}_S$ must be f_S. Finally, we have to verify that $\underline{k}=\langle k^{op},f_S\rangle$ is a morphism of TrS: $f_S(\delta'_{e'}(s'))=f_S(\bigcup_{x:(h^{op})^{-1}(x)=e'}\delta_x(s'))=\bigcup_{x:(h^{op})^{-1}(x)=e'}f_S(\delta_x(s'))$; so, $f_S(\delta'_{e'}(s'))\subseteq\bigcup_{x:(h^{op})^{-1}(x)=e'}\delta''_{f_G^{-1}(x)}(f_S(s'))$; finally, since $f_G^{-1}(x)=(k^{op})^{-1}((h^{op})^{-1}(x))$, the latter colapses onto $\delta''_{(k^{op})^{-1}(e')}(f_S(s'))$. □

Definition 3.1.7. Let $trs=\langle G,S,\delta\rangle$ be a transition system. The *menu map* $M_{trs}:S\rightarrow\mathbf{2}^E$ induced by trs is defined by $M_{trs}(s)=\{e\in E:\delta_e(s)\neq\emptyset\}$. □

Fact 3.1.8. Let h:trs→trs' be a transition system morphism. Then, the following condition holds: $h_G^{-1}(M_{trs}(s)) \subseteq M_{trs'}(h_S(s))$.

Proof: Let $e' \in h_G^{-1}(M_{trs}(s))$. Then: $e'=h_G^{-1}(e)$ for some $e \in M_{trs}(s)$, $\delta_e(s) \neq \emptyset$, $h_S(\delta_e(s)) \neq \emptyset$, $\delta'_{h_G^{-1}(e)}(h_S(s)) \neq \emptyset$, $h_G^{-1}(e) \in M_{trs'}(h_S(s))$, and, finally, $e' \in M_{trs'}(h_S(s))$. □

3.2 Computations

In order to prepare the ground for establishing the envisaged temporal institution in the next section, it is convenient to study the properties of computations of transition systems.

Definition 3.2.1. A *computation system* cos=<G,S,Ξ> is a triple consisting of a set G, a set S, and a subset Ξ of $\{\xi \in [\mathbb{Z} \to S \otimes E \otimes S]: \forall n \in \mathbb{Z}\ (\xi_{n+1})_1=(\xi_n)_3\}$, where $E=\mathbf{2}^G$. □

As in transition systems, the elements of G are called *prime events*, those of E *events* and those of S *states*. Each element ξ of Ξ is called a *computation*: a sequence of triples, each of them a *transition* of the form <s,e,r> with s,r∈S and e∈E.

Definition 3.2.2. A *computation system morphism* h:<G,S,Ξ>→<G',S',Ξ'> is a pair $<h_G,h_S>$ where $h_G:G' \to G$ and $h_S:S \to S'$ are total maps such that $\{h(\xi):\xi \in \Xi\} \subseteq \Xi'$, where $h(\xi)_n=<h_S((\xi_n)_1),h_G^{-1}((\xi_n)_2),h_S((\xi_n)_3)>$ for each $n \in \mathbb{Z}$. □

We denote in the sequel $\{h(\xi):\xi \in \Xi\}$ by h(Ξ). Thus, the computation system morphism condition is written $h(\Xi) \subseteq \Xi'$.

Fact/Definition 3.2.3. Computation systems and their morphisms constitute a concrete category CoS over $Set^{op} \times Set$ with respect to the functor GS':CoS→Set^{op}×Set mapping each cos=<G,S,Ξ> onto the pair (G,S), and each h:<G,S,Ξ>→<G',S',Ξ'> onto the pair $(h_G^{op}:G \to G', h_S:S \to S')$. □

Definition 3.2.4. The *indexing functor* Gen':CoS→Set^{op} is $\pi_1 \circ GS'$. □

Therefore, the functor Gen' maps each cos=<G,S,Ξ> onto G, and each h:<G,S,Ξ>→<G',S',Ξ'> onto $h_G^{op}:G \to G'$.

Fact 3.2.5. The indexing functor Gen':CoS→Set^{op} is a cofibration. Given h:G→G' of Set^{op} and a computation system cos=<G,S,Ξ>, the cocartesian morphism for h and cos is $\underline{h}_{cos}$:cos→cos' defined as follows: (i) cos'=<G',S,Ξ'> with $\Xi'=\{\underline{h}_{cos}(\xi): \xi \in \Xi\}$; and (ii) $\underline{h}_{cos}=<(\underline{h}_{cos})_G,(\underline{h}_{cos})_S>$ with $(\underline{h}_{cos})_G:G' \to G$ given by $(\underline{h}_{cos})_G=h^{op}$ and $(\underline{h}_{cos})_S:S \to S'$ given by $(\underline{h}_{cos})_S=id_S$.

Proof: Clearly, $\underline{h}_{cos}$ is a morphism in CoS. Moreover, $\underline{h}_{cos}$ is cocartesian: given a computation system cos''=<G'',S'',Ξ''>, f:<G,S,Ξ>→<G'',S'',Ξ''> of CoS and k:G'→G'' of Set^{op} such that $f_G=h^{op} \circ k^{op}$, there is a unique

$\underline{k}$:<G',S',Ξ'>→<G'',S',Ξ''> such that f= $\underline{k}$ ∘ $\underline{h}_{cos}$ with $\underline{k}_G=k^{op}$. Indeed, for the composition to hold $\underline{k}_S$ must be f_S. Finally, we have to verify that $\underline{k}$=<k^{op},f_S> is a morphism of CoS: trivial once we recognize that by the definition of $\underline{k}$ we have f(ξ)=$\underline{k}$($\underline{h}_{cos}$(ξ)) for every ξ∈Ξ. □

The set of computations Ξ_{trs} induced by trs=<G,S,δ> is {ξ∈[ℤ→S⊗E⊗S]: ∀n∈ℤ $(\xi_n)_3 \in \delta_{(\xi_n)_2}((\xi_n)_1)$ and $(\xi_{n+1})_1=(\xi_n)_3$}. In this way we can establish a functor from TrS into CoS:

Fact/Definition 3.2.6. The functor C:TrS→CoS maps each transition system trs=<G,S,δ> onto the computation system <G,S,Ξ_{trs}> and each transition system morphism h from trs=<G,S,δ> into trs'=<G',S',δ'> onto the computation system morphism h:<G,S,Ξ_{trs}>→<G',S',$\Xi_{trs'}$> induced by the same pair <h_G,h_S>.

Proof: It is necessary to verify that the transition system morphism condition implies the computation system morphism condition. Indeed, let ξ∈Ξ_{trs}. We prove that h(ξ)∈$\Xi_{trs'}$: ∀n $(\xi_n)_3 \in \delta_{(\xi_n)_2}((\xi_n)_1)$, ∀n $h_S((\xi_n)_3) \in h_S(\delta_{(\xi_n)_2}((\xi_n)_1))$, ∀n $h_S((\xi_n)_3) \in \delta'_{h_G(((\xi_n)_2)}(h_S((\xi_n)_1))$, and, so, h(ξ)∈$\Xi_{trs'}$. □

Definition 3.2.7. The category TCoS of *computation systems generated by transition systems* is the full subcategory of CoS whose elements are images of elements of TrS by C. The *unfolding functor* U:TrS→TCoS is defined as C. □

Fact 3.2.8. The functor U has a right adjoint T:TCoS→TrS.

Proof: For each <G,S,Ξ> choose a <G,S,δ> such that U(<G,S,δ>)= <G,S,Ξ> and consider the identity <id_G,id_S>:U(<G,S,δ>)→<G,S,Ξ>. We prove that the pair <<G,S,δ>,<id_G,id_S>> is co-free over <G,S,Ξ> with respect to U. Given a transition system trs'=<G',S',δ'> and a computation morphism f:U(<G',S',δ'>)→<G,S,Ξ> there is a unique transition system morphism g:<G',S',δ'>→<G,S,δ> such that <id_G,id_S> ∘ U(g)=f, namely g=<f_G,f_S>. Indeed, $f_S(\delta'_{e'}(s')) \subseteq \delta_{f_G(e')}(f_S(s'))$ because: if $t' \in \delta_{e'}(s')$ there is ξ∈$\Xi_{trs'}$ such that ξ_n=<s',∅,s'> for n<0, ξ_0=<s',e',t'>, and ξ_n=<t',∅,t'> for n>0; so, $f(\xi)_n$=<f_S(s'),∅,f_S(s')> for n<0, $f(\xi)_0$=<f_S(s'),f_G^{-1}(e'),f_S(t')>, and $f(\xi)_n$=<f_S(t'),∅,f_S(t')> for n>0; thus, finally, since f(ξ) must be a computation of <G,S,Ξ>=U(<G,S,δ>), $f_S(t') \in \delta_{f_G(e')}(f_S(s'))$. □

The functor T:TCoS→TrS maps each computation system cos=<G,S,Ξ> in TCoS onto the transition system <G,S,δ> such that $\delta_e(s)$= {$(\xi_n)_3$: ξ∈Ξ, n∈ℤ, $(\xi_n)_1$=s, $(\xi_n)_2$=e} and each computation system morphism h:<G,S,Ξ>→<G',S',Ξ'> onto the transition system morphism h:<G,S,δ>→<G',S',δ'> induced by the same pair <h_G,h_S>. Let TGen': TCoS→Set^{op} be the restriction of Gen' to TCoS. Then, we have:

Fact 3.2.9. U$T:TCoS→TrS is a fibered adjunction with respect to

<TGen',Gen> over Set^{op}.

Proof: Clearly, Gen ∘ T=TGen'; TGen' ∘ U=Gen; TGen'(<id_G,id_S>: U ∘ T(cos) → cos) is the identity morphism at TGen'(cos); and Gen(<id_G,id_S>:trs→T ∘ U(trs)) is the identity morphism at Gen(trs). □

Fact 3.2.10. U preserves cocartesian morphisms.

Proof: This is just an application of the general result on preservation of (co)cartesian morphisms by fibered adjoints — see [Winskel 87]. □

As we shall see in the next section, this result is useful when proving the satisfaction condition.

4 The institution

4.1 The category of signatures

Definition 4.1.1. The category Sig of *signatures* is the category Set. □

Each element g of a signature G is intended to correspond to a prime event. Although there are no elements in a signature for referring to events in general (composed of several prime events), no limitation arises since appropriate formulae can be used to this end: see below.

Fact 4.1.2. The category Sig is small complete and cocomplete. □

4.2 The syntax functor

Definition 4.2.1. The *propositional, linear temporal language* L_G induced by a signature G is the least set satisfying the following requirements:

- $\nabla g, \Diamond g \in L_G$, provided that $g \in G$;
- $(\neg \gamma), (\mathbf{X}\gamma), (\mathbf{G}\gamma) \in L_G$, provided that $\gamma \in L_G$;
- $(\gamma_1 \Rightarrow \gamma_2), (\gamma_1 \mathbf{U} \gamma_2) \in L_G$, provided that both $\gamma_1, \gamma_2 \in L_G$. □

Therefore, the language only provides the means for referring directly to the enabling or occurrence of prime events. Events in general may be referred to by use of logical connectives. As usual, we extend L_G with the following abbreviations:

- $(\gamma_1 \vee \gamma_2) = ((\neg \gamma_1) \Rightarrow \gamma_2)$
- $(\gamma_1 \wedge \gamma_2) = (\neg ((\neg \gamma_1) \vee (\neg \gamma_2)))$
- $(\gamma_1 \mid \gamma_2) = (((\neg \gamma_1) \wedge \gamma_2) \vee (\gamma_1 \wedge (\neg \gamma_2)))$
- $(\gamma_1 \Leftrightarrow \gamma_2) = ((\gamma_1 \wedge \gamma_2) \vee ((\neg \gamma_1) \wedge (\neg \gamma_2)))$
- $(\mathbf{F}\gamma) = (\neg (\mathbf{G}(\neg \gamma)))$

For instance $(\nabla g_1 \wedge \nabla g_2)$ refers to the occurrence of an event e such that both $g_1, g_2 \in e$.

Definition 4.2.2. The *translation map* $h^\wedge: L_G \to L_{G'}$ induced by a signature morphism $h: G \to G'$ is inductively defined as follows:

- $h^\wedge(\nabla g) = \nabla h(g)$;
- $h^\wedge(\Diamond g) = \Diamond h(g)$;
- $h^\wedge(\neg \gamma) = (\neg h^\wedge(\gamma))$;
- $h^\wedge(\gamma_1 \Rightarrow \gamma_2) = (h^\wedge(\gamma_1) \Rightarrow h^\wedge(\gamma_2))$;
- $h^\wedge(\mathbf{X}\gamma) = (\mathbf{X}h^\wedge(\gamma))$;
- $h^\wedge(\mathbf{G}\gamma) = (\mathbf{G}h^\wedge(\gamma))$;
- $h^\wedge(\gamma_1 \mathbf{U} \gamma_2) = (h^\wedge(\gamma_1) \mathbf{U} h^\wedge(\gamma_2))$. □

Fact/Definition 4.2.3. The *syntax functor* $Sen: Sig \to Set$ maps each signature G onto the propositional, linear temporal language L_G, and each signature morphism $h: G \to G'$ onto the translation map $h^\wedge: L_G \to L_{G'}$. □

4.3 The semantic functor

Definition 4.3.1. Let G be a signature. The category TrS_G of *G-transition systems* and *G-transition morphisms* is the fiber $Gen^{-1}(G)$. The category CoS_G of *G-computation systems* and *G-computation system morphisms* is the fiber $(Gen')^{-1}(G)$. □

Fact/Definition 4.3.2. The *translation functor* $h^\sim: TrS_{G'} \to TrS_G$ induced by a signature morphism $h: G \to G'$ maps each G'-transition system $trs' = \langle G', S', \delta' \rangle$ onto the codomain of the cocartesian morphism $\underline{h}\,;^{op}_{trs'}$ for h^{op} and trs', and maps each G'-transition morphism $g': trs'_1 \to trs'_2$ onto the unique G-transition system morphism $h^\sim(g'): h^\sim(trs'_1) \to h^\sim(trs'_2)$ such that $h^\sim(g') \circ \underline{h}\,;^{op}_{trs'_1} = \underline{h}\,;^{op}_{trs'_2} \circ g'$. □

That is, $h^\sim$ is the functor induced by h^{op} with respect to the cofibration Gen.

Fact/Definition 4.3.3. The *semantic functor* $Mod: Sig \to Cat^{op}$ maps each signature G onto the subcategory TrS_G of TrS, and each signature morphism $h: G \to G'$ onto the translation functor $h^\sim: TrS_{G'} \to TrS_G$. □

Given a signature morphism $h: G \to G'$ and a G'-transition system $trs' = \langle G', S', \delta' \rangle$, the G-transition system $h^\sim(trs')$ is called the *h-reduct* of trs'. Recalling the characterization of the cofibration Gen, it is easy to conclude: (i) the state space of the h-reduct is identical to the given state space; (ii) the reduct transition map for each $e \in G$ is defined by $\delta_e(s) = \bigcup_{x: h^{-1}(x)=e} \delta_x(s)$: that is, when e happens on s in the h-reduct, the resulting state may be r iff there is an event $x = h(e)$ such that the resulting state in trs' when x happens on s may be r. Furthermore, the cocartesian morphism $\underline{h}\,;^{op}_{trs'}$ for h^{op} and trs' is the pair $\langle h, id_S \rangle$.

For proving the satisfaction condition we shall also need:

Fact/Definition 4.3.4. The *translation functor* $h^\simeq: CoS_{G'} \to CoS_G$ induced by a

signature morphism $h:G\to G'$ maps each G'-computation system $cos'=\langle G',S',\Xi'\rangle$ onto the codomain of the cocartesian morphism $\underline{h}\,;^{op}_{cos'}$ for h^{op} and cos', and maps each G'-computation system morphism $g':cos'_1\to cos'_2$ onto the unique G-computation morphism $h^{\approx}(g):h^{\sim}(cos'_1)\to h^{\sim}(cos'_2)$ such that $h^{\approx}(g')\circ \underline{h}\,;^{op}_{cos'_1} = \underline{h}\,;^{op}_{cos'_2}\circ g'$. □

That is, $h^{\approx}$ is the functor induced by h^{op} with respect to the cofibration Gen'.

4.4 Satisfaction

Definition 4.4.1. Let G be a signature, φ a formula of L_G, and trs a G-transition system. The *satisfaction* of φ by $\xi\in\Xi_{trs}$ at $n\in\mathbb{Z}$ (written $\xi\models_n\varphi$) is inductively defined as follows:

- $\xi\models_n\nabla g$ iff $g\in(\xi_n)_2$;
- $\xi\models_n\Diamond g$ iff there is $e\in M_{trs}((\xi_n)_1)$ such that $g\in e$;
- $\xi\models_n(\neg\gamma)$ iff not $\xi\models_n\gamma$;
- $\xi\models_n(\gamma_1\Rightarrow\gamma_2)$ iff $\xi\models_n\gamma_2$ or not $\xi\models_n\gamma_1$;
- $\xi\models_n(\mathbf{X}\gamma)$ iff $\xi\models_{n+1}\gamma$;
- $\xi\models_n(\mathbf{G}\gamma)$ iff $\xi\models_m\gamma$ for every $m\geq n$;
- $\xi\models_n(\gamma_1\mathbf{U}\gamma_2)$ iff $\xi\models_m\gamma_2$ for some $m\geq n$ and $\xi\models_i\gamma_1$ for every i such that $n\leq i<m$.

Definition 4.4.2. Let G be a signature, φ a formula of L_G, and trs a G-transition system. We say that trs *satisfies* φ (written $trs\models_G\varphi$) iff $\xi\models_n\varphi$ for every $\xi\in\Xi_{trs}$ and every $n\in\mathbb{Z}$.

4.5 Structurality

Having chosen the relevant category of signatures and established the suitable syntax and semantic functors, it remains to check if the satisfaction condition is fulfilled. To this end we need the following lemmas:

Fact 4.5.1. Let G,G' be signatures, $h:G\to G'$ a signature morphism, and trs' a G'-transition system. Then, $\Xi_{h^{\sim}(trs')}=\{\underline{h}\,;^{op}_{\Xi_{trs'}}(\xi'):\xi'\in\Xi_{trs'}\}$.

Proof: Since the left adjoint of a fibered adjunction preserves cocartesian morphisms, $U(h^{\sim}(trs'))=h^{\approx}(U(trs'))$. Therefore, since the set of computations of $h^{\approx}(U(trs'))$ is given by $\{\underline{h}\,;^{op}_{\Xi_{trs'}}(\xi'):\xi'\in\Xi_{trs'}\}$ and the set of computations of $U(h^{\sim}(trs'))$ is $\Xi_{h^{\sim}(trs')}$ the thesis holds. □

Fact 4.5.2. Let G,G' be signatures, $h:G\to G'$ a signature morphism, and trs' a G'-transition system. Then, $M_{h^{\sim}(trs')}(s)=h^{-1}(M_{trs'}(s))$.

Proof: $e\in M_{h^{\sim}(trs')}(s)$ iff $(\bigcup_{x:h^{-1}(x)=e'}\delta'_x(s))\neq\emptyset$ iff $\exists x\ e=h^{-1}(x)$ and $\delta'_x(s)\neq\emptyset$ iff

$e \in \{h^{-1}(x)\colon \delta'_x(s) \neq \emptyset\}$ iff $e \in h^{-1}(M_{trs'}(s))$. □

We are now ready to establish the *satisfaction condition*:

Fact 4.5.3. Let G,G' be signatures, $h\colon G \to G'$ a signature morphism, φ a formula of L_G, and trs' a G'-transition system. Then, $trs' \models_{G'} Sen(h)(\varphi)$ iff $Mod(h)(trs') \models_G \varphi$.

Proof:

We prove by induction on the structure of the formula φ that $\xi' \models_n h^\wedge(\varphi)$ iff $\underline{h}\,;^{op}_{trs'(\xi')} \models_n \varphi$:

• $\xi' \models_n h^\wedge(\nabla g)$ iff $\xi' \models_n \nabla h(g)$ iff $h(g) \in (\xi'_n)_2$ iff $g \in h^{-1}((\xi'_n)_2)$ iff $g \in ((\underline{h}\,;^{op}_{trs'(\xi')})_n)_2$ iff $\underline{h}\,;^{op}_{trs'(\xi')} \models_n \nabla g$.

• $\xi' \models_n h^\wedge(\Diamond g)$ iff $\xi' \models_n \Diamond h(g)$ iff $\exists\, e' \in M_{trs'}((\xi'_n)_1)\ h(g) \in e'$ iff $\exists\, e \in h^{-1}(M_{trs'}((\xi'_n)_1))\ g \in e$ iff $\exists\, e \in M_{h^\sim(trs')}([(\xi'_n)_1])\ g \in e$ iff $\exists\, e \in M_{h^\sim(trs')}(((\underline{h}\,;^{op}_{trs'(\xi')})_n)_1)\ g \in e$ iff $\underline{h}\,;^{op}_{trs'(\xi')} \models_n \Diamond g$.

• $\xi' \models_n h^\wedge(\neg\gamma)$ iff $\xi' \models_n (\neg\ h^\wedge(\gamma))$ iff not $\xi' \models_n h^\wedge(\gamma)$ iff not $\backslash O \backslash AR(\underline{h}\,;^{op})_{trs'(\xi')} \models_n \gamma$ iff $\underline{h}\,;^{op}_{trs'(\xi')} \models_n (\neg\gamma)$.

• $\xi' \models_n h^\wedge(\gamma_1 \Rightarrow \gamma_2)$ iff $\xi' \models_n (h^\wedge(\gamma_1) \Rightarrow h^\wedge(\gamma_2))$ iff $\xi' \models_n h^\wedge(\gamma_2)$ or not $\xi' \models_n h^\wedge(\gamma_1)$ iff $\underline{h}\,;^{op}_{trs'(\xi')} \models_n \gamma_2$ or not $\underline{h}\,;^{op}_{trs'(\xi')} \models_n \gamma_1$ iff $\underline{h}\,;^{op}_{trs'(\xi')} \models_n (\gamma_1 \Rightarrow \gamma_2)$.

• $\xi' \models_n h^\wedge(\mathbf{X}\gamma)$ iff $\xi' \models_n (\mathbf{X} h^\wedge(\gamma))$ iff $\xi' \models_{n+1} h^\wedge(\gamma)$ iff $\underline{h}\,;^{op}_{trs'(\xi')} \models_{n+1} \gamma$ iff $\underline{h}\,;^{op}_{trs'(\xi')} \models_n (\mathbf{X}\gamma)$.

• $\xi' \models_n h^\wedge(\mathbf{G}\gamma)$ iff $\xi' \models_n (\mathbf{G} h^\wedge(\gamma))$ iff $\forall m \geq n\ \xi' \models_m h^\wedge(\gamma)$ iff $\forall m \geq n\ \underline{h}\,;^{op}_{trs'(\xi')} \models_m \gamma$ iff $\underline{h}\,;^{op}_{trs'(\xi')} \models_n (\mathbf{G}\gamma)$.

• $\xi' \models_n h^\wedge(\gamma_1 \mathbf{U} \gamma_2)$ iff $\xi' \models_n (h^\wedge(\gamma_1) \mathbf{U} h^\wedge(\gamma_2))$ iff $\xi' \models_m h^\wedge(\gamma_2)$ for some $m \geq n$ and $\xi' \models_i h^\wedge(\gamma_1)$ for every i such that $n \leq i < m$ iff $\underline{h}\,;^{op}_{trs'(\xi')} \models_m \gamma_2$ for some $m \geq n$ and $\underline{h}\,;^{op}_{trs'(\xi')} \models_i \gamma_1$ for every i such that $n \leq i < m$ iff $\underline{h}\,;^{op}_{trs'(\xi')} \models_n (\gamma_1 \mathbf{U} \gamma_2)$. □

5 Concluding remarks

We have shown how to build an institution for a simple logic of behaviour using a cofibration from a given (flat) category of transition systems into the envisaged category of signatures. The adopted propositional, linear temporal logic was chosen for its simplicity while still distinguishing between event occurrence and enabling. The satisfaction condition was proved using a fibered adjunction between transition systems and their computations.

It is important to note that some desirable universal requirements (eg like those related to the frame principle as used in [Sernadas *et al* 92b]) may destroy the satisfaction condition if incorporated into the logic. In that case, a weaker form of structurality may still be required, along the lines of

[Maibaum *et al* 90]. Note also the "open nature" of transition systems. For instance, $(\neg \nabla g)$ is semantically entailed by $(\nabla g \Rightarrow (\mathbf{X}\nabla g'))$, since the empty event should always be enabled. A related problem concerns "fairness assumptions". The functor U we adopted for extracting the computations from each transition system is rather unfair. It is conceivable to adopt slightly different unfolding functors introducing some degree of fairness (for instance, precluding computations where the empty event is repeated for ever while other events are enabled). One wonders what will be the impact in the satisfaction condition.

Given a specification (theory presentation) $<G,\Gamma>$ within the proposed institution, one may ask what should be its semantics. The traditional "loose" approach of associating to $<G,\Gamma>$ the set Γ^* of all G-transition systems that satisfy each axiom in Γ is not appropriate if we want to find a "canonical" model representing in some "extreme" way all models of Γ (along the lines of [Fiadeiro *et al* 93]). Such a model is useful when providing a semantic map for a specification language built within the proposed institution.

For the semantics of life-cycles (sequences of events) it is very simple to obtain such a "canonical" model: just collect all life-cycles that satisfy the specification. But for the semantics of transition systems (or their computations) we have to face the additional problem of choosing the state space. In [Caleiro 93] it is shown how to obtain such a canonical transition system from a given temporal specification, by adapting the synthesis algorithm in [Manna and Wolper 84]. The categorial characterization of this "operational" semantics of temporal logic is under investigation.

Acknowledgments

This work was partially supported by the JNICT Projects OBCALC and FAC3, the ESPRIT BRA's IS-CORE and COMPASS, and the ESDI Project OBLOG. The authors are grateful to their many colleagues in these projects for many rewarding interactions, and, specially, to José Luiz Fiadeiro, Hans-Dieter Ehrich and Carlos Caleiro.

References

[Adámek *et al* 90]
J.Adámek, H.Herrlich and G.Strecker, *Abstract and Concrete Categories*, Wiley, 1990.

[Caleiro 93]
C.Caleiro, "Operational Semantics of Temporal Object Specifications", INESC/DMIST Research Report, June 1993, presented at the COMPASS GM'93, Dresden.

[Fiadeiro *et al* 93]
J.Fiadeiro, J.F.Costa, A.Sernadas and T.Maibaum, "Process Semantics of Temporal Logic Specification", in M.Bidoit and C.Choppy (eds), *Recent Trends in Data Type Specification: 8th Workshop on Specification of Abstract Data Types - Selected Papers*, Springer-Verlag, 1993, 40-65.

[Goguen and Burstall 84]
J.A.Goguen and R.M.Burstall, "Introducing Institutions", in E.Clarke (ed), *Logics of Programming Workshop*, Springer-Verlag, 1984, 221-256.

[Goldblatt 87]
R.Goldblatt, *Logics of Time and Computation*, CSLI, 1987.

[Maibaum *et al* 90]
T.Maibaum, J.Fiadeiro and M.Sadler, "Stepwise Program Development in π-Institutions", Imperial College, Department of Computing Research Report, 1990.

[Manna and Pnueli 92]
Z.Manna and A.Pnueli, *The Temporal Logic of Reactive and Concurrent Systems*, Springer-Verlag, 1992.

[Manna and Wolper 84]
Z.Manna and P.Wolper, "Synthesis of Communicating Processes from Temporal Logic Specifications", *ACM Transactions on Programming Languages and Systems*, 1984, 68-93.

[SernadasA and Ehrich 91]
A.Sernadas and H.-D.Ehrich, "What is an Object, After All?", in R.Meersman, W.Kent and S.Khosla (eds), *Object Oriented Databases: Analysis, Design and Construction*, North-Holland, 1991, 39-69.

[SernadasA *et al* 89a]
A.Sernadas, J.Fiadeiro, C.Sernadas and H.-D.-Ehrich, "Basic Building Blocks of Information Systems", in E.Falkenberg and P.Lindgreen (eds), *Information System Concepts: An In-depth Analysis*, North-Holland, 1989, 225-246.

[SernadasA *et al* 89b]
A.Sernadas, J.Fiadeiro, C.Sernadas and H.-D.-Ehrich, "Abstract Object Types: A Temporal Perspective", in B.Banieqbal, H.Barringer and A.Pnueli (eds), *Temporal Logic in Specification*, Springer-Verlag, 1989, 324-350.

[SernadasA *et al* 92a]
A.Sernadas, J.F.Costa, J.L.Fiadeiro and H.-D.Ehrich, "Object Template Institution", Dagstuhl Workshop on Foundations of Information Systems Specification and Design, March 16-19, 1992.

[SernadasA *et al* 92b]
A.Sernadas, C.Sernadas and J.F.Costa, "Object Specification Logic",

INESC/DMIST Research Report, June 1992, *submitted*.

[Tarlecki *et al* 91]
A.Tarlecki, R.M.Burstall and J.A.Goguen, "Some Fundamental Algebraic Tools for the Semantics of Computation. Part III: Indexed Categories", *Theoretical Computer Science*, 1991, 239-264.

[Winskel 84]
G.Winskel, "Synchronization Trees", *Theoretical Computer Science*, 34, 1984.

[Winskel 85]
G.Winskel, "Categories of Models for Concurrency", in S.D.Brookes et al (eds), *Proceedings of the Seminar on Concurrency*, Springer-Verlag, 1985, 246-267.

[Winskel 87]
G.Winskel, "Petri Nets, Algebras, Morphisms and Compositionality", *Information and Computation*, March, 1987.