Complete Calculi for Structured Specifications in Fork Algebra

Carlos Gustavo Lopez Pombo[1,3] and Marcelo Fabiùn Frias[2,3]

[1] Department of computer science, Facultad de ciencias exactas y naturales,
Universidad de Buenos Aires
[2] Department of Computer Engineering,
Buenos Aires Institute of Technology (ITBA)
[3] CONICET
clpombo@dc.uba.ar, mfrias@itba.edu.ar

Abstract. In previous articles we presented $\mathbf{A}r_{\mathbf{g}}entum$, a tool for reasoning across heterogeneous specifications based on the language of fork algebras. $\mathbf{A}r_{\mathbf{g}}entum$'s foundations were formalized in the framework of institutions. The formalization made simple to describe a methodology capable of producing a complete system desription from partial views, eventually written in different logical languages.

Structured specifications were introduced by Sannella and Tarlecki and extensively studied by Borzyszkowski. The latter also presented conditions under which the calculus for structured specifications is complete. Using fork algebras as a "universal" institution capable of representing expressive logics (such as dynamic and temporal logics), requires using a fork language that includes a reflexive-transitive closure operator. The calculus thus obtained does not meet the conditions required by Borzyszkowski.

In this article we present structure building operators (SBOs) over fork algebras, and provide a complete calculus for these operators.

1 Introduction and Motivation

Modeling languages such as the Unified Modeling Language (UML) [1] allow us to model a system through various diagrams. Each diagram provides a view of the system under development. This view-centric approach to software modeling has its advantages and disadvantages. Two advantages are clear: *a*) decentralization of the modeling process. Several engineers may be modeling different views of the same system simultaneously, and *b*) separation of concerns is enforced.

At the same time this modeling process evolved, several results were produced on the interpretability of logics to extensions of the theory of fork algebras [2]. An interpretation of a logic L to fork algebras consists on a mapping $T_L : Sen_L \to Sen_{FA}$ satisfying the following interpretability condition:

$$\Gamma \models_L \alpha \iff \{\, T_L(\gamma) \mid \gamma \in \Gamma \,\} \vdash_{FA} T_L(\alpha) \ .$$

So far, interpretability results have been produced for classical first-order logic with equality [2, Cap. 5], monomodal logics and propositional dynamic logic

[2, Cap. 6], first-order dynamic logic [3], propositional linear temporal logic [4] and first-order linear temporal logic [5]. Other attractive features of this class of algebras are that they are isomorphic to algebras whose domain is a set of binary relations, and that they posses a finite equational calculus [2, Cap. 4].

The idea of having heterogeneous specifications and reasoning across them is not new. A vast amount of work on the subject has been done based on Goguen and Burstall's notion of *institution* [6]. Institutions capture in an abstract way the model theory of a logic. They can be related by means of different kinds of mappings such as institution morphisms [6] and institution representations [7]. These mappings between institutions are extensively discussed by Tarlecki in [8]. Representations allow us to encode poorer institutions into a richer one. In [8], Tarlecki goes even further in presenting the way heterogeneous specifications must be manipulated when he writes:

> "... this suggests that we should strive at a development of a convenient to use proof theory (with support tools!) for a sufficiently rich "universal" institution, and then reuse it for other institutions linked to it by institution representations."

These results constitute the foundations of the $\mathbf{A}r_{\mathbf{g}}entum$ project, presented in [9]. $\mathbf{A}r_{\mathbf{g}}entum$ is a CASE tool aimed at the analysis of heterogeneous models of software. A system description is a collection of theory presentations coming from different logics, and analysis of the heterogeneous model is achieved by interpreting the presentations to fork algebras and analyzing the resulting fork-algebraic specification by available tool support.

An additional concern is how to deal with structured specifications. In practice, specifications are built modularly. In [10], Borzyszkowski gave a set of structure building operations (SBOs) and proved that under certain conditions structured specifications over a given logic can be translated, by means of the application of a representation map, to structured specifications over another logic. Thus, an approach that pretends to act as a foundation of a heterogeneous framework following Tarleckis approach (for example, the fork algebras) should support structured specifications. Also in [10], Borzyszkowski presented a logical system for SBOs, as well as an extensive discussion on the conditions under which that calculus is complete. One of these conditions is that the underlying institution must have a complete calculus. This is indeed the case for fork algebras. The other conditions establish requirements which unfortunately the calculus for fork algebras does not meet.

When working with heterogeneous specifications within a "universal" institution, Borzyszkowski's conditions are hard to satisfy. A logic powerful enough to interpret other logics useful in software specification must be expressive enough. Thus, it is unlikely that such a logical system could satisfy conditions such as compactness, interpolation, infinite conjunction and implications, or interesting combinations of them. In [9] we showed why fork algebras are appropriate as a "universal" institution. For example, surveying interpretability results like those for first-order dynamic logic [3], it is easy to see that any candidate to "universal" institution must have an expressive power capable of characterizing, for

instance, reflexive-transitive closure, thus forcing the introduction of some kind of infinitary rule.

In this paper we present a complete calculus for SBOs over fork algebras, and discuss its scope and limitations under several conditions.

The article is organized as follows: In Sec. 2 we present fork algebras as the logical system we will use as universal institution. In Sec. 3 we provide the basic definitions such as institution, entailment system, and structure building operations. In Sec. 4 we develop the contribution of the article by analyzing the calculus proposed by Borzyszkowski and discussing its possibilities and limitations. Finally, in Sec. 5 we draw some conclusions.

2 Fork Algebras

Full proper closure fork algebras with urelements (denoted by fPCFAU) are extensions of relation algebras [11]. In order to introduce this class, we introduce first the class of *star proper closure fork algebras with urelements* (denoted by $\star\mathsf{PCFAU}$).

Definition 1. *Let U be a nonempty set. A $\star\mathsf{PCFAU}$ is a two sorted structure $\langle U, 2^{U\times U}, \cup, \cap, {}^{-}, \emptyset, U\times U, \circ, Id, {}^{\smile}, \nabla, {}^{\diamond}, {}^{*}, \star \rangle$ such that*

- *$\star : U \times U \to U$ is one to one, but not surjective,*
- *Id is the identity relation on the set U,*
- *$\cup$, $\cap$ and ${}^{-}$ stand for set union, intersection and complement relative to $U \times U$, respectively,*
- *$x^{\diamond}$ is the set choice operator defined by the condition:*

$$x^{\diamond} \subseteq x \text{ and } |x^{\diamond}| = 1 \quad \Longleftrightarrow \quad x \neq \emptyset,$$

- *$\circ$ is relational composition, ${}^{\smile}$ is transposition, and * is reflexive-transitive closure,*
- *∇, the* fork *operator, is defined by the condition:*

$$S \nabla T = \{\, \langle x, y \star z\rangle \mid \langle x, y\rangle \in S \ \wedge \ \langle x, z\rangle \in T \,\} \ .$$

Notice that $x^{\diamond}$ denotes an arbitrary pair in x. This is why $x^{\diamond}$ is called a *choice* operator. Function $\star$ is used to encode pairs. For example, in the case of the interpretability of forst-order logic in fork algebras, $\star$ is used to group elements of the domain to represent a valuation of the variables ($a_1 \star a_2 \star ... \star a_n$ represents valuations ν satisfying $\nu(v_i) = a_i$). The fact it is not surjective implies the existence of elements that do not encode pairs. These elements, called *urelements*, will be used to represent the elements from the carriers of the translated logics.

Definition 2. *We define $\mathsf{fPCFAU} = \mathsf{Rd}\ \star\mathsf{PCFAU}$, where Rd takes reducts to structures of the form $\langle 2^{U\times U}, \cup, \cap, {}^{-}, \emptyset, U\times U, \circ, Id, {}^{\smile}, \nabla, {}^{\diamond}, {}^{*} \rangle$ (the sort U and the function $\star$ are forgotten).*

We will refer to the carrier of an algebra $\mathcal{A} \in \mathsf{fPCFAU}$ as $|\mathcal{A}|$.

The variety generated by fPCFAU (the class of *full proper closure fork algebras with urelements*) has a complete ([3, Theorem 1]) equational calculus (the ω-calculus for closure fork algebras with urelements – ω-CCFAU) to be introduced next. In order to present the calculus, we provide the grammar for formulas, the axioms of the calculus, and the proof rules. For the sake of simplifying the notation, we will denote the relation $U \times U$ by 1, and the relation $\overline{1 \nabla 1} \cap Id$ by Id_{U}. Relation Id_{U} is the subset of the identity relation that relates the urelements.

Definition 3. *Let $\mathcal{V}$ be a set of relation variables, then the set of ω-CCFAU terms is the smallest set $Term(\mathcal{V})$ satisfying:*

- $\{ \emptyset, 1, Id \} \subseteq Term(\mathcal{V})$,
- *If $x, y \in Term(\mathcal{V})$, then $\{ \breve{x}, x^*, x^\diamond, x \cup y, x \cap y, x \circ y, x \nabla y \} \subseteq Term(\mathcal{V})$.*

Definition 4. *Let $\mathcal{V}$ be a set of relation variables, then the set of ω-CCFAU formulas is the set of identities $t_1 = t_2$, with $t_1, t_2 \in Term(\mathcal{V})$.*

Definition 5. *The identities described in Forms. (1) – (5) are axioms*[1] *of ω-CCFAU.*

1. *A set of identities axiomatizing the relational calculus [11].*
2. *The following axioms for the fork operator:*

$$\begin{aligned} x \nabla y &= (x \circ (Id \nabla 1)) \cap (y \circ (1 \nabla Id)) , \\ (x \nabla y) \circ (z \nabla w)^\smile &= (x \circ \breve{z}) \cap (y \circ \breve{w}) , \\ (Id \nabla 1)^\smile \nabla (1 \nabla Id)^\smile &\leq Id. \end{aligned}$$

3. *The following axioms for the choice operator [12, p. 324]:*

$$x^\diamond \circ 1 \circ \breve{x^\diamond} \leq Id, \quad \breve{x^\diamond} \circ 1 \circ x^\diamond \leq Id, \quad 1 \circ (x \cap x^\diamond) \circ 1 = 1 \circ x \circ 1 \ .$$

4. *The following axioms for the Kleene star:*

$$x^* = Id \ \cup \ x \circ x^*, \quad x^* \circ y \leq y \ \cup \ x^* \circ (\overline{y} \ \cap \ x \circ y) \ .$$

5. *An axiom forcing a nonempty set of urelements.*

$$1 \circ Id_{\mathsf{U}} \circ 1 = 1 \ .$$

Definition 6. *The inference rules for the calculus ω-CCFAU are those of equational logic (see for instance [13, p. 94]), extended by adding the following inference rule*[2]*:*

[1] Since the calculus of relations extends the Boolean calculus, we will denote by $\leq$ the ordering induced by the Boolean calculus in ω-CCFAU. As it is usual, $x \leq y$ is a shorthand for $x \cup y = y$.

[2] Given $i > 0$, by x^i we denote the relation inductively defined as follows: $x^1 = x$, and $x^{i+1} = x \circ x^i$.

$$\frac{\vdash Id \leq y \qquad x^i \leq y \vdash x^{i+1} \leq y \quad (\forall i \in \mathbb{N})}{\vdash x^* \leq y}$$

Notice that only extralogical symbols belong to an equational or first-order signature. Symbols such as $=$ in equational logic, or $\vee$ in first-order logic, have a meaning that is univoquely determined by the carriers and the interpretation of the extralogical symbols. Similarly, once the field of a fPCFAU has been fixed, all the operators can be assigned a standard meaning. This gives rise to the following definition of fPCFAU-signature . Given $\mathfrak{A}$ a proper closure fork algebra, then $\leq$ is set inclusion.

Definition 7. *A* fPCFAU *signature is a set of function symbols* $\{f_j\}_{j\in\mathcal{J}}$*. Each function symbol comes equipped with its arity. Notice that since* fPCFAU*s have only one sort, the arity is a natural number.*

The set of fPCFAU signatures will be denoted as $Sign_{\mathsf{fPCFAU}}$. Actually, in order to interpret the logics mentioned in Sec. 1, constant relational symbols (rather than functions in general) suffice. Since new operators may be necessary in order to interpret new logics in the future, signatures will be allowed to contain functions of arbitrary rank.

In order to extend the definitions of terms (Def. 3) and formulas (Def. 4) to fPCFAU signatures, we need to add the following rule:

- If $t_1, \ldots, t_{ari(f_j)} \in Term(\mathcal{V})$, then $f_j(t_1, \ldots, t_{arity(f_j)}) \in Term(\mathcal{V})$ (for all $j \in \mathcal{J}$).

If $\Sigma \in Sign_{\mathsf{fPCFAU}}$, the set of Σ-terms will be denoted as $Term_\Sigma$. In the same way, Sen_Σ will denote the set of equalities between Σ-terms (i.e. the set of Σ-formulas).

Definition 8. *Let* $\Sigma = \{f_j\}_{j\in\mathcal{J}} \in Sign_{\mathsf{fPCFAU}}$*, then* $\mathcal{M} = \left\langle \mathcal{P}, \{\underline{f_j}\}_{j\in\mathcal{J}} \right\rangle \in Mod_\Sigma$ *iff* $\mathcal{P} \in$ fPCFAU*, and* $\underline{f_j} : |\mathcal{P}|^{arity(f_j)} \to |\mathcal{P}|$*, for all* $j \in \mathcal{J}$*. Then, we denote by* $m_\mathcal{M} : Term_\Sigma \to |\mathcal{P}|$ *the function that interprets terms in model* $\mathcal{M}$*.*

Definition 9. *Let* $\Sigma \in Sign_{\mathsf{fPCFAU}}$*, then* $\models^\Sigma_{\mathsf{fPCFAU}} \subseteq Mod_\Sigma \times Sen_\Sigma$ *is defined as follows:* $\mathcal{M} \models^\Sigma_{\mathsf{fPCFAU}} t_1 = t_2$ *iff* $m_\mathcal{M}(t_1) = m_\mathcal{M}(t_2)$*.*

3 Institutions and Structured Specifications

The theory of institutions was presented by Goguen and Burstall in [6]. Institutions provide a formal and generic definition of what a logical system is, and of how specifications in a logical system can be structured [14]. Institutions have evolved in a number of directions, from an abstract theory of software specification and development [15] to a very general version of abstract model theory [16], and offered a suitable formal framework for addressing heterogeneity [17,18], including applications to the UML [19]. The following definitions were taken from [7].

Definition 10. [Institution]
An institution *is a structure of the form* $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *satisfying the following conditions:*

- Sign *is a category of signatures,*
- $\mathbf{Sen} : \mathsf{Sign} \to \mathsf{Set}$ *is a functor (let* $\Sigma \in |\mathsf{Sign}|$*, then* $\mathbf{Sen}(\Sigma)$ *returns the set of* Σ*-sentences),*
- $\mathbf{Mod} : \mathsf{Sign}^{\mathsf{op}} \to \mathsf{Cat}$ *is a functor (let* $\Sigma \in |\mathsf{Sign}|$*, then* $\mathbf{Mod}(\Sigma)$ *returns the category of* Σ*-models),*
- $\{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}$*, where* $\models^{\Sigma} \subseteq |\mathbf{Mod}(\Sigma)| \times \mathbf{Sen}(\Sigma)$*, is a family of binary relations,*

and for any signature morphism $\sigma : \Sigma \to \Sigma'$*,* Σ*-sentence* $\phi \in \mathbf{Sen}(\Sigma)$ *and* Σ'*-model* $\mathcal{M}' \in |\mathbf{Mod}(\Sigma)|$ *the following* $\models$*-invariance condition holds:*

$$\mathcal{M}' \models^{\Sigma'} \mathbf{Sen}(\sigma)(\phi) \quad \textit{iff} \quad \mathbf{Mod}(\sigma^{\mathsf{op}})(\mathcal{M}') \models^{\Sigma} \phi \ .$$

Let $\Sigma \in |\mathsf{Sign}|$ and $\Gamma \subseteq \mathbf{Sen}(\Sigma)$, then we define the functor $\mathbf{Mod}(\Sigma, \Gamma)$ as the full subcategory of $\mathbf{Mod}(\Sigma)$ determined by those models $\mathcal{M} \in |\mathbf{Mod}(\Sigma)|$ such that for all $\gamma \in \Gamma$, $\mathcal{M} \models^{\Sigma} \gamma$. In addition, it is possible to define a relation $\models^{\Sigma}$ between sets of formulas and formulas in the following way: let $\alpha \in \mathbf{Sen}(\Sigma)$, then:

$$\Gamma \models^{\Sigma} \alpha \ \text{ if and only if } \ \mathcal{M} \models^{\Sigma} \alpha \ \text{ for all } \mathcal{M} \in |\mathbf{Mod}(\Sigma, \Gamma)|.$$

Definition 11. [Entailment system]
An entailment system *is a structure of the form* $\langle \mathsf{Sign}, \mathbf{Sen}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *satisfying the following conditions:*

- Sign *is a category of signatures,*
- $\mathbf{Sen} : \mathsf{Sign} \to \mathsf{Set}$ *is a functor (let* $\Sigma \in |\mathsf{Sign}|$*, then* $\mathbf{Sen}(\Sigma)$ *returns the set of* Σ*-sentences),*
- $\{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}$*, where* $\vdash^{\Sigma} \subseteq 2^{\mathbf{Sen}(\Sigma)} \times \mathbf{Sen}(\Sigma)$*, is a family of binary relations such that for any* $\Sigma, \Sigma' \in |\mathsf{Sign}|$*,* $\{\phi\} \cup \{\phi_i\}_{i \in \mathcal{I}} \subseteq \mathbf{Sen}(\Sigma)$*,* $\Gamma, \Gamma' \subseteq \mathbf{Sen}(\Sigma)$ *the following conditions are satisfied:*
  1. *reflexivity:* $\{\phi\} \vdash^{\Sigma} \phi$*,*
  2. *monotonicity: if* $\Gamma \vdash^{\Sigma} \phi$ *and* $\Gamma \subseteq \Gamma'$*, then* $\Gamma' \vdash^{\Sigma} \phi$*,*
  3. *transitivity: if* $\Gamma \vdash^{\Sigma} \phi_i$ *for all* $i \in \mathcal{I}$ *and* $\{\phi_i\}_{i \in \mathcal{I}} \vdash^{\Sigma} \phi$*, then* $\Gamma \vdash^{\Sigma} \phi$*, and*
  4. $\vdash$*-translation: if* $\Gamma \vdash^{\Sigma} \phi$*, then for any morphism* $\sigma : \Sigma \to \Sigma'$ *in* Sign*,* $\mathbf{Sen}(\sigma)(\Gamma) \vdash^{\Sigma'} \mathbf{Sen}(\sigma)(\phi)$*.*

Definition 12. *Let* $\langle \mathsf{Sign}, \mathbf{Sen}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *be an entailment system, then* Th*, its category of theories, is a pair* $\langle \mathcal{O}, \mathcal{A} \rangle$ *such that:*

- $\mathcal{O} = \{ \langle \Sigma, \Gamma \rangle \mid \Sigma \in |\mathsf{Sign}| \text{ and } \Gamma \subseteq \mathbf{Sen}(\Sigma) \}$*, and*
- $\mathcal{A} = \left\{ \sigma : \langle \Sigma, \Gamma \rangle \to \langle \Sigma', \Gamma' \rangle \;\middle|\; \begin{array}{l} \langle \Sigma, \Gamma \rangle, \langle \Sigma', \Gamma' \rangle \in \mathcal{O}, \\ \sigma : \Sigma \to \Sigma' \text{ is a morphism in } \mathsf{Sign} \text{ and} \\ \text{for all } \gamma \in \Gamma, \Gamma' \vdash^{\Sigma'} \mathbf{Sen}(\sigma)(\gamma) \end{array} \right\}$.

In addition, if a morphism $\sigma : \langle \Sigma, \Gamma \rangle \to \langle \Sigma', \Gamma' \rangle$ satisfies $\mathbf{Sen}(\sigma)(\Gamma) \subseteq \Gamma'$ it is called *axiom preserving*. This defines the category Th_0 by keeping only those morphisms of Th that are axiom preserving. It is easy to notice that Th_0 is a complete subcategory of Th. Now, if we consider the definition of **Mod**, extended to signatures and set of sentences, we get a functor $\mathbf{Mod} : \mathsf{Th}^{\mathsf{op}} \to \mathsf{Cat}$ defined as follows: let $T = \langle \Sigma, \Gamma \rangle \in |\mathsf{Th}|$, then $\mathbf{Mod}(T) = \mathbf{Mod}(\Sigma, \Gamma)$.

Definition 13. [Logic]
A logic *is a structure of the form* $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *satisfying the following conditions:*

- $\langle \mathsf{Sign}, \mathbf{Sen}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *is an entailment system,*
- $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *is an institution, and*
- *the following* soundness *condition is satisfied: for any* $\Sigma \in |\mathsf{Sign}|$, $\phi \in \mathbf{Sen}(\Sigma)$, $\Gamma \subseteq \mathbf{Sen}(\Sigma)$, $\Gamma \vdash^{\Sigma} \phi \Longrightarrow \Gamma \models^{\Sigma} \phi$.

A logic is complete *if in addition the following condition is also satisfied: for any* $\Sigma \in |\mathsf{Sign}|$, $\phi \in \mathbf{Sen}(\Sigma)$, $\Gamma \subseteq \mathbf{Sen}(\Sigma)$, $\Gamma \vdash^{\Sigma} \phi \Longleftarrow \Gamma \models^{\Sigma} \phi$.

Next we provide some definitions that will be useful in further sections.

Definition 14. [Interpolation and weak interpolation]
An institution $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *has the* interpolation property *if and only if for any* $t'_1 : \Sigma_1 \to \Sigma', t'_2 : \Sigma_2 \to \Sigma'$ *pushout in* Sign *for* $t_1 : \Sigma \to \Sigma_1, t_2 : \Sigma \to \Sigma_2$, *and* $\varphi_i \in \mathbf{Sen}(\Sigma_i)$ *for* $i = 1, 2$, *if* $\mathbf{Sen}(t'_1)(\varphi_1) \models^{\Sigma'} \mathbf{Sen}(t'_2)(\varphi_2)$, *then there exists* $\varphi \in \mathbf{Sen}(\Sigma)$ *(called the interpolant of* φ_1 *and* φ_2*) such that* $\varphi_1 \models^{\Sigma_1} \mathbf{Sen}(t_1)(\varphi)$ *and* $\mathbf{Sen}(t_2)(\varphi) \models^{\Sigma_2} \varphi_2$.

In a similar way, it is said to have the weak interpolation property *if and only if for any* $t'_1 : \Sigma_1 \to \Sigma', t'_2 : \Sigma_2 \to \Sigma'$ *pushout in* Sign *for* $t_1 : \Sigma \to \Sigma_1, t_2 : \Sigma \to \Sigma_2$, *and* $\varphi_i \in \mathbf{Sen}(\Sigma_i)$ *for* $i = 1, 2$, *if* $\mathbf{Sen}(t'_1)(\varphi_1) \models^{\Sigma'} \mathbf{Sen}(t'_2)(\varphi_2)$, *then there exists* $\Gamma \subseteq \mathbf{Sen}(\Sigma)$ *(called the interpolant of* φ_1 *and* φ_2*) such that* $\varphi_1 \models^{\Sigma_1} \mathbf{Sen}(t_1)(\Gamma)$ *and* $\mathbf{Sen}(t_2)(\Gamma) \models^{\Sigma_2} \varphi_2$.

Definition 15. [Weak amalgamation]
An institution $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *has the* weak amalgamation property *if and only if for any* $t'_1 : \Sigma_1 \to \Sigma', t'_2 : \Sigma_2 \to \Sigma'$ *pushout in* Sign *for* $t_1 : \Sigma \to \Sigma_1, t_2 : \Sigma \to \Sigma_2$, *and for any models* $\mathcal{M}_1 \in |\mathbf{Mod}(\Sigma_1)|$ *and* $\mathcal{M}_2 \in |\mathbf{Mod}(\Sigma_2)|$ *such that* $\mathbf{Mod}(t_1)(\mathcal{M}_1) = \mathbf{Mod}(t_2)(\mathcal{M}_2)$, *then there exists* $\mathcal{M}' \in |\mathbf{Mod}(\Sigma')|$ *such that* $\mathbf{Mod}(t'_1)(\mathcal{M}') = \mathcal{M}_1$ *and* $\mathbf{Mod}(t'_2)(\mathcal{M}') = \mathcal{M}_2$.

From now on we will work with specifications as they were defined in [10]. Given an institution $\mathbb{I} = \langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$, for any specification SP over $\mathbb{I}$ we will denote its signature as $\mathbf{Sig}[SP] \in |\mathsf{Sign}|$, and its class of models as $\mathbf{Mod}[SP] \subseteq \mathbf{Mod}(\mathbf{Sig}[SP])$[3]. If $\mathbf{Sig}[SP] = \Sigma$, then we will call SP a Σ-specification, and we will denote the class of Σ-specifications as Spec_{Σ}.

[3] Notice that there are two operators **Mod**, the first one applying to structured specifications and the second one being the model functor of the institution.

Definition 16 [Structure building operations]. *The class of specifications over an institution* $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *are defined as follows*

- *Any pair* $\langle \Sigma, \Gamma \rangle$*, where* $\Sigma \in |\mathsf{Sign}|$ *and* $\Gamma \subseteq \mathbf{Sen}(\Sigma)$*, is a specification, also called* flat specification *or* presentation*, such that:*
  $\mathbf{Sig}[\langle \Sigma, \Gamma \rangle] = \Sigma$, $\mathbf{Mod}[\langle \Sigma, \Gamma \rangle] = |\mathbf{Mod}(\langle \Sigma, \Gamma \rangle)|$
- *Let* $\Sigma \in |\mathsf{Sign}|$*, then given* $SP_1, SP_2 \in \mathsf{Spec}_{\Sigma}$*,* $SP_1 \cup SP_2$ *is a* Σ*-specification such that:*
  $\mathbf{Sig}[SP_1 \cup SP_2] = \Sigma$, $\mathbf{Mod}[SP_1 \cup SP_2] = \mathbf{Mod}[SP_1] \cap \mathbf{Mod}[SP_2]$
- *Let* $\Sigma, \Sigma' \in |\mathsf{Sign}|$*, then given* $SP \in \mathsf{Spec}_{\Sigma}$ *and a morphism* $\sigma : \Sigma \to \Sigma'$*, then* **translate** SP **by** σ *is a* Σ'*-specification such that:*
  $\mathbf{Sig}[\textbf{translate } SP \textbf{ by } \sigma] = \Sigma'$,
  $\mathbf{Mod}[\textbf{translate } SP \textbf{ by } \sigma] = \{\, \mathcal{M}' \mid \mathbf{Mod}(\sigma)(\mathcal{M}') \in \mathbf{Mod}[SP] \,\}$
- *Let* $\Sigma, \Sigma' \in |\mathsf{Sign}|$*, then given* $SP' \in \mathsf{Spec}_{\Sigma'}$ *and a morphism* $\sigma : \Sigma \to \Sigma'$*, then* **derive from** SP' **by** σ *is a* Σ*-specification such that:*
  $\mathbf{Sig}[\textbf{derive from } SP' \textbf{ by } \sigma] = \Sigma$,
  $\mathbf{Mod}[\textbf{derive from } SP' \textbf{ by } \sigma] = \{\, \mathbf{Mod}(\sigma)(\mathcal{M}') \mid \mathcal{M}' \in \mathbf{Mod}[SP'] \,\}$

The operations introduced in the previous definition are referred as structure building operations or SBOs, and express a mechanism to put specifications together in a structured way. The operators **Sig** and **Mod** help us retrieve both the signature and the corresponding class of models for a given structured specification.

Definition 17. *Given* SP_1 *and* SP_2 *specifications, we say that* SP_1 *is equivalent to* SP_2 *(denoted* $SP_1 \equiv SP_2$*) if and only if* $\mathbf{Sig}[SP_1] = \mathbf{Sig}[SP_2]$ *and* $\mathbf{Mod}[SP_1] = \mathbf{Mod}[SP_2]$.

Definition 18. *Given* SP *a* Σ*-specification, and* $\alpha \in |\mathbf{Sen}(\Sigma)|$*.* α *is a semantic consequence of* SP *(denoted* $SP \models_{\Sigma} \alpha$*) if and only if* $\mathbf{Mod}[SP] \models^{\Sigma} \alpha$.[4]

As it was presented in [10] structured specifications have a normal form of the shape **derive from** SP' **by** t, where SP' is a flat specification. This normal form is obtained by the application of the operator **nf** [10, Def. 3.7][5]. Also in [10] the following theorem is proved.

Theorem 1. *Let* SP *be a* Σ*-specification over the institution* $\mathbb{I}$*, if* $\mathbb{I}$ *has the weak amalgamation property, then* $\mathbf{nf}(SP) \equiv SP$.

[4] As $\mathbf{Mod}[SP]$ is a class of models, we define $\mathbf{Mod}[SP] \models^{\Sigma} \alpha$ if and only if for all $\mathcal{M} \in \mathbf{Mod}[SP]$, $\mathcal{M} \models^{\Sigma} \alpha$. Also notice the difference between $\models_{\Sigma}$ and $\models^{\Sigma}$, the first one being the satisfaction relation between structured specifications and formulas, and the second one being the satisfaction relation of the underlying institution.

[5] The intuition behind the operator **nf** is that it flattens the specification by translating the axioms to the "richest" signature using the pushouts in Sign followed by the derivation of the resulting flat specification to a signature having the symbols that must remain visible.

4 Complete Calculi for Structured Specifications in Fork Algebras

In [10] Borzyszkowski presented a calculus for structured specifications and gave sufficient conditions under which it is complete. In this section we will explore conditions under which this calculus is complete when specifications are structured over Fork Algebras.

4.1 Scope and Limitations of Borzyszkowski's Calculus for Structured Specifications

We start by reviewing the scope and limitations of Borzyszkowski's calculus for structured specifications, by analyzing the conditions presented in [10] when they are instantiated for the logic of full proper closure fork algebras with urelements.

Definition 19. *Let* $\mathbb{I} = \langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *be the logic of full proper closure fork algebras with urelements. Then, the following rules, define a* Sign*-indexed family of entailment relations.*

$$\frac{\{SP \vdash_{\Sigma} \psi\}_{\psi \in \Delta} \qquad \Delta \vdash^{\Sigma} \varphi}{SP \vdash_{\Sigma} \varphi}\ [CR] \qquad \frac{\Gamma \vdash^{\Sigma} \varphi}{\langle \Sigma, \Gamma \rangle \vdash_{\Sigma} \varphi}\ [basic]$$

$$\frac{SP' \vdash_{\Sigma'} \mathbf{Sen}(\sigma)(\varphi)}{\textbf{derive from } SP' \textbf{ by } \sigma \vdash_{\Sigma} \varphi}\ [derive]$$

$$\frac{SP_1 \vdash_{\Sigma} \varphi}{SP_1 \cup SP_2 \vdash_{\Sigma} \varphi}\ [sum1] \qquad \frac{SP_2 \vdash_{\Sigma} \varphi}{SP_1 \cup SP_2 \vdash_{\Sigma} \varphi}\ [sum2]$$

$$\frac{SP \vdash_{\Sigma} \varphi}{\textbf{translate } SP \textbf{ by } \sigma \vdash_{\Sigma'} \mathbf{Sen}(\sigma)(\varphi)}\ [translate]$$

Definition 20. *A specification is said to be* finite *if and only if any flat specification* $\langle \Sigma, \Gamma \rangle$ *occurring as part of a structured specification satisfies that* Γ *is finite.*

Proposition 1. *Let* $\mathbb{I} = \langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *be the logic of full proper closure fork algebras with urelements. Then,*

1. *if we restrict the morphisms in* $|\mathsf{Sign}|$ *to be injections,* $\mathbb{I}$ *has the weak interpolation property,*
2. *if we restrict* $|\mathsf{Th}_0|$ *to finite presentations, then for any* $t'_1 : \Sigma_1 \to \Sigma', t'_2 : \Sigma_2 \to \Sigma'$ *pushout in* Sign *for* $t_1 : \Sigma \to \Sigma_1, t_2 : \Sigma \to \Sigma_2$, $\varphi_i \in \mathbf{Sen}(\Sigma_i)$ *for* $i = 1, 2$ *such that* $\mathbf{Sen}(t'_1)(\varphi_1) \models^{\Sigma'} \mathbf{Sen}(t'_2)(\varphi_2)$, *there exists* $\Gamma \subseteq \mathbf{Sen}(\Sigma)$ *such that* $\varphi_1 \models^{\Sigma_1} \mathbf{Sen}(t_1)(\Gamma)$, $\mathbf{Sen}(t_2)(\Gamma) \models^{\Sigma_2} \varphi_2$, *and* Γ *is finite,*
3. $\mathbb{I}$ *has the weak amalgamation property, and*
4. $\mathbb{I}$ *has conjunction and implication.*

Proof. 1. By [20, Coro. 4].

2. By [20, Coro. 4] and considering that theory presentations in $|\mathsf{Th}_0|$ are formed by unconditional equations, and by equational completeness.
3. Let $t'_1 : \Sigma_1 \to \Sigma', t'_2 : \Sigma_2 \to \Sigma'$ be a pushout in Sign for $t_1 : \Sigma \to \Sigma_1, t_2 : \Sigma \to \Sigma_2$, and $\mathcal{M}_1$ and $\mathcal{M}_2$ the models $\langle M_1, \{f_i^{\mathcal{M}_1}\}_{i\in\mathcal{I}_1}\rangle \in |\mathbf{Mod}(\Sigma_1)|$ and $\langle M_2, \{f_i^{\mathcal{M}_2}\}_{i\in\mathcal{I}_2}\rangle \in |\mathbf{Mod}(\Sigma_2)|$ respectively such that $\mathbf{Mod}(t_1)(\mathcal{M}_1) = \mathbf{Mod}(t_2)(\mathcal{M}_2)$. Let $\mathcal{M} = \langle M, \{f_i^{\mathcal{M}}\}_{i\in\mathcal{I}}\rangle \in |\mathbf{Mod}(\Sigma)|$ such that $\mathcal{M} = \mathbf{Mod}(t_1)(\mathcal{M}_1) = \mathbf{Mod}(t_2)(\mathcal{M}_2)$, then define $\mathcal{M}' = \langle M, \{f'^{\mathcal{M}'}_i\}_{i\in\mathcal{I}'}\rangle$ such that:
   - $\mathcal{I}' = t'_1(\mathcal{I}_1) \cup t'_2(\mathcal{I}_2)$,
   - $f'^{\mathcal{M}'}_{t_1\circ t'_1(i)} = f_i^{\mathcal{M}}$, for all$i \in \mathcal{I}$ (notice that $t_1 \circ t'_1 = t_2 \circ t'_2$,
   - $f'^{\mathcal{M}'}_{t'_1(i)} = f_i^{\mathcal{M}_1}$, for all $i \in \mathcal{I}_1/t_1(\mathcal{I})$,
   - $f'^{\mathcal{M}'}_{t'_2(i)} = f_i^{\mathcal{M}_2}$, for all $i \in \mathcal{I}_2/t_2(\mathcal{I})$ (as t'_1, t'_2 is a pushout in Sign for t_1, t_2, then $t'_1(\mathcal{I}_1/t_1(\mathcal{I})) \cap t'_2(\mathcal{I}_2/t_2(\mathcal{I})) = \emptyset$).

   Then, by construction $\mathbf{Mod}(t'_1)(\mathcal{M}') = \mathcal{M}_1$ and $\mathbf{Mod}(t'_2)(\mathcal{M}') = \mathcal{M}_2$.
4. By [21, p. 26], and considering that fPCFAU are full, thus simple, any boolean combination of equations α is equivalent to an equation $T(\alpha) = 1$.

Theorem 2. *Let* $\mathbb{I} = \langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma\in|\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma\in|\mathsf{Sign}|}\rangle$ *be the logic of full proper closure fork algebras with urelements such that* Sign *is restricted to have only injective morphisms,* $\Sigma \in |\mathsf{Sign}|$*, and* SP *be a finite specification such that* $\mathbf{Sig}[SP] = \Sigma$*, then for all* $\alpha \in |\mathbf{Sen}(\Sigma)|$ $SP \vdash_{\Sigma} \alpha$ *if and only if* $SP \models_{\Sigma} \alpha$*.*

Proof. The proof follows the lines of [10, Thm. 3.9]. Rather than using the satisfaction of the interpolation property, we use the satisfaction of the weak interpolation property (which holds by Prop. 1.1), the fact that $\mathbb{I}$ has de weak amalgamation property (Prop. 1.3), that $\mathbb{I}$ has conjunction and implication (Prop. 1.4) and that whenever specifications are finite, the interpolants are also finite (Prop. 1.2).

In Thm. 2 we proved that Borzyszkowski's calculus is complete for the logic of fork algebras. Yet the result is limited to finite presentations. This fact establishes a limitation for fork algebras. Having a complete calculus required us having an inference rule with infinitary many hypothesis in order to characterize reflexive-transitive closure. Thus, working with finite specifications limits the properties that can be proved to those that are consequences of a finite number of axioms. This does not mean that we will not be able to prove any property involving reflexive-transitive closure. There are many properties involving this operator for which a finite set of axioms is sufficient. In Sec. 4.2 we will overcome this limitation.

4.2 A Complete Calculus for Infinite Structured Specifications in Fork Algebras

In this section we will present a complete calculus for structured (not necessarily finite) specifications in fork algebras. The completeness of the calculus will depend on the fact full proper closure fork algebras with urelements (the semantic models we are using) have a complete calculus [22].

As we mentioned above, the use of fork algebras extended with reflexive-transitive closure requires a proof theory that does not meet the conditions imposed by Borzyszkowski in [10]. The calculus we will present will provide the methodological insight in order to carry out proofs in the infinitary setting.

The following definition presents the calculus for infinite structured specifications. It differs from the calculus presented in Def. 19 in two ways: *a*) we added a rule (*[equiv]*) allowing to replace a specification by another, provided that they are equivalent, and *b*) rules *[CR]*, *[sum1]* and *[sum2]* were replaced by a single, and slightly more complex, rule for $\cup$ (*[sum]*).

Definition 21. *Let* $\mathbb{I} = \langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *be the logic of full proper closure fork algebras with urelements. Then, the following rules, define a* Sign*-indexed family of entailment relations.*

$$\frac{\Gamma \vdash^{\Sigma} \varphi}{\langle \Sigma, \Gamma \rangle \vdash_{\Sigma} \varphi} \textit{ [basic]} \qquad \frac{SP_2 \vdash_{\Sigma} \varphi \qquad SP_1 \equiv SP_2}{SP_1 \vdash_{\Sigma} \varphi} \textit{ [equiv]}$$

$$\frac{SP' \vdash_{\Sigma'} \mathbf{Sen}(\sigma)(\varphi)}{\mathbf{derive\ from}\ SP'\ \mathbf{by}\ \sigma \vdash_{\Sigma} \varphi} \textit{ [derive]}$$

$$\frac{\{SP_1 \vdash_{\Sigma} \psi\}_{\psi \in \Delta} \qquad \langle \Sigma, \Delta \rangle \cup SP_2 \vdash_{\Sigma} \varphi}{SP_1 \cup SP_2 \vdash_{\Sigma} \varphi} \textit{ [sum]}$$

$$\frac{SP \vdash_{\Sigma} \varphi}{\mathbf{translate}\ SP\ \mathbf{by}\ \sigma \vdash_{\Sigma'} \mathbf{Sen}(\sigma)(\varphi)} \textit{ [translate]}$$

Theorem 3. *Let* $\langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *be the logic of full proper closure fork algebras with urelements. Let* $\Sigma, \Sigma' \in |\mathsf{Sign}|$, $\Gamma' \in \mathbf{Sen}(\Sigma')$, $\sigma : \Sigma \to \Sigma'$ *a morphism in* $|\mathsf{Sign}|$, $SP \in \mathsf{Spec}_{\Sigma}$ *and* $\langle \Sigma, \Gamma \rangle \in |\mathsf{Th}_0|$. *Then,* $SP \vdash_{\Sigma} \varphi$ *if and only if* $SP \models_{\Sigma} \varphi$.

Proof. The proof of soundness (i.e. that if $\langle \Sigma, \Gamma \rangle \cup SP \vdash_{\Sigma} \varphi$, then $\langle \Sigma, \Gamma \rangle \cup SP \models_{\Sigma} \varphi$) follows by observing that by Def. 16 all the rules are sound. Completeness trivially follows using the rules *[derive]*, *[equiv]* (considering the fact that $\mathbf{nf}(SP) \equiv SP$), and *[basic]*.

Observing the proof a question arises. Is a calculus like this of any utility? This question has two possible answers. From a theoretical point of view the completeness of this calculus reduces directly to the completeness of the calculus of the underlying logic, thus proofs using the structure building operators reduce to proofs using the calculus for flat specifications. From a practical perspective, theorem proving is an essential tool for formally verifying the behavior of software systems. From this methodological point of view the calculus can be very useful in guiding an engineer in proving properties of structured specifications. We stick to the second argument to explain the changes we introduced in Def. 21 with respect to Def. 19.

Borzyszkowski's completeness proof [10] suggests that the proofs of properties of a union of specifications should be organized by resorting to rules *[CR]*, *[sum1]*

and *[sum2]*. This is possible because there is an implicit use of the interpolation property in the elimination of the union of two specifications. In this sense, interpolation is a strong requirement in Borzyszkowski's calculus, even when it is disguised as a combination of weaker properties. In the case of a logic that does not meet this condition, that construction is not possible because the interpolant is not a formula, but a (possibly infinite) set of formulas. To solve this we added rule *[sum]*, which explicits the construction used by Borzyszkowski. On the negative side, the use of the rule *[sum]* eliminates a union between two structured specifications, but introduces another one between a structured specification and a flat one. This responds to the need of using the (possibly infinite) interpolant to complete the proof. This newly added union prevents us from using the structure of the specification as a guide to develop the proof. The inclusion of the rule *[equiv]* gives us a solution (besides the fact that it is needed in the proof of completeness), by enabling the replacement of a given specification by an equivalent one in which the structure can be useful in developing the proof. The next proposition shows some examples of the possibilities that the use of this rule provides.

Proposition 2. [Properties of SBOs]
Let $\mathbb{I} = \langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *be the logic of full proper closure fork algebras with urelements.*

1. $\langle \Sigma', \mathbf{Sen}(\sigma)(\Gamma) \rangle \cup$ **translate** SP **by** $\sigma \equiv$ **translate** $\langle \Sigma, \Gamma \rangle \cup SP$ **by** σ,
2. **derive from** $\langle \Sigma', \mathbf{Sen}(\sigma)(\Gamma) \rangle \cup SP'$ **by** $\sigma \equiv \langle \Sigma, \Gamma \rangle \cup$ **derive from** SP' **by** σ,
3. $\langle \Sigma, \Gamma \rangle \cup (SP_1 \cup SP_2) \equiv (\langle \Sigma, \Gamma \rangle \cup SP_1) \cup SP_2$,
4. $SP_1 \cup SP_2 \equiv SP_2 \cup SP_1$,
5. $(SP_1 \cup SP_2) \cup SP_3 \equiv SP_1 \cup (SP_2 \cup SP_3)$,
6. $\langle \Sigma, \Gamma_1 \rangle \cup \langle \Sigma, \Gamma_2 \rangle \equiv \langle \Sigma, \Gamma_1 \cup \Gamma_2 \rangle$.

Proof. The proofs of 4, 5 and 6 are trivial by Def. 16, and the proof of 3 is an instance of 5. 1 and 2 follows by Def. 16 and set-theoretical reasoning on the classes of models.

We call the reader's attention to the fact that even when we developed this ideas for the particular case of fork algebras, they apply to any language with similar characteristics. Fork algebras served just as a motivation in the formalization of a complete proof calculus for languages that can be considered as "universal" institutions. On the other hand, the reader should notice that the properties presented in Prop. 2 hold, at least, for any institution for which the definition of the operators **Sig**[] and **Mod**[] remain as it was presented in Def. 16.

4.3 A Categorical Characterization of the Structure Building Operations

In this paper we have discussed some problems related to the possibility of having a complete calculus for structured specifications. We now identify another drawback, but this time related to the way structured specifications are defined.

Structured specifications are defined on top of an institution (an appropriate way of formalizing the fact that a specific methodology for software description is defined over an underlying logic). Recalling the definition of structured specifications (as a consequence of the definition of SBOs) and their semantics, it is easy to see that SBOs cannot be characterized by universal constructions in the category Th_0. This is because the operations **derive from** and **translate** alter the relation between the set of axioms in the specification, and its class of models. A problem of this separation is that even when the underlying logic serves as a tool in defining the semantics associated to a structured specification, it is not possible to find a theory (in the underlying logic) which acts as a counterpart of a given structured specification.

The solution we propose is the study of the properties that a logic must have in order to internalize the SBOs as constructions in the category Th_0. In this sense, a possible approach is the search for a logical system in which the effect of applying the operations **derive from** and **translate** on a theory can be characterized by some transformation of the axioms of the specification. Because of the space limitation we will not fully develop this approach in the present work, but we will state some preliminary ideas.

We already mentioned that the motivation behind choosing fork algebras is their expressive power, revealed by the existence of several representation maps from different logics. Fork algebras have a rich model theory in which the logical structure of models is capable of representing the logical structure of models of several logics ubiquitous in software design. For example, [3, Def. 15] shows how first-order dynamic logic is interpreted in fork algebras. In all the extralogical symbols (those that appear in the signature) are interpreted by extending the fork algebraic signature with new constants. As was shown in Def. 8 , fork algebraic constants are interpreted as elements in the domain of models.

Consider the extension of the language of fork algebras presented in Defs. 3 and 4 but replacing the latter definition by the following:

Definition 22. *Let $\mathcal{V}$ be a set of relation variables, then the set $Form(\mathcal{V})$ of first-order ω-*CCFAU *formulas is the smallest set $\mathcal{F}$ such that:*

- *if $t_1, t_2 \in Term(\mathcal{V})$, then $t_1 = t_2 \in \mathcal{F}$,*
- *if $\alpha, \beta \in \mathcal{F}$ and $C \in \mathcal{V}$, then $\neg\alpha, \alpha \vee \beta, (\exists C)\alpha \in \mathcal{F}$.*

This language is just a first order extension of the original equational one presented in Sec. 2. The main properties presented for the equational version also hold for this new version (for example, the fact that there exists a complete calculus for a concrete class of models in which the domain is formed by binary relations). If we restrict the category of signatures to those that only include relational constants and injective morphisms, mapping $\mathcal{T}$ establishes a relation between structured specifications and flat theories in Th_0. We will use the following definition:

Definition 23. *Let $\Sigma = \langle \{C_i\}_{i \in \mathcal{I}} \rangle$, $\Sigma' = \langle \{C'_i\}_{i \in \mathcal{I}'} \rangle$ and $\sigma : \Sigma \to \Sigma'$. We denote by $\mathbf{Sen}(\sigma)^\star$ the backward translation from Σ'-formulas to Σ-formulas*

such that for all $C'_i \in \Sigma'$, if there exists $j \in \mathcal{I}$ such that $\sigma(j) = i$, then C'_i is replaced by C_j in the translation, and left as C'_i otherwise.

Definition 24. $\mathbf{Ax} : \mathsf{Th}_0 \to \mathsf{Set}$ *is the forgetful functor that for any theory in* Th_0 *returns its corresponding set of axioms.*

Definition 25. *Let* $\mathbb{I} = \langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *be the logic of full proper closure fork algebras with urelements,* $\Sigma = \langle \{C_i\}_{i \in \mathcal{I}} \rangle$ *and* $\Sigma' = \langle \{C'_i\}_{i \in \mathcal{I}'} \rangle$ *signatures in* $|\mathsf{Sign}|$*, and* $\sigma : \Sigma \to \Sigma'$ *a morphism in* Sign*.*

- $\mathcal{T}(\langle \Sigma, \Gamma \rangle) = \langle \Sigma, \Gamma \rangle$,
- $\mathcal{T}(\langle \Sigma, \Gamma_1 \rangle \cup \langle \Sigma, \Gamma_2 \rangle) = \langle \Sigma, \Gamma_1 \cup \Gamma_2 \rangle$[6],
- $\mathcal{T}(\mathbf{derive\ from}\ SP'\ \mathbf{by}\ \sigma) = \langle \Sigma, \widehat{\Gamma} \rangle$*, where*

$$\widehat{\Gamma} = Ax \cup \left\{ \left(\exists \{C'_i\}_{i \in \mathcal{I}'/\sigma(\mathcal{I})} \right) \bigwedge_{\alpha \in \mathbf{Sen}(\sigma)^{\star}(\mathbf{Ax}(\mathcal{T}(SP))/Ax)} \alpha \right\} \tag{1}$$

$$Ax = \mathbf{Sen}(\sigma)^{\star}(\mathbf{Ax}(\mathcal{T}(SP)) \cap \mathbf{Sen}(\langle \{C'_{\sigma(i)}\}_{i \in \mathcal{I}} \rangle)) \tag{2}$$

  Equation 2 characterizes those axioms in the structured specification that are expressible over Σ symbols (i.e. those symbols that were not hidden by the operation). Equation 1 states that the set of axioms of the resulting theory is formed by the formulas in Ax and an existential formula replacing those axioms that were not expressible over Σ.
- $\mathcal{T}(\mathbf{translate}\ SP\ \mathbf{by}\ \sigma) = \langle \Sigma', \mathbf{Sen}(\sigma)(\mathbf{Ax}(\mathcal{T}(SP))) \rangle$.

Using the previous definition it is easy to prove the following theorem.

Theorem 4. *Let* $\mathbb{I} = \langle \mathsf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \{\vdash^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|}, \{\models^{\Sigma}\}_{\Sigma \in |\mathsf{Sign}|} \rangle$ *be the logic of full proper closure fork algebras with urelements. Let* $\Sigma \in |\mathsf{Sign}|$*, and* SP *a* Σ*-specification. Then,* $\mathbf{Mod}[SP] = \mathbf{Mod}(\mathcal{T}(SP))$.

A more general result can be obtained if we observe that this result is a direct consequence of the fact that the calculus associated with the underlying institution supports an extension satisfying that: *a*) the semantics of the extralogical symbols are elements of a sort of the models (in our case extralogical symbols are constants and their interpretation are binary relations), and *b*) has some kind of existential quantifier and conjunction.

5 Conclusions

Motivated by the use of extensions of fork algebras we analyzed the work of Borzyszkowski on structured specifications and showed that the conditions imposed to logical systems in order to have a complete calculus are too restrictive.

[6] Notice that $\langle \Sigma, \Gamma_1 \cup \Gamma_2 \rangle$ is the apex of the colimit of the diagram formed by the theories $\langle \Sigma, \Gamma_1 \rangle$, $\langle \Sigma, \Gamma_2 \rangle$, $\langle \Sigma, \emptyset \rangle$, and the morphisms $id_{\Sigma 1} : \langle \Sigma, \emptyset \rangle \to \langle \Sigma, \Gamma_1 \rangle$ and $id_{\Sigma 2} : \langle \Sigma, \emptyset \rangle \to \langle \Sigma, \Gamma_2 \rangle$ in Th_0.

We also presented a complete calculus for structured specifications over the logic of fork algebras. Finally, we described some ideas on how to completely characterize structured specifications in the underlying logic. This last topic was treated in a superficial way due to space limitations but we believe that work in this direction will contribute to characterize in an appropriate way the relation between structured specifications and their underlying logic.

References

1. Booch, G., Rumbaugh, J., Jacobson, I.: The unified modeling language user guide. Addison–Wesley Longman Publishing Co., Inc., Boston (1998)
2. Frias, M.F.: Fork algebras in algebra, logic and computer science. Advances in logic, vol. 2. World Scientific Publishing Co., Singapore (2002)
3. Frias, M.F., Baum, G.A., Maibaum, T.S.E.: Interpretability of first-order dynamic logic in a relational calculus. In: de Swart, H. (ed.) RelMiCS 2001. LNCS, vol. 2561, pp. 66–80. Springer, Heidelberg (2002)
4. Frias, M.F., Lopez Pombo, C.G.: Time is on my side. In: Procs. of RelMiCS 7, pp. 105–111 (2003)
5. Frias, M.F., Lopez Pombo, C.G.: Interpretability of first-order linear temporal logics in fork algebras. JLAP 66(2), 161–184 (2006)
6. Goguen, J.A., Burstall, R.M.: Introducing institutions. In: Hutchison, D., Shepherd, W.D., Mariani, J.A. (eds.) Local Area Networks: An Advanced Course. LNCS, vol. 184, pp. 221–256. Springer, Heidelberg (1985)
7. Meseguer, J.: General logics. In: Procs. of the Logic Colloquium 1987, vol. 129, pp. 275–329. North Holland, Amsterdam (1989)
8. Tarlecki, A.: Moving between logical systems. In: Haveraaen, M., Dahl, O.-J., Owe, O. (eds.) Abstract Data Types 1995 and COMPASS 1995. LNCS, vol. 1130, pp. 478–502. Springer, Heidelberg (1996)
9. Lopez Pombo, C.G., Frias, M.F.: Fork algebras as a sufficiently rich universal institution. In: Johnson, M., Vene, V. (eds.) AMAST 2006. LNCS, vol. 4019, pp. 235–247. Springer, Heidelberg (2006)
10. Borzyszkowski, T.: Logical systems for structured specifications. TCS 286, 197–245 (2002)
11. Tarski, A.: On the calculus of relations. JSL 6(3), 73–89 (1941)
12. Maddux, R.D.: Finitary algebraic logic. Zeitschrift fur Mathematisch Logik und Grundlagen der Mathematik 35, 321–332 (1989)
13. Burris, S., Sankappanavar, H.P.: A course in universal algebra. Graduate Texts in Mathematics. Springer, Berlin (1981)
14. Sannella, D., Tarlecki, A.: Specifications in an arbitrary institution. Information and computation 76(2-3), 165–210 (1988)
15. Tarlecki, A.: Abstract specification theory: an overview. In: Procs. of the NATO Advanced Study Institute on Models, Algebras and Logic of Engineering Software. NATO Science Series, pp. 43–79. IOS Press, Marktoberdorf (2003)
16. Diaconescu, R. (ed.): Institution-independent Model Theory, Studies in Universal Logic, vol. 2. Birkhäuser, Basel
17. Mossakowski, T., Maeder, C., Luttich, K.: The heterogeneous tool set, Hets. In: Grumberg, O., Huth, M. (eds.) TACAS 2007. LNCS, vol. 4424, pp. 519–522. Springer, Heidelberg (2007)

18. Tarlecki, A.: Towards heterogeneous specifications. In: Gabbay, D., de Rijke, M. (eds.) Frontiers of Combining Systems. Studies in Logic and Computation, vol. 2, pp. 337–360. Research Studies Press (2000)
19. Cengarle, M.V., Knapp, A., Tarlecki, A., Wirsing, M.: A heterogeneous approach to UML semantics. In: Degano, P., De Nicola, R., Meseguer, J. (eds.) Concurrency, Graphs and Models. LNCS, vol. 5065, pp. 383–402. Springer, Heidelberg (2008)
20. Roşu, G., Goguen, J.A.: On equational craig interpolation. Journal of Universal Computer Science 6(1), 194–200 (2000)
21. Tarski, A., Givant, S.: A formalization of set theory without variables. American Mathematical Society Colloqium Publications, Providence (1987)
22. Lopez Pombo, C.G.: Fork algebras as a tool for reasoning across heterogeneous specifications. PhD thesis, Universidad de Buenos Aires (2007)