Verifying Generative CASL Architectural Specifications

Piotr Hoffman

Warsaw University, Institute of Informatics,
Banacha 2, 02-097 Warszawa, Poland
piotrek@mimuw.edu.pl

Abstract. We present a proof-calculus for architectural specifications, complete w.r.t. their generative semantics. Architectural specifications, introduced in the CASL specification language, are a formal mechanism for expressing implementation steps in program development. They state that to implement a needed unit, one may implement some other units and then assemble them in the prescribed manner; thus they capture modular design steps in the development process. We focus on developing verification techniques applicable to full CASL architectural specifications, which involves, inter alia, getting around the lack of amalgamation in the CASL institution.

Introduction

The formal development of a program consists of three phases: writing a requirements specification, implementing it, and proving the correctness of the implementation with respect to the specification (see [ST97]). Architectural specifications [BST99] are meant to be a tool aiding the developer in breaking down the implementation task into independent subtasks. These subtasks can be either further broken down, or directly coded in a chosen programming language. The benefits of dividing a programming task into independent subtasks are obvious: work on parts of the project can proceed in parallel, the project as a whole is easier to maintain and comprehend. Also, the correctness proof can now be constructed from the correctness proofs for the individual subtasks and a correctness proof of the subdivision (i.e., architectural specification) itself.

Our goal is to provide one crucial ingredient necessary for architectural specifications to be used in practice in the formal development of software, namely verification of their correctness. What we do is reduce the correctness problem to the problem of proving semantic consequence in an institution (usually the underlying institution of the architectural specification at issue).

In Sect. 1 we define the syntax and formal semantics of architectural specifications in an institution-independent fashion. In Sect. 2 we then devise a calculus for proving properties of architectural specifications and state its correctness and completeness w.r.t. a generative semantics. This calculus in many respects builds on what has been presented in [Hof01].

Sect. 3 and 4 (together with Sect. 6) form the novel part of the paper. Here, we develop techniques for discharging proof-obligations which arise when using the proof-calculus of Sect. 2. These techniques are institution-independent, i.e., they work in any institution having certain, abstractly formulated, expressibility properties. To extend the applicability of these techniques to institutions without amalgamation, a representation of the institution of interest in some "better" institution is introduced.

In Sect. 5 we define the full CASL logic [CASL00,CASL], including subsorting and structured specifications, and construct a representation of this logic in a many-sorted logic. Finally, in Sect. 6 we prove that this many-sorted logic and the used representation of CASL have all the properties required in order to apply the techniques presented in Sect. 3 and 4.

1 Preliminaries

Some notation: by $f\,[a/x]$ we denote the function which extends f with the mapping $x \mapsto a$; at the same time, if X is a syntactic object, then $X[B/A]$ substitutes all As in X by Bs; for any notion $\mathcal{N}$, we write $\mathcal{N}^*$ for the same notion applied to tuples.

The definition of architectural specifications will be parametrized by an underlying logical system, formalized as an *institution* (see [GB92]). An institution $\mathbf{I}$ is a quadruple $(Sig, \mathbf{Mod}, Sen, \models)$, where:

- Sig is a category of *signatures*;
- $\mathbf{Mod} : Sig^{op} \to \mathcal{CAT}$ is a functor into the quasi-category of all categories;
- $Sen : Sig \to \mathbf{Set}$ is a functor into the category of all small sets (we sometimes drop the functoriality requirement);
- $\models$ is a family $\{\models_\Sigma\}_{\Sigma \in \mathbf{Ob}(Sig)}$, where $\models_\Sigma$ is a relation on $\mathbf{Mod}(\Sigma) \times Sen(\Sigma)$.

These data are subject to the so-called *satisfaction condition*, i.e., for any $\sigma : \Sigma \to \Delta$ in Sig, $\phi \in Sen(\Sigma)$ and $M \in \mathbf{Mod}(\Delta)$ the following equivalence holds:

$$M \models_\Delta Sen(\sigma)(\phi) \iff \mathbf{Mod}(\sigma)(M) \models_\Sigma \phi$$

Objects of $\mathbf{Mod}(\Sigma)$ are called *Σ-models* and elements of $Sen(\Sigma)$ are called *Σ-sentences*. We usually denote the functor $\mathbf{Mod}(\sigma)$ by $\cdot|_\sigma$ and call it the *(σ-) reduct*. If $N|_\sigma = M$, then N is a *σ-extension* of M (and M is *σ-extendible*). For any $\sigma : \Sigma \to \Delta$ in Sig and $\phi \in Sen(\Sigma)$, by $\sigma(\phi)$ we denote $Sen(\sigma)(\phi)$ and we call it the *translation* of ϕ along σ. We will overload the symbols $\mathbf{Mod}$ and $\models$ as follows. For any set Φ of Σ-sentences, $\mathbf{Mod}_\Sigma(\Phi)$ denotes the class of all Σ-models satisfying all sentences from Φ. If Ψ is a set of Σ-sentences, then we write $\Phi \models_\Sigma \Psi$ if $\mathbf{Mod}_\Sigma(\Phi) \subseteq \mathbf{Mod}_\Sigma(\Psi)$.

By Pres($\mathbf{I}$) we denote the category of *presentations*, i.e., pairs (Σ, Φ), where Φ is a finite set of Σ-sentences, a morphism $\sigma : (\Sigma, \Phi) \to (\Delta, \Psi)$ being a signature morphism $\sigma : \Sigma \to \Delta$ such that $\sigma(\Phi) \models_\Delta \Psi$. By $Sig[\_]$ and $Ax[\_]$ we denote the respective components of a presentation.

$ASP ::= \textbf{units}\ UD_1 \ldots UD_n\ \textbf{result}\ UE$
$UD ::= A : USP$
$UE ::= \bullet T \mid \lambda\ A : SP \bullet T$
$T ::= A \mid A(T) \mid T\ \textbf{and}\ T \mid \textbf{reduce}\ T\ \textbf{by}\ \sigma \mid \textbf{let}\ A = T\ \textbf{in}\ T$

Fig. 1. Language of architectural specifications.

We parametrize the definition of architectural specifications by an underlying institution $\mathbf{I} = (Sig, \mathbf{Mod}, Sen, \models)$ and a subcategory of Sig called the *inclusion subcategory*, which we require to be a partial order on the objects of Sig. An inclusion of Σ into Δ is denoted $\iota_{\Sigma,\Delta}$. If this inclusion exists, we say that Σ is a subsignature of Δ. The language of architectural specifications is described by the grammar in Figure 1 (here, the As come from an infinite set of identifiers, the σs are signature morphisms in $\mathbf{I}$, and the SPs are objects in $\mathrm{Pres}(\mathbf{I})$; USPs are introduced below).

Syntactic differences aside, this language of architectural specifications differs somewhat from what is used in CASL. On the one hand, the reduct is more restrictive in CASL. On the other hand, we have disallowed a number of constructs available in CASL: multi-parameter units, unit definitions, fitting morphisms in applications, declarations with imports, and translation. Except for the imports, this does not constitute a big difference, it just makes the analysis and presentation more clear. Also, we have chosen our specifications to be presentations. As we will see in Sect. 5, this is not a real restriction either, since structured specifications fit well into such a framework.

In order to introduce a formal semantics of architectural specifications, we need a few definitions. A *parametric signature* is a pair $P\Sigma = (\Sigma, \Delta)$ such that Σ is a subsignature of Δ. A *unit signature* $U\Sigma$ is either a (regular) signature or a parametric signature. A *parametric unit* over a parametric signature (Σ, Δ) is a partial function U from $\mathbf{Mod}(\Sigma)$ to $\mathbf{Mod}(\Delta)$ which is *persistent*, i.e., for any $M \in \mathrm{dom}(U)$ we have $U(M)|_{\iota_{\Sigma,\Delta}} = M$. A *unit* over a unit signature is a model over that signature, if it is a (regular) signature, or a parametric unit over that signature, if it is a parametric signature. A *unit specification USP* is either a presentation (Σ, Φ), with $[\![USP]\!] = \mathbf{Mod}_\Sigma(\Phi)$, or a *parametric unit specification* $(\Sigma, \Phi) \to (\Delta, \Psi)$, where Σ is a subsignature of Δ, with $[\![USP]\!]$ being the class of all parametric units U over (Σ, Δ) such that $\mathrm{dom}(U) = \mathbf{Mod}_\Sigma(\Phi)$ and for all $M \in \mathrm{dom}(U)$, $U(M) \in \mathbf{Mod}_\Delta(\Psi)$. A *static environment* δ is a pair (B, P), where B maps identifiers to signatures and P maps identifiers to parametric signatures. The domains of B and P are required to be finite and disjoint. A *unit environment* fitting a static environment (B, P) is a map e, sending any identifier $X \in \mathrm{dom}(B)$ to a model over $B(X)$ and any identifier $X \in \mathrm{dom}(P)$ to a parametric unit over $P(X)$. A *unit context* fitting a static environment δ is any set E of environments fitting δ. A *model function of type* (E, Σ) is a total function F from the context E into $\mathbf{Mod}(\Sigma)$. *Unit functions of type* $(E, U\Sigma)$, denoted UF, are defined analogically.

To deal with the **and**-construct, we need two notions. First, we define the *sum* of signatures Σ, Δ to be their least upper bound $\Gamma = \Sigma \cup \Delta$ with respect to the inclusion order, if it exists. Then for any models M over Σ and N over Δ we define their *amalgamation*, denoted $M \oplus N$, to be a Γ-model P satisfying $P|_{\iota_{\Sigma,\Gamma}} = M$ and $P|_{\iota_{\Delta,\Gamma}} = N$, if it exists and is unique. Models are called *amalgamable* if their amalgamation exists.

We use the natural semantics style, with judgments written $\_\_ \vdash \_\_ \rhd \_\_$. We always implicitly add premises stating that objects denoted by δ are static environments, in any pair δ, E, E fits δ, etc.

The semantics in Fig. 2and 3 assigns:

- to a unit declaration, a static environment δ and a unit context E fitting δ;
- to a term, given a static environment δ and a unit context E fitting δ, a signature Σ and a model function F of type (E, Σ);
- to a unit expression, given a static environment δ and a unit context E fitting δ, a unit signature $U\Sigma$ and a unit function UF of type $(E, U\Sigma)$;
- to an architectural specification, a static environment δ, a unit signature $U\Sigma$ and a unit function UF of type $(\mathrm{dom}(UF), U\Sigma)$, where $\mathrm{dom}(UF)$ fits δ.

This semantics, though considerably simplified, is consistent with the formal semantics of CASL, found in [CASL00].

2 Proving Properties of Architectural Specifications

An architectural specification describes the decomposition of a programming task. This may be expressed by a statement of the form $\vdash ASP :: USP$, which says "ASP describes a correct procedure of building a unit $U \in [\![USP]\!]$". This means that, first of all, the procedure itself is correct, i.e., ASP has a denotation $\delta, U\Sigma, UF$; and, second, the procedure gives the required result, i.e., any unit in the image of UF is in $[\![USP]\!]$.

A *context* consists of a finite number of declarations of two types:

- $A :_\Sigma \Phi$, where (Σ, Φ) is a presentation; we then say that $A \in \mathrm{dom}(\Gamma)$ and $\Gamma(A) = \Sigma$,
- $\sigma : A \to B$, where $\Gamma(A) = \mathrm{dom}(\sigma)$ and $\Gamma(B) = \mathrm{cod}(\sigma)$.

A *parametric context* Γ_{par} consists of a finite number of declarations of the form $A :_{\Sigma\to\Delta} \Phi \to \Psi$, where $(\Sigma, \Phi) \to (\Delta, \Psi)$ is a unit specification. For both kinds of contexts, if Γ_1 and Γ_2 coincide on the intersection of their domains, then $\Gamma_1 \cup \Gamma_2$ is defined naturally.

We say that a *model family* $\{M_X\}_{X\in\mathrm{dom}(\Gamma)}$ is *consistent* with a context Γ, written $\{M_X\}_{X\in\mathrm{dom}(\Gamma)} \models \Gamma$, if:

- $X :_\Sigma \Phi$ in Γ implies $M_X \in \mathbf{Mod}_\Sigma(\Phi)$,
- $\sigma : X \to Y$ in Γ implies $M_X = M_Y|_\sigma$.

$$\boxed{\vdash ASP \rhd \delta, U\Sigma, UF}$$

$$\frac{\begin{array}{c}\vdash UD_1 \rhd (B_1, P_1), E_1 \quad \ldots \quad \vdash UD_n \rhd (B_n, P_n), E_n \\ \delta = (B_1 \cup \cdots \cup B_n, P_1 \cup \cdots \cup P_n) \text{ is an } n\text{-element static environment} \\ E = \{\ e_1 \cup \cdots \cup e_n \mid e_1 \in E_1, \ldots, e_n \in E_n\ \} \\ \delta, E \vdash UE \rhd U\Sigma, UF\end{array}}{\vdash \mathbf{units}\ UD_1 \ldots UD_n\ \mathbf{result}\ UE \rhd \delta, U\Sigma, UF}$$

$$\boxed{\vdash UD \rhd \delta, E}$$

$$\frac{}{\vdash A : (\Sigma, \Phi) \rhd (A \mapsto \Sigma, \emptyset), \{\ A \mapsto M \mid M \in \mathbf{Mod}_{\Sigma}(\Phi)\ \}}$$

$$\frac{}{\vdash A : SP \rightarrow SP' \rhd (\emptyset, A \mapsto Sig[SP] \rightarrow Sig[SP']), \{A \mapsto U \mid U \in [\![SP \rightarrow SP']\!]\ \}}$$

$$\boxed{\delta, E \vdash UE \rhd U\Sigma, UF}$$

$$\frac{(B, P), E \vdash T \rhd \Sigma, F}{(B, P), E \vdash \bullet T \rhd \Sigma, F}$$

$$\frac{\begin{array}{c}(B\,[\Sigma/A]\,, P), \{\, e\,[M/A] \mid e \in E,\ M \in \mathbf{Mod}_{\Sigma}(\Phi)\,\} \vdash T \rhd \Delta, F \\ \Sigma \text{ is a subsignature of } \Delta \\ \text{for all } e \in E \text{ and } M \in \mathbf{Mod}_{\Sigma}(\Phi),\ F(e\,[M/A])|_{\iota_{\Sigma,\Delta}} = M\end{array}}{\begin{array}{c}(B, P), E \vdash \lambda A : (\Sigma, \Phi) \bullet T \rhd \\ (\Sigma, \Delta),\ \lambda e \in E \cdot \lambda M \in \mathbf{Mod}_{\Sigma}(\Phi) \cdot F(e\,[M/A])\end{array}}$$

$$\boxed{\delta, E \vdash T \rhd \Sigma, F}$$

$$\frac{A \in \mathrm{dom}(B)}{(B, P), E \vdash A \rhd B(A), \lambda e \in E \cdot e(A)}$$

$$\frac{\begin{array}{c}A \in \mathrm{dom}(P) \text{ and } P(A) = (\Sigma, \Delta) \\ (B, P), E \vdash T \rhd \Sigma, F \\ \text{for all } e \in E,\ F(e) \in \mathrm{dom}(e(A))\end{array}}{(B, P), E \vdash A(T) \rhd \Delta, \lambda e \in E \cdot e(A)(F(e))}$$

$$\frac{\begin{array}{c}\delta, E \vdash T_1 \rhd \Sigma_1, F_1 \quad \text{and} \quad \delta, E \vdash T_2 \rhd \Sigma_2, F_2 \\ \Delta = \Sigma_1 \cup \Sigma_2 \\ \text{for all } e \in E,\ F_1(e) \oplus F_2(e) \text{ exists}\end{array}}{\delta, E \vdash T_1\ \mathbf{and}\ T_2 \rhd \Delta, \lambda e \in E \cdot F_1(e) \oplus F_2(e)}$$

Fig. 2. The semantics of architectural specifications.

$$\frac{\delta, E \vdash T \rhd \Sigma', F}{\delta, E \vdash \mathbf{reduce}\ T\ \mathbf{by}\ \sigma : \Sigma \to \Sigma' \rhd \Sigma, \lambda e \in E \cdot F(e)|_\sigma}$$

$$\frac{\begin{array}{c}(B, P), E \vdash T \rhd \Sigma, F \\ A \notin \mathrm{dom}(B) \cup \mathrm{dom}(P) \\ (B\,[\Sigma/A]\,, P), \{\, e\,[F(e)/A] \mid e \in E\,\} \vdash T' \rhd \Sigma', F'\end{array}}{(B, P), E \vdash \mathbf{let}\ A = T\ \mathbf{in}\ T' \rhd \Sigma', \lambda e \in E \cdot F'(e\,[F(e)/A])}$$

Fig. 3. The semantics of architectural specifications, continued.

A calculus for deriving statements $\vdash ASP :: USP$ is given in Fig. 4 and 5. This calculus is correct, but its completeness depends on additional assumptions. A proof by induction over the structure of (the **result**-term of) the architectural specification (see [Hof01] for a few further hints) is omitted.

Theorem 1. *For any architectural specification ASP and unit specification USP, if $\vdash ASP :: USP$, then $\vdash ASP \rhd \delta, U\Sigma, UF$ for some $\delta, U\Sigma, UF$ such that $UF(e) \in [\![USP]\!]$ for all $e \in \mathrm{dom}(UF)$.* □

Theorem 2. *For any architectural specification ASP and unit specification USP, if $\vdash ASP \rhd \delta, U\Sigma, UF$ for some $\delta, U\Sigma, UF$ such that $UF(e) \in [\![USP]\!]$ for all $e \in \mathrm{dom}(UF)$ and:*

- *no parametric unit in ASP is applied more than once, and*
- *no parametric unit declaration in ASP is inconsistent,*

then $\vdash ASP :: USP$. □

The first of the requirements of Th. 2 effectively means that our calculus is complete w.r.t. a *generative* semantics, i.e., one in which two applications of the same parametric unit to the same argument may yield different results (the semantics of Fig. 2 and 3 is non-generative, but both coincide if no unit is applied more than once). There are also methodological reasons for choosing a generative semantics (e.g., SML is generative [Pau96]; for a discussion of generativity see [Hof01], Sect. 4 *in fine*). The second of the above requirements is purely technical and of little importance, since an inconsistent declaration makes an architectural specification useless in any case.

Of course, the above theorems do not solve our problem fully, since we still need some method for discharging the following types of premises:

type I "for any model family $\{M_X\}_{X \in \mathrm{dom}(\Gamma)} \models \Gamma$, we have $M_A \models_{\Gamma(A)} \phi$", and
type II "for any model family $\{M_X\}_{X \in \mathrm{dom}(\Gamma)} \models \Gamma$, $M_A \oplus M_B$ exists".

(the premise $\phi \models_\Sigma \psi$ is merely a special case of a type I premise).

If the signature category has coproducts, then a type II premise may be transformed into a type IIa premise: "for any model family $\{M_X\}_{X \in \mathrm{dom}(\Gamma)} \models \Gamma$, M_C has a unique η-extension", where η is the universal morphism from a coproduct $\Gamma(A) \sqcup \Gamma(B)$ to the sum $\Gamma(A) \cup \Gamma(B)$.

$$\boxed{\vdash ASP :: USP}$$

$$\frac{\begin{array}{c}\vdash UD_1 :: \Gamma^1_{par}, \Gamma^1 \quad \ldots \quad \vdash UD_n :: \Gamma^n_{par}, \Gamma^n \\ \mathrm{dom}(\Gamma^i_{par}) \cap \mathrm{dom}(\Gamma^j_{par}) = \emptyset \text{ and } \mathrm{dom}(\Gamma^i) \cap \mathrm{dom}(\Gamma^j) = \emptyset \text{ for all } 1 \leq i < j \leq n \\ \Gamma_{par} = \Gamma^1_{par} \cup \cdots \cup \Gamma^n_{par},\ \Gamma = \Gamma^1 \cup \cdots \cup \Gamma^n \text{ and } \mathrm{dom}(\Gamma_{par}) \cap \mathrm{dom}(\Gamma) = \emptyset \\ \Gamma_{par}, \Gamma \vdash UE :: USP\end{array}}{\vdash \textbf{units } UD_1 \ldots UD_n \textbf{ result } UE :: USP}$$

$$\boxed{\vdash UD :: \Gamma_{par}, \Gamma}$$

$$\frac{}{\vdash A : (\Sigma, \Phi) :: \emptyset, \{A :_\Sigma \Phi\}} \qquad \frac{}{\vdash A : (\Sigma, \Phi) \to (\Delta, \Psi) :: \{A :_{\Sigma \to \Delta} \Phi \to \Psi\}, \emptyset}$$

$$\boxed{\Gamma_{par}, \Gamma \vdash UE :: USP}$$

$$\frac{\begin{array}{c}\Gamma_{par}, \Gamma \vdash T :: \Gamma', A \\ \text{for any family } \{M_X\}_{X \in \mathrm{dom}(\Gamma')} \models \Gamma', \text{ we have } M_A \models_{\Gamma'(A)} \Phi\end{array}}{\Gamma_{par}, \Gamma \vdash T :: (\Sigma, \Phi)}$$

$$\frac{\begin{array}{c}\Phi_0 \models_\Sigma \Phi \text{ and } \Phi \models_\Sigma \Phi_0 \\ \Gamma_{par}, \Gamma \cup \{A :_\Sigma \Phi\} \vdash T :: \Gamma', B \\ \Sigma \text{ is a subsignature of } \Gamma'(B) \\ \text{for any model family } \{M_X\}_{X \in \mathrm{dom}(\Gamma')},\ M_A \oplus M_B \text{ exists} \\ \text{for any model family } \{M_X\}_{X \in \mathrm{dom}(\Gamma')} \models \Gamma', \text{ we have } M_B \models_{\Gamma'(B)} \Psi\end{array}}{\Gamma_{par}, \Gamma \vdash \lambda A : (\Sigma, \Phi_0) \bullet T :: (\Sigma, \Phi) \to (\Gamma'(B), \Psi)}$$

$$\boxed{\Gamma_{par}, \Gamma \vdash T :: \Gamma', A}$$

$$\frac{A \in \mathrm{dom}(\Gamma)}{\Gamma_{par}, \Gamma \vdash A :: \Gamma, A}$$

$$\frac{\begin{array}{c}\Gamma_{par}, \Gamma \vdash T :: \Gamma', A' \\ A :_{\Gamma'(A') \to \Delta} \Phi \to \Psi \text{ in } \Gamma_{par} \text{ and } B \notin \mathrm{dom}(\Gamma') \\ \text{for any model family } \{M_X\}_{X \in \mathrm{dom}(\Gamma')} \models \Gamma', \text{ we have } M_{A'} \models_{\Gamma'(A')} \Phi\end{array}}{\Gamma_{par}, \Gamma \vdash A\ (\ T\ ) :: \Gamma' \cup \{B :_\Delta \Psi,\ \iota_{\Gamma'(A') \subseteq \Delta} : A' \to B\}, B}$$

$$\frac{\begin{array}{c}\Gamma_{par}, \Gamma \vdash T_1 :: \Gamma_1, A_1 \text{ and } \Gamma_{par}, \Gamma \vdash T_2 :: \Gamma_2, A_2 \\ \Delta = \Gamma_1(A_1) \cup \Gamma_2(A_2) \\ \mathrm{dom}(\Gamma_1) \cap \mathrm{dom}(\Gamma_2) = \mathrm{dom}(\Gamma) \text{ and } B \notin \mathrm{dom}(\Gamma_1) \cup \mathrm{dom}(\Gamma_2) \\ \text{for any model family } \{M_X\}_{X \in \mathrm{dom}(\Gamma_1 \cup \Gamma_2)} \models \Gamma_1 \cup \Gamma_2,\ M_{A_1} \oplus M_{A_2} \text{ exists}\end{array}}{\Gamma_{par}, \Gamma \vdash T_1 \textbf{ and } T_2 :: \Gamma_1 \cup \Gamma_2 \cup \{B :_\Delta \emptyset,\ \iota_{\Gamma_1(A_1) \subseteq \Delta} : A_1 \to B,\ \iota_{\Gamma_2(A_2) \subseteq \Delta} : A_2 \to B\}, B}$$

Fig. 4. A proof-calculus for architectural specifications.

$$\frac{\begin{array}{c}\Gamma_{par}, \Gamma \vdash T :: \Gamma', A \\ B \notin \mathrm{dom}(\Gamma')\end{array}}{\Gamma_{par}, \Gamma \vdash \textbf{reduce } T \textbf{ by } \sigma : \Sigma \to \Gamma'(A) :: \Gamma' \cup \{B :_{\Sigma} \emptyset, \sigma : B \to A\}, B}$$

$$\frac{\begin{array}{c}\Gamma_{par}, \Gamma \vdash T :: \Gamma', B \\ \Gamma_{par}, \Gamma' \cup \{A :_{\Gamma'(A)} \emptyset, \mathrm{id}_{\Gamma'(A)} : A \to B\} \vdash T' :: \Gamma'', E \\ A \notin \mathrm{dom}(\Gamma') \text{ and } D \notin \mathrm{dom}(\Gamma'')\end{array}}{\Gamma_{par}, \Gamma \vdash \textbf{local } A = T \textbf{ within } T' :: \Gamma''[D/A], E[D/A]}$$

Fig. 5. A proof-calculus for architectural specifications, continued.

3 Type I Premises

In this section we develop a technique for checking premises of the form "for any model family $\{M_X\}_{X \in \mathrm{dom}(\Gamma)} \models \Gamma$, we have $M_A \models_{\Gamma(A)} \phi$". It should be noted that this problem (and this applies to type IIa premises just as well) is so complex since we are not allowed to assume that the underlying institution has logical amalgamation (see below). The reason is the Casl institution lacking logical amalgamation. As suggested in [SMTKH01], to circumvent this problem, we represent the underlying institution in a second institution which enjoys the logical amalgamation property. In the sequel, we will work with two institutions, $\mathbf{I} = (Sig, \mathbf{Mod}, Sen, \models)$ and $\mathbf{I}' = (Sig', \mathbf{Mod}', Sen', \models')$.

An institution $\mathbf{I}$ has *logical amalgamation* if for any context Γ one may compute a sink $\{\alpha_X : \Gamma(X) \to \Sigma\}_{X \in \mathrm{dom}(\Gamma)}$ and a finite set of Σ-sentences Φ such that the set $\{ \{M|_{\alpha_X}\}_{X \in \mathrm{dom}(\Gamma)} \mid M \in \mathbf{Mod}_{\Sigma}(\Phi) \}$ is equal to the set of all model families consistent with Γ. The pair $(\{\alpha_X\}_{X \in \mathrm{dom}(\Gamma)}, \Phi)$ is called an *amalgamating sink*. The idea here is that with logical amalgamation we are able to replace quantification of the form "for any model family $\{M_X\}_{X \in \mathrm{dom}(\Gamma)} \models \Gamma, \ldots$" by "for any model $M \models_{\Sigma} \Phi, \ldots$", with references to M_A in the former replaced by $M|_{\alpha_A}$ in the latter.

A *representation of* $\mathbf{I}$ *in* $\mathbf{I}'$ is a triple $R = (p, s, m)$, where:

- $p : Sig \to \mathrm{Pres}(\mathbf{I}')$ is functor,
- $s : Sen \to p; Sig[\_]; Sen'$ is a natural transformation,
- $m : p^{op}; \mathbf{Mod}' \to \mathbf{Mod}$ is a natural isomorphism.

These data must satisfy the following *representation condition* for all $\Sigma \in \mathbf{Ob}(Sig)$, $M' \in \mathbf{Mod}'(p(\Sigma))$, $\phi \in Sen(\Sigma)$:

$$M' \models'_{Sig[p(\Sigma)]} s_{\Sigma}(\phi) \iff m_{\Sigma}(M') \models_{\Sigma} \phi$$

The above-defined form of representation is in fact very strong, effectively implying that $\mathbf{I}$ is a "subinstitution" of $\mathbf{I}'$ definable in terms of $\mathbf{I}'$-sentences.

For any set Φ of Σ-sentences over $\mathbf{I}$, by $\overline{s_{\Sigma}}(\Phi)$ we denote the set $Ax[p(\Sigma)] \cup s_{\Sigma}(\Phi)$ of $Sig[p(\Sigma)]$-sentences over $\mathbf{I}'$. For any $\mathbf{I}$-context Γ, by $R(\Gamma)$ we denote the context obtained by mapping any declaration $A :_{\Sigma} \Phi$ in Γ to $A :_{Sig[p(\Sigma)]} \overline{s_{\Sigma}}(\Phi)$, and any declaration $\sigma : A \to B$ in Γ to $p(\sigma) : A \to B$.

Lemma 1. *Let Γ be an $\mathbf{I}$-context and $R = (p, s, m) : \mathbf{I} \to \mathbf{I}'$ a representation. Then the map taking any model family $\{M_X\}_{X \in \mathrm{dom}(\Gamma)}$ consistent with $R(\Gamma)$ to the model family $\{m_{\Gamma(X)}(M_X)\}_{X \in \mathrm{dom}(\Gamma)}$ is a well defined bijection onto the set of all model families consistent with Γ.* □

Theorem 3. *Assume that $R : \mathbf{I} \to \mathbf{I}'$ is a representation and that $\mathbf{I}'$ has logical amalgamation. Then, for any $\mathbf{I}$-context Γ with $\Gamma(X) = \Sigma_X$ for all $X \in \mathrm{dom}(\Gamma)$, an amalgamating sink for $R(\Gamma)$ may be computed. Moreover, for any such sink $(\{\tau_X\}_{X \in \mathrm{dom}(\Gamma)}, \Phi)$, for any $A \in \mathrm{dom}(\Gamma)$ and any Σ_A-sentence ϕ, the following conditions are equivalent:*

1. *for any model family $\{M_X\}_{X \in \mathrm{dom}(\Gamma)} \models \Gamma$, we have $M_A \models_{\Sigma_A} \phi$,*
2. *$\Phi \models'_{\Sigma} \tau_A(s_{\Sigma_A}(\phi))$.*

Proof. ((1) $\Rightarrow$ (2)) Take $Q \models'_{\Sigma} \Phi$. By definition of amalgamating sink, we have $\{ Q|'_{\tau_X} \mid X \in \mathrm{dom}(\Gamma) \} \models' R(\Gamma)$. Hence, $\{ m_{\Sigma_X}(Q|'_{\tau_X}) \mid X \in \mathrm{dom}(\Gamma) \} \models \Gamma$. Thus, by (1), $m_{\Sigma_A}(Q|'_{\tau_A}) \models_{\Sigma_A} \phi$. This implies that $Q|'_{\tau_A} \models'_{Sig[p(\Sigma_A)]} s_{\Sigma_A}(\phi)$, and so $Q \models'_{Sig[p(\Sigma_A)]} \tau_A(s_{\Sigma_A}(\phi))$ as required.

((2) $\Rightarrow$ (1)) Take a model family $\{M_X\}_{X \in \mathrm{dom}(\Gamma)} \models \Gamma$. Then by the lemma $\{m^{-1}_{\Sigma_X}(M_X)\}_{X \in \mathrm{dom}(\Gamma)} \models' R(\Gamma)$. By definition of amalgamating sink, there exists a model $Q \models'_{\Sigma} \Phi$ such that $m^{-1}_{\Sigma_X}(M_X) = Q|'_{\tau_X}$ for all $X \in \mathrm{dom}(\Gamma)$. By (2), we have $Q \models'_{\Sigma} \tau_A(s_{\Sigma_A}(\phi))$. We may infer that $m^{-1}_{\Sigma_A}(M_A) = Q|'_{\tau_A} \models'_{Sig[p(\Sigma_A)]} s_{\Sigma_A}(\phi)$, and so $M_A \models_{\Sigma_A} \phi$. □

4 Type IIa Premises

We now develop techniques for checking the second type of premise, i.e., "for any model family $\{M_X\}_{X \in \mathrm{dom}(\Gamma)} \models \Gamma$, M_A is uniquely η-extendible", where η comes from a certain class $\mathcal{E}$ of signature morphisms.

We say that in $\mathbf{I}$ *morphism equalities are expressible* if for any $\alpha, \beta : \Sigma \to \Delta$ one may compute a finite set Φ of Δ-sentences such that for any finite set of Δ-sentences Φ_0 we have $\Phi_0 \models_{\Delta} \Phi$ iff for any $M \models_{\Delta} \Phi_0$ the equality $M|_{\alpha} = M|_{\beta}$ holds. We then say that Φ *expresses* $\alpha = \beta$. Observe that we require much less than $M \models_{\Delta} \Phi$ iff $M|_{\alpha} = M|_{\beta}$ (such a Φ would usually not exist).

Let $\mathcal{E}$ be a class of signature morphisms in $\mathbf{I}$ and $R = (p, s, m) : \mathbf{I} \to \mathbf{I}'$ a representation. Then *$\mathcal{E}$-extendability is expressible in R* if for any $\eta : \Sigma \to \Sigma'$ from $\mathcal{E}$, setting $p(\Sigma) = (\Delta, \Phi)$ and $p(\Sigma') = (\Delta', \Phi')$, one may compute:

1. pairs $(\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n)$, where $\alpha_i, \beta_i : \Delta_i \to \Delta$ are signature morphisms in $\mathbf{I}'$, such that for any Δ-model M, M is $p(\eta)$-extendible iff $M|'_{\alpha_1} = M|'_{\beta_1}$, $\ldots$, $M|'_{\alpha_n} = M|'_{\beta_n}$, and
2. a finite set Ψ' of Δ'-sentences such that for any Δ-model N:

(a) if N has a unique $p(\eta)$-extension satisfying Φ', then any of its $p(\eta)$-extensions satisfies Ψ', and
(b) if N has a $p(\eta)$-extension satisfying Ψ', then it has a unique $p(\eta)$-extension satisfying Φ'.

We then say that (α_1, β_1), ..., (α_n, β_n) *and* Ψ' *express* η*-extendability.* The idea behind this definition is as follows. Condition (1) captures $p(\eta)$-extendability. Condition (2) is designed to transform this into unique η-extendability: point (b) ensures unique η-extendability, while point (a) checks that Ψ' is not overly restrictive.

Theorem 4. *Let $R : \mathbf{I} \to \mathbf{I}'$ be a representation and assume that $\mathbf{I}'$ has logical amalgamation, $\mathcal{E}$-extendability is expressible in R and morphism equalities are expressible in $\mathbf{I}'$. For any $\mathbf{I}$-context Γ with $\Gamma(X) = \Sigma_X$ for all $X \in \mathrm{dom}(\Gamma)$, for any $A \in \mathrm{dom}(\Gamma)$ and any $\eta \in \mathcal{E}$ with $\mathrm{dom}(\eta) = \Sigma_A$ and $\mathrm{cod}(\eta) = \Delta$ the following objects may be computed:*

(A) an amalgamating sink $(\{\tau_X\}_{X \in \mathrm{dom}(\Gamma)}, \Phi)$ for $R(\Gamma)$,
(B) pairs $(\alpha_1, \beta_1), \ldots, (\alpha_n, \beta_n)$ and a set Ψ expressing η-extendability,
(C) sets $\Phi_1, \ldots, \Phi_n$ expressing the morphism equalities $(\alpha_1; \tau_A = \beta_1; \tau_A)$, ..., $(\alpha_n; \tau_A = \beta_n; \tau_A)$, respectively,
(D) an amalgamating sink $(\{\sigma_X\}_{X \in \mathrm{dom}(R\Gamma')}, \Phi')$ for $R\Gamma' = R(\Gamma) \cup \{B :_{Sig[p(\Delta)]} \emptyset,\ p(\eta) : A \to B\}$, where $B \notin \mathrm{dom}(\Gamma)$.

Moreover, for any such objects, the following conditions are equivalent:

1. *for any model family $\{M_X\}_{X \in \mathrm{dom}(\Gamma)} \models \Gamma$, M_A has a unique η-extension,*
2. *(a) $\Phi \models'_\Sigma \Phi_1, \ldots, \Phi \models'_\Sigma \Phi_n$, and*
   (b) $\Phi' \models' \sigma_B(\Psi)$.

Proof. ((1) $\Rightarrow$ (2a)) Take $1 \leq i \leq n$ and arbitrary $M \models'_\Sigma \Phi$. By (A), we have $\{\ M|'_{\tau_X} \mid X \in \mathrm{dom}(\Gamma)\ \} \models' R(\Gamma)$. Thus $\{m_{\Sigma_X}(M|'_{\tau_X}) \mid X \in \mathrm{dom}(\Gamma)\ \} \models \Gamma$, and, by condition (1), there exists a (unique) model N such that $N|_\eta = m_{\Sigma_A}(M|'_{\tau_A})$, which implies that $m^{-1}_{\Sigma_A}(N)|'_{p(\eta)} = M|'_{\tau_A}$. This proves that $M|'_{\tau_A}$ is $p(\eta)$-extendible, and so by (B) we have $M|'_{\alpha_i;\tau_A} = M|'_{\beta_i;\tau_A}$.

By (C), this proves that $\Phi \models'_\Sigma \Phi_i$.

((1) $\Rightarrow$ (2b)) Take any model $M \models' \Phi'$. By (D), we have $\{\ M|'_{\sigma_X} \mid X \in \mathrm{dom}(\Gamma')\ \} \models' R\Gamma'$, and so $\{\ M|'_{\sigma_X} \mid X \in \mathrm{dom}(\Gamma)\ \} \models' R(\Gamma)$. Hence, we have $\{\ m_{\Sigma_X}(M|'_{\sigma_X}) \mid X \in \mathrm{dom}(\Gamma)\ \} \models \Gamma$, and, by condition (1), $m_{\Sigma_A}(M|'_{\sigma_A})$ has a unique η-extension P.

This implies that $M|'_{\sigma_A}$ has a unique $p(\eta)$-extension satisfying Φ'. First, $m^{-1}_{\Sigma_B}(P)$ is such an extension, since $m^{-1}_{\Sigma_B}(P)|'_{p(\eta)} = m^{-1}_{\Sigma_A}(P|_\eta) = m^{-1}_{\Sigma_A}(m_{\Sigma_A}(M|'_{\sigma_A}) = M|'_{\sigma_A}$ and by the definition of m we have $m^{-1}_{\Sigma_B}(P) \in \mathbf{Mod}'(p(\Delta))$. Second, if $Q|'_{p(\eta)} = M|'_{\sigma_A}$ and $Q \in \mathbf{Mod}'(p(\Delta))$, then $m_{\Sigma_B}(Q)$ is well-defined, and $m_{\Sigma_B}(Q)|_\eta = m_{\Sigma_A}(Q|'_{p(\eta)}) = m_{\Sigma_A}(M|'_{\sigma_A})$, so by uniqueness of P we get $P = m_{\Sigma_B}(Q)$, i.e., $Q = m^{-1}_{\Sigma_B}(P)$.

Thus, by (B), we may conclude that any $p(\eta)$-extension of $M|'_{\sigma_A}$ satisfies Ψ', in particular $M|'_{\sigma_B} \models'_{\Delta} \Psi'$, i.e., $M \models' \sigma_B(\Psi')$.

((2) $\Rightarrow$ (1)) Let $\{M_X\}_{X \in \mathrm{dom}(\Gamma)}$ be a model family consistent with Γ.

We will first prove that for any $1 \le i \le n$ we have $m^{-1}_{\Sigma_A}(M_A)|'_{\alpha_i} = m^{-1}_{\Sigma_A}(M_A)|'_{\beta_i}$. Consider the model family $\{m^{-1}_{\Sigma_X}(M_X)\}_{X \in \mathrm{dom}(\Gamma)}$. It is consistent with $R(\Gamma)$, so, by (A), there exists a model $Q \models'_{\Sigma} \Phi$ such that for all $X \in \mathrm{dom}(\Gamma)$ we have $Q|'_{\tau_X} = m^{-1}_{\Sigma_X}(M_X)$. From condition (2a) we infer that $Q \models'_{\Sigma} \Phi_i$ for $1 \le i \le n$. Now (C) gives us $Q|'_{\alpha_i;\tau_A} = Q|'_{\beta_i;\tau_A}$ and, hence, $m^{-1}_{\Sigma_A}(M_A)|'_{\alpha_i} = m^{-1}_{\Sigma_A}(M_A)|'_{\beta_i}$ for $1 \le i \le n$.

From (B) we know that there exists a model N with $N|'_{p(\eta)} = m^{-1}_{\Sigma_A}(M_A)$.

We now prove that $N \models'_{Sig[p(\Delta)]} \Psi$. Define $N_X = m^{-1}_{\Sigma_X}(M_X)$ for $X \in \mathrm{dom}(\Gamma)$ and $N_B = N$. Then $\{N_X\}_{X \in \mathrm{dom}(R\Gamma')} \models' R\Gamma'$. Thus, by (D), there exists a model $Q \models' \Phi'$ such that $Q|'_{\sigma_X} = N_X$ for all $X \in \mathrm{dom}(R\Gamma')$. By condition (2b) we then have $Q \models' \sigma_B(\Psi)$, and so $N = N_B = Q|'_{\sigma_B} \models'_{Sig[p(\Delta)]} \Psi$.

By (B) we may infer that there exists a unique model $M' \in \mathbf{Mod}'(p(\Delta))$ such that $M'|'_{p(\eta)} = N|'_{p(\eta)}$.

We claim that $P = m_{\Delta}(M')$ is the unique model satisfying $P|_{\eta} = M_A$. First, it is well defined, since $M' \in \mathbf{Mod}'(p(\Delta))$. Second, $P|_{\eta} = m_{\Delta}(M')|_{\eta} = m_{\Sigma_A}(M'|'_{p(\eta)}) = m_{\Sigma_A}(N|'_{p(\eta)}) = M_A$. Third, for any P with $P|_{\eta} = M_A$ we have $m^{-1}_{\Delta}(P)|'_{p(\eta)} = m^{-1}_{\Sigma_A}(P|_{\eta}) = m^{-1}_{\Sigma_A}(M_A) = N|'_{p(\eta)}$, and by the uniqueness of M' we have $M' = m^{-1}_{\Delta}(P)$, which means $P = m_{\Delta}(M')$. □

5 The Logic of CASL

In this section we define the institution of CASL logic and prove some basic facts about it (this definition is consistent with what may be found in the CASL semantics [CASL00]). We do this in three steps.

In step one, we construct the institution $\mathrm{CASL}_0 = (Sig_0, \mathbf{Mod_0}, Sen_0, \models^0)$. Its features include multi-sorted first-order logic, both total and partial function symbols, predicates, and sort-generation constraints. It is defined as follows:

- a signature Σ in Sig_0 is a tuple (S, TF, PF, P), where:
  - S is a finite set of *sorts*,
  - $TF = \{TF_{w,s}\}_{w \in S^*, s \in S}$ is a family of finite sets of *names of total function symbols*,
  - $PF = \{PF_{w,s}\}_{w \in S^*, s \in S}$ is a family of finite sets of *names of partial function symbols*,
  - $P = \{P_w\}_{w \in S^*}$ is a family of finite sets of *names of predicates* .

  We additionally require that always $TF_{w,s} \cap PF_{w,s} = \emptyset$. A function symbol $f \in TF_{w,s} \cup PF_{w,s}$ is denoted $f_{w,s}$. A predicate $b \in P_w$ is denoted b_w.
- a signature morphism $\sigma : (S, TF, PF, P) \to (S', TF', PF', P')$ in Sig_0 is a triple $(\sigma^S, \sigma^F, \sigma^P)$, where:
  - $\sigma^S : S \to S'$ is a function,

  - $\sigma^F = \{\sigma^F_{w,s}\}_{w\in S^*, s\in S}$ is a family of functions, $\sigma^F_{w,s} : TF_{w,s} \cup PF_{w,s} \to TF'_{(\sigma^S)^*(w),\sigma^S(s)} \cup PF'_{(\sigma^S)^*(w),\sigma^S(s)}$ preserving the totality of symbols,
  - $\sigma^P = \{\sigma^P_w\}_{w\in S^*}$ is a family of functions, with $\sigma^P_w : P_w \to P'_{(\sigma^S)^*(w)}$.
- for any signature Σ, $\mathbf{Mod_0}(\Sigma)$ is the category of models having:
  - as models M, functions taking:
    - any sort s in Σ to a non-empty carrier set $M[cr, s]$,
    - any function symbol $f_{w,s}$ in Σ, where $w = s_1 \ldots s_n$, to a partial function $M[fn, f_{w,s}] : M[cr, s_1] \times \cdots \times M[cr, s_n] \to M[cr, s]$, the function being total if $f_{w,s}$ is,
    - any predicate b_w in Σ, where $w = s_1 \ldots s_n$, to a set $M[pr, b_w] \subseteq M[cr, s_1] \times \cdots \times M[cr, s_n]$.
  - as (homo)morphisms $h : M \to M'$, families $\{h_s\}_{s\in S^*}$ of functions $h_s : M[cr, s] \to M'[cr, s]$ such that for any tuple of sorts $w = s_1 \ldots s_n$ in Σ and any tuple $(x_1, \ldots, x_n) \in M[cr, s_1] \times \cdots \times M[cr, s_n]$:
    - for any sort s and function symbol $f_{w,s}$ in Σ, if $h_s(M[fn, f_{w,s}](x_1, \ldots, x_n))$ is defined, then so is $M'[fn, f_{w,s}](h_{s_1}(x_1), \ldots, h_{s_n}(x_n))$ and they are both equal,
    - for any predicate b_w in Σ, if $(x_1, \ldots, x_n) \in M[pr, b_w]$, then $(h_{s_1}(x_1), \ldots, h_{s_n}(x_n)) \in M'[pr, b_w]$.
- for any signature morphism $\sigma = (\sigma^S, \sigma^F, \sigma^P) : \Sigma = (S, TF, PF, P) \to \Sigma' = (S', TF', PF', P')$, the reduct functor $\cdot|_\sigma : \mathbf{Mod_0}(\Sigma') \to \mathbf{Mod_0}(\Sigma)$ takes:
  - any model $M' \in \mathbf{Mod_0}(\Sigma')$ to the model $M \in \mathbf{Mod_0}(\Sigma)$ defined by:
    - for all sorts s in Σ, $M[cr, s] = M'[cr, \sigma^S(s)]$,
    - for all function symbols $f_{w,s}$ in Σ, $M[fn, f_{w,s}] = M'[fn, \sigma^F_{w,s}(f)_{(\sigma^S)^*(w),\sigma^S(s)}]$,
    - for all predicates b_w in Σ, $M[pr, b_w] = M'[pr, \sigma^P_w(b)_{(\sigma^S)^*(w)}]$.
  - any Σ'-homomorphism $h' = \{h'_{s'}\}_{s'\in S'} : M' \to N'$ to the Σ-homomorphism $h = h'|_\sigma : M'|_\sigma \to N'|_\sigma$ defined by $h = \{h'_{\sigma^S(s)}\}_{s\in S}$.
- for any signature Σ, $Sen_0(\Sigma)$ is the set of all closed first-order formulas, with atomic formulas built using variables of sorts in Σ, function symbols and predicates in Σ, and equality. Additionally, triples (S_0, F_0, θ), called *sort-generation constraints*, are also in this set, provided that:
  - $\theta : \Sigma_0 \to \Sigma$ is a signature morphism,
  - S_0 is the set of *generated sorts*, i.e., a set of sorts from Σ_0,
  - F_0 is the set of *constructors*, i.e., a set of function symbols from Σ_0.
- for any signature morphism $\sigma : \Sigma \to \Sigma'$, the function $Sen_0(\sigma) : Sen_0(\Sigma) \to Sen_0(\Sigma')$ is defined naturally on first-order sentences and by the rule that $\sigma((S_0, F_0, \theta)) = Sen_0(\sigma)((S_0, F_0, \theta)) = (S_0, F_0, \theta; \sigma)$ on sort-generation constraints.
- the satisfaction relation $M \models^0_\Sigma \phi$ for first-order ϕ is defined as normal first-order satisfaction, with $=$ interpreted strongly and predicates on non-defined values being false. For sort-generation constraints we have $M \models^0_\Sigma (S_0, F_0, \theta)$ if for any $s_0 \in S_0$ and any $x_0 \in (M|_\theta)[cr, s_0]$ there exists a term t_0 over $\mathrm{dom}(\theta)$ of sort s_0 built using function symbols from F_0 and with no variables of sorts from S_0, and a valuation v_0 into $M|_\theta$ such that t_0 under valuation v_0 has value x_0.

Proposition 1. CASL$_0$ *is an institution.* □

For any morphism $\sigma : \Sigma \to \Delta$, the "image" of Σ is denoted $\sigma(\Sigma)$, and its complement $\Delta \setminus \sigma(\Sigma)$. We say that σ is an *inclusion* if $\sigma : \Sigma \to \sigma(\Sigma)$ is the identity.

In the second step we construct the institution CASL$_\leq = (Sig_\leq, \mathbf{Mod}_\leq, Sen_\leq, \models^\leq)$. Here, signatures are additionally equipped with a subsort preorder which, on the model level, is interpreted by injective mappings embedding a subsort, say *Zero*, in a supersort, say *NonNeg* or *NonPos*. By means of an overloading relation it is then ensured that if two function symbols use the same name, say $f : NonNeg \to Elem$ and $f : NonPos \to Elem$, then both are interpreted so that they "coincide" on the carrier of *Zero* (the common subsort). The formal definition follows:

- a signature Σ in $Sig_\leq$ is a tuple $(S, TF, PF, P, \leq)$ such that (S, TF, PF, P) is a signature in Sig_0 and $\leq$ is a reflexive-transitive *subsort preorder* on $S \times S$; for such a signature the *overloading relations* $\sim_F$ and $\sim_P$ are defined by:
  - $f_{w_1,s_1} \sim_F f_{w_2,s_2}$ if there exist w and s with $w \leq^* w_1, w_2$ and $s_1, s_2 \leq s$,
  - $b_{w_1} \sim_P b_{w_2}$ if there exists w with $w \leq^* w_1, w_2$.
- a signature morphism $\sigma : \Sigma = (S, TF, PF, P, \leq) \to \Sigma' = (S', TF', PF', P', \leq')$ in $Sig_\leq$ is a signature morphism $\sigma = (\sigma^S, \sigma^F, \sigma^P) : (S, TF, PF, P) \to (S', TF', PF', P')$ in Sig_0 such that:
  - $s \leq s'$ implies $\sigma^S(s) \leq' \sigma^S(s')$ for all sorts s, s' in Σ,
  - $f_{w_1,s_1} \sim_F f_{w_2,s_2}$ implies $\sigma^F_{w_1,s_1}(f)_{(\sigma^S)^*(w_1),\sigma^S(s_1)} \sim'_F \sigma^F_{w_2,s_2}(f)_{(\sigma^S)^*(w_2),\sigma^S(s_2)}$ for all function symbols f_{w_1,s_1}, f_{w_2,s_2} in Σ,
  - $b_{w_1} \sim_P b_{w_2}$ implies $\sigma^P_{w_1}(b)_{(\sigma^S)^*(w_1)} \sim'_P \sigma^P_{w_2}(b)_{(\sigma^S)^*(w_2)}$ for all predicates b_{w_1}, b_{w_2} in Σ.
- for any signature $\Sigma = (S, TF, PF, P, \leq)$ in $Sig_\leq$, by $\Sigma^\#$ we denote the Sig_0 signature (S, TF', PF', P'), where:
  - $TF'_{w,s} = TF_{w,s} \uplus \{em_{s' \leq s} | w = s' \leq s\}$ (the *embeddings*),
  - $PF'_{w,s} = PF_{w,s} \uplus \{prj_{s' \geq s} | s \leq s' = w\}$ (the *projections*),
  - $P'_w = P_w \uplus \{in_{s,s'} | s' \leq s = w\}$ (the *membership predicates*).

  This naturally extends to a functor $(\_)^\# : Sig_\leq \to Sig_0$.
  By $\Sigma^{\#\#}$ we denote the presentation $(\Sigma^\#, \Phi)$, where Φ consists of the *defining sentences*, which state that:
  1. an embedding of s into s is the identity,
  2. an embedding of s into s' is injective,
  3. the composition of embeddings of s into s' and s' into s'' is equal to the embedding of s into s',
  4. a projection of s' onto s is the least right-sided inverse of the embedding of s into s',
  5. a membership predicate $in_{s,s'}$ holds on an element x of sort s iff the projection of s onto s' is defined on x,
  6. if $f_{w_1,s_1} \sim_F f_{w_2,s_2}$, $w \leq^* w_1, w_2$ and $s_1, s_2 \leq s$, then for any tuple x^* of elements of sorts w, embedding x^* in w_1, applying f_{w_1,s_1} and embedding that in s gives the same result, as embedding x^* in w_2, applying f_{w_2,s_2} and embedding that in s,

7. if $b_{w_1} \sim_P b_{w_2}$ and $w \leq^* w_1, w_2$, then for any tuple x^* of elements of sorts w, embedding x^* in w_1 and checking, whether it is in b_{w_1}, gives the same result as embedding x^* in w_2 and checking, whether it is in b_{w_2}.

Again, this naturally extends to a functor $(\_)^{\#\#} : Sig_{\leq} \to \mathrm{Pres}(\textsc{Casl}_0)$.

- the model functor $\mathbf{Mod}_{\leq}$ is defined as the composition $((\_)^{\#\#})^{op}; \mathbf{Mod_0}$,
- the sentence functor $Sen_{\leq}$ is defined as the composition $(\_)^{\#}; Sen_0$,
- the satisfaction relation is defined by $M \models^{\leq}_{\Sigma} \phi$ if $M \models^{0}_{\Sigma^{\#}} \phi$, for $M \in \mathbf{Mod}_{\leq}(\Sigma)$ and $\phi \in Sen_{\leq}(\Sigma)$.

Proposition 2. *The triple R_0 consisting of:*

- $(\_)^{\#\#} : Sig_{\leq} \to \mathrm{Pres}(\textsc{Casl}_0)$,
- $\{\mathrm{id}_{Sen_{\leq}(\Sigma)}\}_{\Sigma \in \mathbf{Ob}(Sig_{\leq})} : Sen_{\leq} \to (\_)^{\#}; Sen_0$,
- $\{\mathrm{id}_{\mathbf{Mod}_{\leq}(\Sigma)}\}_{\Sigma \in \mathbf{Ob}(Sig_{\leq})} : ((\_)^{\#\#})^{op}; \mathbf{Mod_0} \to \mathbf{Mod}_{\leq}$,

is a representation of $\textsc{Casl}_{\leq}$ *in* $\textsc{Casl}_0$. □

We now present a construction developed in [ST88]. Given an institution $\mathbf{I} = (Sig, \mathbf{Mod}, Sen, \models)$ and a class $\mathcal{D}$ of signature morphisms, define the institution of *structured specifications over* $\mathbf{I}$, denoted $Str_{\mathcal{D}}(\mathbf{I}) = (Sig, \mathbf{Mod}, Sen', \models')$, by structural induction. For any signature Σ, a sentence in $Sen'(\Sigma)$ is:

- a presentation (Σ, Φ) over $\mathbf{I}$; we define $M \models'_{\Sigma} (\Sigma, \Phi)$ if $M \models_{\Sigma} \Phi$,
- **translate** ϕ **by** σ, with $\sigma : \Delta \to \Sigma$ and ϕ a Δ-sentence over $Str_{\mathcal{D}}(\mathbf{I})$; we define $M \models'_{\Delta}$ **translate** ϕ **by** σ if $M|_{\sigma} \models'_{\Sigma} \phi$,
- $\phi_1 \cup \phi_2$, where ϕ_1 and ϕ_2 are Σ-sentences over $Str_{\mathcal{D}}(\mathbf{I})$; we define $M \models'_{\Sigma} \phi_1 \cup \phi_2$ if $M \models'_{\Sigma} \phi_1$ and $M \models'_{\Sigma} \phi_2$,
- **derive** σ **from** ϕ, where $\sigma : \Sigma \to \Delta$ comes from $\mathcal{D}$ and ϕ is a Δ-sentence over $Str_{\mathcal{D}}(\mathbf{I})$; we define $M \models'_{\Delta}$ **derive** σ **from** ϕ if M has a σ-extension $N \models'_{\Sigma} \phi$,
- **free** ϕ **along** σ, with $\sigma : \Delta \to \Sigma$ and ϕ a Σ-sentence over $Str_{\mathcal{D}}(\mathbf{I})$; we define $M \models'_{\Delta}$ **free** ϕ **along** σ if $M \models'_{\Delta} \phi$ and for any model $N \models'_{\Delta} \phi$ and any homomorphism $h : M|_{\sigma} \to N|_{\sigma}$ there exists a unique homomorphism $g : M \to N$ such that $g|_{\sigma} = h$.

The translation of a Σ-sentence ϕ over $Str_{\mathcal{D}}(\mathbf{I})$ along a signature morphism $\sigma : \Sigma \to \Delta$ is defined to be **translate** ϕ **by** σ.

Now, the *full* $\textsc{Casl}$ *institution*, denoted $\textsc{Casl}_1$, is simply $Str_{\mathcal{I}}(\textsc{Casl}_{\leq})$, where $\mathcal{I}$ is the class of all signature morphisms σ, which are inclusions.

The following proposition allows us to lift representations of institutions to representations of structured specifications over them:

Proposition 3. *For any class $\mathcal{D}$ of signature morphisms in $\mathbf{I}$ and representation $R = (p, s, m) : \mathbf{I} \to \mathbf{I}'$ the following is a representation $R' = (p', s', m') : Str_{\mathcal{D}}(\mathbf{I}) \to Str_{\mathcal{D}'}(\mathbf{I}')$, where $\mathcal{D}' = \{p(\sigma) \mid \sigma \in \mathcal{D}\}$:*

- $p'(\Sigma) = (Sig[p(\Sigma)], \{p(\Sigma)\})$,
- $m' = m$,

– *s' is defined by structural induction:*
  - $s'_\Sigma((\Sigma, \Phi)) = (Sig[p(\Sigma)], \overline{s_\Sigma}(\Phi))$,
  - $s'_\Sigma(\textbf{translate } \phi \textbf{ by } \sigma) = \Big(\textbf{translate } s'_{\mathrm{dom}(\sigma)}(\phi) \textbf{ by } p(\sigma)\Big) \cup p(\mathrm{cod}(\Sigma))$,
  - $s'_\Sigma(\phi_1 \cup \phi_2) = s'_\Sigma(\phi_1) \cup s'_\Sigma(\phi_2)$*;*
  - $s'_\Sigma(\textbf{derive } \sigma \textbf{ from } \phi) = \textbf{derive } p(\sigma) \textbf{ from } s'_{\mathrm{cod}(\sigma)}(\phi)$,
  - $s'_\Sigma(\textbf{free } \phi \textbf{ along } \sigma) = \textbf{free } s'_{\mathrm{cod}(\sigma)}(\phi) \textbf{ along } p(\sigma)$.

□

Let $\text{StrCasl}_0 = Str_{\mathcal{I}}(\text{Casl}_0)$. The *standard representation* $R = (p, m, s) : \text{Casl}_1 \to \text{StrCasl}_0$ is the representation obtained via the above proposition from the representation $R_0 : \text{Casl}_\leq \to \text{Casl}_0$. Note that any Casl structured specification, as defined in the official semantics, may be easily represented as a sentence over Casl_1; the institution StrCasl_0, on the other hand, is equivalent to Casl structured specifications in which no subsorting is ever used.

6 Applying the Theorems to Casl

In this section we apply Th. 3 and 4 to Casl and the standard representation.

From [Hof01] (Prop. 1 and Cor. 1) we know that if the signature category is finitely cocomplete, the colimits are computable, and the model functor preserves finite limits, then the institution has logical amalgamation. The category Sig_0 is indeed finitely cocomplete [Mos98], the colimits being computable. The functor $\mathbf{Mod_0}$ preserves finite limits, too (cf. [GB84] for the first-order case). Thus:

Proposition 4. *The institution* StrCasl_0 *has logical amalgamation.* □

The same is of course true for Casl_0. However, neither $\text{Casl}_\leq$, nor Casl_1 have logical amalgamation (see [SMTKH01]), and this is the reason for introducing the representation R.

We now take a look at the assumptions of Th. 4. We will need the following auxiliary fact:

Proposition 5. *In the institution* StrCasl_0 *isomorphic models are elementarily equivalent, i.e., they satisfy precisely the same sentences.* □

Proposition 6. *In* StrCasl_0 *morphism equalities are expressible.*

Proof. Let $\alpha, \beta : \Sigma \to \Delta$ be arbitrary signature morphisms. If there exists a sort s in Σ with $\alpha^S(s) \neq \beta^S(s)$, then let ϕ be the sentence "*false*". Otherwise let ϕ be the conjunction of:

– "$\forall x^* : (\alpha^S)^*(w) \cdot \alpha^S_{w,s}(f)_{(\alpha^S)^*(w),\alpha^S(s)}(x^*) = \beta^S_{w,s}(f)_{(\alpha^S)^*(w),\alpha^S(s)}(x^*)$", for all $f_{w,s}$ in Σ,
– "$\forall x^* : (\alpha^S)^*(w) \cdot \alpha^P_w(b)_{(\alpha^S)^*(w)}(x^*) \iff \beta^P_w(b)_{(\alpha^S)^*(w)}(x^*)$", for b_w in Σ.

We claim that for any finite set Φ_0 of Δ-sentences, we have $\Phi_0 \models_\Delta \phi$ iff for any $M \models_\Delta \Phi_0$ we have $M|_\alpha = M|_\beta$.

The proof "$\Rightarrow$" is trivial. For "$\Leftarrow$", take Φ_0 satisfying the premise and let $M \models_\Delta \Phi_0$.

There exists a model N with all carriers distinct (here we use the non-emptyness of carriers), isomorphic to M. By Prop. 5, M and N satisfy precisely the same Δ-sentences. Thus $N \models_\Delta \Phi_0$ and therefore $N|_\alpha = N|_\beta$. This implies that for any sort s in Σ $N[cr, \alpha^S(s)] = N[cr, \beta^S(s)]$, and so $\alpha^S(s) = \beta^S(s)$.

Thus, ϕ is a conjunction of sentences, as defined above. We only need to prove that this conjunction holds in M. But this is obvious, since $M|_\alpha = M|_\beta$, so for any $f_{w,s}$ in Σ we must have $M[fn, \alpha^F_{w,s}(f)_{(\alpha^S)^*(w),\alpha^S(s)}] = M[fn, \beta^F_{w,s}(f)_{(\beta^S)^*(w),\beta^S(s)}]$, which implies that M must satisfy the appropriate element of the conjunction; a similar argument applies to predicates. $\square$

Let $\mathcal{E}$ be the class of morphisms $\eta : \Sigma \to \Sigma'$ in CASL_1 such that η^S, η^F and η^P are surjective functions and the subsort preorder in Σ' is the least preorder generated by the η^S-image of the subsort preorder of Σ. Note that $\eta \in \mathcal{E}$ does not imply that $p(\eta)$ is an epimorphism. We do have:

Proposition 7. *Any universal morphism $\sigma : \Sigma_1 \sqcup \Sigma_2 \to \Sigma_1 \cup \Sigma_2$ in CASL_1 is in $\mathcal{E}$.* $\square$

Proposition 8. *$\mathcal{E}$-extendability is expressible in R.*

Proof. Let $\eta : \Sigma \to \Sigma'$ be in $\mathcal{E}$; set $p(\Sigma) = (\Delta, \Phi)$ and $p(\Sigma') = (\Delta', \Phi')$. Let $\theta = p(\eta) : \Delta = (S, TF, PF, P) \to \Delta' = (S', TF', PF', P')$. By $C_\emptyset$ we denote a function constantly equal $\emptyset$ and with an appropriate domain.

To prove condition (1) of the definition we now define three families of pairs of morphisms:

- for any sorts s, s' in Δ such that $\theta^S(s) = \theta^S(s')$, let $\alpha, \beta : (\{s\}, C_\emptyset, C_\emptyset, C_\emptyset) \to \Delta$ be such that α takes s to s and β takes s to s',
- for any function symbols $f_{w,s}$ and $f'_{w',s'}$ in Δ, where $w = s_1 \ldots s_n$ and $w' = s'_1 \ldots s'_n$, such that $\theta^F_{w,s_0}(f) = \theta^F_{w',s'_0}(f')$, let $\alpha, \beta : (\{t_1, \ldots, t_n, t_0\}, C_\emptyset, C_\emptyset[\{f\}/(t_1 \ldots t_n, t_0)], C_\emptyset) \to \Delta$ be such that α takes f to f and t_i to s_i, and β takes f to f' and t_i to s'_i (for $0 \leq i \leq n$; the t_i are chosen distinct),
- for any predicates b_w and $b'_{w'}$ in Δ, where $w = s_1 \ldots s_n$ and $w' = s'_1 \ldots s'_n$, such that $\theta^P_w(b) = \theta^P_{w'}(b')$, let $\alpha, \beta : (\{t_1, \ldots, t_n\}, C_\emptyset, C_\emptyset, C_\emptyset[\{b\}/t_1 \ldots t_n]) \to \Delta$ be such that α takes b to b and t_i to s_i, and β takes b to b' and t_i to s'_i (for $1 \leq i \leq n$; the t_i again chosen distinct).

We claim that a Δ-model M is θ-extendible iff for all of the above-defined pairs (α, β) we have $M|'_\alpha = M|'_\beta$.

The "only if" part is fairly obvious: if we have a Δ'-model N with $N|'_\theta = M$ and (α, β) is one of the above pairs, then, since $\alpha; \theta = \beta; \theta$, we have $M|'_\alpha = (N|'_\theta)|'_\alpha = N|'_{\alpha;\theta} = N|'_{\beta;\theta} = (N|'_\theta)|'_\beta = M|'_\beta$.

For the "if" part, suppose that a Δ-model M indeed satisfies all of the required equations. We define a Δ'-model N:

- for any sort s' in Δ', $N[cr, s'] = M[cr, s]$, for any sort s in Δ with $s' = \theta^S(s)$ (since $\eta \in \mathcal{E}$, such an s must exist),

- for any function symbol $f'_{w',s'}$ in Δ', $N[fn, f'_{w',s'}]$ is:
  - $M[fn, f_{w,s}]$ if $w' = (\theta^S)^*(w)$, $s' = \theta^S(s)$ and $f' = \theta^F_{w,s}(f)$ for some function symbol $f_{w,s}$ in Δ,
  - any function of the appropriate type, otherwise,
- for any predicate $b'_{w'}$ in Δ', $N[pr, b'_{w'}]$ is:
  - $M[pr, b_w]$ if $w' = (\theta^S)^*(w)$ and $b' = \theta^P_w(b)$ for some predicate b_w in Δ,
  - the empty relation, otherwise.

This definition is correct precisely because for any of the above pairs of morphisms (α, β) we have $M|'_\alpha = M|'_\beta$. Directly from the definition it follows that $M = N|'_\theta$. This proves condition (1) of the definition. Thus, we move to condition (2).

For any sorts s, t in Δ', an *s, t-path* q is a sequence $s_1 \leq \cdots \leq s_n$ ($n > 1$) such that $s = \theta^S(s_1)$, $t = \theta^S(s_n)$ and that $s_i = s_j$, $i < j$ imply $i = 1$, $j = n$. For any Δ'-term T of sort s, by $q(T)$ we denote the "composition-of-embeddings" term $em_{\theta^S(s_{n-1}) \leq \theta^S(s_n)}(\dots em_{\theta^S(s_1) \leq \theta^S(s_2)}(T) \dots)$. These notions are further extended to tuples w of sorts and q^* of sequences (the sequence at each coordinate may be of different length). Observe that since $\eta \in \mathcal{E}$, for any sorts $s \leq' t$ in Δ' there exists an s, t-path. Also, the set of all paths is finite.

As the set Ψ' from condition (2) we take $\theta(\Phi)$ plus:

(A) for any s, t-paths q_1 and q_2, the sentence "$\forall x : s \cdot q_1(x) = q_2(x)$",
(B) for any $f_{w_1,s_1} \sim'_F f_{w_2,s_2}$ in Σ', any w, w_1-paths q_1^*, w, w_2-paths q_2^*, s_1, s-path q_1 and s_2, s-path q_2, the sentence "$\forall x^* : w \cdot q_1(f_{w_1,s_1}(q_1^*(x^*))) = q_2(f_{w_2,s_2}(q_2^*(x^*)))$",
(C) for any $b_{w_1} \sim'_P b_{w_2}$ in Σ', any w, w_1-paths q_1^* and w, w_2-paths q_2^*, the sentence "$\forall x^* : w \cdot b_{w_1}(q_1^*(x^*)) \iff b_{w_2}(q_2^*(x^*))$".

Let N be a Δ-model. We will prove that:

1. if N has a θ-extension satisfying Φ', then any of its θ-extensions satisfies Ψ',
2. if N has a θ-extension satisfying Ψ', then it has a unique θ-extension satisfying Φ'.

Observe that by definition of Ψ', for Δ'-models P, Q, if $P|'_\theta = Q|'_\theta$, then $P \models'_{\Delta'} \Psi'$ iff $Q \models'_{\Delta'} \Psi'$. Also, $\Phi' \models'_{\Delta'} \Psi'$. Now, if $N' \models'_{\Delta'} \Phi'$ and $N = N'|'_\theta = N''|'_\theta$, then $N' \models'_{\Delta'} \Psi'$ and $N'|'_\theta = N''|'_\theta$, hence $N'' \models'_{\Delta'} \Psi'$. This proves point (1).

As for point (2), take any model N and its θ-extension N' satisfying Ψ'. Define a Δ'-model M' as follows:

- carriers and function symbols (embeddings, etc., inclusive) and predicates in $\theta(\Delta)$ are interpreted in M' as in N',
- for any embedding $em_{s \leq t}$ in $\Delta' \setminus \theta(\Delta)$ we set $M'[fn, em_{s \leq t}] = N'[fn, em_{\theta^S(s_1) \leq \theta^S(s_2)}]; \dots ; N'[fn, em_{\theta^S(s_{n-1}) \leq \theta^S(s_n)}]$, where $q = s_1 \leq \cdots \leq s_n$ is an s, t-path,
- for any projection $prj_{t \geq s}$ in $\Delta' \setminus \theta(\Delta)$ we set $M'[fn, prj_{t \geq s}]$ to be the least right-sided inverse of $M'[fn, em_{s \leq t}]$,
- for any predicate $in_{s,t}$ in $\Delta' \setminus \theta(\Delta)$ we set $M'[pr, in_{s,t}] = \{x \in M'[cr, s] \mid M'[fn, prj_{s \geq t}](x) \text{ is defined } \}$.

The fact that N' satisfies the sentences from point (A) ensures that this definition is correct. It is also clear that M' is a θ-extension of N. We now check that $M' \models'_{\Delta'} \Phi'$.

Defining sentence no. 1 (defining sentences have been introduced in the previous section) holds, since all self-embeddings are in $\theta(\Delta)$. Defining sentences 3, 4 and 5 hold by definition of M'. Sentence 2 is a consequence of 3 and of the original embeddings being injective in N. Sentences 6 and 7 are consequences of the sentences from points (B) and (C).

Since it is obvious that any θ-extension of N satisfying defining sentences 1-7 is forced to be defined as above, this completes the proof. $\square$

Props. 4, 6, 7 and 8 allow us to state the following:

Corollary 1. *Theorems 3 and 4 apply to* CASL_1 *and the standard representation* $R : \mathrm{CASL}_1 \to \mathrm{StrCASL}_0$. $\square$

Actually, even more may be said. The cited propositions together with Theorems 3 and 4 define define an algorithm, which reduces the problem of checking whether a CASL_1 architectural specification has a denotation (i.e., is correct), to the problem of proving semantic consequence in the $\mathrm{StrCASL}_0$ institution, that is, in the institution of structured specifications over multi-sorted CASL. This statement is somewhat weakened by the fact that, for this reduction to work, the architectural specification at issue must contain no inconsistent parametric unit declarations and no parametric unit in it may be applied more than once.

This algorithm is of particular value, because proving semantic consequence between structured specifications has gained a lot of attention; recent papers to be mentioned include [BCH99,Borz98,MAH01].

7 Conclusion

In this paper a system — correct and complete w.r.t. a generative semantics — for proving properties of architectural specifications has been presented. Of course this completeness is relative to the calculus used for proving semantic consequence. This system is general enough to include the full CASL logic, even though it does not enjoy the amalgamation property.

This work builds on the work of [Hof01]. There, techniques for the relatively simple case of first-order logic were introduced. Here, we provide abstract, institution-independent formulations of needed properties and of theorems. Moreover, it is proven that the full CASL logic satisfies those properties, thus giving us a verification algorithm for CASL architectural specifications, and at the same time showing to what a wide scope of logics our techniques apply.

It should, however, be borne in mind, that the algorithm of [Hof01] was complete w.r.t. a non-generative semantics, while the algorithm presented here is complete w.r.t. a generative semantics (a generative semantics easily reduces to a non-generative one).

A clear aim is to get all the advantages of the current approach while retaining completeness w.r.t. a non-generative semantics, and thus fully solve the verification problem for CASL architectural specifications. This would also require taking a closer look at the remaining CASL architectural constructs (imports may cause additional trouble). What might also be interesting is using a representation into a different institution, namely that of *enriched* CASL [SMTKH01,SMT01,SMTKH]; this way, one would not remove any information from specifications, as is the case with the standard representation (in enriched CASL a subsort category replaces the subsort preorder).

Acknowlegdements This research has been partially supported by KBN grant no. 7 T11C 002 21 and by EU AGILE project IST-2001-32747.

References

[BCH99] M. Bidoit, M. V. Cengarle, R. Hennicker: Proof Systems for Structured Specifications and Their Refinements. In E. Astesiano, H.-J. Kreowski, B. Krieg-Brückner (eds.): Algebraic Foundations of Systems Specification; pp. 385–434. Springer, 1999.

[Borz98] T. Borzyszkowski: Completeness of a logical system for structured specifications. In F. Parisi Presicce (ed.): Proc. WADT'97, LNCS 1376, pp. 107–121 (1998).

[BST99] M. Bidoit, D. Sannella, A. Tarlecki. Architectural Specifications in CASL. In: Proc. AMAST'98, LNCS 1548, pp. 341–357 (1999); final ver. to appear in Formal Aspects of Computing.

[CASL] E. Astesiano, M. Bidoit, H. Kirchner, B. Krieg-Brückner, P. Mosses, D. Sannella, A. Tarlecki. CASL: The Common Algebraic Specification Language. TCS (to appear).

[CASL00] CASL — Summary and Semantics (note S-9), ver. 1.0. In [CoFI] (2000).

[CoFI] CoFI, The Common Framework Initiative for Algebraic Specification and Development. Documents accessible at `http://www.brics.dk/Projects/CoFI`.

[GB84] J. Goguen, R. Burstall: Some fundamental algebraic tools for the semantics of computation. Part 1: Comma categories, colimits, signatures, and theories. TCS 31(2), pp. 175–209 (1984)

[GB92] J. Goguen, R. Burstall: Institutions: abstract model theory for specification and programming. Journal of the ACM 39, pp. 95–146 (1992).

[Hof01] P. Hoffman: Verifying Architectural Specifications. In M. Cerioli, G. Reggio (eds.): Proc. WADT'01, LNCS 2267, pp. 152–175 (2001).

[MAH01] T. Mossakowski, S. Autexier, D. Hutter: Extending Development Graphs With Hiding. In H. Hussmann (ed.): Proc. FASE'01, LNCS 2029, pp. 269–283 (2001).

[Mos98] T. Mossakowski: Cocompleteness of the CASL signature category. In [CoFI], note S-7 (1998).

[Pau96] L. Paulson: ML for the Working Programmer. Cambridge Univ. Press, 1991.

[SMT01] L. Schröder, T. Mossakowski, A. Tarlecki: Amalgamation in CASL via Enriched Signatures. In F. Orejas, P. Spirakis, J. van Leeuwen (eds.): Proc. ICALP'01, LNCS 2076, pp. 993–1004 (2001).

[SMTKH01] L. Schröder, T. Mossakowski, A. Tarlecki, B. Klin, P. Hoffman: Semantics of Architectural Specifications in CASL. In H. Hussmann (ed.): Proc. FASE'01, LNCS 2029, pp. 253–268 (2001).

[SMTKH] L. Schröder, T. Mossakowski, A. Tarlecki, B. Klin, P. Hoffman: Amalgamation in the Semantics of CASL. TCS (to appear).

[ST88] D. Sannella, A. Tarlecki: Specifications in an Arbitrary Institution. Information and Computation vol. 76, 2/3, pp. 165–210 (1988).

[ST97] D. Sannella, A. Tarlecki: Essential Concepts of Algebraic Specification and Program Development. Formal Aspects of Computing 9, pp. 229–269 (1997).