Heterogeneous colimits

Mihai Codescu
DFKI Lab Bremen
D-28359 Bremen, Germany
Mihai.Codescu@dfki.de

Till Mossakowski
DFKI Lab Bremen and University of Bremen
D-28359 Bremen, Germany
Till.Mossakowski@dfki.de

Abstract

Colimits are a useful tool for the combination of specifications and logical theories. We generalize the notion of colimit to a heterogeneous multi-logic setting. For practically realistic cases, the notion has to be weakened. We describe an algorithm that approximates the weaker notion but obtains a colimit whenever possible. This algorithm is being implemented as part of the Heterogeneous Tool Set HETS.

1. Introduction

The notion of colimit has been used as a means for combining logical theories and software specifications, see e.g. [4] for the theory and [21] for a tool computing colimits of specifications that has been successfully used in industrial applications.

A major property of colimits of specifications is *amalgamation*. Roughly speaking, this property states that models of given specifications can be combined to yield a uniquely determined model of a colimit specification, provided that the original models coincide on common components. The amalgamation property (called 'exactness' in [6]) is a major technical assumption in the study of specification semantics [17] and is important in many respects. To give a few examples: it is a prerequisite for good behaviour w.r.t. parametrization and conservative extensions [6], and the proof system for development graphs with hiding [15], which allow a management of change for structured specifications, is sound only for logics with amalgamation.

One implicit assumption behind what has been said so far is that the theories or specifications all live within the one and the same formalism. However, viewpoint specifications [3] and heterogeneous ontologies [18], which both have received attention recently, involve different specification formalisms. In such heterogeneous settings (typically based on a graph of logics), colimits and amalgamation can be obtained under certain conditions [5], but often, these conditions are too strong to be met in practice. Hence, we start from weaker conditions, using both amalgamable colimits as well weakly amalgamable cocones.

We then describe an algorithmic method of obtaining weakly amalgamable cocones of heterogeneous diagrams, as an approximation of heterogeneous colimits, better suited for practical situations. We also present the way the graph of logics and logic translations, which is the base for heterogeneous specifications, gets a 2-categorical structure by adding the concept of modification, relating translations that are essentially the same. These are currently being integrated into the Heterogeneous Tool Set HETS [16].

2. Institutions and Comorphisms

Institutions [9] are the concept that is the basis of the theory of structured and heterogeneous specifications. They formalize in a model-oriented way the notion of logical system, abstracting away the details of signatures, sentences and models by not imposing other restriction on them than the satisfaction condition, which has the meaning that truth is invariant under change of notation and enlargement of context.

Definition 1 *An institution $I = (\mathbb{S}ign, \mathbb{S}en, \mathbb{M}od, \models)$ consists of:*

- *a category $\mathbb{S}ign$ of* signatures*,*
- *a functor $\mathbb{S}en\colon \mathbb{S}ign \longrightarrow \mathbb{S}et$, giving for each signature Σ the set of* sentences $\mathbb{S}en(\Sigma)$ *and for each signature morphism $\varphi\colon \Sigma \longrightarrow \Sigma'$ a* sentence translation map $Sen(\varphi)\colon \mathbb{S}en(\Sigma) \longrightarrow \mathbb{S}en(\Sigma')$*, where we may write $Sen(\varphi)(e)$ as $\varphi(e)$,*
- *a functor $\mathbb{M}od\colon \mathbb{S}ign^{op} \longrightarrow \mathbb{C}at$ giving for each signature Σ the category of* models $\mathbb{M}od(\Sigma)$ *and for each signature morphism $\varphi\colon \Sigma \longrightarrow \Sigma'$, the* reduct functor $\mathbb{M}od(\varphi)\colon \mathbb{M}od(\Sigma') \longrightarrow \mathbb{M}od(\Sigma)$*, where we may write $\mathbb{M}od(\varphi)(M')$ as $M' \restriction_\varphi$;*

- *a binary relation* $\models_\Sigma \subseteq |\mathbb{M}od(\Sigma)| \times \mathbb{S}en(\Sigma)$, *for each signature* Σ, *called the* satisfaction relation

such that the following satisfaction condition *holds:*

$$M' \restriction_\varphi \models_\Sigma e \iff M' \models_{\Sigma'} \varphi(e)$$

for each signature morphism $\varphi\colon \Sigma \longrightarrow \Sigma'$, *each* Σ*-sentence* e *and each* Σ'*-model* M'.

Institutions have also been characterized [9] as functors into a certain category of 'twisted relations' or rooms, denoted $\mathbf{Room}$, whose objects $(S, \mathcal{M}, \models)$ consist of a set of S of *sentences*, a category $\mathcal{M}$ of *models* and a satisfaction relation $\models \subseteq |\mathcal{M}| \times S$ and whose arrows $(\alpha, \beta)\colon (S_1, \mathcal{M}_1, \models_1) \longrightarrow (S_2, \mathcal{M}_2, \models_2)$ consist of a sentence translation function $\alpha\colon S_1 \longrightarrow S_2$ and a model reduction functor $\beta\colon \mathcal{M}_2 \longrightarrow \mathcal{M}_1$, such that

$$M_2 \models_2 \alpha(\varphi_1) \Leftrightarrow \beta(M_2) \models_1 \varphi_1$$

holds for each $M_2 \in \mathcal{M}_2$ and each $\varphi_1 \in S_1$ (*satisfaction condition*).

Then, an institution can be defined as a functor $\mathcal{I}\colon \mathbb{S}ign \longrightarrow \mathbf{Room}$. This definition allows us to express results in a more concise manner.

Example 2 *[18]* Relational Schemes *(for relational databases). A signature of* $\mathbf{Rel}$ *consists of a set of relation symbols, where each relation symbol is indexed with a string of field names. Signature morphisms map relation symbols and field names. A model consists of a domain (set), and an* n*-ary relation for each relation symbol with* n *fields. A model reduction just forgets the parts of a model that are not needed. A sentence is a relationship between two field names of two relation symbols. Sentence translation is just renaming. A relationship is satisfied in a model if for each element occurring in the source field component of a tuple in the source relation, the same element also occurs in the target field component of a tuple in the target relation.* □

Example 3 *[9]* First-order Logic. *In the institution* $\mathbf{FOL}^=$ *of many-sorted first-order logic with equality, signatures are many-sorted first-order signatures, consisting of sorts and typed function and predicate symbols. Signature morphisms map symbols such that typing is preserved. Models are many-sorted first-order structures. Sentences are first-order formulas. Sentence translation means replacement of the translated symbols. Model reduct means reassembling the model's components according to the signature morphism. Satisfaction is the usual satisfaction of a first-order sentence in a first-order structure.* □

Example 4 *[13] The institution* $\mathbf{PFOL}^=$ *of partial first-order logic with equality. Signatures are many-sorted first-order signatures enriched by partial function symbols. Models are many-sorted partial first-order structures. Sentences are first-order formulas involving existential equations (both sides are defined and equal) and strong equations (both sides are equally defined, and equal in case of definedness).* □

Example 5 *[2]* Higher-order Logic. *The institution* $\mathbf{HOL}^=$ *of many-sorted higher-order logic with equality extends* $\mathbf{FOL}^=$ *with higher-order types, which are interpreted as appropriate subsets of the function types, where appropriate means that all* λ*-terms can be interpreted (Henkin semantics). Sentences extend first-order sentences by* λ*-abstraction and arbitrary application.* □

Example 6 *[1]* Description Logics. *Signatures of the description logic* $\mathcal{ALC}$ *consist of a set of* B *of atomic concepts and a set* R *of roles, while signature morphisms provide respective mappings. Models are single-sorted first-order structures that interpret concepts as unary and roles as binary predicates. Sentences are subsumption relations* $C_1 \sqsubseteq C_2$ *between concepts, where concepts follow the grammar*

$$C ::= B \mid \top \mid \bot \mid C_1 \sqcup C_2 \mid C_1 \sqcap C_2 \mid \neg C \mid \forall R.C \mid \exists R.C$$

Sentence translation and reduct is defined similarly as in $\mathbf{FOL}^=$. *Satisfaction is the standard satisfaction of description logics.* □

Example 7 *[20] The institution* $\mathbf{PLNG}$ *of a programming language. It is built over an algebra of built-in data types and operations of a programming language. Signatures are given as function (functional procedure) headings; sentences are function bodies; and models are maps that for each function symbol, assign a computation (either diverging, or yielding a result) to any sequence of actual parameters. A model satisfies a sentence iff it assigns to each sequence of parameters the computation of the function body as given by the sentence. Hence, sentences determine particular functions in the model uniquely. Finally, signature morphisms, model reductions and sentence translations are defined similarly to those in* $FOL^=$. □

Within an arbitrary but fixed institution, we can easily define the usual notion of *logical consequence* or *semantical entailment*. Given a set of Σ-sentences Γ and a Σ-sentence φ, we say that φ *follows from* Γ, written $\Gamma \models_\Sigma \varphi$, iff for all Σ-models M, we have $M \models_\Sigma \Gamma$ implies $M \models_\Sigma \varphi$. (Here, $M \models_\Sigma \Gamma$ means that $M \models_\Sigma \psi$ for each $\psi \in \Gamma$.)

A *theory* is a pair (Σ, Γ) where Γ is a set of Σ-sentences. A theory morphism $(\Sigma, \Gamma) \longrightarrow (\Sigma', \Gamma')$ is a signature morphism $\sigma : \Sigma \longrightarrow \Sigma$ such that $\Gamma' \models_{\Sigma'} \sigma(\Gamma)$. Given an institution I, the *institution of theories* I^{th} has as signature category the category of theories of I. The remaining components are inherited from I, but with models of a theory restricted to those actually satisfying its axioms.

Institution comorphisms [8] typically express that an institution is included or encoded into another one. Other kind of mappings between institutions have also been introduced, but we restrict ourselves here only to comorphisms.

Definition 8 *Given institutions* $I_1\colon \mathbb{S}ign_1 \longrightarrow \mathbf{Room}$ *and* $I_2\colon \mathbb{S}ign_2 \longrightarrow \mathbf{Room}$*, an* institution comorphism $(\Phi,\rho)\colon I_1 \longrightarrow I_2$ *consists of a functor* $\Phi\colon \mathbb{S}ign_1 \longrightarrow \mathbb{S}ign_2$ *and a natural transformation* $\rho\colon I_1 \longrightarrow I_2 \circ \Phi$*.*

The natural transformation ρ may be thought of as a pair of natural transformations $(\alpha\colon \mathbb{S}en_1 \longrightarrow \mathbb{S}en_2 \circ \Phi, \beta\colon \mathbb{M}od_2 \circ \Phi^{op} \longrightarrow \mathbb{M}od_1)$. Then, for any signature $\Sigma \in |Sign_1|$, the satisfaction condition for ρ_Σ becomes

$$M \models_2 \alpha_\Sigma(e) \iff \beta_\Sigma(M) \models_1 e$$

for any Σ-sentence e and any $\Phi(\Sigma)$-model M, with the meaning that truth is invariant under translation along comorphisms.

Together with obvious compositions and identities, this gives us the category $\mathbf{CoIns}$ of institutions and institution comorphisms.

Example 9 *The obvious inclusions from* $\mathbf{FOL}$ *to* $\mathbf{PFOL}^=$ *and from* $\mathbf{FOL}$ *to* $\mathbf{HOL}^=$ *are an institution comorphisms.*

Example 10 *The comorphism from the institution of relational schemes to* FOL *maps signatures and models in the straightforward way, while each sentence* $R(f_1,..,f_n) \rightarrow R'(f'_1,...,f'_m)$ *linking fields* f_i *and* f'_j *is translated to the first-order formula* $R(x_1,...,x_n) \implies \exists y_1...y_{j-1}y_{j+1}...y_m\ R'(y_1,...,y_{j-1},x_i,y_{j+1},...,y_m)$*.* □

Example 11 *The translation of* $\mathcal{ALC}$ *to* $\mathbf{FOL}$ *is straightforward; see [1].* □

Example 12 *There is an institution comorphism from* $\mathbf{PFOL}^=$ *to* $\mathbf{FOL}^{th}$ *that codes out partiality via error elements. Details can be found in [13]. By composing with the inclusion from* $\mathbf{FOL}^{th}$ *to* $(\mathbf{HOL}^=)^{th}$*, we get a comorphism from* $\mathbf{PFOL}^=$ *to* $(\mathbf{HOL}^=)^{th}$*.* □

Example 13 *There is a so-called institution semi-morphism toPFOL from* $\mathbf{PLNG}$ *to* $\mathbf{PFOL}^=$ *[20]. It extracts an algebraic signature* $\Phi(\Sigma)$ *with partial operations out of a* $\mathbf{PLNG}$*-signature* Σ *by adding the signature of built-in data types and operations of the programming language. For any function declared, any* $\mathbf{PLNG}$*-model* M *determines its computations on given arguments, from which we can extract a partial function that maps any sequence of arguments to the result of the computation (if any). These are used to expand the built-in algebra of data types and operations of the programming language with an interpretation for the extra function names in the signature obtained, thus obtaining a* $\mathbf{PFOL}^=$*-model* $\beta(M)$*.*

In our setting, this can be modelled as a span of comorphisms

$$\begin{tikzcd}\mathbf{PLNG} & \mathbf{PFOL}^= \circ \Phi \arrow[l, "toPFOL^-"'] \arrow[r, "toPFOL^+"] & \mathbf{PFOL}^=\end{tikzcd}$$

$$\begin{array}{lclcl}
\mathbb{S}ign^{\mathbf{PLNG}} & \overset{id}{\leftarrow} & \mathbb{S}ign^{\mathbf{PLNG}} & \overset{\Phi}{\rightarrow} & \mathbb{S}ign^{PFOL} \\
\mathbb{S}en^{\mathbf{PLNG}} & \overset{incl}{\leftarrow} & \emptyset & \overset{incl}{\rightarrow} & \mathbb{S}en^{PFOL} \circ \Phi \\
\mathbb{M}od^{\mathbf{PLNG}} & \overset{\beta}{\rightarrow} & \mathbb{M}od^{\mathbf{PFOL}^=} \circ \Phi^{op} & \overset{id}{\leftarrow} & \mathbb{M}od^{PFOL} \circ \Phi^{op}
\end{array}$$

Here, the "middle" institution $\mathbf{PFOL}^= \circ \Phi$ *is the institution with signature category inherited from* $\mathbf{PLNG}$*, no sentences, and models inherited from* $\mathbf{PFOL}^=$ *via* Φ*.* □

Example 14 *There is an institution comorphism from* $\mathbf{PLNG}$ *to* $(\mathbf{HOL}^=)^{th}$ *that codes the semantics of* $\mathbf{PLNG}$ *within higher-order logic.* □

Definition 15 *An institution comorphism is* model-expansive*, if each model translation* β_Σ *is surjective on objects.*

All of the above comorphisms, except the second one from Example 13, are model expansive. Model-expansive comorphisms allow for transportation of logical entailment questions and hence re-use hence proof systems:

Proposition 16 *Given a model-expansive comorphism* $(\Phi,\alpha,\beta)\colon I \longrightarrow J$*,*

$$\Gamma \models^I \varphi \text{ iff } \alpha_\Sigma(\Gamma) \models^J \alpha(\varphi)$$

3. Colimits and Amalgamation

We briefly recall the categorical notion of colimit. Colimits are a mean of combining interconnected objects consistently to this interconnection (see motivations in [9]). A *diagram* in a category C is a functor $D\colon G \longrightarrow C$, where G can be thought of as the graph of interconnections between the objects the functor D selects. A *cocone* of a diagram $D\colon G \longrightarrow C$ consists of an object c of C and a family of morphisms $\alpha_i\colon D(i) \longrightarrow c$, for each object i of G, such that for each edge of the diagram, $e\colon i \longrightarrow i'$ we have that $D(e);\alpha_{i'} = \alpha_i$. A *colimiting cocone* (or colimit) $(c, \{\alpha_i\}_{i\in|G|})$ can be intuitively understood as a minimal cocone, i.e. has the property that for any other cocone $(d, \{\beta_i\}_{i\in|G|})$ there exists a unique morphism $\gamma\colon c \longrightarrow d$

such that $\alpha_i ; \gamma = \beta_i$. By dropping the uniqueness condition and requiring only that a morphism γ should exist, we obtain a *weak* colimit.

When G is the category $\bullet \longleftarrow \bullet \longrightarrow \bullet$ with 3 objects and 2 non-identity arrows, the G-colimits are called *pushouts*.

Since specifications are actually *theories* over some institution (i.e. pairs (Σ, E) with Σ a signature and E a set of Σ-sentences) we are actually interested in computing colimits of theories rather than just signatures. To obtain a colimit of theories, it suffices to compute the colimit of signatures and then the set of sentences of the colimit theory is defined as the union of all component theories in the diagram, translated along the signature morphisms of the colimiting cocone.

In the sequel, fix an arbitrary institution $I = (\mathbb{S}ign, \mathbb{S}en, \mathbb{M}od, \models)$.

Definition 17 *Given a diagram $D \colon J \longrightarrow \mathbb{S}ign^I$, a family of models $(M_j)_{j \in |J|}$ is called D-*compatible *if $M_k \restriction_{D(\delta)} = M_j$ for each $\delta \colon j \longrightarrow k \in J$. A cocone $(\Sigma, (\mu_j)_{j \in |J|})$ over the diagram in $D \colon J \longrightarrow \mathbb{S}ign^I$ is called* weakly amalgamable *if for each D-compatible family of models $(M_j)_{j \in |J|}$, there is a Σ-model M with $M \restriction_{\mu_j} = M_j$ $(j \in |J|)$* [1]*. If this model is unique, the cocone is called* amalgamable*. I (or $\mathbb{M}od$) admits* (finite) (weak) amalgamation *if (finite) colimit cocones are (weakly) amalgamable. An important special case is the one of pushouts: I is called* (weakly) semi-exact*, if it has pushouts and admits (weak) amalgamation for these.* □

Amalgamation resp. exactness can be lifted to comorphisms as follows:

Definition 18 *Let $\rho = (\Phi, \alpha, \beta) \colon I \longrightarrow J$ be an institution comorphism and let $\mathcal{D}$ be a class of signature morphisms in I. Then ρ is said to have the* (weak) $\mathcal{D}$-amalgamation property*, if for each signature morphism $\sigma \colon \Sigma_1 \longrightarrow \Sigma_2 \in \mathcal{D}$, the diagram*

$$\begin{tikzcd}
\mathbb{M}od^I(\Sigma_2) \arrow[d, "\mathbb{M}od^I(\sigma)"'] & \mathbb{M}od^J(\Phi(\Sigma_2)) \arrow[l, "\beta_{\Sigma_2}"'] \arrow[d, "\mathbb{M}od^J(\Phi(\sigma))"] \\
\mathbb{M}od^I(\Sigma_1) & \mathbb{M}od^J(\Phi(\Sigma_1)) \arrow[l, "\beta_{\Sigma_1}"']
\end{tikzcd}$$

admits (weak) amalgamation, i.e. any for any two models $M_2 \in \mathbb{M}od^I(\Sigma_2)$ and $M_1' \in \mathbb{M}od^J(\Phi(\Sigma_1))$ with $M_2 \restriction \sigma = \beta_{\Sigma_1}(M_1')$, there is a unique (not necessarily unique) $M_2' \in \mathbb{M}od^J(\Phi(\Sigma_2))$ with $\beta_{\Sigma_2}(M_2') = M_2$ and $M_2' \restriction \Phi(\sigma) = M_1'$. In case that $\mathcal{D}$ consists of all signature morphisms, the (weak) $\mathcal{D}$-amalgamation property is also called (weak) exactness. □

[1]Recall that we use $M \restriction_\mu$ for $Mod(\mu)(M)$ (see Definition 1).

4. Grothendieck Institutions

The Grothendieck construction for indexed institutions (based on institution morphisms) has been defined in [5]; we here describe the dual, comorphism-based variant [12]. The idea is to begin with a graph of logics and logics translations and then to flatten this graph, using a so-called Grothendieck construction.

Definition 19 *Given an index category Ind, an* indexed coinstitution *is a functor $\mathcal{I} \colon Ind^{op} \longrightarrow \mathbf{CoIns}$ into the category of institutions and institution comorphisms.*

In an indexed coinstitution $\mathcal{I}$, we use the notation $\mathcal{I}^i = (\mathbb{S}ign^i, \mathbb{S}en^i, \mathbb{M}od^i, \models^i)$ for $\mathcal{I}(i)$, (Φ^d, ρ^d) for the comorphism $\mathcal{I}(d)$.

Definition 20 *Given an indexed coinstitution $\mathcal{I} \colon Ind^{op} \longrightarrow \mathbf{CoIns}$, define the* Grothendieck institution *$\mathcal{I}^\#$ as follows:*

- *signatures in $\mathcal{I}^\#$ are pairs (i, Σ), where $i \in |Ind|$ and Σ a signature in $\mathcal{I}^i$,*
- *signature morphisms $(d, \sigma) \colon (i, \Sigma_1) \longrightarrow (j, \Sigma_2)$ consist of a morphism $d \colon j \longrightarrow i \in Ind$ and a signature morphism $\sigma \colon \Phi^d(\Sigma_1) \longrightarrow \Sigma_2$ in $\mathcal{I}^j$,*
- *composition is given by $(d_2, \sigma_2) \circ (d_1, \sigma_1) = (d_1 \circ d_2, \sigma_2 \circ \Phi^{d_2}(\sigma_1))$,*
- *$\mathcal{I}^\#(i, \Sigma) = \mathcal{I}^i(\Sigma)$, and $\mathcal{I}^\#(d, \sigma) =$*
$$\mathcal{I}^i(\Sigma_1) \xrightarrow{\rho^d} \mathcal{I}^j(\Phi^d(\Sigma_1)) \xrightarrow{\mathcal{I}^j(\sigma)} \mathcal{I}^j(\Sigma_2)\,.$$

That is, sentences, models and satisfaction for a Grothendieck signature (i, Σ) are defined component wise, while the sentence and model translations for a Grothendieck signature morphism are obtained by composing the translation given by the inter-institution comorphism with that given by the intra-institution signature morphism. We also denote the Grothendieck institution by $(\mathbb{S}ign^\#, \mathbb{S}en^\#, \mathbb{M}od^\#, \models^\#)$.

The following results regarding cocompleteness and exactness of Grothendieck institutions have been proved in [12].

Theorem 21 *Let $\mathcal{I} \colon Ind^{op} \longrightarrow \mathbf{CoIns}$ be an indexed coinstitution and K be some small category such that*

1. *Ind is K-complete (that is, has limits of all diagrams over K),*
2. *Φ^d is K-cocontinuous for each $d \colon i \longrightarrow j \in Ind$ (meaning that it preserves colimits), and*

3. *the indexed category of signatures of $\mathcal{I}$ is locally K-cocomplete (the latter meaning that $\mathbb{S}ign^i$ is K-cocomplete for each $i \in |Ind|$).*

Then the signature category $\mathbb{S}ign^\#$ of the Grothendieck institution has K-colimits. □

An indexed coinstitution $\mathcal{I}\colon Ind^{op} \longrightarrow$ **CoIns** is called *(weakly) locally semi-exact*, if each institution I^i is (weakly) semi-exact ($i \in |Ind|$).

$\mathcal{I}$ is called *(weakly) semi-exact* if for each pullback in Ind

$$\begin{tikzcd}
i & j1 \arrow[l, "d_1"'] \\
j2 \arrow[u, "d_2"] & k \arrow[l, "e_2"] \arrow[u, "e_1"']
\end{tikzcd}$$

the square

$$\begin{tikzcd}
\mathbb{M}od^i(\Sigma) & \mathbb{M}od^{j1}(\Phi^{d_1}(\Sigma)) \arrow[l, "\beta^{d_1}_\Sigma"'] \\
\mathbb{M}od^{j2}(\Phi^{d_2}(\Sigma)) \arrow[u, "\beta^{d_2}_\Sigma"] & \mathbb{M}od^k(\Phi^{e_1}(\Phi^{d_1}(\Sigma))) \arrow[l, "\beta^{e_2}_\Sigma"'] \arrow[u, "\beta^{e_1}_\Sigma"']
\end{tikzcd}$$

is a (weak) pullback for each signature Σ in $\mathbb{S}ign^i$.

Theorem 22 *Assume that the indexed coinstitution $\mathcal{I}\colon Ind^{op} \longrightarrow$ **CoIns** fulfills the assumptions of Theorem 21. Then the Grothendieck institution $\mathcal{I}^\#$ is (weakly) semi-exact if and only if*

1. *$\mathcal{I}$ is (weakly) locally semi-exact,*
2. *$\mathcal{I}$ is (weakly) semi-exact, and*
3. *for all $d\colon i \longrightarrow j \in Ind$, $\mathcal{I}^d$ is (weakly) exact.*

The importance of this theorem show up in connection with Prop. 16:

Corollary 23 *The proof system for development graphs with hiding [15] can be re-used for Grothendieck institutions satisfying the assumptions of Theorem 22, provided that each of the involved institutions can be mapped (via a model-expansive comorphism) into a proof-supported institution.*

5. The Heterogeneous Tool Set (HETS)

The heterogeneous tool set (Hets) is a parsing, static analysis and proof management tool combining various such tools for individual specification languages, thus providing a tool for heterogeneous multi-logic specification. The heterogeneous tool set is a both flexible, multi-lateral *and* formal (i.e. based on a mathematical semantics) integration tool. Unlike other tools, it treats logic translations (formalised as institution comorphisms) as first-class citizens. The architecture of the heterogeneous tool set is shown in Fig. 1 (see [16] for detailed discussion).

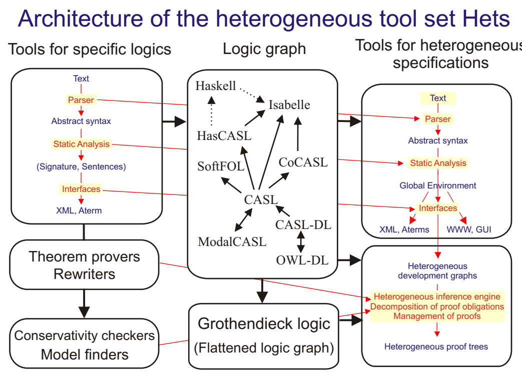

Figure 1. Architecture of the heterogeneous tool set

Heterogeneous colimits are needed for several purposes in Hets:

- The proof calculus for development graphs [15] has been generalised to Grothendieck institutions [12]. In order to handle hiding (of parts of specifications) correctly, during a proof, the hidden parts have to be revealed. This is done using colimits, and in the case of the Grothendieck institutions, these colimits are of course heterogeneous.
- In order to prove correctness of a heterogeneous theory morphism (refinement) $\sigma\colon N_1 \longrightarrow N_2$, we have to translate N_1 and N_2 into the same logic so we can make the proof. The logical framework approach assumes that the theories of N_1 and N_2 are encoded into some logic that is fixed once and forall. By contrast, in HETS we can rather flexibly find a logic that is a "common upper bound" (=some weak form of colimit) of the logics of both N_1 and N_2 and that moreover has best possible tool support.
- Colimits also appear in the semantics of instantiation of parameterised specifications.
- Colimits play a role for alignments of ontologies [22], and recently, also heterogeneous ontologies have been studied [18]. Therefore, we have added a menu for directly computing heterogeneous colimits with Hets.

6. Example: Heterogeneous Ontologies

Example 24 *Let us consider the following formalisation of bibliographical data from [18]: we have a description logic T-Box formalized as an $\mathcal{ALC}$ signature Σ_1 with atomic concepts $B = \{Researcher, Article, Journal\}$ and roles $R = \{name, author, title, hasArticle, impactFactor\}$. The axioms Ax_{DLbib} are*

$Researcher \sqsubseteq \exists name.\top$
$Article \sqsubseteq \exists author.\top \sqcap \exists title.\top$
$Journal \sqsubseteq \exists name.\top$
$\sqcap \exists hasArticle.\top \sqcap \exists impactFactor.\top$

This is assumed to be a fragment of a larger $\mathcal{ALC}$ ontology with signature Σ_2.

On the other hand, there is a similar formalisation using the relational schema with signature Σ_3 and axioms Ax_{RelBib} presented in Figure 2 as a fragment of a larger relational schema Σ_4 of some relational database.

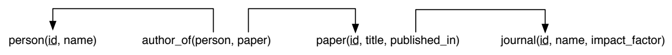
person(id, name)
author_of(person, paper)
paper(id, title, published_in)
journal(id, name, impact_factor)

Figure 2. Relational schema of an information system

[18] link these two ontologies by mapping both into a given reference ontology T living in first-order logic. However, with this approach, the question whether the axiomatizations Ax_{DLbib} and Ax_{RelBib} have comparable strength cannot be studied at all.

Hence, we here follow a different approach. Instead of using a common reference theory, we specify an interface theory $Interface$ in **FOL** *that relates the two ontologies as follows here:*

$\forall\, p, j, n, f, a, t : s$
$.\, journal(j, n, f) \Leftrightarrow$
$Journal(j) \wedge name(j, n) \wedge impactFactor(j, f)$
$.\, paper(a, t, j) \Leftrightarrow$
$Article(a) \wedge Journal(j) \wedge hasArticle(j, a) \wedge title(a, t)$
$.\, author\_of(p, a) \Leftrightarrow$
$Researcher(p) \wedge Article(a) \wedge author(p, a)$
$.\, person(p, n) \Leftrightarrow Researcher(p) \wedge name(p, n)$

The signature Σ of this interface theory is the union of the translations (as given by the comorphisms from Examples 10 and 11) of Σ_1 and Σ_3 to first-order logic.

Assume we want to check whether all models of the theory Ax_{DLbib} in Σ_2 are models of a theory Ax_{RelBib} in Σ_4. Both theories would have to be translated into a common language. We construct the diagram in figure 3 (notice that we marked distinctively the inclusions) and we compute its colimit, then we try to prove that $\theta_1(Ax_{DLbib}) \models \theta_2(Ax_{RelBib})$, where we denote $\theta_1 = \gamma_1; \delta_1$. □

$$\begin{tikzcd}
 & & (\mathbf{FOL}, \Sigma'') & & \\
 & (\mathbf{FOL}, \Sigma') \arrow[ur, dotted, "{(id_{\mathbf{FOL}}, \delta_1)}"] & & & \\
(\mathcal{ALC}, \Sigma_2) \arrow[ur, dotted, "{(ALC2FOL, \gamma_1)}"] & & (\mathbf{FOL}, \Sigma) \arrow[ul, dotted, "{(id_{\mathbf{FOL}}, \gamma_2)}"'] & & (\mathbf{Rel}, \Sigma_4) \arrow[uull, dotted, "{(Rel2FOL, \theta_2)}"'] \\
 & (\mathcal{ALC}, \Sigma_1) \arrow[ul, hook, "{(id_{ALC}, \phi_1)}"'] \arrow[ur, "{(ALC2FOL, \sigma_1)}"'] & & (\mathbf{Rel}, \Sigma_3) \arrow[ul, "{(Rel2FOL, \sigma_2)}"] \arrow[ur, hook, "{(id_{Rel}, \phi_2)}"'] &
\end{tikzcd}$$

Figure 3.

Using the Heterogeneous Tool Set HETS, we found out that only two of the axioms in Ax_{RelBib} are provable in this way; the second relationship pointing from the $paper$ field of the $author\_of$ relation to the id field of the $paper$ relation cannot be deduced from the description logic axiomatization Ax_{DLbib}. The reason for this is that Ax_{DLbib} does not state that an $Article$ must appear in $Journal$. In order to extend Ax_{DLbib} accordingly, the inverse of the role $hasArticle$ would be needed.

7. Relaxing Colimits and Amalgamation

The example of the last section shows that theorems 21 and 22 have too strong premises to be applied in all practical situations. Given a diagram $J \rightarrow Ind$, its limit must be the index of some institution that can serve to encode, via comorphisms, all the institutions indexed by the diagram. The existence of such an institution may not be a problem, but the uniqueness condition imposed by the limit property is more problematic. This means that any two such "universal" institutions must have isomorphic indices and hence be isomorphic themselves. This might work well in some circumstances, but may not desirable in others: after all, a number of non-isomorphic logics, such as classical higher-order logic, the calculus of constructions and rewriting logic have been proposed as such a "universal" logic. Also, the assumptions of Theorem 22 may not hold in all the cases - e.g. institutions with subsorts [19] are not weakly semi-exact.

Therefore, we drop the uniqueness restriction by replacing weak exactness with quasi-exactness, i.e. amalgamable colimits with weakly amalgamable cocones. Also, the new framework will allow non-exact institutions and comorphisms to be included in the indexed coinstitution serving as basis of the Grothendieck construction.

A problem occurs when using this approach, namely a great number of comorphisms with the same behaviour are introduced via compositions. Therefore, we use the institution comorphism modifications to identify comorphisms with the same sentence and model translation maps.

Hence, we strengthen the original notion from [5] to *discrete* modifications:

Definition 25 *Given institution comorphisms* $(\Phi, \rho)\colon I_1 \longrightarrow I_2$ *and* $(\Phi', \rho')\colon I_1 \longrightarrow I_2$*, a* discrete institution comorphism modification $\theta\colon (\Phi, \rho) \longrightarrow (\Phi', \rho')$ *is a natural transformation* $\theta\colon \Phi \longrightarrow \Phi'$ *such that* $(I_2 \cdot \theta) \circ \rho = \rho'$.

Together with obvious identities and compositions, discrete modifications can serve as 2-cells, and thus $\mathbf{CoIns}$ is turned into a 2-category.

We obtain a congruence on Grothendieck signature morphisms: the congruence is generated by

$$(d', \mathcal{I}^u_\Sigma\colon \Phi^{d'}(\Sigma) \longrightarrow \Phi^d(\Sigma)) \equiv (d, id\colon \Phi^d(\Sigma) \longrightarrow \Phi^d(\Sigma))$$

for $\Sigma \in \mathbb{S}ign^i$, $d, d' : j \longrightarrow i \in Ind$, and $u : d \Rightarrow d' \in Ind$. This congruence has the following crucial property:

Proposition 26 $\equiv$ *is contained in the kernel of* $\mathcal{I}^\#$ *(considered as a functor).*

Let $q^{\mathcal{I}} : \mathbb{S}ign^\# \longrightarrow \mathbb{S}ign^\# / \equiv$ be the quotient functor induced by $\equiv$definition of quotient category). Note that it is the identity on objects. We easily obtain that the functor $\mathcal{I}^\#$ factors through the quotient category $\mathbb{S}ign^\#/\equiv$.

Corollary 27 $\mathcal{I}^\# : \mathbb{S}ign^\# \longrightarrow \mathbf{Room}$ *leads to a quotient Grothendieck institution* $\mathcal{I}^\#/\equiv : \mathbb{S}ign^\#/\equiv \longrightarrow \mathbf{Room}$.

By abuse of notation, we denote $\mathcal{I}^\#/\equiv$ by $(\mathbb{S}ign^\#/\equiv, \mathbb{S}en^\#, \mathbb{M}od^\#, \models^\#)$.

Consider the span of comorphisms $\mathbf{PLNG} \xleftarrow{toPFOL^-} \mathbf{PFOL}^= \circ \Phi \xrightarrow{toPFOL^+} \mathbf{PFOL}^=$ for which we want to obtain a weakly amalgamable cocone. But this can e.g. be given by coding of both $\mathbf{PFOL}^=$ and $\mathbf{PLNG}$ into a common logic such as higher order logic (see Examples 12 and 14) . However, the resulting square does not commute, since on the way from $\mathbf{PFOL}^= \circ \Phi$ to $\mathbf{HOL}^=$ via $\mathbf{PFOL}^=$, the operational semantics of the programming language is expressed in $\mathbf{HOL}^=$. But there is a diagram of two-cells:

$$\begin{tikzcd}
 & \mathbf{HOL}^= & \\
\mathbf{PLNG} \arrow[ur, "PLNG2HOL"] \arrow[r, phantom, "\overset{id}{\Longrightarrow}"] & & \mathbf{PFOL}^= \arrow[ul, "PFOL2HOL"'] \arrow[l, phantom, "\overset{\theta}{\Longleftarrow}"] \\
 & \mathbf{PFOL}^= \circ \Phi \arrow[ul, "toPFOL^-"] \arrow[ur, "toPFOL^+"'] \arrow[uu] &
\end{tikzcd}$$

which is weakly amalgamable in the following sense:

Definition 28 *Given a 2-indexed coinstitution* $\mathcal{I}\colon Ind^{op} \longrightarrow \mathbf{CoIns}$*, a square consisting of two lax triangles of index morphisms*

$$\begin{tikzcd}
 & i & \\
j2 \arrow[ur, "d2"] \arrow[r, phantom, "\overset{u2}{\Longrightarrow}"] & & j1 \arrow[ul, "d1"'] \arrow[l, phantom, "\overset{u1}{\Longleftarrow}"] \\
 & k \arrow[ul, "e2"] \arrow[ur, "e1"'] \arrow[uu, "d"] &
\end{tikzcd}$$

is called (weakly) amalgamable, if each pair consisting of a $\Phi^{d2}(\Sigma)$*- and a* $\Phi^{d1}(\Sigma)$*-model with the same* Σ*-reduct is (weakly) amalgamable to a pair consisting of a* $\Phi^{e2}(\Phi^{d2}(\Sigma))$*- and a* $\Phi^{e1}(\Phi^{d1}(\Sigma))$*-model having the same* $\Phi^d(\Sigma)$*-reduct.*

$\mathcal{I}$ *is called* lax-quasi-exact*, if each for pair of arrows* $j1 \xrightarrow{d_1} i \xleftarrow{d_2} j2$ *in* Ind*, there is some weakly amalgamable square of lax triangles as above, such that additionally* $\mathcal{I}^k$ is quasi-semi-exact.

Theorem 29 *For a 2-indexed coinstitution* $\mathcal{I}\colon Ind^* \longrightarrow \mathbf{CoIns}$*, assume that*

- *$\mathcal{I}$ is lax-quasi-exact, and*
- *all institution comorphisms in $\mathcal{I}$ are weakly exact.*

Then $\mathcal{I}^\#/\equiv$ *is quasi-semi-exact.*

8. Algorithms for the Relaxed Setting

Call a diagram *connected* if the graph underlying its index category is connected when the identity arrows are deleted. A diagram is *thin*, or a *preorder*, if its index category is thin, i.e. there is at most one arrow between two objects. A preorder is *finitely bounded inf-complete* if any two elements with a common lower bound have an infimum.

Corollary 30 *Let* $\mathcal{I}$ *satisfy the assumptions of Theorem 29. Then* $\mathcal{I}^\#/\equiv$ *admits weak amalgamation of connected finitely bounded inf-complete diagrams.*

Proof. Let $D\colon J \longrightarrow \mathbb{S}ign^\#$ be a connected diagram and let Max be the set of maximal nodes in J. We successively construct new diagrams out of J. Take two nodes in Max that have a common lower bound (if two such nodes do not exist, the diagram is not connected). By Theorem 29, there is a weak amalgamating cocone for the sub-diagram consisting of the two maximal nodes and their infimum (together with the arrows from it into the maximal nodes). Extend the diagram with the cocone. The diagram thus obtained now has a set of maximal nodes whose size is decreased by one. By iterating this construction, we get a diagram with one maximal node. The maximal node then is just the tip of a weakly amalgamating cocone for the original diagram.

Analogous to Cor. 23, we have:

Corollary 31 *The proof system for development graphs with hiding [15] can be re-used for Grothendieck institutions satisfying the assumptions of Cor. 30, provided that each of the involved institutions can be mapped (via a model-expansive comorphism) into a proof-supported institution.*

This result leads to a weakly amalgamable square in the Grothendieck institution as follows:

$$\begin{tikzcd}
 & (\mathbf{PFOL}^{=} \circ \Phi, \Sigma_p) \arrow[dl, "(toPFOL^{-}{,}id)"'] \arrow[dr, "(toPFOL^{+}{,}id)"] & \\
(\mathbf{PLNG}, \Sigma_P) \arrow[dr, "(PLNG2HOL{,}id)"'] & & (\mathbf{PFOL}^{=}, \Sigma_S) \arrow[dl, "(PFOL2HOL{,}\theta_{\Sigma_S})"] \\
 & (\mathbf{HOL}^{=}, PLNG2HOL(\Sigma_P)) &
\end{tikzcd}$$

The algorithm implemented in HETS for obtaining weakly amalgamable cocones of heterogeneous diagrams has some differences with the construction presented in the Corollary 30. From the practical point of view, it is more convenient not to check whether the entire $\mathcal{I}$ is lax-quasi-exact or if all comorphisms existing in the logic graph are weakly exact, but to test this each time a pair of maximal nodes is chosen. Since the tests may fail to hold for a particular situation, we use backtracking on pairs of maximal nodes and weakly amalgamable squares of lax triangles to explore all possible choices. Also, for homogeneous diagrams, weakly amalgamable cocones are computed within the institution, without further assumptions on the shape of diagram.

Another difference is that the situation when the preorder is not finitely bounded inf-complete is also considered, i.e. the two maximal nodes do not have an infimum, but several maximal common lower bounds. For simplicity, we may assume that only two such maximal common lower bounds exist, see following diagram:

$$\begin{tikzcd}
(i_1, \Sigma_1) & & (i_2, \Sigma_2) \\
(i_3, \Sigma_3) \arrow[u, "(d_1{,}\varphi_1)"] \arrow[urr, "(d_2{,}\varphi_2)"'] & & (i_4, \Sigma_4) \arrow[u, "(d_4{,}\varphi_4)"'] \arrow[ull, "(d_3{,}\varphi_3)"]
\end{tikzcd}$$

Then, one of these common lower bounds, in our case say (i_3, Σ_3) is selected for building the span for which we compute the weak amalgamating cocone, as in Theorem 29. Assume this is $(i_1, \Sigma_1) \xrightarrow{(e_1,\rho_1)} (j, \Sigma) \xleftarrow{(e_2,\rho_2)} (i_2, \Sigma_2)$. If the equality $d_3; e_1 = d_4; e_2$ holds and $Sign^j$ has coequalizers, we consider the following double arrow

$$\begin{tikzcd}
\Phi^{d_3;e_1}(\Sigma_4) \arrow[r, bend left, "\Phi^{e_1}(\varphi_3);\rho_1"] \arrow[r, bend right, "\Phi^{e_2}(\varphi_4);\rho_2"'] & \Sigma \arrow[r, dotted, "\gamma"'] & \Sigma'
\end{tikzcd}$$

for which we compute the coequalizer (γ, Σ'). Then the diagram is extended with $(i_1, \Sigma_1) \xrightarrow{(e_1,\rho_1;\gamma)} (j, \Sigma') \xleftarrow{(e_2,\rho_2;\gamma)} (i_2, \Sigma_2)$. If for a particular choice of maximal bound, one of the assumptions fails to hold, a new common lower bound is selected until all have been considered.

Definition 32 *A weakly amalgamable square of lax triangles*

$$\begin{tikzcd}
 & i & \\
j2 \arrow[ur, "d2"] \arrow[r, phantom, "\overset{u2}{\Longrightarrow}"] & {} & j1 \arrow[ul, "d1"'] \arrow[l, phantom, "\overset{u1}{\Longleftarrow}"] \\
 & k \arrow[ul, "e2"] \arrow[ur, "e1"'] \arrow[uu, "d"] &
\end{tikzcd}$$

is sufficiently large *for a set of spans in Ind of shape $j_1 \xleftarrow{f1_a} k_a \xrightarrow{f2_a} j_2$ if I^k has coequalizers and for each of the spans in the set, $f1_a; d1 = f2_a; d2$.*

A diagram is *compatible with squares* if for any two maximal nodes of indexes j_1 and j_2 there exists a choice of a maximal common lower bound with the index i and a weakly amalgamable square of lax triangles which is sufficiently large for the set of spans obtained from the other maximal common bounds.

Corollary 33 *Let $\mathcal{I}$ satisfy the assumptions of Theorem 29. Then $\mathcal{I}^{\#}/\equiv$ admits weak amalgamation of connected finitely thin diagrams if when extending the diagram with new maximal nodes the compatibility with squares is preserved.*

To conclude, the algorithm's steps are the following:

1. Check whether the diagram is homogeneous. If it is, compute a weakly amalgamable cocone in the underlying institution.
2. If the diagram is not connected or not thin, the algorithm fails.
3. Let Max be the set of maximal nodes of the diagram. If Max has only one element, then this is a weakly amalgamable cocone of the original diagram.

4. Pick two maximal nodes that have a common lower bound (the diagram is connected, so we know they exist).

5. Check whether the maximal nodes have an infimum. If they do, compute a weakly amalgamable cocone of the span obtained from the arrows from this infimum to the maximal nodes and extend the diagram with it. Then return to step 3. If we fail to compute the weakly amalgamable cocone (i.e. we do not have the square of lax triangles), return to step 4 to make a new choice.

6. If the two maximal nodes do not have an infimum, compute the list of all maximal common lower bounds of the two nodes.

7. Pick a maximal lower bound and compute a weakly amalgamable cocone of the span obtained from it and the two maximal nodes.

8. For all the others maximal lower bounds, check whether the coequalizers can be computed (as explained above). If this succeeds, extend the diagram with the new node and the arrows to it and go back to step 3. If it fails, return to step 7 to pick another bound. If all the options have failed, return to step 4 to pick new maximal nodes.

Notice that the algorithm could find several weakly amalgamable cocones, if there are more squares of lax triangles available for two maximal nodes and a bound. We prefer to display all possible answers and let the user select a cocone, since a certain logic may have better problem-specific tool support. This is also the main advantage over an algorithm that translates the entire diagram to some "universal" logic and computes its colimit.

During the implementation of the algorithm, we also needed to test whether two arbitrary compositions of institution comorphism modifications (as natural transformation, therefore both horizontal and vertical compositions[2]) are equal. These two kinds of compositions are related by the so-called "Interchange Law" stating that for any natural transformations $\gamma, \mu, \eta, \epsilon$

$$(\gamma * \eta) \circ (\mu * \epsilon) = (\gamma \circ \mu) * (\eta \circ \epsilon)$$

when the compositions on the left side are defined. Using this law and rules for cancelling identities, we develop a term rewrite system (see figure 4) which we prove terminating and confluent. Then, two arbitrary terms denoting valid compositions are equal if they rewrite to the same normal form.

While termination of the term rewrite system is easy to notice (since for all the rules, on the right side either the depth of term or the number of horizontal compositions decreases), proving confluence is a little more difficult. We used the Church-Rosser checker [7] written in Maude to obtain the critical pairs of the term rewriting system and then noticed that all of them are eliminated in the case of type-correct terms (i.e. modifications that actually compose).

[2]We denote $*$ the horizontal composition and $\circ$ the vertical one.

$$\begin{array}{l}
(\gamma * \eta) \circ (\mu * \epsilon) \rightarrow (\gamma \circ \mu) * (\eta \circ \epsilon) \\
1_F \circ \gamma \rightarrow \gamma \\
\gamma \circ 1_G \rightarrow \gamma \\
1_F * 1_G \rightarrow 1_{F;G}
\end{array}$$

Figure 4. Rewrite rules for deciding equality of comorphism modifications

9. Conclusion

We have presented an algorithm that generalises the computation of colimits of specifications to a heterogeneous setting. It has turned out that the notion of (amalgamable) colimit has to be replaced by that of weakly amalgamable cocone in order to obtain a framework that is general enough to cover practically interesting cases. Moreover, the algorithm provides a true colimit whenever this is possible. We have illustrated the approach with two examples: one involving the relation between specification and programming. For this example, the 2-categorical machinery is needed in order to construct a weakly amalgamable cocone (which in turn is essential for proving refinements in the proof calculus for development graphs with hiding [15]). The other example concerns ontologies for bibliographical information, and links the schema of a relational database with an ontology specified in description logic. Here, the heterogeneous situation is simpler, because the involved formalisms can be mapped to first-order logic, where also an interface theory lives. However, the logical structure is a bit more complex: in a sense, a refinement between *two* weakly amalgamable cocones needs to be proved. This approach has the advantage (compared with the integration into a common reference ontology pursued in [18]) that the involved axiomatisations can be directly compared w.r.t. their strength. We have pointed out where the strength differs, and how this can be changed if wanted. The integration into a common reference ontology in [18] is of much weaker nature: it just states that the two axiomatizations have a common upper bound. It should be stressed that the cocone computed for this example falls outside the scope of the standard theorems from the literature [5], because **FOL** generally is not the colimit institution for this diagram.

Concerning related work, [10] tackle the same problem as the present paper, but involving the invention of new institutions, without making clear how these will be equipped with proof systems. Moreover, amalgamation is not studied at all.

The algorithm is being implemented as part of the Heterogeneous Tool Set HETS. Future work should provide more applications to specific examples of heterogeneous specifications and ontologies.

Acknowledgments Thanks for Oliver Kutz for discussions about the ontology example. This work has been supported by the German Federal Ministry of Education and Research (Project 01 IW 07002 FormalSafe).

References

[1] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.

[2] T. Borzyszkowski. Moving specification structures between logical systems. In J. L. Fiadeiro, editor, *WADT 1998*, *LNCS* 1589, p. 16–30. Springer, 1999.

[3] B. Braatz, M. Klein, and G. Schröter. Semantical integration of object-oriented viewpoint specification techniques. In *SoftSpez Final Report*, *LNCS* 3147, p. 602–626, 2004.

[4] R. M. Burstall and J. A. Goguen. Putting theories together to make specifications. In *IJCAI*, p. 1045–1058, 1977.

[5] R. Diaconescu. Grothendieck institutions. *Applied categorical structures*, 10:383–402, 2002.

[6] R. Diaconescu, J. Goguen, and P. Stefaneas. Logical support for modularisation. In *Logical Environments*, p. 83–130. Cambridge, 1993.

[7] F. Durán and J. Meseguer. A Church-Rosser checker tool for Maude equational specifications. Technical report, Universidad de Málaga and SRI International, July 2000.

[8] J. Goguen and G. Roşu. Institution morphisms. *Formal Aspects of Computing*, 13:274–307, 2002.

[9] J. A. Goguen and R. M. Burstall. Institutions: Abstract Model Theory for Specification and Programming. *Journal of the ACM*, 39:95–146, 1992.

[10] E. H. Haeusler, A. Martini, and U. Wolter. Some models of heterogeneous and distributed specifications based on universal constructions. In J.-Y. Béziau and A. Costa-Leite, editors, *Perspectives on Universal Logic*, p. 297–318. Italy, 2007.

[11] N. Martí-Oliet, J. Meseguer, and M. Palomino. Theoroidal maps as algebraic simulations. In J. L. Fiadeiro, P. D. Mosses, and F. Orejas, editors, *WADT 2004*, *LNCS* 3423, p. 126–143. Springer, 2004.

[12] T. Mossakowski. Comorphism-based Grothendieck logics. In K. Diks and W. Rytter, editors, *MFCS 2002*, *LNCS* 2420, p. 593–604. Springer, 2002.

[13] T. Mossakowski. Relating CASL with other specification languages: the institution level. *Theoret. Comp. Sci.*, 286:367–475, 2002.

[14] T. Mossakowski. Institutional 2-cells and grothendieck institutions. In K. Futatsugi, J.-P. Jouannaud, and J. Meseguer, editors, *Algebra, Meaning and Computation*, *LNCS* 4060, p. 124–149. Springer, 2006.

[15] T. Mossakowski, S. Autexier, and D. Hutter. Development graphs – proof management for structured specifications. *JLAP*, 67(1-2):114–145, 2006.

[16] T. Mossakowski, C. Maeder, and K. Lüttich. The Heterogeneous Tool Set. In O. Grumberg and M. Huth, editors, *TACAS 2007*, *LNCS* 4424, p. 519–522. Springer-Verlag, 2007.

[17] D. Sannella and A. Tarlecki. Specifications in an arbitrary institution. *Inform. and Comput.*, 76:165–210, 1988.

[18] W. M. Schorlemmer and Y. Kalfoglou. Institutionalising Ontology-Based Semantic Integration. *Journal of Applied Ontology*, 2007. To appear.

[19] L. Schröder, T. Mossakowski, A. Tarlecki, P. Hoffman, and B. Klin. Amalgamation in the semantics of CASL. *Theoret. Comp. Sci.*, 331(1):215–247, 2005.

[20] A. Tarlecki. Moving between logical systems. In M. Haveraaen, O. Owe, and O.-J. Dahl, editors, *WADT 1995*, *LNCS* 1130, p. 478–502. Springer Verlag, 1996.

[21] K. E. Williamson, M. Healy, and R. A. Barker. Industrial applications of software synthesis via category theory-case studies using specware. *Autom. Softw. Eng*, 8(1):7–30, 2001.

[22] A. Zimmermann, M. Krötzsch, J. Euzenat, and P. Hitzler. Formalizing Ontology Alignment and its Operations with Category Theory. In B. Bennett and C. Fellbaum, editors, *FOIS 2006*, volume 150 of *Frontiers in Artificial Intelligence and Applications*, p. 277–288. IOS Press, NOV 2006.