A Relatively Complete Calculus for Structured Heterogeneous Specifications[⋆]

Till Mossakowski[1] and Andrzej Tarlecki[2]

[1] Faculty of Computer Science, Otto-von-Guericke University of Magdeburg
[2] Institute of Informatics, University of Warsaw

Abstract. Proof calculi for structured specifications have been developed independently of the underlying logical system (formalised as institution). Typically, completeness of these calculi requires interpolation properties of the underlying logic. We develop a relatively complete calculus for structured heterogeneous specifications that does not need interpolation.

1 Introduction

The theory of *institutions* [GB92] provides an excellent framework where the theory of specification and formal software development may be presented in an adequately general and abstract way [ST88a, ST12]. The initial work within this area captured specifications built and developments carried out in an arbitrary but fixed logical system formalised as an institution. However, the practice of software specification and development goes much beyond this. Different logical systems may be appropriate or most convenient for specification of different modules of the same system, of different aspects of system behaviour, or of different stages of system development. This leads to the need for a number of logical systems to be used in the same specification and development project, linked by appropriate notions of morphisms between institutions [GR02]. This observation spurred a substantial amount of research work already, and motivates the research presented here.

In such a framework, one works in a *heterogeneous logical environment* formed by a number of logical systems formalised as institutions and linked with each other in a way captured by various maps between institutions. One such logical environment is the CafeOBJ cube [DF02], another one the HETS family of institutions [Mos05], supported by a tool to build and work with heterogeneous specifications [MML07]. The standard ways of building structured specifications within an institution may then be complemented by heterogeneous specification building constructs, that allow one to move specifications from one institution to another, and then combine specifications originally built in different institutions [Tar00, DF02, MML07, Mos05, MT09].

We study here proof systems for so obtained structured heterogeneous specifications. Of course, we build on the calculi that deal with homogeneous specifications, constructed within a single institution. This topic has been well-studied [ST88a, Bor02, Dia08, ST12], with completeness of the resulting systems being the

⋆ This work has been partially supported by the German Research Foundation (DFG) via the SFB/TR 8 “Spatial Cognition”, project I1-[OntoSpace] (TM) and by the Polish Ministry of Science and Higher Education, grant N206 493138 (AT).

main problematic issue. The completeness results, where they can be achieved, either rely on strong interpolation properties, or sacrifice compositionality of the proof systems, allowing the structure of the specifications to be flattened out entirely. We propose a middle way here, keeping as much as possible of the specification structure, and still ensuring completeness of the resulting calculus. We argue that in many practical situations the structure that is kept is relevant, and the minimal massaging of the specifications we suggest brings no real harm.

We extend this idea to heterogeneous specifications, where the required interpolation property cannot be really expected, and our approach is in fact the only realistically possible. A technical tool here is the notion of modification between institution maps (which we adapt from [Dia02]) and lax compatibility of such maps, which serves us to formulate the necessary (and realistic) compatibility conditions that make the completeness of the resulting calculus for structured heterogeneous specifications achievable.

2 Structured Specifications and Proofs

Let us begin by recalling the notion of an institution, as a formalisation of an arbitrary logical system [GB92], assuming that the reader is familiar with all the intuitions that this notion brings in (see [Mac98] for an introduction to category theory).

Definition 2.1. *An* institution $\mathcal{I}$ *consists of:*

- *a category* $\mathbf{Sign}_{\mathcal{I}}$ *of* signatures*;*
- *a functor* $\mathbf{Sen}_{\mathcal{I}} \colon \mathbf{Sign}_{\mathcal{I}} \to \mathbf{Set}$,[1] *giving a set* $\mathbf{Sen}(\Sigma)$ *of* Σ-sentences *for each signature* $\Sigma \in |\mathbf{Sign}_{\mathcal{I}}|$, *and a function* $\mathbf{Sen}(\sigma) \colon \mathbf{Sen}(\Sigma) \to \mathbf{Sen}(\Sigma')$, *denoted by* $\sigma(\_)$, *that yields* σ-translation *of* Σ*-sentences to* Σ'*-sentences for each signature morphism* $\sigma \colon \Sigma \to \Sigma'$*;*
- *a functor* $\mathbf{Mod}_{\mathcal{I}} \colon \mathbf{Sign}_{\mathcal{I}}^{op} \to \mathbf{Class}$,[2] *giving a class* $\mathbf{Mod}(\Sigma)$ *of* Σ-models *for each signature* $\Sigma \in |\mathbf{Sign}_{\mathcal{I}}|$, *and a functor* $\mathbf{Mod}(\sigma) \colon \mathbf{Mod}(\Sigma') \to \mathbf{Mod}(\Sigma)$, *denoted by* $\_|_{\sigma}$, *that yields* σ-reducts *of* Σ'*-models for each signature morphism* $\sigma \colon \Sigma \to \Sigma'$*; and*
- *for each* $\Sigma \in |\mathbf{Sign}_{\mathcal{I}}|$, *a* satisfaction relation $\models_{\mathcal{I},\Sigma} \subseteq \mathbf{Mod}_{\mathcal{I}}(\Sigma) \times \mathbf{Sen}_{\mathcal{I}}(\Sigma)$

such that for any signature morphism $\sigma \colon \Sigma \to \Sigma'$, Σ*-sentence* $\varphi \in \mathbf{Sen}_{\mathcal{I}}(\Sigma)$ *and* Σ'*-model* $M' \in \mathbf{Mod}_{\mathcal{I}}(\Sigma')$*:*

$$M' \models_{\mathcal{I},\Sigma'} \sigma(\varphi) \iff M'|_{\sigma} \models_{\mathcal{I},\Sigma} \varphi \qquad [\textit{Satisfaction condition}]$$

The satisfaction condition expresses that truth is invariant under change of notation and context.

Example 2.2. Propositional Logic. The institution $\mathbf{Prop}$ of propositional logic has sets Σ (of propositional symbols) as signatures, and functions $\sigma \colon \Sigma_1 \to \Sigma_2$ between such sets as signature morphisms. A Σ-model M is a mapping from Σ to $\{\mathit{true}, \mathit{false}\}$. The

[1] The category $\mathbf{Set}$ has all sets as objects and all functions as morphisms.

[2] $\mathbf{Class}$ is the quasi-category of all classes, where "quasi" means that it lives in a higher set-theoretic universe. If model morphisms are needed, one may use categories instead of classes.

reduct of a Σ_2-model M_2 along $\sigma\colon \Sigma_1 \to \Sigma_2$ is the Σ_1-model given by the composition $\sigma; M_2$.[3] Σ-sentences are built from Σ with the usual propositional connectives, and sentence translation along a signature morphism just replaces the propositional symbols along the morphism. Finally, satisfaction of a sentence in a model is defined by the standard truth-table semantics. It is straightforward to see that the satisfaction condition holds.

Example 2.3. Untyped First-order Logic. In the institution $\mathbf{UFOL}^=$ of untyped first-order logic with equality, signatures are first-order signatures, consisting of a set of function symbols with arities, and a set of predicate symbols with arities. Signature morphisms map symbols so that arities are preserved. Models are first-order structures, and 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.

Many-sorted First-order Logic. The institution $\mathbf{FOL}^=$ of many-sorted first-order logic with equality is similar to $\mathbf{UFOL}^=$. Signatures are many-sorted first-order signatures, consisting of sorts and typed function and predicate symbols. The rest is similar to $\mathbf{UFOL}^=$. For details, see [GB92].

Many-sorted Partial First-order Logic. The institution $\mathbf{PFOL}^=$ is similar to $\mathbf{FOL}^=$, but functions can be partial. Atomic formulas evaluate to false if some component term involves some undefinedness. See [CoF04].

CASL extends $\mathbf{PFOL}^=$ with subsorting and induction (for datatypes), see [CoF04].

Many-sorted Equational Logic ($\mathbf{EqL}$) is the sublogic of $\mathbf{FOL}^=$ restricting signatures to those without predicate symbols and sentences to universally quantified equations.

In any institution $\mathcal{I}$, standard logical notions, like the model class $Mod(\Gamma)$ for any set Γ of sentences, semantic (logical) consequence $\Gamma \models \varphi$ for any set Γ of sentences and sentence φ over the same signature, are defined as usual. In particular, a *theory* is a pair $T = \langle \Sigma, \Gamma \rangle$, where $\Sigma \in \mathbf{Sign}$ and $\Gamma \subseteq \mathbf{Sen}(\Sigma)$. *Theory morphisms* are signature morphisms mapping axioms to logical consequences, leading to a category $\mathbf{Th}$ of theories. It is easy to extend this to an *institution of theories* $\mathcal{I}^{th} = (\mathbf{Th}, \mathbf{Sen}, \mathbf{Mod}, \models)$ over $\mathcal{I}$.

Definition 2.4. *A cocone for a diagram in* $\mathbf{Sign}$ *is* (weakly) amalgamable *if it is mapped to a (weak) limit in* $\mathbf{Class}$ *under* $\mathbf{Mod}$. $\mathcal{I}$ *(or* $\mathbf{Mod}$*) admits* (finite) (weak) amalgamation *if (finite) colimits exists in* $\mathbf{Sign}$ *and colimiting cocones are (weakly) amalgamable, i.e. if* $\mathbf{Mod}$ *maps (finite) colimits to (weak) limits. An important special case is pushouts:* $\mathcal{I}$ *(or* $\mathbf{Mod}$*) is* (weakly) semi-exact*, if pushouts exist in* $\mathbf{Sign}$ *and are (weakly) amalgamable.*

Definition 2.5. *An institution* $\mathcal{I}$ *is* quasi-exact *if for each diagram* $D\colon J \longrightarrow \mathbf{Sign}$*, there is some weakly amalgamable cocone over* D. Quasi-semi-exactness *is the restriction of this notion to diagrams of shape* $\bullet \longleftarrow \bullet \longrightarrow \bullet$.

[3] We write composition in any category in the diagrammatic order and denote it by ";".

We recall a variant of Craig interpolation that better fits for logics that may have no implication, namely *Craig-Robinson interpolation* [DM00, Sho67].

Definition 2.6. *Given an institution $\mathcal{I}$, a commuting square in* $\mathbf{Sign}$

$$\begin{tikzcd} \Sigma \arrow[r, "\varphi_1"] \arrow[d, "\varphi_2"'] & \Sigma_1 \arrow[d, "\theta_1"] \\ \Sigma_2 \arrow[r, "\theta_2"'] & \Sigma' \end{tikzcd}$$

admits Craig-Robinson interpolation *whenever for all finite sets of sentences $\Psi_1 \subseteq \mathbf{Sen}(\Sigma_1)$ and $\Psi_2, \Gamma_2 \subseteq \mathbf{Sen}(\Sigma_2)$, if $\theta_1(\Psi_1) \cup \theta_2(\Gamma_2) \models \theta_2(\Psi_2)$ then there exists a finite set Ψ of Σ-sentences such that $\Psi_1 \models \varphi_1(\Psi)$ and $\varphi_2(\Psi) \cup \Gamma_2 \models \Psi_2$.*

$\mathcal{I}$ has Craig-Robinson interpolation *if all signature pushouts admit Craig-Robinson interpolation.*

This is the category-theoretic generalisation of the usual notion of interpolation. In particular, the usual notion of "common language of Ψ_1 and Ψ_2" is generalised to an arbitrary span $\Sigma_1 \xleftarrow{\varphi_1} \Sigma \xrightarrow{\varphi_2} \Sigma_2$ (imagine Σ to be the intersection $\Sigma_1 \cap \Sigma_2$). Craig interpolation is a weaker version of Craig-Robinson interpolation, with $\Gamma_2 = \emptyset$ in Def. 2.6.

Institutions were originally introduced to free the theory of specifications from dependency on any particular logical system. We follow [ST88a] and for any institution $\mathcal{I}$ consider a class $Spec_{\mathcal{I}}$ of specifications built in $\mathcal{I}$ from *basic specifications* (*presentations*, which consist of a signature and a set of sentences over this signature) by means of a number of *specifications-building operations*. Fix an institution $\mathcal{I} = (\mathbf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \models)$. Simultaneously with the notion of structured specification, we define functions *Sig* and *Mod* yielding the signature and the model class for any specification.

***presentations*:** For any signature $\Sigma \in |\mathbf{Sign}|$ and finite set $\Gamma \subseteq \mathbf{Sen}(\Sigma)$ of Σ-sentences, the *presentation* $\langle \Sigma, \Gamma \rangle$ is a specification with:

$$Sig[\langle \Sigma, \Gamma \rangle] := \Sigma \qquad Mod[\langle \Sigma, \Gamma \rangle] := \{M \in \mathbf{Mod}(\Sigma) \mid M \models \Gamma\}$$

***union*:** For any signature $\Sigma \in |\mathbf{Sign}|$, given Σ-specifications SP_1 and SP_2, their *union* $SP_1 \cup SP_2$ is a specification with:

$$Sig[SP_1 \cup SP_2] := \Sigma \qquad Mod[SP_1 \cup SP_2] := Mod[SP_1] \cap Mod[SP_2]$$

***translation*:** For any signature morphism $\sigma\colon \Sigma \longrightarrow \Sigma'$ and Σ-specification SP, $\sigma(SP)$ is a specification with:

$$Sig[\sigma(SP)] := \Sigma' \qquad Mod[\sigma(SP)] := \{M' \in \mathbf{Mod}(\Sigma') M'|_\sigma \in Mod[SP]\}$$

***hiding*:** For any signature morphism $\sigma\colon \Sigma \longrightarrow \Sigma'$ and Σ'-specification SP', $SP'|_\sigma$ is a specification with:

$$Sig[SP'|_\sigma] := \Sigma \qquad Mod[SP'|_\sigma] := \{M'|_\sigma \mid M' \in Mod[SP']\}$$

Typical structuring constructs of many existing specification languages like CASL [CoF04], CafeOBJ [DF02] and others can be mapped to this kernel formalism.

The semantics determines specification equivalence: $SP_1 \equiv SP_2$ iff $\mathit{Sig}[SP_1] = \mathit{Sig}[SP_2]$ and $\mathit{Mod}[SP_1] = \mathit{Mod}[SP_2]$. Furthermore, we get the obvious notion of semantic consequence for structured specifications: given a specification SP, a sentence $\varphi \in \mathbf{Sen}(\mathit{Sig}[SP])$ is a *semantic consequence* of SP, written $SP \models \varphi$, if $M \models \varphi$ for all models $M \in \mathit{Mod}[SP]$. Then, given two specifications SP and SP', SP *refines* to SP', written $SP \rightsquigarrow SP'$, if $\mathit{Sig}[SP] = \mathit{Sig}[SP']$ and $\mathit{Mod}[SP'] \subseteq \mathit{Mod}[SP]$. These two simple notions underlie the standard view of properties that specifications ensure and of systematic development of programs from specifications by step-wise refinements, see [ST88b, ST12].

3 Proofs

We very briefly recalled above the semantic concepts developed within the theory of institutions that underlie the methodology for formal specification and systematic development of software, cf. [ST12]. For practical applications they need a proof-theoretic counterpart, whereby the semantic, hard to establish relationships are augmented by calculi to approximate them in an effective way.

Proof-theoretic entailment to approximate semantic entailment in any institution is captured by the following notion, introduced in the institutional context in [FS88] under the name of π-institution, see also [Mes89, HST94].

Definition 3.1. *Given an institution* $\mathcal{I} = (\mathbf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \models)$*, an* entailment system $\vdash$ *for* $\mathcal{I}$ *consists of a relation* $\vdash_\Sigma \subseteq \mathcal{P}(\mathbf{Sen}(\Sigma)) \times \mathbf{Sen}(\Sigma)$ *for each* $\Sigma \in |\mathbf{Sign}|$*, such that the following properties are satisfied:*

1. reflexivity: *for any* $\varphi \in \mathbf{Sen}(\Sigma)$*,* $\{\varphi\} \vdash_\Sigma \varphi$*,*
2. monotonicity: *if* $\Gamma \vdash_\Sigma \varphi$ *and* $\Gamma' \supseteq \Gamma$ *then* $\Gamma' \vdash_\Sigma \varphi$*,*
3. transitivity: *if* $\Gamma \vdash_\Sigma \varphi_i$ *for* $i \in I$ *and* $\Gamma \cup \{\varphi_i \mid i \in I\} \vdash_\Sigma \psi$*, then* $\Gamma \vdash_\Sigma \psi$*,*
4. $\vdash$-translation: *if* $\Gamma \vdash_\Sigma \varphi$*, then for any* $\sigma \colon \Sigma \longrightarrow \Sigma'$ *in* $\mathbf{Sign}$*,* $\sigma(\Gamma) \vdash_{\Sigma'} \sigma(\varphi)$*,*
5. soundness: *if* $\Gamma \vdash_\Sigma \varphi$ *then* $\Gamma \models_\Sigma \varphi$*.*

The entailment system is complete *if, in addition,* $\Gamma \models_\Sigma \varphi$ *implies* $\Gamma \vdash_\Sigma \varphi$*.*

A *logic* $\mathcal{LOG} = (\mathbf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \models, \vdash)$ is an institution $(\mathbf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \models)$ equipped with an entailment system $\vdash$. In an arbitrary logic, it is possible to design a logic independent proof calculus [ST88a] for proving entailments between specifications and sentences, written in the form $SP \vdash \varphi$, where SP is a structured specification and φ is a formula, see Fig. 1.

$$(\mathit{CR})\ \frac{\{SP \vdash \varphi_i\}_{i\in I}\quad \{\varphi_i\}_{i\in I} \vdash \varphi}{SP \vdash \varphi} \qquad (\mathit{basic})\ \frac{\varphi \in \Gamma}{\langle \Sigma, \Gamma\rangle \vdash \varphi} \qquad (\mathit{sum1})\ \frac{SP_1 \vdash \varphi}{SP_1 \cup SP_2 \vdash \varphi}$$

$$(\mathit{sum2})\ \frac{SP_2 \vdash \varphi}{SP_1 \cup SP_2 \vdash \varphi} \qquad (\mathit{trans})\ \frac{SP \vdash \varphi}{\sigma(SP) \vdash \sigma(\varphi)} \qquad (\mathit{derive})\ \frac{SP \vdash \sigma(\varphi)}{SP|_\sigma \vdash \varphi}$$

Fig. 1. Proof calculus for entailment in structured specifications

$$(\mathit{Basic})\ \frac{SP \vdash \varphi \ \text{ for all } \varphi \in \Gamma}{\langle \Sigma, \Gamma \rangle \rightsquigarrow SP} \qquad (\mathit{Sum})\ \frac{SP_1 \rightsquigarrow SP \quad SP_2 \rightsquigarrow SP}{SP_1 \cup SP_2 \rightsquigarrow SP}$$

$$(\mathit{Trans})\ \frac{SP \rightsquigarrow SP'|_\sigma}{\sigma(SP) \rightsquigarrow SP'} \qquad (\mathit{Derive})\ \frac{SP \rightsquigarrow SP''}{SP|_\sigma \rightsquigarrow SP'} \quad \text{if } \sigma\colon SP' \longrightarrow SP'' \text{ is a conservative extension}$$

Fig. 2. Proof calculus for refinement of structured specifications

Fig. 2 shows an extension of this calculus to refinements between specifications, with judgements written as $SP \rightsquigarrow SP'$, where SP and SP' are structured specifications with a common signature. Note that rule (CR) can be limited to a finitary version for compact institutions (where an institution is compact if $\Gamma \models \varphi$ implies the existence of a finite $\Gamma' \subseteq \Gamma$ with $\Gamma' \models \varphi$). The extended calculus relies on an *oracle for conservative extensions*, where given specifications SP and SP', a signature morphism $\sigma : \mathit{Sig}[SP] \to \mathit{Sig}[SP']$ is a *conservative extension* if it is a specification morphism $\sigma : SP \to SP'$ (i.e., $M'|_\sigma \in \mathit{Mod}[SP]$ for all $M' \in \mathit{Mod}[SP']$) and is *conservative* (for all $M \in \mathit{Mod}[SP]$ there is $M' \in \mathit{Mod}[SP']$ with $M'|_\sigma = M$).

Theorem 3.2 (Soundness [ST88a, Bor02]). *The calculi for specification entailment and refinement between structured specifications given above are sound: if $SP \vdash \varphi$ then $SP \models \varphi$, and if $SP \rightsquigarrow SP'$ then $SP \gtrdot SP'$.*

Theorem 3.3 (Completeness [Bor02, Dia08, ST13]). *Assuming that*

- *the institution has* Craig-Robinson interpolation*,*
- *the institution is* weakly semi-exact*,*
- *the entailment system is* complete*,*

the calculi for specification entailment and refinement between structured specifications are sound and complete: $SP \vdash \varphi$ iff $SP \models \varphi$, and $SP \rightsquigarrow SP'$ iff $SP \gtrdot SP'$.

Actually, as discussed in [Bor02, ST13], the assumption of Craig-Robinson interpolation and weak amalgamation can be restricted to those pushouts for which it is really needed. Typically, we can limit the classes of morphisms used to build structured specification by hiding and translation, respectively. Under suitable technical conditions, Craig-Robinson interpolation is needed then for pushouts of spans formed by morphisms permitted in hiding on the left and those permitted in translations on the right. Still, the requirement that the institution admits Craig-Robinson interpolation is the strongest assumption in Thm. 3.3. While it holds in many logics, there are prominent examples where is fails. For example, even Craig interpolation fails in $\mathbf{QS5}$, the first-order version of the modal logic $\mathbf{S5}$ [Fin79], which is just one instance of many failures of interpolation in various versions of modal logics. Interpolation also fails in some typical logical systems used in specification formalisms, with interpretation of some types or concepts fixed semantically; for instance, interpolation fails for the logic of CASL due to CASL-style subsorting [Bor00]. Even the standard first-order logic may cause problems here. While untyped first-order logic $\mathbf{UFOL}^=$ has Craig-Robinson interpolation, its many-sorted version $\mathbf{FOL}^=$ admits Craig-Robinson interpolation for pushouts of spans where at least one morphism is injective on sorts. To use Thm. 3.3

even in the refined version hinted at above for specifications in $\mathbf{FOL}^=$ we would have to limit the use of hiding to signature morphisms that are injective on sorts — a seriously limiting restriction. Things get even worse with many-sorted equational logic $\mathbf{EqL}$, which admits Craig-Robinson interpolation for pushouts of spans with the left morphism satisfying a strong "encapsulation" property, see [Dia08] (Craig, but not necessarily Craig-Robinson interpolation, is also ensured here for pushouts of spans with the right morphism being injective).

As shown in [ST13], Craig-Robinson interpolation is necessary for the completeness of the above calculi, and moreover, the calculus for specification entailments cannot be improved without sacrificing its *compositionality* (consequences of a structured specification are deduced from the consequences of its immediate components).

When we sacrifice compositionality of the calculus, a sound and complete calculus may be obtained also for institutions without interpolation when we agree that specifications are "massaged" before calculating their consequences, so allowing the calculus to reach arbitrarily deep into the specification structure. This is often done using *normal forms* of specifications. The well-known normal form result is that each structured specification SP as considered here can be turned into an equivalent normal form $\mathit{nf}(SP) = \langle \Sigma', \Gamma' \rangle|_\sigma$, thus entirely flattening the specification to a theory with a single use of hiding (this requires the institution to have relevant signature pushouts that admit weak amalgamation). Then an obvious rule

$$(\mathit{nf})\ \frac{\Gamma' \vdash \sigma(\varphi)}{SP \vdash \varphi} \qquad \text{if } \mathit{nf}(SP) = \langle \Sigma', \Gamma' \rangle|_\sigma$$

yields a sound and complete calculus for specification entailments.

However, this normal form and its use in the above proof rule entirely forgets about any structure that was given in SP. We show how some key aspects of the structure may be maintained without losing the completeness of the calculus. To achieve this we define a *structured normal form* $\mathit{snf}(SP)$ for any structured specification SP, which only pushes out the hiding operations, while retaining the key structure given by union and translation (and, very informally, renaming hidden symbols to avoid unintended name clashes). The definition below requires existence and suitable choice of the relevant signature pushouts:

$$\frac{}{\mathit{snf}(\langle \Sigma, \Gamma \rangle) = \langle \Sigma, \Gamma \rangle|_{id}}$$

$$\frac{\mathit{snf}(SP_1) = SP_1'|_{\sigma_1} \quad \mathit{snf}(SP_2) = SP_2'|_{\sigma_2}}{\mathit{snf}(SP_1 \cup SP_2) = (\theta_1(SP_1') \cup \theta_2(SP_2'))|_{\sigma_1;\theta_1}} \text{ if } \begin{tikzcd} Sig[SP_1] \arrow[r, "\sigma_1"] \arrow[d, "\sigma_2"] & Sig[SP_1'] \arrow[d, "\theta_1"] \\ Sig[SP_2'] \arrow[r, "\theta_2"] & \Sigma' \end{tikzcd} \text{ is a pushout}$$

$$\frac{\mathit{snf}(SP) = SP'|_{\sigma_1}}{\mathit{snf}(\sigma_2(SP)) = (\theta_1(SP'))|_{\theta_2}} \text{ if } \begin{tikzcd} Sig[SP] \arrow[r, "\sigma_1"] \arrow[d, "\sigma_2"] & Sig[SP'] \arrow[d, "\theta_1"] \\ \Sigma_2 \arrow[r, "\theta_2"] & \Sigma' \end{tikzcd} \text{ is a pushout}$$

$$\frac{\mathit{snf}(SP) = SP'|_\sigma}{\mathit{snf}(SP|_\theta) = SP|_{\theta\,;\,\sigma}}$$

Proposition 3.4. *In any weakly semi-exact institution, SP and $\mathit{snf}(SP)$ are equivalent.*

Moreover, we can obtain a stronger completeness result:

Theorem 3.5. *Under the assumptions that the institution is* weakly semi-exact *and the entailment system is* complete*, the calculi for specification entailments and refinement between structured specifications extended by the following* structured normal form rule*:*

$$(\mathit{snf})\ \frac{SP' \vdash \sigma(\varphi)}{SP \vdash \varphi} \qquad \text{if } \mathit{snf}(SP) = SP'|_\sigma$$

are sound and complete.

Let us stress again that using structured normal forms is much better than using normal forms: the latter flatten out specification structure completely, while the former keep the structure almost intact — only hiding, not so frequently used in typical specifications, is moved outside. In many institutions, the obvious choice of signature pushouts involved in the definition of *snf* leads to the structured normal forms where all the visible names are kept as in the original structured specification, while only the hidden operations may need to be renamed so that name clashes are avoided. This allows proof search strategies in structured specifications, as discussed for instance in [SB83, HST94], to be easily mimicked in their corresponding structured normal forms.

Consequently, the above proof calculus for specification entailments with the rule (snf) offers a well-balanced choice, maintaining the key advantages of compositionality and keeping the need for restructuring specifications to the necessary minimum.

A complete oracle for conservative extensions is very powerful: it can be used to trivially obtain a complete refinement calculus. Namely, in order to decide whether $SP_1 \rightsquigarrow SP_2$, it suffices to check whether $SP_1 \cup SP_2$ is a conservative extension of SP_2. Nevertheless, our completeness theorem is meaningful and useful. This is because the completeness proof uses the oracle for conservative extensions only in a limited way. The extensions considered are those obtained from hidings (pushed along some morphism into a "big" signature collecting everything). This means, for example, that if we use hiding only to hide symbols that have been defined using some logic-specific definition scheme, we will need the oracle for conservative extensions only for checking this definition scheme — and typically all such "definitional extensions" are conservative. We cannot expect in general to check conservativity independently of the underlying institution; institution-specific rules are needed. See e.g. [CMM13] for checking conservativity in CASL.

4 Heterogeneous Specifications

So far, we have covered specifications built and their refinements carried out in an arbitrary but fixed logical system formalised as an institution. In practice though, different logical systems may be appropriate or most convenient for specification of different modules of the same system, of different aspects of system behaviour, or of different stages of system development. This leads to the need for a number of logical systems to be used in the same specification and development project. This makes sense though only if the logical systems involved (formalised as institutions) are linked appropriately, with links captured by various notions of morphisms between institutions [GR02], yielding heterogeneous specification environments such as those of CafeOBJ [DF02] and HETS [MML07].

Definition 4.1. *An* institution comorphism $\rho\colon \mathcal{I} \to \mathcal{I}'$ *consists of:*

- *a functor* $\Phi\colon \mathbf{Sign} \to \mathbf{Sign}'$*;*
- *a natural transformation* $\alpha\colon \mathbf{Sen} \to \Phi\,;\mathbf{Sen}'$*, and*
- *a natural transformation* $\beta\colon \Phi^{op}\,;\mathbf{Mod}' \to \mathbf{Mod}$*,*

such that for any $\Sigma \in |\mathbf{Sign}|$*, for any* $\varphi \in \mathbf{Sen}(\Sigma)$ *and* $M' \in \mathbf{Mod}'(\Phi(\Sigma))$*:*

$$M' \models'_{\Phi(\Sigma)} \alpha_\Sigma(\varphi) \iff \beta_\Sigma(M') \models_\Sigma \varphi \qquad [\textit{Satisfaction condition}]$$

Institution comorphisms compose in the obvious, component-wise manner. The category of institutions with institution comorphisms is denoted by $co\mathcal{INS}$*.*

Example 4.2. Consider the translation of propositional logic into untyped first-order logic, mapping propositions to unary predicates plus a global constant a. An atomic sentence p is mapped to $p(a)$; this is inductively extended to all sentences. A first-order model is translated to a propositional model by inspecting whether the interpretation of a is contained in a given predicate. This can easily be organised as an institution comorphism.

Example 4.3. Many examples for comorphisms arise from *subinstitutions*, where we follow [Mes89] and define them as comorphisms $\rho = \langle\Phi, \alpha, \beta\rangle$ such that the signature translation Φ is an embedding of categories, all sentence translations α_Σ are injective and and all model translations β_Σ are isomorphisms. For example, propositional logic and many-sorted equational logic are both subinstitutions of many-sorted first-order logic (but not of untyped first-order logic).

Example 4.4. The encoding of $\mathbf{PFOL}^=$ into $\mathbf{FOL}^=$ that adds definedness predicates to signatures and restricts carrier sets of models to these predicates can easily be formalised as an institution comorphism [Mos02b].

The following properties of institution comorphisms ensure a good interaction with logical consequence:

Definition 4.5. *An institution comorphism is* model expansive*, if all the model translation functors are surjective on objects.*

An institution comorphism is (weakly) exact*, if the naturality squares for the model translation are (weak) pullbacks.*

For example, the comorphism from propositional logic to $\mathbf{UFOL}$ from Example 4.2 is model-expansive and weakly exact. Any subinstitution comorphism is both model-expansive and exact.

The notion of a *heterogeneous logical environment* (called *indexed coinstitutions* in [Mos02a], dualising the *indexed institutions* of [Dia02]) may be formalised as a collection of institutions linked by institution comorphisms.

Definition 4.6. *A* heterogeneous logical environment *$\mathcal{HLE}$ is a collection of institutions and institution comorphisms between them, that is, a diagram $\mathcal{HLE}\colon \mathcal{G} \to co\mathcal{INS}$*[4] *in the category $co\mathcal{INS}$.*

Working in a heterogeneous logical environment, we can enrich the collection of specification-building operations by translation along institution comorphisms, see [ST12]. Somewhat less naturally, we can also define hiding w.r.t. institution comorphisms, but the target signature has to be given explicitly then. Namely, given an institution comorphism $\rho\colon \mathcal{I} \to \mathcal{I}'$, we define:

***heterogeneous translation*:** For any $\mathcal{I}$-specification SP, $\rho(SP)$ is a specification with:

$$Sig[\rho(SP)] := \Phi(Sig[SP]) \qquad Mod[\rho(SP)] := \beta^{-1}_{Sig[SP]}(Mod[SP])$$

***heterogeneous hiding*:** For any $\mathcal{I}'$-specification SP' and signature Σ with $Sig[SP'] = \Phi(\Sigma)$, $SP'|^{\Sigma}_{\rho}$ is a specification with:

$$Sig[SP'|^{\Sigma}_{\rho}] := \Sigma \qquad Mod[SP'|^{\Sigma}_{\rho}] := \beta_{\Sigma}(Mod[SP'])$$

These new, inter-institutional specification-building operations may be arbitrarily mixed with other (intra-institutional) operations, yielding heterogeneous specifications. Parts of such specifications may be given in different institutions of the heterogeneous logical environment we work in. However, each such a specification as a whole eventually focuses on a particular institution in this environment, where its overall semantics (signature and the class of models) is given. In essence, viewed from a certain perspective, such *focused heterogeneous specifications* do not differ much from the structured specifications built within a single institution. For instance, the view of a software specification and development process as presented in [ST12] directly adapts to the use of such specifications without much (semantic) change. We will make this view more formal now.

Definition 4.7. *Consider institutions $\mathcal{I}$ and $\mathcal{I}'$ and signatures $\Sigma \in |\mathbf{Sign}|$ and $\Sigma' \in |\mathbf{Sign}'|$. A* heterogeneous signature comorphism *is a pair $\langle\rho, \sigma'\rangle\colon \Sigma \to \Sigma'$ that consists of an institution comorphism $\rho\colon \mathcal{I} \to \mathcal{I}'$ and a signature morphism $\sigma'\colon \Phi(\Sigma) \to \Sigma'$ in $\mathbf{Sign}'$. It induces the* heterogeneous reduct *$\_|_{\langle\rho,\sigma'\rangle}\colon \mathbf{Mod}'(\Sigma') \to \mathbf{Mod}(\Sigma)$ defined as the composition $\mathbf{Mod}'(\sigma')\,;\beta_{\Sigma}$, i.e., $M'|_{\langle\rho,\sigma'\rangle} = \beta_{\Sigma}(M'|_{\sigma'})$, for all $M' \in \mathbf{Mod}'(\Sigma')$. Heterogeneous sentence translations are defined similarly.*

Heterogeneous signature comorphisms compose as expected: $\langle\rho_1, \sigma_1\rangle; \langle\rho_2, \sigma_2\rangle = \langle\rho_1; \rho_2, \Phi_2(\sigma_1); \sigma_2\rangle$. For any heterogeneous logical environment $\mathcal{HLE}\colon \mathcal{G} \to co\mathcal{INS}$ this yields the heterogeneous category $\mathbf{Sign}^{\mathcal{HLE}}$ of signatures in institutions in $\mathcal{HLE}$ with heterogeneous comorphisms that involve institution comorphisms in $\mathcal{HLE}$. Then model functors extend to $\mathbf{Mod}^{\mathcal{HLE}}\colon (\mathbf{Sign}^{\mathcal{HLE}})^{op} \to \mathbf{Class}$ using the reducts defined above. Similarly, we obtain $\mathbf{Sen}^{\mathcal{HLE}}\colon \mathbf{Sign}^{\mathcal{HLE}} \to \mathbf{Set}$.

Proposition 4.8 ([Mos02a]). *The constructions in Def. 4.7 augmented with the family of satisfaction relations defined component-wise yield an institution*

$$\mathcal{I}^{\mathcal{HLE}} = \langle \mathbf{Sign}^{\mathcal{HLE}}, \mathbf{Sen}^{\mathcal{HLE}}, \mathbf{Mod}^{\mathcal{HLE}}, \models^{\mathcal{HLE}}\rangle.$$

[4] We introduce the following notation: the objects $n \in |\mathcal{G}|$ carry institutions $\mathcal{HLE}(n) = \mathcal{I}^n = \langle \mathbf{Sign}^n, \mathbf{Sen}^n, \mathbf{Mod}^n, \langle\models^n\rangle_{\Sigma\in|\mathbf{Sign}^n|}\rangle$ linked by institution comorphisms $\mathcal{HLE}(e) = \rho^e = \langle\Phi^e, \alpha^e, \beta^e\rangle\colon \mathcal{HLE}(n) \to \mathcal{HLE}(m)$ for each morphism $e\colon n \to m$ in $\mathcal{G}$.

The institution $\mathcal{I}^{\mathcal{HLE}}$ is known as the *Grothendieck institution* [Dia02, Mos02a].

For full formality, signatures in the heterogeneous categories of signatures defined above should really be written as pairs $\langle i, \Sigma \rangle$, with $i \in |\mathcal{G}|$.

The inter-institutional specification-building operations given above arise as hiding w.r.t. and translation along heterogeneous signature comorphisms within the Grothendieck institution $\mathcal{I}^{\mathcal{HLE}}$. Namely, given an institution comorphism $\rho\colon \mathcal{I} \to \mathcal{I}'$ and $\mathcal{I}$-specification SP, $\rho(SP)$ can be captured as $\langle \rho, id_{\Sigma'} \rangle(SP)$, where $\Sigma' = \Phi(Sig[SP])$. Similarly, for $\mathcal{I}'$-specification SP' and $\mathcal{I}$-signature Σ such that $\Phi(\Sigma) = Sig[SP']$, $SP'|^{\Sigma}_{\rho}$ becomes now $SP'|_{\langle \rho, id_{\Sigma} \rangle}$. Conversely, the "intra-institutional" specification-building operations introduced in Sect. 2 in the Grothendieck institution $\mathcal{I}^{\mathcal{HLE}}$ may be presented using the inter-institutional operations introduced above in combination with intra-institutional operations in component institutions. In particular, translation along $\langle \rho, \sigma \rangle$ is the composition of heterogeneous translation along ρ with (intra-institutional) translation along σ, and analogously for hiding w.r.t. $\langle \rho, \sigma \rangle$.

Consequently, the proof calculi introduced in Sect. 3 can be directly used for heterogeneous specifications by considering them for specifications built in the Grothendieck institution. The soundness (as given by Thm 3.2) carries over without change. The completeness theorems (Thm. 3.3 and 3.5) carry over as well, but the problem is that the assumptions under which they guarantee completeness of the calculi typically fail in Grothendieck institutions for many logical environments. Craig-Robinson interpolation was problematic even for truly homogeneous logical systems — it will fail in Grothendieck institutions for heterogeneous logical environments that contain even one institution where it fails. If all the institutions in the environment have interpolation, it still is likely to fail for the Grothendieck institution, even if [Dia04, Dia08] offer results which carry over Craig-Robinson interpolation from component institutions to the Grothendieck institution — under rather strong assumptions though.

The other key assumption in Thm. 3.3 and, especially, Thm 3.5, the weak amalgamation property, carries over from the heterogeneous logical environment to the Grothendieck institution rather naturally:

Proposition 4.9 ([Mos02a]). *Let $\mathcal{HLE}\colon \mathcal{G} \to co\mathcal{INS}$ be a heterogeneous logical environment consisting of comorphisms with cocontinuous signature translation functors. Its Grothendieck institution is (weakly) semi-exact if and only if*

- *$\mathcal{HLE}$ is (weakly) locally semi-exact, i.e., each institution in $\mathcal{HLE}$ is (weakly) semi-exact,*
- *$\mathcal{HLE}$ is (weakly) semi-exact, i.e., pushouts in $\mathcal{G}$ exist and are for each signature, (weak) pullbacks of model translation functors, and*
- *all institution comorphisms in $\mathcal{HLE}$ are (weakly) exact.*

Unfortunately, again, the conditions of Prop. 4.9 are not fulfilled in many typical logical environments. For example, neither the CASL institution nor the HETS logical environment are weakly semi-exact. Indeed, in HETS, there are many spans of institution comorphisms which can only be complemented to squares that do not even commute — see [Mos06] for an example.

5 Lax Heterogeneous Logical Environments

Sometimes it is useful to indicate that two institution comorphisms differ only in an inessential way. This in particular applies when the comorphisms arise as compositions of other comorphisms. We therefore introduce the notion of *modification*. Modifications are also useful for solving the problem mentioned at the end of the previous section (see Def. 5.6 below). Moreover, they naturally arise when representing comorphisms in some "universal" logic, along with the representation of source and target logic. Following [Mos02a], we dualise and strengthen the original notion from [Dia02] to *discrete* modifications (but we omit the qualifier "discrete" henceforth):

Definition 5.1. *Given two institution comorphisms* $\rho_1, \rho_2 \colon \mathcal{I} \longrightarrow \mathcal{J}$*, an* institution comorphism modification $\tau\colon \rho_1 \longrightarrow \rho_2$ *is a natural transformation* $\tau\colon \Phi_1 \longrightarrow \Phi_2$ *such that* $\alpha_1; (\mathbf{Sen}_{\mathcal{J}} \cdot \tau) = \alpha_2$ *and* $(\mathbf{Mod}_{\mathcal{J}} \cdot \tau); \beta_2 = \beta_1$.

Together with obvious identities and compositions, modifications can serve as 2-cells, leading to a 2-category which we also denote by $co\mathcal{INS}$.

Example 5.2. There are two ways to go from equational logic to first-order logic: one is the obvious subinstitution comorphism ρ_1 from Example 4.3, the other one is the composition ρ_2 of the obvious subinstitution comorphism ρ_2' from equational logic to partial first-order logic with the encoding ρ_2'' of partial first-order logic into first-order logic from Example 4.4. (Actually, the latter ends in $\mathbf{FOL}^{th}$.) These comorphisms are different: ρ_2 adds some (superfluous) coding of partiality. The comorphism modification $\tau\colon \rho_1 \longrightarrow \rho_2$ is just the pointwise inclusion of an algebraic signature viewed as first-order signature into the theory coding a partial variant of that signature.

$$\begin{tikzcd}
 & \mathbf{FOL}^{th} \\
\mathbf{EqL} \arrow[ur, "\rho_1"] \arrow[r, "\rho_2'"'] & \mathbf{PFOL} \arrow[u, "\rho_2''"']
\end{tikzcd}$$

with the 2-cell $\tau \colon \rho_1 \Rightarrow \rho_2$.

This motivates the following extension of the notion of heterogeneous logical environment:

Definition 5.3. *A* lax heterogeneous logical environment *is a 2-functor* $\mathcal{HLE}\colon \mathcal{G} \to co\mathcal{INS}$*, where both* $\mathcal{G}$ *and* $co\mathcal{INS}$ *are 2-categories.*[5]

We can then use the institution comorphism modifications to obtain a congruence on Grothendieck signature morphisms: the congruence is generated by

$$\langle d', \tau^u_\Sigma \colon \Phi^{d'}(\Sigma) \longrightarrow \Phi^d(\Sigma)\rangle \equiv \langle d, id \colon \Phi^d(\Sigma) \longrightarrow \Phi^d(\Sigma)\rangle : \langle i, \Sigma\rangle \to \langle j, \Phi^d(\Sigma)\rangle$$

for $\Sigma \in \mathbf{Sign}^i$, $d, d' \colon i \longrightarrow j \in \mathcal{G}$, and $u : d' \Rightarrow d \in \mathcal{G}$. This congruence has the following crucial property:

[5] Extending the notation introduced in footnote 4, a 2-cell $u\colon d \Rightarrow d'$ determines the corresponding modification $\mathcal{HLE}(u) = \tau^u \colon \rho^d \Rightarrow \rho^{d'}$.

Proposition 5.4. *Equivalent signature morphisms have identical sentence translation and model reduct functors.*

Let $q^{\mathcal{HLE}}\colon \mathbf{Sign}^{\mathcal{HLE}} \longrightarrow \mathbf{Sign}^{\mathcal{HLE}}/\!\equiv$ be the quotient functor induced by $\equiv$ (see [Mac98] for the definition of quotient category). Note that it is the identity on objects. We easily obtain that the model and sentence functors of the Grothendieck institution $\mathcal{I}^{\mathcal{HLE}}$ factor through the quotient category $\mathbf{Sign}^{\mathcal{HLE}}/\!\equiv$:

Corollary 5.5. *The components of the Grothendieck institution $\mathcal{I}^{\mathcal{HLE}}$ factor through the equivalence $\equiv$, yielding the* quotient Grothendieck institution*, which by abuse of notation we write as $\mathcal{I}^{\mathcal{HLE}}/\!\equiv\; = \langle \mathbf{Sign}^{\mathcal{HLE}}/\!\equiv, \mathbf{Sen}^{\mathcal{HLE}}, \mathbf{Mod}^{\mathcal{HLE}}, \models^{\mathcal{HLE}} \rangle$.*

When considering e.g. the comorphism going from partial first-order logic $\mathbf{PFOL}^{=}$ to first-order logic $\mathbf{FOL}^{=}$, and the composite comorphism going from $\mathbf{PFOL}^{=}$ to CASL and then to $\mathbf{FOL}^{=}$, we end up in different comorphisms, which are however related by a comorphism modification. The above identification process in the Grothendieck institution now tells us that it does not matter which way we choose.

Definition 5.6. *Given a lax heterogeneous logical environment $\mathcal{HLE}\colon \mathcal{G} \longrightarrow co\mathcal{INS}$, a square consisting of two lax triangles of index morphisms*

$$\begin{tikzcd}
 & i \arrow[dl, "d1"'] \arrow[dr, "d2"] \arrow[dd, "d"] & \\
j1 \arrow[dr, "e1"'] & \overset{u1}{\Longleftarrow} \qquad \overset{u2}{\Longrightarrow} & j2 \arrow[dl, "e2"] \\
 & k &
\end{tikzcd}$$

is called (weakly) amalgamable, if the following outer square is a (weak) pullback

$$\begin{tikzcd}
\mathbf{Mod}^{i}(\Sigma) & & \mathbf{Mod}^{j1}(\Phi^{d1}(\Sigma)) \arrow[ll, "\beta^{d1}_{\Sigma}"'] \\
 & \mathbf{Mod}^{k}(\Phi^{d}(\Sigma)) \arrow[ul, "\beta^{d}_{\Sigma}"'] & \mathbf{Mod}^{k}(\Phi^{e1}(\Phi^{d1}(\Sigma))) \arrow[l, "\mathbf{Mod}^{k}(\tau^{u1}_{\Sigma})"'] \arrow[u, "\beta^{e1}_{\Sigma}"'] \\
\mathbf{Mod}^{j2}(\Phi^{d2}(\Sigma)) \arrow[uu, "\beta^{d2}_{\Sigma}"] & \mathbf{Mod}^{k}(\Phi^{e2}(\Phi^{d2}(\Sigma))) \arrow[l, "\beta^{e2}_{\Sigma}"'] \arrow[u, "\mathbf{Mod}^{k}(\tau^{u2}_{\Sigma})"'] & \bullet \arrow[l, dotted] \arrow[u, dotted]
\end{tikzcd}$$

where the lower right square is a pullback.

$\mathcal{HLE}$ is called lax-quasi-exact*, if each pair of arrows $j1 \xleftarrow{d_1} i \xrightarrow{d_2} j2$ in $\mathcal{G}$ may be completed to a weakly amalgamable square of lax triangles*

$$\begin{tikzcd}
 & i \arrow[dl, "d_1"'] \arrow[dr, "d_2"] \arrow[dd] & \\
j1 \arrow[dr] & \Longleftarrow \qquad \Longrightarrow & j2 \arrow[dl] \\
 & k &
\end{tikzcd}$$

□

6 A Proof Calculus for Heterogeneous Specifications

We obtain a proof calculus for entailment between heterogeneous specifications and sentences by extending the proof calculus in for structured specifications in Sect. 3, Fig. 1, with the following rules:

$$(\mathit{het\text{-}trans})\ \frac{SP \vdash \varphi}{\rho(SP) \vdash \alpha(\varphi)} \qquad (\mathit{het\text{-}derive})\ \frac{SP \vdash \alpha(\varphi)}{SP|_{\rho}^{\Sigma} \vdash \varphi}$$

$$(\mathit{borrowing})\ \frac{\rho(SP) \vdash \alpha(\varphi)}{SP \vdash \varphi} \qquad \text{if } \rho \text{ is model-expansive}$$

$$(\mathit{Het\text{-}snf})\ \frac{SP' \vdash \sigma(\alpha(\varphi))}{SP \vdash \varphi} \qquad \text{if } \mathit{hsnf}(SP) = (SP'|_{\sigma})|_{\rho}^{\Sigma}$$

(where hsnf is snf for the Grothendieck institution) and the calculus for refinements between heterogeneous specifications in Fig. 2 is extended as follows:

$$(\mathit{Het\text{-}Trans})\ \frac{SP \rightsquigarrow SP'|_{\rho}^{\Sigma}}{\rho(SP) \rightsquigarrow SP'} \qquad (\mathit{Het\text{-}Derive})\ \frac{SP \rightsquigarrow SP''}{SP|_{\rho}^{\Sigma} \rightsquigarrow SP'} \qquad \text{if } \rho\colon SP' \longrightarrow SP'' \text{ is a conservative extension}$$

Conservativity of $\rho = (\Phi, \alpha, \beta)\colon SP' \longrightarrow SP''$ means that for each model $M' \in \mathit{Mod}(SP')$, there is a model $M'' \in \mathit{Mod}(SP'')$ with $\beta(M'') = M'$.

Theorem 6.1. *For a lax heterogeneous logical environment $\mathcal{HLE}\colon \mathcal{G} \longrightarrow co\mathcal{INS}$ (with some of the institutions also being logics), the proof calculi for heterogeneous specifications are sound for $\mathcal{I}^{\mathcal{HLE}}/\!\equiv$. If*

1. *$\mathcal{HLE}$ is lax-quasi-exact,*
2. *all institution comorphisms in $\mathcal{HLE}$ are weakly exact,*
3. *there is a set $\mathcal{L}$ of institutions in $\mathcal{HLE}$ that come as* complete *logics,*
4. *all institutions in $\mathcal{L}$ are quasi-semi-exact,*
5. *from each institution in $\mathcal{HLE}$, there is some model-expansive comorphism in $\mathcal{HLE}$ going into some logic in $\mathcal{L}$,*

then the proof calculus for entailments between heterogeneous specifications and sentences is complete over $\mathcal{I}^{\mathcal{HLE}}/\!\equiv$. If, moreover, the rule system is extended with a (sound and complete) oracle for conservative extension, then the proof calculus for refinements between heterogeneous specifications is also complete.

The oracle for conservative extensions cannot be resigned (not even in the homogeneous case, see [MAH06]). One crucial achievement here is that, in contrast to Prop. 4.9, we need *neither* cocontinuity *nor* exactness of the comorphisms. Moreover, we need quasi-exactness only for some of the logics; this allows us to include logics which are not quasi-exact, such as CASL. Our proof calculus is related to, but different from and conceptually simpler than the one for heterogeneous development graphs in [Mos02a, Mos05]: it is defined along the structure of heterogeneous structured specifications. A similar proof calculus has been implemented in the heterogeneous tool set HETS [MML07].

7 Final Remarks

Building on the standard approach to structured specifications in an arbitrary institution, we extend it to deal with heterogeneous specifications constructed in a heterogeneous logical environment, formalised as a diagram of institutions with institution comorphisms. The focus in this paper is on the proof systems for consequences of so obtained structured heterogeneous specifications and for refinements between such specifications. We put forward a modification of the standard proof systems of homogeneous structured specifications that strike a proper balance between compositionality and the need for completeness. This system is then extended to heterogeneous specifications. The key result is the (soundness and) completeness of the system under assumptions considerably milder than those that guarantee completeness of purely compositional calculi.

In order to make the work in this paper practically useful for formal software development with heterogeneous logics, the implementation of heterogeneous specifications and proofs in HETS [MML07, Mos05] needs to be generalised to the lax case (see Sect. 5). It also would be important to generalise the present work to further practically relevant notions of maps between institutions, studied in [GR02]. Future work will apply the presented approach to the heterogeneous logical environment arising from UML (see [CKTW08] for initial promising steps in this direction).

References

[Bor00] Borzyszkowski, T.: Generalized interpolation in CASL. Information Processing Letters 79, 19–24 (2000)

[Bor02] Borzyszkowski, T.: Logical systems for structured specifications. Theoretical Computer Science 286, 197–245 (2002)

[CKTW08] Cengarle, M.V., Knapp, A., Tarlecki, A., Wirsing, M.: A heterogeneous approach to UML semantics. In: Degano, P., De Nicola, R., Meseguer, J. (eds.) Concurrency, Graphs and Models. LNCS, vol. 5065, pp. 383–402. Springer, Heidelberg (2008)

[CM97] Cerioli, M., Meseguer, J.: I borrow your logic? (transporting logical structures along maps). Theor. Comput. Sci. 173(2), 311–347 (1997)

[CMM13] Codescu, M., Mossakowski, T., Maeder, C.: Checking conservativity with HETS. In: Heckel, R., Milius, S. (eds.) CALCO 2013. LNCS, vol. 8089, pp. 315–321. Springer, Heidelberg (2013)

[CoF04] Mosses, P.D. (ed.): CASL Reference Manual. LNCS, vol. 2960. Springer, Heidelberg (2004), `http://www.cofi.info`

[DF02] Diaconescu, R., Futatsugi, K.: Logical foundations of CafeOBJ. Theoretical Computer Science 285, 289–318 (2002)

[Dia02] Diaconescu, R.: Grothendieck institutions. J. Applied Categorical Structures 10, 383–402 (2002)

[Dia04] Diaconescu, R.: Interpolation in Grothendieck Institutions. Theoretical Computer Science 311(1-3), 439–461 (2004)

[Dia08] Diaconescu, R.: Institution-independent Model Theory. Birkhäuser (2008)

[DM00] Dimitrakos, T., Maibaum, T.: On a generalized modularization theorem. Information Processing Letters 74, 65–71 (2000)

[Fin79] Fine, K.: Failures of the Interpolation Lemma in Quantified Modal Logic. J. of Symbolic Logic 44(2), 201–206 (1979)

[FS88] Fiadeiro, J., Sernadas, A.: Structuring theories on consequence. In: Sannella, D., Tarlecki, A. (eds.) Abstract Data Types 1987. LNCS, vol. 332, pp. 44–72. Springer, Heidelberg (1988)

[GB92] Goguen, J.A., Burstall, R.M.: Institutions: Abstract model theory for specification and programming. Journal of the ACM 39(1), 95–146 (1992)

[GR02] Goguen, J.A., Rosu, G.: Institution morphisms. Formal Aspects of Computing 13(3-5), 274–307 (2002)

[HST94] Harper, R., Sannella, D., Tarlecki, A.: Structured presentations and logic representations. Annals of Pure and Applied Logic 67, 113–160 (1994)

[Mac98] Mac Lane, S.: Categories for the Working Mathematician, 2nd edn. Springer (1998)

[MAH06] Mossakowski, T., Autexier, S., Hutter, D.: Development graphs – proof management for structured specifications. J. of Logic and Algebraic Programming 67(1-2), 114–145 (2006)

[Mes89] Meseguer, J.: General logics. In: Logic Colloquium 1987, pp. 275–329. North Holland (1989)

[MML07] Mossakowski, T., Maeder, C., Lüttich, K.: The Heterogeneous Tool Set, HETS. In: Grumberg, O., Huth, M. (eds.) TACAS 2007. LNCS, vol. 4424, pp. 519–522. Springer, Heidelberg (2007)

[Mos02a] Mossakowski, T.: Comorphism-based Grothendieck logics. In: Diks, K., Rytter, W. (eds.) MFCS 2002. LNCS, vol. 2420, pp. 593–604. Springer, Heidelberg (2002)

[Mos02b] Mossakowski, T.: Relating CASL with other specification languages: the institution level. Theoretical Computer Science 286, 367–475 (2002)

[Mos05] Mossakowski, T.: Heterogeneous Specification and the Heterogeneous Tool Set. Habilitation thesis, Universität Bremen (2005)

[Mos06] Mossakowski, T.: Institutional 2-cells and grothendieck institutions. In: Futatsugi, K., Jouannaud, J.-P., Meseguer, J. (eds.) Algebra, Meaning, and Computation. LNCS, vol. 4060, pp. 124–149. Springer, Heidelberg (2006)

[MT09] Mossakowski, T., Tarlecki, A.: Heterogeneous logical environments for distributed specifications. In: Corradini, A., Montanari, U. (eds.) WADT 2008. LNCS, vol. 5486, pp. 266–289. Springer, Heidelberg (2009)

[SB83] Sannella, D., Burstall, R.: Structured theories in LCF. In: Ausiello, G., Protasi, M. (eds.) CAAP 1983. LNCS, vol. 159, pp. 377–391. Springer, Heidelberg (1983)

[Sho67] Shoenfield, J.: Mathematical Logic. Addison-Wesley (1967)

[ST88a] Sannella, D., Tarlecki, A.: Specifications in an arbitrary institution. Information and Computation 76, 165–210 (1988)

[ST88b] Sannella, D., Tarlecki, A.: Toward formal development of programs from algebraic specifications: Implementations revisited. Acta Informatica 25, 233–281 (1988)

[ST12] Sannella, D., Tarlecki, A.: Foundations of Algebraic Specification and Formal Software Development. Monographs in Theoretical Computer Science. An EATCS Series. Springer (2012)

[ST13] Sannella, D., Tarlecki, A.: Property-oriented semantics of structured specifications. In: Mathematical Structures in Computer Science (2013)

[Tar00] Tarlecki, A.: Towards heterogeneous specifications. In: Gabbay, D., de Rijke, M. (eds.) Frontiers of Combining Systems 2, Studies in Logic and Computation, pp. 337–360. Research Studies Press (2000)