Heterogeneous Theories and the Heterogeneous Tool Set

Till Mossakowski

BISS, Dept. of Computer Science, University of Bremen

1 Introduction

Heterogeneous multi-logic theories arise in different contexts: they are needed for the specification of large software systems, as well as for mediating between different ontologies. This is because large theories typically involve different aspects that are best specified in different logics (like equational logics, description logics, first-order logics, higher-order logics, modal logics), but also because different formalisms are in practical use (like RDF, OWL, EML). Using heterogeneous theories, different formalims being developed at different sites can be related, i.e. there is a formal interoperability among languages and tools. In many cases, specialized languages and tools have their strengths in particular aspects. Using heterogeneous theories, these strengths can be combined with comparably small effort. By contrast, a true combination of all the involved logics into a single logic would be too complex (or even inconsistent) in many cases.

We propose to use *institutions* as a formalization of the notion of logical system. Institutions can be related by so-called institution morphsims and comorphisms. Any graph of institutions and (co)morphisms can be flattened to a so-called *Grothendieck institution*, which is kind of disjoint union of all the logics, enriched with connections via the (co)morphisms.

This semantic basis for heterogeneous theories is complemented by the heterogeneous tool set, which provides tool support. Based on an object-oriented interface for institutions (using type classes in Haskell), it implements the Grothendieck institution and provides a heterogeneous parser, static analysis and proof support for heterogeneous theories. This is based on parsers, static analysers and proof support for the individual institutions, and on a heterogeneous proof calculus for theories in the Grothendieck institution.

2 Institutions

Institutions are the central abstract notion that is the basis for a theory of structured specification and proving independent of the underlying logical system. Naturally, this notion is also the basis for heterogeneous theories. While institutions capture model theory, entailment systems are a related abstract notion capturing proof theory. Finally, an institution equipped with an entailment is called a *logic*.

Dagstuhl Seminar Proceedings 04391
Semantic Interoperability and Integration
http://drops.dagstuhl.de/opus/volltexte/2005/43

Many different logics, including first-order [11], higher-order [4], polymorphic [18, 20], modal [5, 23, 6], temporal [8], process [8], behavioural [3], and object-oriented [21, 9, 13, 22, 1] logics have been shown to be institutions. Recently, there has been interest in institutions in connection with XML and databases [1].

A specification formalism is usually based on some notion of signature, model, sentence and satisfaction. These are the usual ingredients of Barwise's abstract model theory [2]. Contrary to Barwise's notions, *institutions* of Goguen and Burstall [11] do not assume that signatures are algebraic signatures and thus cover a much larger variety of logics. Indeed, the theory of institutions assumes nothing about signatures except that they form a class and that there are *signature morphisms*, which can be composed in some way. This amounts to stating that signatures form a *category*.

There is also nothing special assumed about the form of the *sentences* and *models*. Given a signature Σ, the Σ-sentences form just a set, while the Σ-models form a category (taking into account that there may be *model morphisms*).

Signature morphisms lead to *translations* of sentences and of models (thus, the assignments of sentences and of models to signatures are functors). There is a contravariance between the sentence and the model translation: sentences are translated *along* signature morphisms, while models are translated *against* signature morphisms.

Informally, this can be motivated as follows. Forget for a moment the above generality and think of signatures as of sets of certain symbols. Think of sentences over a signature Σ as derivation trees over some grammar, decorated at the nodes with the symbols from Σ. Then sentence translation along a signature morphism $\sigma\colon \Sigma \longrightarrow \Sigma'$ keeps the structure of the derivation tree, but replaces the symbols decorating the nodes, using σ. This explains why sentences are translated *along* signature morphisms.

Concerning models over a signature: they have to interpret the symbols from the signature somehow. Thus, a Σ-model can be seen as a map M going from the symbols of Σ to some semantical domain. Now given a Σ'-model M' and a signature morphism $\sigma\colon \Sigma \longrightarrow \Sigma'$, by composing the interpretation map M' with σ we get a new interpretation map, let us call it $M'|_\sigma$, which is a Σ-model! ($M'|_\sigma$ is also called the *σ-reduct* of M'.) This explains why models are translated *against* signature morphisms.

Of course, these explanations just have motivating purpose: there can be institutions with a completely different view of signatures, models and sentences. However, they shed some light on how many typical institutions work.[1]

Finally, institutions have a *satisfaction relation* between models and sentences, which has to be invariant under the simultaneous translation of sentences and models w.r.t. a given signature morphism.

This leads to the following formal definition [11].

Definition 1. An *institution* $I = (\mathbf{Sign}^I, \mathbf{Sen}^I, \mathbf{Mod}^I, \models^I)$ consists of

- a category $\mathbf{Sign}^I$ of *signatures*,

[1] Indeed, the above explanation has been formalized as so-called *parchments* [16].

- a functor $\mathbf{Sen}^I : \mathbf{Sign}^I \longrightarrow \mathbf{Set}$ giving, for each signature Σ, the set of *sentences* $\mathbf{Sen}^I(\Sigma)$, and for each signature morphism $\sigma : \Sigma \longrightarrow \Sigma'$, the *sentence translation map* $\mathbf{Sen}^I(\sigma) : \mathbf{Sen}^I(\Sigma) \longrightarrow \mathbf{Sen}^I(\Sigma')$, where often $\mathbf{Sen}^I(\sigma)(\varphi)$ is written as $\sigma(\varphi)$,
- a functor $\mathbf{Mod}^I : (\mathbf{Sign}^I)^{op} \longrightarrow \mathcal{CAT}$[2]giving, for each signature Σ, the category of *models* $\mathbf{Mod}^I(\Sigma)$, and for each signature morphism $\sigma : \Sigma \longrightarrow \Sigma'$, the *reduct functor* $\mathbf{Mod}^I(\sigma) : \mathbf{Mod}^I(\Sigma') \longrightarrow \mathbf{Mod}^I(\Sigma)$, where often $\mathbf{Mod}^I(\sigma)(M')$ is written as $M'|_\sigma$ (the σ-reduct of M'),
- a satisfaction relation $\models^I_\Sigma \subseteq |\mathbf{Mod}^I(\Sigma)| \times \mathbf{Sen}^I(\Sigma)$ for each $\Sigma \in \mathbf{Sign}^I$,

such that for each $\sigma : \Sigma \longrightarrow \Sigma'$ in $\mathbf{Sign}^I$ the following *satisfaction condition* holds:

$$M' \models^I_{\Sigma'} \sigma(\varphi) \Leftrightarrow M'|_\sigma \models^I_\Sigma \varphi$$

for each $M' \in \mathbf{Mod}^I(\Sigma')$ and $\varphi \in \mathbf{Sen}^I(\Sigma)$. □

Example 2. The institution $Eq^=$ of equational logic. Signatures are many-sorted algebraic signatures consisting of a set of sorts and a set of function symbols (where each function symbol has a string of argument sorts and a result sort). Signature morphisms map sorts and function symbols in a compatible way. Models are just many-sorted algebras, i.e. each sort is interpreted as a carrier set, and each function symbol is interpreted as a function between the carrier sets specified by the argument and result sorts. Reducts are constructed as sketched above. Sentences are equations between many-sorted terms, and sentence translation means replacement of the translated symbols. Finally, satisfaction is the usual satisfaction of an equation in an algebra. □

Example 3. The institution $FOL^=$ of many-sorted first-order logic with equality. Signatures are many-sorted first-order signatures, i.e. many-sorted algebraic signatures enriched with predicate symbols. Models are many-sorted first-order structures. Sentences are first-order formulas, and again sentence translation means replacement of the translated symbols. Satisfaction is the usual satisfaction of a first-order sentence in a first-order structure. □

Example 4. The institution $PFOL^=$ 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 containing existential equations, strong equations, definedness statements and predicate applications as atomic formulas. Satisfaction is defined using total valuations of variables, while valuation of terms is partial due to the existence of partial functions. An existential equation holds if both sides are defined and equal, whereas a strong equation also holds if both sides are undefined. A definedness statement holds if the term is defined. A predicate application holds if the terms contained in it are defined, and the corresponding tuple of values is in the interpretation of the predicate. This is extended to first-order formulas as usual. □

[2] $\mathcal{CAT}$ be the (quasi-)category of categories and functors.

Many familiar basic concepts from logic can be defined over any institution:

Definition 5. Given a set of Σ-sentences Γ and a Σ-sentence φ, then φ is a *semantic consequence* of Γ, written $\Gamma \models_\Sigma \varphi$, iff for all Σ-models M, we have $M \models_\Sigma \Gamma$ implies $M \models_\Sigma \varphi$, where $M \models_\Sigma \Gamma$ means $M \models_\Sigma \psi$ for each $\psi \in \Gamma$. Two sentences are *semantically equivalent*, written $\varphi_1 \models\mid \varphi_2$, if they are satisfied by the same models. Two models are *elementary equivalent*, written $M_1 \equiv M_2$, if they satisfy the same sentences. An institution is *compact* iff $\Gamma \models_\Sigma \varphi$ implies $\Gamma' \models_\Sigma \varphi$ for some finite subset Γ' of Γ. A *theory* is a pair (Σ, Γ) where Γ is a set of Σ-sentences, and is *consistent* iff it has at least one model. A theory morphism $(\Sigma, \Gamma) \longrightarrow (\Sigma', \Gamma')$ is a signature morphism $\sigma\colon\ \Sigma \longrightarrow \Sigma$ such that $\Gamma' \models_{\Sigma'} \sigma(\Gamma)$. A theory morphism $\sigma\colon\ (\Sigma, \Gamma) \longrightarrow (\Sigma', \Gamma')$ is *conservative* iff each (Σ, Γ)-model has at least one expansion (along σ) to a (Σ', Γ')-model.[3] □

A *logic* is an institution equipped with an entailment system $(\vdash_\Sigma)_{\Sigma \in |\mathbf{Sign}|}$, see [14] for details. Logics are required to be sound: if $\Gamma \vdash_\Sigma \varphi$ then $\Gamma \models_\Sigma \varphi$. The converse implication is called completeness.

3 Institution Comorphisms

We now come to the task of relating different institutions. Institution *comorphisms* [10] relate two given institutions. A typical situation is that an institution comorphism expresses the fact that an institution is embedded or encoded into another one.

An *institution comorphism* from an institution I to an institution J consists of the following components:

- a translation Φ of I-signatures to J-signatures. Given an I-signature Σ, the task is to find a J-encoding $\Phi(\Sigma)$ of Σ in some way. In particular, the model category of $\Phi(\Sigma)$ should approximate the model category of Σ somehow.
- a translation α of I-sentences to J-sentences. The reason why the sentence translation goes *along* with the signature translation is similar to the reason why the sentence translation *within* an institution goes along with the signature morphism. Namely, if a signature Σ in I is encoded by the presentation $\Phi(\Sigma)$ in J, it is expected that each symbol in Σ is translated to some corresponding symbol in $\Phi(\Sigma)$. Now if we assume that a Σ-sentence φ is a derivation tree decorated with some symbols from Σ, the translation $\alpha_\Sigma(\varphi)$ just keeps the structure of the tree and translates the symbols according to the correspondence of symbols in Σ and $\Phi(\Sigma)$.
- a translation β of J-models to I-models, giving the above mentioned relation between Σ-models in I and $\Phi(\Sigma)$-models in J. Here, we again have the

[3] Besides this model-theoretic notion of conservativeness, there also is a weaker consequence-theoretic notion: $\Gamma' \models \sigma(\varphi)$ implies $\Gamma \models \varphi$, and a proof-theoretic notion coinciding with the consequence-theoretic one for complete logics: $\Gamma' \vdash \sigma(\varphi)$ implies $\Gamma \vdash \varphi$. We here prefer the model-theoretic notion, since this agrees with a model semantics of theories.

contravariance of the model translation, as in the definition of institution. Often it happens that there is also a model translation γ in the opposite direction. However, while β is formalized as a natural transformation, γ is not always natural (see [12] for a counterexample). Naturality of β is essential for heterogeneous theories, see [15].

We impose a satisfaction condition on comporphisms as well: we require that a translated model satisfies a sentence iff the original model satisfies the translated sentence.

More formally, given institutions I and J, an *institution comorphism* $\rho = (\Phi, \alpha, \beta)\colon I \longrightarrow J$ consists of

- a functor $\Phi\colon \mathbf{Sign}^I \longrightarrow \mathbf{Sign}^J$,
- a natural transformation $\alpha\colon \mathbf{Sen}^I \longrightarrow \mathbf{Sen}^J \circ \Phi$,
- a natural transformation $\beta\colon \mathbf{Mod}^J \circ \Phi^{op} \longrightarrow \mathbf{Mod}^I$

such that the following *satisfaction condition* is satisfied for all $\Sigma \in \mathbf{Sign}^I$, $M' \in \mathbf{Mod}^J(\Phi(\Sigma))$ and $\varphi \in \mathbf{Sen}^I(\Sigma)$:

$$M' \models^J_{\Phi(\Sigma)} \alpha_\Sigma(\varphi) \Leftrightarrow \beta_\Sigma(M') \models^I_\Sigma \varphi.$$

Together with obvious compositions and identities, this gives us the category **CoIns** of institution and institution comorpisms.

In more detail, this means that each signature $\Sigma \in \mathbf{Sign}^I$ is translated to a signature $\Phi(\Sigma) \in \mathbf{Sign}^J$, and each signature morphism $\sigma\colon \Sigma \longrightarrow \Sigma' \in \mathbf{Sign}^I$ is translated to a signature morphism $\Phi(\sigma)\colon \Phi(\Sigma) \longrightarrow \Phi(\Sigma') \in \mathbf{Sign}^J$. Moreover, for each signature $\Sigma \in \mathbf{Sign}^I$, we have a sentence translation map $\alpha_\Sigma\colon \mathbf{Sen}^I(\Sigma) \longrightarrow \mathbf{Sen}^J(\Phi(\Sigma))$ and a model translation functor $\beta_\Sigma : \mathbf{Mod}^J(\Phi(\Sigma)) \longrightarrow \mathbf{Mod}^I(\Sigma)$. Naturality of α and β means that for any signature morphism $\sigma\colon \Sigma \longrightarrow \Sigma' \in \mathbf{Sign}^I$,

$$\begin{array}{ccc}
\mathbf{Sen}^I(\Sigma) & \xrightarrow{\alpha_\Sigma} & \mathbf{Sen}^J(\Phi(\Sigma)) \\
\downarrow{\scriptstyle \mathbf{Sen}^I(\sigma)} & & \downarrow{\scriptstyle \mathbf{Sen}^J(\Phi(\sigma))} \\
\mathbf{Sen}^I(\Sigma') & \xrightarrow{\alpha_{\Sigma'}} & \mathbf{Sen}^J(\Phi(\Sigma'))
\end{array}$$

and

$$\begin{array}{ccc}
\mathbf{Mod}^I(\Sigma) & \xleftarrow{\beta_\Sigma} & \mathbf{Mod}^J(\Phi(\Sigma)) \\
\uparrow{\scriptstyle \mathbf{Mod}^I(\sigma)} & & \uparrow{\scriptstyle \mathbf{Mod}^J(\Phi(\sigma))} \\
\mathbf{Mod}^I(\Sigma') & \xleftarrow{\beta_{\Sigma'}} & \mathbf{Mod}^J(\Phi(\Sigma'))
\end{array}$$

commute.

Example 6. There is an institution comorphism going from equational logic to first-order logic with equality. An algebraic signature is translated to a first-order signature by just taking the set of predicate symbols to be empty. Sentence translation is just inclusion of equations into first-order sentences. A first-order model with empty set of predicates is translated by just considering it as an algebra. □

The notion of institution comorphism can be varied in several ways by changing the directions of the arrows or even, in the case of semi-morphisms, omitting the arrows [10, 24]. For simplicity, we will stick to comorphisms as introduced above.

4 Grothendieck Institutions

Heterogeneous theories can be viewed as theories in a Grothendieck construction. Diaconescu's Grothendieck institution construction [7] basically flattens a diagram of institution and morphisms. We here recall the Grothendieck institution for the comorphism-based case [17]:

Definition 7. An *indexed coinstitution* is a functor $\mathcal{I}: Ind^{op} \longrightarrow \mathbf{CoIns}$ into the category **CoIns** of institutions and institution comorphisms[4].

Conceptually, an indexed coinstitution is just a graph of institutions and institution comorphisms (together with some way to compose the comorphisms).

The basic idea of the Grothendieck institution is that all signatures of all institutions are put side by side, and a signature morphism in this large realm of signatures consists of an intra-institution signature morphism plus an inter-institution translation (along some institution comorphism). The other components are then defined in a straightforward way.

Definition 8. Given an *indexed coinstitution* $\mathcal{I}: Ind^{op} \longrightarrow \mathbf{CoIns}$, define the *Grothendieck institution* $\mathcal{I}^{\#}$ as follows:

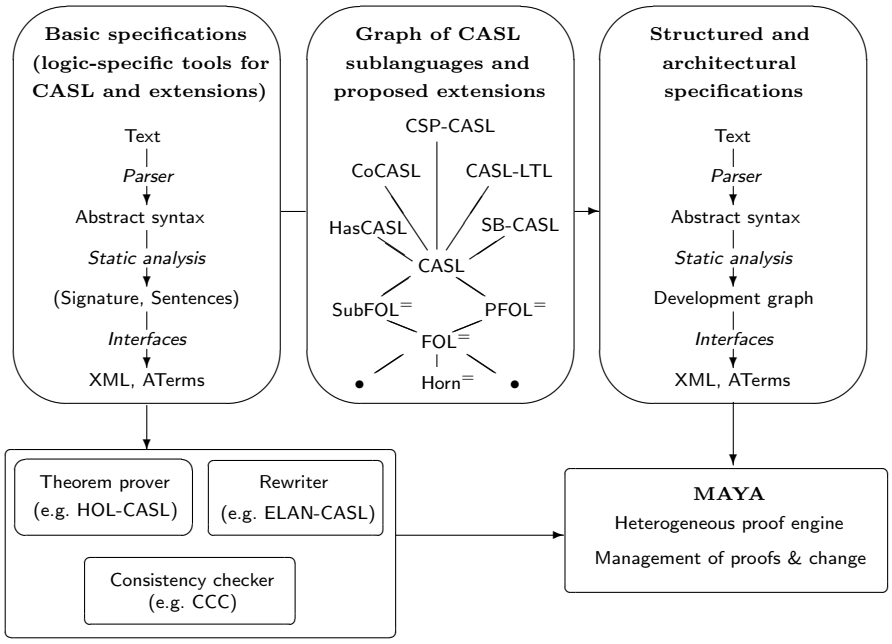
- signatures in $\mathcal{I}^{\#}$ are pairs (Σ, i), where $i \in |Ind|$ and Σ a signature in the institution $\mathcal{I}(i)$,
- signature morphisms $(\sigma, e): (\Sigma_1, i) \longrightarrow (\Sigma_2, j)$ consist of a morphism $e: j \longrightarrow i \in Ind$ and a signature morphism $\sigma: \Phi^{\mathcal{I}(e)}(\Sigma_1) \longrightarrow \Sigma_2$ (here, $\mathcal{I}(e): \mathcal{I}(i) \longrightarrow \mathcal{I}(j)$ is the institution comorphism corresponding to the arrow $e: j \longrightarrow i$ in the indexed coinstitution, and $\Phi^{\mathcal{I}(e)}$ is its signature translation component),
- the (Σ, i)-sentences are the Σ-sentences in $\mathcal{I}(i)$, and sentence translation along (σ, e) is the composition of sentence translation along σ with sentence translation along $\mathcal{I}(e)$,

[4] Indeed, the name is justified by the fact that the category of institutions and institution comorphisms is isomorphic to the category of coinstitutions and coinstitution morphisms. A coinstitution is an institution with model translations covariant to signature morphisms, while sentence translations are contravariant.

– the (Σ, i)-models are the Σ-models in $\mathcal{I}(i)$, and model reduction along (σ, e) is the composition of model translation along $\mathcal{I}(e)$ with model reduction along σ, and
– satisfaction w.r.t. (Σ, i) is satisfaction w.r.t. Σ in $\mathcal{I}(i)$. □

The importance of the definition of Grothendieck institution lies in the fact that any of the institution independent logical notions introduced in Def. 5 directly carries over to Grothendieck institutions. Two such notions are e.g. consistency of theories and conservativity of theory extensions (morphisms); these notions play an important role for formal ontologies.

5 The Heterogeneous Tool set (HETS)



The Heterogeneous Tool Set HETS is a tool implementing the theory developed so far. Its architecture is depicted above. HETS has an abstract interface corresponding to concept of institution (or more precisely, entailment system — since model theory is not directly implementable) in Haskell.

HETS implements this by providing a type class `Logic`. `Logic` is a multiparameter type classes with functional dependencies [19]. Such a type class can be thought of as a formal parameter signature. `Logic` contains types for signatures, signature morphisms, sentences, abstract syntax of basic specifications etc., and functions for parsing, printing, static analysis, and proving. Based on this abstract interface, we have implemented *heterogeneous* tools for parsing and static analysis of heterogeneous CASL. The static semantic analysis yields a so called

development graph (a kind of module graph) over the Grothendieck institution, and we are currently implementing the corresponding proof calculus.

Technically, heterogeneity is realized as follows. On top of the type class `Logic`, an existential datatype is constructed. Usually, existential types are used to realize e.g. heterogeneous lists, where each element may have a different type. We use lists of (components of) institutions and comorphisms instead. This leads to an implementation of the Grothendieck institution over an indexed coinstitution.

We have instantiated this general framework with institution-specific analysis tools for CASL, HASCASL, Haskell, CSP-CASL and MODALCASL. We are currently adding support for OWL-DL. Future work will interface existing theorem proving tools with specific institutions in HETS. We already have implemented an experimental interface to the theorem prover Isabelle.

The Heterogeneous Tool Set is available at `www.tzi.de/cofi/hets`.

Acknowledgements

Thanks to Andrzej Tarlecki, Joseph Goguen, Grigore Rosu, Serge Autexier and Dieter Hutter for useful cooperation and discussions, and to Răzvan Diaconescu for inventing Grothendieck institutions.

This work has been supported by the *Deutsche Forschungsgemeinschaft* under Grant KR 1191/5-2.

References

1. S. Alagi. Institutions: integrating objects, XML and databases. *Information and Software Technology*, 44:207–216, 2002.
2. J. Barwise. Axioms for abstract model theory. *Annals of Mathematical Logic*, 7:221–265, 1974.
3. M. Bidoit and R. Hennicker. Using an institution encoding for proving consequences of structured COL-specifications. Talk at the WADT 2002, Frauenchiemsee.
4. T. Borzyszkowski. Moving specification structures between logical systems. In J. L. Fiadeiro, editor, *Recent Trends in Algebraic Development Techniques, 13th International Workshop, WADT'98, Lisbon, Portugal, April 1998, Selected Papers*, volume 1589 of *Lecture Notes in Computer Science*, pages 16–30. Springer, 1999.
5. C. Cirstea. Institutionalising many-sorted coalgebraic modal logic. In *CMCS 2002*, Electronic Notes in Theoretical Computer Science. Elsevier Science, 2002.
6. R. Diaconescu. *Institution-independent model theory*. Manuscript, University of Bucharest.
7. R. Diaconescu. Grothendieck institutions. *Applied categorical structures*, 10:383–402, 2002.
8. J. L. Fiadeiro and J. F. Costa. Mirror, mirror in my hand: A duality between specifications and models of process behaviour. *Mathematical Structures in Computer Science*, 6(4):353–373, 1996.
9. J. Goguen and R. Diaconescu. An Oxford survey of order sorted algebra. *Mathematical Structures in Computer Science*, 4(3):363–392, Sept. 1994.

10. J. Goguen and G. Rosu. Institution morphisms. *Formal aspects of computing*, 13:274–307, 2002.
11. J. A. Goguen and R. M. Burstall. Institutions: Abstract model theory for specification and programming. *Journal of the Association for Computing Machinery*, 39:95–146, 1992. Predecessor in: LNCS 164, 221–256, 1984.
12. H.-J. Kreowski and T. Mossakowski. Equivalence and difference of institutions: Simulating Horn clause logic with based algebras. *Mathematical Structures in Computer Science*, 5:189–215, 1995.
13. A. Lopes and J. L. Fiadeiro. Preservation and reflection in specification. In *Algebraic Methodology and Software Technology*, pages 380–394, 1997.
14. J. Meseguer. General logics. In *Logic Colloquium 87*, pages 275–329. North Holland, 1989.
15. T. Mossakowski. The heterogeneous tool set. Available at www.tzi.de/cofi/hets, University of Bremen.
16. T. Mossakowski. Using limits of parchments to systematically construct institutions of partial algebras. In M. Haveraaen, O. Owe, and O.-J. Dahl, editors, *Recent Trends in Data Type Specifications. 11th Workshop on Specification of Abstract Data Types*, volume 1130 of *Lecture Notes in Computer Science*, pages 379–393. Springer Verlag, 1996.
17. T. Mossakowski. Comorphism-based Grothendieck logics. In K. Diks and W. Rytter, editors, *Mathematical foundations of computer science*, volume 2420 of *LNCS*, pages 593–604. Springer, 2002.
18. M. Nielsen and U. Pletat. Polymorphism in an institutional framework, 1986. Technical University of Denmark.
19. S. Peyton Jones, M. Jones, and E. Meijer. Type classes: exploring the design space. In *Haskell Workshop*. 1997.
20. L. Schröder, T. Mossakowski, and C. Lüth. Type class polymorphism in an institutional framework. In J. Fiadeiro, editor, *Recent Trends in Algebraic Development Techniques, 17th International Workshop (WADT 2004)*, Lecture Notes in Computer Science. Springer; Berlin; http://www.springer.de, 2004. To appear.
21. A. Sernadas, J. F. Costa, and C. Sernadas. An institution of object behaviour. In H. Ehrig and F. Orejas, editors, *Recent Trends in Data Type Specification*, volume 785 of *Lecture Notes in Computer Science*, pages 337–350. Springer-Verlag, 1994.
22. A. Sernadas and C. Sernadas. Denotational semantics of object specification within an arbitrary temporal logic institution. Research report, Section of Computer Science, Department of Mathematics, Instituto Superior Técnico, 1049-001 Lisboa, Portugal, 1993. Presented at IS-CORE Workshop 93.
23. A. Sernadas, C. Sernadas, C. Caleiro, and T. Mossakowski. Categorical fibring of logics with terms and binding operators. In D. Gabbay and M. d. Rijke, editors, *Frontiers of Combining Systems 2*, Studies in Logic and Computation, pages 295–316. Research Studies Press, 2000.
24. A. Tarlecki. Moving between logical systems. In M. Haveraaen, O. Owe, and O.-J. Dahl, editors, *Recent Trends in Data Type Specifications. 11th Workshop on Specification of Abstract Data Types*, volume 1130 of *Lecture Notes in Computer Science*, pages 478–502. Springer Verlag, 1996.