Carnap, Goguen, and the Hyperontologies

Logical Pluralism and Heterogeneous Structuring in Ontology Design

Oliver Kutz, Till Mossakowski, and Dominik Lücke

To Joseph Goguen

Mathematics Subject Classification (2010). Primary 68T30; Secondary 03C95.

Keywords. Ontologies; Reasoning; Modularity; Logical Pluralism; Combination Techniques; Algebraic Specification; Institution Theory.

Abstract. We present a general framework for the design of formal ontologies, resting on two main principles: firstly, we endorse Rudolf Carnap's principle of logical tolerance by giving central stage to the concept of logical heterogeneity, i.e. the use of a plurality of logical languages within one ontology design. Secondly, to structure and combine heterogeneous ontologies in a semantically well-founded way, we base our work on abstract model theory in the form of institutional semantics, as forcefully put forward by Joseph Goguen and Rod Burstall.

The theoretical foundation in institution theory establishes a close link to algebraic specification theory. We explore this link by systematically applying tools and techniques from this area to corresponding ontology structuring and design tasks, in particular employ the structuring mechanisms of the heterogeneous algebraic specification language HetCasl for defining an abstract notion of structured heterogeneous ontology, leading to the idea of a hyperontology, a heterogeneous, distributed, highly modular and structured ontology. This approach enables the designer to split up a heterogeneous ontology into semantically meaningful parts and employ dedicated reasoning tools to them.

Moreover, we distinguish, on a structural and semantic level, several different kinds of combining and aligning heterogeneous ontologies, namely integration, connection, and refinement. The notion of heterogeneous refinement can, in particular, be used to provide both a general notion of sub-ontology as well as a notion of heterogeneous equivalence of ontologies. Finally, we sketch how different modes of reasoning over ontologies are related to these different structuring aspects.

This paper is an expanded version of [103], with material drawn from [109, 104, 105, 107].

Contents

1. Introduction

Only later, when I became acquainted with the entirely different language forms of *Principia Mathematica*, the modal logic of C. I. Lewis, the intuitionistic logic of Brouwer and Heyting, and the typeless systems of Quine and others, did I recognise the infinite variety of possible language forms. On the one hand, I became aware of the problems connected with the finding of language forms suitable for given purposes; on the other hand, I gained the insight that one cannot speak of "the correct language form", because various forms have different advantages in different respects. The latter insight led me to the principle of tolerance.

Rudolf Carnap, *Intellectual Autobiography*, p. 68, (1963)

There is a population explosion among the logical systems used in computing science. [...] However, it seems that many general results used in the applications are actually completely independent of what underlying logic is chosen.

Joseph A. Goguen, Rod M. Burstall, *Institutions: Abstract Model Theory for Specification and Programming*, (1992)

Given a species of structure, say widgets, then the result of interconnecting a system of widgets to form a super-widget corresponds to taking the *colimit* of the diagram of widgets in which the morphisms show how they are interconnected.

Joseph A. Goguen, *A Categorical Manifesto*, 6th dogma, (1991)

Traditionally, investigations of human reasoning have been pursued mainly at the level of reasoning performed by single individuals or small groups (e.g. in a dialogue situation). *Ontologies* can be thought of as establishing common agreements among numerous experts on the logical meaning of certain terms in a particular field. It turns out that experience in this rather young field (when understood as a part of modern knowledge engineering and logic rather than general philosophy) supports the relativistic perspective suggested by the title of this paper. Just as in the fields of philosophical logic and non-classical logic, formal ontology draws on a vast array of formal logics with varying expressive capabilities and directed towards different kinds, or modes, of logical reasoning. We here explore their various roles in ontology design, the reasoning they support, and ways in which ontologies in different such logics can be combined.

More specifically, this paper is about the foundations of ontological engineering. It is not, however, about 'best practices' or discusses specific approaches to or methodologies for ontology design, such as the OntoClean methodology [78]. Rather, we outline a general methodological and theoretical environment for the design and construction of formal ontologies, and the problem of corresponding reasoning support. This framework rests, on a very general level, on two main principles: firstly, we endorse Rudolf Carnap's principle of logical tolerance by giving central stage to the concept of logical heterogeneity, i.e. the use of a plurality of logical languages within one ontology design. Secondly, to structure and combine heterogeneous ontologies in a semantically well-founded way, we base our work on abstract model theory in the form of institutional semantics, as forcefully put forward by Joseph Goguen and Rod Burstall.

We begin by briefly discussing the notion of an 'ontology' itself as it is assumed in modern knowledge engineering. A fundamental distinction, introduced by Guarino and Giaretta in [77], should first be kept in mind, namely the distinction between 'Ontology' (with an uppercase 'O') understood as the philosophical discipline, and the use of 'ontologies' (with a lowercase 'o' and a plural reading) as a knowledge engineering artefact.

Amongst the many definitions of 'Ontology' as a philosophical discipline, we present the definition given by Leibniz because of its clarity and brevity:

> Ontology or the science of something and of nothing, of being and not-being, of the thing and the mode of the thing, of substance and accident. [112, Text n. 126 p. 527]

The contemporary notion of an 'ontology' as a technical artefact,[1] on the other hand, grew out of a hypothesis which lies at heart of modern knowledge representation and reasoning, namely Simon & Newell's *physical symbol system hypothesis*: "A physical symbol system has the necessary and sufficient means for general intelligent action." [146].[2] The general idea of a 'formal ontology' that emerged from this tradition within Artificial Intelligence, with Marvin Minsky's semantic frames as one of the first milestones [134], is a symbolic representation of a specific domain of human general or commonsense knowledge that is based on general principles and insights from the philosophical discipline 'Ontology', based on logical representation languages, and trimmed or fine-tuned towards intended technical applications and reasoning scenarios.

An often cited definition of the term 'ontology' in this latter sense was given by Tom Gruber in 1992, taken from [72] and slightly abridged. We will discuss this in some detail and relate it to what we believe are the most important aspects of universality w.r.t reasoning with and over ontologies.

> A conceptualization is an abstract, simplified view of the world that we wish to represent for some purpose. Every knowledge base, knowledge-based system, or knowledge-level agent is committed to some conceptualization, explicitly or implicitly.
>
> An **ontology** is an explicit specification of a conceptualization. The term is borrowed from philosophy, where an

[1]Different notions of 'ontology' as a technical artefact have been extensively discussed in the literature, and we here do not intend to repeat these discussions. For instance, to explicate the definition given by Gruber in some more detail [77] distinguished seven different readings of the term 'ontology', namely: (1) Ontology as a philosophical discipline; (2) Ontology as a an informal conceptual system; (3) Ontology as a formal semantic account; (4) Ontology as a specification of a "conceptualization"; (5) Ontology as a representation of a conceptual system via a logical theory; (5.1) characterized by specific formal properties; (5.2) characterized only by its specific purposes; (6) Ontology as the vocabulary used by a logical theory; (7) Ontology as a (meta-level) specification of a logical theory.

[2]See [131] for an extensive analysis and historical detail.

> Ontology is a systematic account of Existence. For AI systems, "what exists" is that which can be represented. When the knowledge of a domain is represented in a declarative formalism, the set of objects that can be represented is called the universe of discourse. This set of objects, and the describable relationships among them, are reflected in the representational vocabulary with which a knowledge-based program represents knowledge. Thus, in the context of AI, we can describe the ontology of a program by defining a set of representational terms. In such an ontology, definitions associate the names of entities in the universe of discourse (e.g., classes, relations, functions, or other objects) with human-readable text describing what the names mean, and formal axioms that constrain the interpretation and well-formed use of these terms. Formally, an ontology is the statement of a logical theory.

Whilst this definition is widely cited, it has also been criticised for not being precise enough, in particular concerning the usage of the notion of a 'conceptualization' adapted from [61].[3] However, it still nicely captures the main differences between the usage of the term 'ontology' in philosophy vs. computer science and artificial intelligence. Namely, consider the following snippets from this definition:

1. 'simplified view of the world that we wish to represent for some purpose': an ontology as a technical artefact is not intended to cover the world in its entirety, but only chosen (small) parts, on specific levels of abstraction, and for given purposes;
2. "'what exists" is that which can be represented': ontological commitments are dependent on the expressive capabilities of selected representational formalisms;
3. 'representational vocabulary' and 'human-readable text': there is a 'tension' between the logical vocabulary used, and the natural language concepts and terms it is meant to capture;[4]

[3]Guarino, in [75], for instance, analysis that it assumes an *extensional* interpretation, i.e. referring to particular states of affairs, whilst the intended interpretation of the term 'conceptualisation' should be an *intensional* one, i.e. singling out the class of all admissible interpretations of the relevant terms used. (As in Montague semantics, where the 'meaning' of a concept C can be defined as a function mapping possible worlds to extensions of C at that world). I.e. in the terminology of [77], a conceptualisation is a semantic structure that reflects a particular conceptual system, an ontological theory is a set of axioms intended to express ontological knowledge, and an ontological commitment the set of its models, which should moreover not be in conflict with the conceptualisation.

[4]For instance, Brachman, in 1979, introduced a classification of the primitives used in KR systems at the time [29], distinguishing the following four levels: (i) 'Implementational', (ii) 'Logical', (iii) 'Conceptual', and (iv) 'Linguistic'. Guarino, in [74, 76] added to these four layers yet another layer, namely the 'Epistemological Layer' for the primitives, situated between the 'Logical' and the 'Conceptual' layers.

4. 'an ontology is the statement of a logical theory': on a technical level, an ontology is seen as equivalent to a logical theory, written in a certain formalism.

Three questions in particular we would like to extract from this discussion which are directly related to questions concerning the universality of human reasoning when applied to ontology engineering:

(1) Can there be a 'universal ontology', i.e. a symbolic representation of ontological or commonsense knowledge that fits all kinds of domains and possible application and reasoning scenarios?

(2) Correspondingly, even if not all relevant knowledge could be encoded in one single formal ontology, is there one perfect, 'true' formal ontology language that is adequate for all purposes, e.g. is first-order logic such a language?

(3) Irrespective of whether or not there is such a language, is there a singular kind, or mode, of formal reasoning that needs to be performed with and over formal ontologies, independently of the intended application area and chosen representational formalism?

In condensed form, our answers to these three questions will turn out to be 'No!', 'No!', and 'No!', and the core of this paper will provide a detailed technical substantiation of this claim—in the next three section, we will sketch informally our reasons for supporting a relativistic approach to ontology design, discussing in particular the need for a plurality of ontologies vs. a single monolithic ontology, the necessity of employing several different formal representation languages, as well as the importance of supporting logically diverse reasoning mechanisms.[5] In Section 1.4, then, we will provide a more technical summary of the contributions of this paper and explicate the terms **Carnapian Goguenism** and **Hyperontology**.

1.1. From Universal Common Sense to Domain Ontology.

In Artificial Intelligence, there is a long tradition in attempting to provide 'universal' solutions to generic problems, from the concept of a 'general problem solver' (GPS) articulated in 1957 by Simon & Newell [145], the Cyc ontology to capture all of human common sense knowledge started by Douglas Lenat in 1984 [113], to attempts starting in the 1990s at constructing foundational ontologies, such as BFO, GFO, and Dolce. Whilst the latter have already given up on being universally applicable, Cyc's vision is to explicitly 'assimilate' various data sources and thus to "extend the flexibility and power of the Cyc product to serve as the universal ontology and knowledge repository in any application requiring knowledge based reasoning" [130]. The belief that such an approach can work might be summarised by a quote attributed to Lenat: "Intelligence is ten million rules."

In all cases, it turned out that, whilst being partly successful, the applicability of these approaches was far from universal; rather, such universal

[5]In particular, we hold that questions (2) and (3) are inextricably intertwined.

approaches typically suffer, at the very least, from algorithmic combinatorial explosion: in the case of the GPS, this led to a diversification of algorithmic approaches, in the case of Cyc to more modest goals at when and where this kind of knowledge base could be employed, and in cases of foundational ontologies to a diversification not only of targeted application areas for a specific foundational ontology, but also to different versions of such ontologies, such as Dolce (first-order) and Dolce-Lite (DL). This corresponds to a relativisation of the physical symbol system hypothesis, resulting in a more pragmatic use of ontologies in focused application domains and contexts, and a grounding of ontological concepts (e.g. using sensor data), thus better capturing the situated, embodied and embedded nature of human concepts (compare Goguen's [63] for a similar analysis).

Ontologies generally play an increasingly important role in various areas of Knowledge Representation, ranging from the life sciences and engineering domains to linguistic semantics. Ontologies may accordingly be classified with respect to their intended usage. Guarino [75] (p. 7), for instance, distinguishes between 'top-level ontology' (also called upper or foundational ontology, axiomatising the most general concepts such as 'space', 'time', and 'matter'), 'domain ontology' and 'task ontology' (axiomatising the vocabulary related to a generic domain, such as medicine, or specific activities respectively, such as 'diagnosing'), and 'application ontology' (describing concepts depending both on a particular domain and task). Such different kinds of ontologies might not only require different levels of logical expressivity, but indeed might require different kinds of reasoning support. In the process, ontologies are being designed in a broad spectrum of logical languages, with considerably varying expressivity and supporting quite different reasoning methods.

As a matter of fact, first-order logics and their fragments are very popular for the specification of ontologies. Many (domain) ontologies are written in description logics (DLs), like $\mathcal{SHOIN}(D)$ (underlying the web ontology language $\mathcal{OWL}$ in its variant $\mathcal{OWL}$-DL 1.0) and $\mathcal{SROIQ}(D)$ (underlying $\mathcal{OWL}$-DL 2.0) [90]. These logics are characterised by having a rather fine-tuned expressivity, exhibiting (still) decidable satisfiability problems, whilst being amenable to highly optimised implementations.

In this context, it can be a rather difficult task for an ontology designer to choose an appropriate logic and formalism for a specific ontology design beforehand—and failing in making the right choice might lead to the necessity of re-designing large parts of an ontology from scratch, or limit future expandability.

1.2. Logical Pluralism and Syntactic Heterogeneity.

Heterogeneity (or plurality) of ontology languages is thus clearly an important issue, and is the first main topic of this paper.

Carnap, in '*Die logische Syntax der Sprache*' [33, §17], famously put forward his principle of logical tolerance as follows:

Es ist nicht unser Geschäft, Verbote zu erlassen, sondern zu Konventionen zu gelangen.

In der Logik gibt es keine Moral. Jeder mag seine Logik, d.h. seine Sprachform, aufbauen wie er will. Nur muß er, wenn er mit uns diskutieren will, deutlich angeben, wie er es machen will, syntaktische Bestimmungen geben statt philosophischer Erörterungen.[6]

In philosophy, the idea of **logical pluralism** has a long history. Roughly speaking, this is the position that challenges the singularity of classical logic as the *one true logic*, and instead maintains that, for various purposes and applications, or contexts, other logics (i.e. other consequence relations) than classical logic are the *right ones*. While this idea can be traced back through the whole history of logic, the 20th century saw an abundance of such challenges to classical logic (see e.g. [81, 82]), such as Brouwer's intuitionism (and the various schools of constructivist logic that developed from there), substructural logics (e.g. relevant and linear logic), paraconsistency, modal, fuzzy, and many-valued logic, etc.
In a similar spirit, in his study of 'pre-semantics'[7], Belnap [17] argues for the usefulness of Carnapian tolerance thus

> In the first place, pre-semantics helps us become clear that some of the deepest semantic ideas are quite independent of notational systems (grammars). Second, in the tolerant spirit of Carnap, we believe that one is likely to want a variety of complementary (noncompeting) pre-semantic analyses—and most especially, a variety of pre-semantic treatments of one and the same "language". [. . .] There can and should be multiple useful, productive, insightful and pertinent analyses of the same target. Pre-semantics therefore emphasizes the usefulness of thinking in terms of a variety of pre-semantic systems.

Obviously, the idea of logical pluralism can be (and constantly is) heavily debated within philosophical logic (compare e.g. [14, 15], [157, 158], and [178]). However, the need for the design and application of a lot of special purpose non-classical logics has been articulated in many areas of computer science and linguistics, in particular in knowledge representation, artificial intelligence, semantics of natural language, and the logical foundations of programming languages and software specification. Here is one such voice [133]:

> [. . .] it is a fact of life that no single perspective, no single formalization or level of abstraction suffices to represent a system and reason about its behavior. [. . .]

[6]"It is not our business to set up prohibitions, but to arrive at conventions. [. . .] In logic there are no morals. Everyone is at liberty to build up his own logic, i.e. his own language, as he wishes. All that is required of him is that, if he wishes to discuss it, he must state his methods clearly, and give syntactical rules instead of philosophical arguments." Translated 1937.

[7]Belnap [17] argues that "Semantics presupposes grammar." and therefore that pre-semantics is the study of "pure theories" of values such as Carnap's *intensions* that do not involve grammar directly, and that are, in this reading, understood as pre-semantic rather than 'semantic' in the strict sense.

> no logical formalism (specification language, prototyping language, etc.) will be best for all purposes. What exists is a space of possibilities (the universe of logics) in which careful choice of the formalisms that best suit some given purposes can be exercised.

Indeed, from this situation, the new field of 'universal logic' has emerged (see e.g. [22], the journal 'Logica Universalis', and the book series 'Studies in Universal Logic'). Universal logic is to logic what universal algebra is to algebra: it is concerned with studying the most general features of logics or classes of logics.

Logical pluralism de facto permeates almost all areas of knowledge engineering; its wide acceptance is clearly a fact we have to acknowledge. Indeed, we believe that from a methodological viewpoint, and especially from its sheer practical usefulness, the advantages of adopting a position of logical pluralism in ontology engineering can hardly be seriously challenged, and maintain, as Belnap [17] put it, that "One does not have to 'believe in alternative logics' to repudiate the sort of absolutism that comes not from logic itself, but from narrow-gauge metaphysics or epistemology.".

1.3. Logical Reasoning Modes for Ontologies.

When acknowledging the situation that there are different kinds of ontologies targeted at various application domains, and being formulated in varying logical languages, it is obvious that there are also diverse requirements concerning reasoning support. Indeed, there are many cases where either weaker DLs are enough, such as sub-Boolean $\mathcal{EL}$, and more specialised (and faster) algorithms can be employed, or, contrarily, the expressivity has to be extended beyond the scope of standard description logics. An example for the former would be the NCI thesaurus (containing about 45.000 concepts) which is intended to become the reference terminology for cancer research [173], an example for the latter many foundational ontologies such as Sumo [147], Dolce [129, 58], GFO [84], BFO [71], and GUM [11, 10]. These are typically specified in some variant of first-order logic, and their first-order theories tend to be rather large (Dolce, for instance, consists of a few hundred axioms, and Sumo of several thousand). But also for domain ontologies, the web ontology language $\mathcal{OWL}$ is not always sufficient. Although $\mathcal{OWL}$ is being constantly refined and extended, its main target application is the Semantic Web and related areas, and it can thus not be expected to be fit for any purpose: there will always be new, typically interdisciplinary application areas for ontologies where the employed (or required) formal languages do not directly fit into the $\mathcal{OWL}$ landscape. A notorious example for this is $\mathcal{OWL}$'s lack of expressive power to properly define the mereological parthood relation which is essential e.g. in many medical or biological ontologies.[8]

[8]We will discuss formalisations of parthood extensively in Sec. 2.1, but see [94] for an overview.

The discussion so far offers a *pragmatic* motivation for logical pluralism: whilst first-order logic is predominant, there are many variants and fragments in use, with different expressivity, user communities, and tool support. However, there is also a more *fundamental* motivation for logical pluralism, namely the open-ended nature of a field such as ontology engineering that strongly depends on a rapidly changing array of application domains in which ontologies are deployed, with continuously changing demands on expressive capabilities and supported reasoning tasks. Sometimes, for example, first-oder expressivity is deemed insufficient. There are versions of DOLCE that use modal logic and second-order constructs. Moreover, there are cases where combinations with or connections to formalism with different semantics have to be covered, such as temporal, spatial, or epistemic logics, cf. e.g. [3, 4, 106, 48, 31].

Although classical reasoning is still predominant in currently implemented systems, it has been realised in recent years that there are many application scenarios for ontologies where there is an explicit need to leave the classical mode of reasoning and switch to a different mode, most importantly to non-monotonic and default reasoning, paraconsistent reasoning, or various kinds of reasoning with uncertainty. Paraconsistent logic, for instance, is quite important in ontological engineering, in particular when dealing with large amounts of 'data': in many contexts, one does not want a local inconsistency among some facts to have the effect of a global inconsistency [124]. Another non-classical mode of reasoning arises when incomplete or uncertain information suggests probabilistic or fuzzy reasoning, as for instance studied in [116, 117], or employing rough mereology to deal with problems of ontological granularity [95]. Finally, non-monotonic and default reasoning becomes important when dealing with 'rules' in addition to axioms, when working with databases that use closed-world semantics, or when one wants to, e.g., allow exceptions in concept definitions.

Pluralism of ontology languages also gains importance when considering the problem of cognitive adequacy of formalism. Some spatial calculi use linguistic terms to define spatial relations (for instance the Region Connection Calculus RCC8 discussed below in Section 2.1.2), aiming at being cognitively adequate in the sense that the formal terms provided reflect distinctions humans would typically make and find relevant. For instance, RCC8 distinguishes relations such as 'overlap' and 'disjointness', but also 'tangential proper part'. The simpler RCC5 does not consider the boundary of a region, and is thus seen as more cognitively adequate in situations where humans would not consider this distinction relevant (see [**?**] for such an analysis concerning Allen's temporal interval calculus). Another case would be modal and temporal operators, whose usage could be seen as cognitively more adequate than first-order logic as these operators directly reflect natural language (and its semantics)—quantifying over time points, for most people, perhaps is not the most straightforward way of writing down a temporal statement. It could also be argued that paraconsistent logic is more cognitively adequate than classical reasoning for the simple fact that, in everyday communication and reasoning, we do not accept logical explosion.[9]

1.4. From Carnapian Goguenism to Constructing Hyperontologies

Adopting a position of logical pluralism and notational tolerance by no means implies that we believe there should be no standards for ontology languages. To the contrary, establishing standards like the RDF framework (see [115] for an institutional analysis), the $\mathcal{OWL}$ languages [70][10], Common Logic [39, 44][11], or indeed the CASL family [139][12], is essential for deploying ontologies in the real world. But a general approach to ontology engineering should not only provide the means to adopt (at the same time) several standards for ontology languages, it also needs to deliver techniques to systematically combine and integrate logical modules written in different such languages.

The second focus of the paper is thus on the problems of **modularity and structuring** of ontologies, and the related problem area of combining, in various ways, ontology components, or modules.

A first problem here is the mere size of ontologies making the design process potentially quite hard and error prone (at least for humans). This issue has been only partly cured in $\mathcal{OWL}$ by the `imports` construct, and leaves the problem of 'debugging' large ontologies as an important issue [92]. Also, simple operations such as the re-use of parts of an ontology in a different 'context' whilst *renaming* (parts of) the signature are not possible in the $\mathcal{OWL}$ languages. Although matching [51], aligning [183], and re-using (parts of) ontologies has received considerable attention recently, in particular concerning algorithms for deciding conservativity in various description logics [118, 40, 96], there is little work on formal structuring in the presence of heterogeneity.

We believe that a lot can be learned in this respect from techniques developed for (algebraic) specification in software engineering, and will provide a systematic account that parallels structuring techniques from algebraic specification with typical problems found in ontology design. The cure that we propose to the above issues relies strongly on the concept of *heterogeneity*: facing the fact that several logics and formalisms are used for designing ontologies, we suggest heterogeneous structuring constructs that allow the combination of ontologies in various but always formally and semantically well-founded ways. Our approach is based on the theory of institutions and formal structuring techniques from algebraic specification theory, and generalises and extends ideas of [20], [2], [127], and in particular [64, 65].

[9]Cognitive adequacy is also related to the *succinctness* of logical formalisms. For instance, many first-order statements can be equivalently formalised in various modal or temporal formalisms, but the first-order formulae may be exponentially shorter (and thus more intricate).
[10]See also `http://www.w3.org/2007/OWL/wiki/OWL_Working_Group`
[11]See also `http://common-logic.org/`
[12]See also `http://www.cofi.info/`

Two ideas, then, in their combination, perfectly summarise our approach to ontological engineering: Carnap's principle of **logical tolerance** [33], and Goguen's approach to **logical integration** [62, 64].

According to the former, we acknowledge and approve that different ontology designs will require different logical languages,[13] whilst the latter maintains that the logical re-combination of ontology modules should rely on an understanding of the structural or diagrammatic properties of the combination.

The position that takes logical pluralism and thus relativises it to ontological engineering we might call **onto-logical pluralism**. The position that takes onto-logical pluralism and applies structuring and modularity principles based on abstract model theory (institution theory), we call **Carnapian Goguenism**.

Just as it has become a truism that good software engineering strongly involves modular design as well as a choice of appropriate programming languages for the various modules, we maintain that different kinds of ontologies require different logical languages, with varying expressivity, and indeed with different and possibly not directly compatible semantics, and that the combination of such *heterogeneous* modules requires sophisticated techniques of structuring and modular re-combination. This position directly leads to the following three methodological questions:

1. How can large and complex ontologies be built up from parts, being formulated in different logical languages, and in what ways can those parts be related?
2. How can the structure of the overall ontology be represented, and how can various logical properties of the parts be preserved?
3. How can we perform (automated) logical reasoning over such structured ontologies, and how, or when, can we reduce reasoning in the overall ontology to the ontology's component modules?

In this paper, we will give answers to all three questions which may be sketched as follows. We will

Structured Ontologies. develop a rather abstract view of heterogeneously structured ontology, encompassing essentially all logics being used in ontology design today and allowing for the modelling of the most complex relationships between ontologies. Technically, we have formalised several logics that are important from an ontology design perspective as so-called *institutions* [66, 67], including the description logic $\mathcal{SROIQ}(D)$ and various variants of first-order logic and different modal logics, and

[13]Carnap's version of logical tolerance ('Toleranzprinzip der Syntax') argues for the freedom to choose appropriate logical languages for various tasks. Carnap's position also implies, as [160] put it, that "Any difference in logical consequence is due to a difference in languages." This is, however, a weaker (or different) position than full blown logical tolerance as it is sometimes understood, which maintains that there are indeed different 'right' consequence relations for different applications for *the same* language, see [34, 35] and [160].

supply *institution comorphisms* (logic translations) as mappings between them.

Classification of Combination Techniques. systematise the field of 'combining ontologies' by identifying three classes of such combinations: *refinements*, *integrations*, and *connections*. The differentiating criteria are the use of signatures in the overall combination and the corresponding model-theoretic properties.

Hyperontologies. introduce a notion of heterogeneously structured ontology, which also affords distributed networks of ontologies written in different formalisms, which we call *hyperontologies*.

Reasoning with Combinations. analyse how various well-known ontology design and combination techniques fit into these abstract categories, including structuring through conservative extensions, ontology alignments,[14] $\mathcal{E}$-connections, and database-scheme–ontology reconciliation.

Tool Support. discuss how the tool HETS (Heterogeneous Tool Set) can support various reasoning and ontology engineering tasks and indicate the current and planned tool support for existing ontology languages and reasoners.

The main features of our approach to ontology design may then be summarised as follows:

- The ontology designer can use description logics to specify most parts of an ontology, and can use first-order (or even higher-order) logic where needed. Moreover, the overall ontology can be assembled from (and can be split up into) semantically meaningful parts ('modules') that are systematically related by structuring mechanisms. These parts can then be re-used and/or extended in different settings.
- Institution theory provides logic translations between different ontology languages, translating the syntax and semantics of different formalisms.
- Various concepts of 'ontological module' are covered, including simple imports (extensions) and union of theories, as well as conservative and definitional extensions. We here consider conservative extensions as a way of ('a priori') structuring ontologies, rather than as a methodology to ('a posteriori') cutting large ontologies into pieces which lie conservatively within the whole. However, we consider the latter (algorithmic) approach as assistive to, for instance, verifying a desired conservative design.
- Structuring into modules is made explicit in the ontology and generates so-called proof obligations, e.g. for conservativity. Proof obligations can also be used to keep track of desired consequences of an ontology (module), especially during the design process.
- Re-using (parts of) ontologies whilst renaming (parts of) the signature is handled by *symbol maps* and *hiding symbols*: essentially, this allows the internalisation of (strict) alignment mappings.

[14]Previously discussed in [109].

- The approach allows heterogeneous refinements: it is possible to prove that an ontology O_2 is a refinement of another ontology O_1, formalised in a different logic. For instance, one can check if a domain ontology is a refinement of (a part of) a foundational one. There are two interesting by-products of the definition of heterogeneous refinement: firstly, it provides a rather general definition of heterogeneous sub-ontology, and secondly, it can be used to give a definition of equivalence of ontologies across different ontology languages.
- The tool HETS provides parsing, static analysis and proof management for heterogeneous logical theories. It can visualise the module structure of complex logical theories, using so-called development graphs. For individual nodes (corresponding to logical theories) in such a graph, the concept hierarchy can be displayed. Moreover, HETS is able to prove intended consequences of theories, prove refinements between theories, or demonstrate their consistency, and compute normal forms and colimits (also for heterogeneous specifications). This is done by integrating several first-order provers and model-finders (SPASS, DARWIN), a higher-order prover (ISABELLE), as well as the DL reasoners PELLET and FACT++.

The structure of the paper is as follows:

In Section 2, we suggest a heterogeneous framework for the design of ontologies, based on the theory of institutions and the notion of development graph, and define the notion of 'abstract structured heterogeneous ontology'. We here also introduce several logics used in ontology design, formalise them as institutions, and discuss their varying expressivity and applicability (mostly) with respect to the mereological parthood relation. Section 3 then discusses a number of combination techniques and scenarios known from the literature from the abstract viewpoint of institution theory, including reference ontologies, alignments, as well as $\mathcal{E}$-connections and distributed description logics (DDLs). In particular, we turn our attention to relationships between ontologies that are (logically) particularly strict, namely the notion of refinement, the derived notion of heterogeneous sub-ontology, and a notion of heterogeneous equivalence based on refinements. In Section 4, we supply some details on the implementation of our abstract framework based on the tool HETS, and discuss its reasoning support for various ontology languages ranging from the DL $\mathcal{SROIQ}(D)$ to first-order and higher-order logic, including 'non-standard' operations such as co-limit computation and support for checking conservativities. We here also give two examples of refinements together with concrete specifications, based on a relational-schema and description logic formalisations of a bibliographical database, as well as examples for ontology integration and alignments. Finally, Section 5 summarises the paper and discusses future work.

2. Heterogeneous Ontologies and Structuring

The study of modularity principles can be carried out to a quite large extent independently of the details of the underlying logical system that is used. The notion of **institutions** was introduced by Goguen and Burstall in the late 1970s exactly for this purpose (see [67]). They capture in a very abstract and flexible way the notion of a logical system by describing how, in any logical system, signatures, models, sentences (axioms) and satisfaction (of sentences in models) are related. Here, a signature collects the non-logical (i.e. 'user-defined') symbols that both occur in the sentences and are interpreted in the models. E.g. in first-order logic, these are predicate symbols, constant and function symbols, and possibly sorts. Signature morphisms can be thought of as mappings between two signatures. An insight of institution theory is the importance of indexing by context (i.e. signature) and change of context (i.e. signature morphisms), such that satisfaction is invariant under change of context. Another important insight is the fact that a surprisingly large body of logical notions and results can be developed in a way that is completely independent of the specific nature of the underlying institution.[15]

We assume some acquaintance with the basic notions of category theory and refer to [1] or [126] for an introduction.

Definition 2.1. An **institution** is a quadruple $I = (\mathbf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \models)$ consisting of the following:

- a category $\mathbf{Sign}$ of *signatures*,
- a functor $\mathbf{Sen}\colon \mathbf{Sign} \longrightarrow \mathbf{Set}$[16] giving, for each signature Σ, the set of *sentences* $\mathbf{Sen}(\Sigma)$, and for each signature morphism $\sigma\colon \Sigma \longrightarrow \Sigma'$, the *sentence translation map* $\mathbf{Sen}(\sigma)\colon \mathbf{Sen}(\Sigma) \longrightarrow \mathbf{Sen}(\Sigma')$, where often $\mathbf{Sen}(\sigma)(\varphi)$ is written as $\sigma(\varphi)$,
- a functor $\mathbf{Mod}\colon \mathbf{Sign}^{op} \longrightarrow \mathcal{CAT}$[17] giving, for each signature Σ, the category of *models* $\mathbf{Mod}(\Sigma)$, and for each signature morphism $\sigma\colon \Sigma \longrightarrow \Sigma'$, the *reduct functor* $\mathbf{Mod}(\sigma)\colon \mathbf{Mod}(\Sigma') \longrightarrow \mathbf{Mod}(\Sigma)$, where often $\mathbf{Mod}(\sigma)(M')$ is written as $M'\restriction_\sigma$, and $M'\restriction_\sigma$ is called the *σ-reduct* of M', while M' is called a *σ-expansion* of $M'\restriction_\sigma$,
- a satisfaction relation $\models_\Sigma \subseteq |\mathbf{Mod}(\Sigma)| \times \mathbf{Sen}(\Sigma)$ for each $\Sigma \in |\mathbf{Sign}|$,

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

$$(\star) \qquad M' \models_{\Sigma'} \sigma(\varphi) \text{ iff } M'\restriction_\sigma \models_\Sigma \varphi$$

for each $M' \in |\mathbf{Mod}(\Sigma')|$ and $\varphi \in \mathbf{Sen}(\Sigma)$, expressing that truth is invariant under change of notation and context.[18] $\dashv$

[15] For an extensive treatment of model theory in this setting, see [46].

[16] **Set** is the category having all small sets as objects and functions as arrows.

[17] $\mathcal{CAT}$ is the category of categories and functors. Strictly speaking, $\mathcal{CAT}$ is not a category but only a so-called quasicategory, which is a category that lives in a higher set-theoretic universe.

[18] Note, however, that non-monotonic formalisms can only indirectly be covered this way, but compare, e.g., [79].

2.1. A Variety of Logics

Examples of institutions include, among others, first- and higher-order classical logic, description logics, and various non-classical logics such as (quantified) modal logics or paraconsistent logics. In the following, we give the institutional formulations of the most important logics used in ontology engineering, and use this occasion to illustrate the varying expressivity and applicability of these logics in ontology design, and give some historical background. As a running example, we use the well-known ontological problem of formalising the mereological **parthood relation** [171]. This notion is central not only to foundational ontologies such as Dolce [129], but is important also in many more applied areas that use ontologies. [94], for instance, discuss the ontological, linguistic and cognitive aspects of different mereological and meronymic part-whole relations introduced in the literature and organise them in a basic formal taxonomy in order to assist an ontology designer in the *ontologically correct* usage and selection of these different relations. We here, in contrast, focus mostly on the classical parthood relation and analyse and compare the varying expressive capabilities w.r.t. parthood of the logics that we introduce.

2.1.1. Variants and Fragments of Classical First-Order Logic.

Example. **Propositional Logic.** The institution **Prop** of propositional logic has sets Σ (of propositional symbols) as signatures, and functions $\sigma : \Sigma_1 \to \Sigma_2$ between such sets as signature morphisms. A Σ-model M is a mapping from Σ to $\{\mathit{true}, \mathit{false}\}$. The reduct of a Σ_2-model M_2 along $\sigma : \Sigma_1 \to \Sigma_2$ is the Σ_1-model given by the composition $M_2 \circ \sigma$. 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 straight-forward to see that the satisfaction condition holds.

Propositional reasoning is at the core of ontology design. Boolean expressivity is sufficient to axiomatise the taxonomic structure of an ontology by imposing disjointness and sub- or super-concept relationships via implication and negation, as well as e.g. non-empty overlap of concepts. Here, the most immediate connection to ontology intuitively is perhaps given when propositional logic is thought of as diagrammatic reasoning (over unary predicates interpreted as sets) in terms of Venn or Euler diagrams [179]. This style of reasoning about concepts is already implicit, among others, in Aristotle's Syllogisms and in particular in Ramon Llull's work in the 13th century that constituted a major milestone towards modern propositional logic by studying, e.g., material implication and conjunction [60].[19]

[19]Gardner's book [60] contains not only material on Llull's life and his contributions to logic, but in particular their relationship to diagrammatic reasoning in the style of Venn.

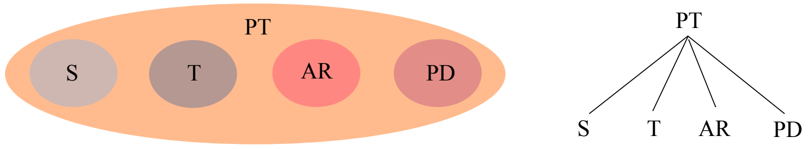

FIGURE 1. A taxonomy for basic mereology as Euler diagram (left) and taxonomic tree (right).

As concerns the parthood relation, we might want to fix the basic taxonomic information as given in the DOLCE ontology. That is, assume basic mereology is defined over the categories PT (for *particular*), PD (for *perdurant*), T (for *time interval*), S (for *space region*), and AR (for *abstract region*). PD, S, T, AR are assumed to be pairwise disjoint. Moreover, PT serves as top-concept, i.e. we have:

$$S \vee T \vee AR \vee PD \rightarrow PT, \quad S \wedge T \rightarrow \bot, \quad T \wedge AR \rightarrow \bot, \quad \text{etc.}$$

The basic taxonomy is illustrated in Fig. 1.

$\dashv$

Example. **Untyped First-order Logic.** In the institution $\mathbf{FOL}^{=}$ 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 such 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.

Untyped first-order logic is capable of expressing basic mereology. More precisely, we here sketch the axioms of the theory of classical extensional parthood CET, which is part of the DOLCE ontology. First of all, the parthood relation P is declared to be a partial order, that is, satisfies reflexivity, antisymmetry, and transitivity. Moreover, CET satisfies the following axioms, for each of the categories $X = S, T, AR, PD$:

(Proper Part)
$\forall x, y \in X.PP(x, y) \quad \longleftrightarrow \quad P(x, y) \wedge \neg P(y, x)$

(Overlap)
$\forall x, y \in X.Ov(x, y) \quad \longleftrightarrow \quad \exists z \in X.(P(z, x) \wedge P(z, y))$

(Atom)
$\forall x \in X.At(x) \quad \longleftrightarrow \quad \neg\exists y \in X.(PP(y, x))$

(Atomic Part)
$\forall x, y \in X.AtP(x; y) \quad \longleftrightarrow \quad P(x, y) \wedge At(x)$

(Binary Sum)
$\forall x, y, z \in X.Sum(z, x, y) \quad \longleftrightarrow \quad \forall w \in X.(Ov(w, z) \Leftrightarrow (Ov(w, x) \vee Ov(w, y)))$

(Binary Difference)
$\forall x, y \in X.\neg\forall x, y, z \in X.Dif(z, x, y) \quad \longleftrightarrow \quad \forall w \in X.(P(w, z) \leftrightarrow (P(w, x) \wedge \neg Ov(w, y)))$

(Extensionality
& Existence of the Difference)
$P(x, y) \longrightarrow \exists z \in X.(Dif(z, x, y))$

(Existence of the Sum)
$\forall x, y \in X.\exists z \in X.(Sum(z, x, y))$

Note that the first six axioms are definitional in nature, namely defining the predicates PP, Ov, At, AtP, Sum and Dif, whilst only the last two ones make existential statements. ⊣

Example. **Many-sorted First-order Logic.** The institution $\mathbf{FOL}^{\mathsf{ms}=}$ of many-sorted first-order logic with equality is similar to FOL. Signatures are many-sorted first-order signatures, consisting of sorts and typed function and predicate symbols. The rest is similar to FOL. For details, see [67].

Although not strictly more expressive than single-sorted $\mathbf{FOL}^{=}$, introducing a sort structure allows a cleaner and more principled design of first-order ontologies. This was indeed one of the reasons for using $\mathbf{FOL}^{\mathsf{ms}=}$ in the Casl formulation of the Dolce ontology (see below for details of the Casl language). The taxonomy introduced in Fig. 1 can be restated by introducing PD, S, T, AR, PT as five different sorts, with the corresponding sub-sort relationships. Moreover, axioms involving different sorts can be stated more succinctly, and static type checking gives more control over correct modelling. Finally, sorting makes a modular construction of models possible, such that consistency proofs for large ontologies become feasible, see [104]. ⊣

Example. **Common Logic.** Common logic has not been formalised as an institution in the literature so far, but doing so is quite straight-forward. A common logic signature Σ (called vocabulary in Common Logic terminology) consists of a set of names, with a subset called the set of discourse names, and a set of sequence markers. A signature morphism consists of two maps between these sets, such that the property of being a discourse name is preserved and reflected.[20] A Σ-model consists of a set UR, the universe of reference, with a non-empty subset $UD \subseteq UR$, the universe of discourse, and four mappings:

- *rel* from UR to subsets of $UD^* = \{< x_1, \ldots, x_n > | x_1, \ldots, x_n \in UD\}$ (i.e., the set of finite sequences of elements of UD);
- *fun* from UR to total functions from UD^* into UD;
- *int* from names in Σ to UR, such that $int(v)$ is in UD if and only if v is a discourse name;
- *seq* from sequence markers in Σ to UD^*.

Model reducts leave UR, UD, *rel* and *fun* untouched, while *int* and *seq* are composed with the appropriate signature morphism component. A Σ-sentence is a first-order sentence, where predications and function applications are written in a higher-order like syntax:

$$t(s)$$

Here, t is an arbitrary term, and s is a sequence term, which can be a sequence of terms $t_1 \ldots t_n$, or a sequence marker. However, a predication $t(s)$ is interpreted like the first-order formula $R(t, s)$, and a function application $t(s)$ like the first-order term $F(t, s)$, where R and F are fictitious symbols denoting the semantic objects *rel* and *fun*. In this way, Common Logic provides a first-order simulation of a higher-order language. Quantification variables are partitioned into those for individuals and those for sequences. Sentence translation along signature morphisms is done by simple replacement of names and sequence markers. Interpretation of terms and formulae is as in first-order logic, with the difference that the terms at predicate resp. function symbol positions are interpreted with *rel* resp. *fun* in order to obtain the predicate resp. function, as discussion above. A further difference is the presence of sequence terms (namely sequence markers and juxtapositions of terms), which denote sequences in UD^*, with term juxtaposition interpreted by sequence concatenation. Note that sequences are essentially a second-order feature. For details, see [39]. As an example, consider the DOLCE formula $\forall\phi(\phi(x))$, corresponding to $\bigwedge_{\psi\in\Pi}(\psi(x))$, where predicate variables ϕ, ψ range over a finite set Π of explicitly introduced universals. In Common Logic, this is written, using standard logical syntax (note that Common Logic is agnostic about concrete syntax)

$$\forall\phi.\Pi(\phi) \rightarrow \phi(x)$$

or in the often used Lisp-like syntax of the Common Logic Interchange Format CLIF:

```
(forall (?phi) (if (pi ?phi) (?phi ?x)))
```

Another example might once again be taken from the area of mereology. Ridder [161] as well as Herre [85] introduce a notion of *second order fusion*

[20]That is, a name is a discourse name if and only if its image under the signature morphism is.

as follows, building on the predicate Ov (Overlap) introduced earlier:

$$Fus(X, a) :\longleftrightarrow \forall y.(Ov(y, a) \leftrightarrow \exists z.(X(z) \wedge Ov(z, a))$$

saying that a is the fusion of X, i.e. of the items contained in X. Here, X is a variable for a unary predicate, i.e. the formula is in the language of monadic second-order logic, but can also be regarded as a formula of Common Logic.

Sequence markers add even more flexibility. For example, it is possible to express that a list of predicates is mutually disjoint as follows:

$$\begin{array}{l} \textit{mutually-disjoint}(P) \\ \textit{mutually-disjoint}(P\ \ Q\ \ \ldots) \leftrightarrow \\ \qquad (\forall x.\neg(P(x) \wedge Q(x))) \wedge \textit{mutually-disjoint}(P\ \ \ldots) \wedge \textit{mutually-disjoint}(Q\ \ \ldots) \end{array}$$

The disjointness axioms for Dolce's categories can then concisely be stated as

$$\textit{mutually-disjoint}(S\ \ T\ \ AR\ \ PD)$$

$\dashv$

Example. **Relational Schemes.** This logic, first introduced in [103], is about schemes for relational databases and their integrity constraints. A signature in this institution consists of a set of sorts and a set of relation symbols, where each relation symbol is indexed with a string of sorted field names as in:

```
paper(key id:pointer, title:string, published_in:pointer)
journal(key id:pointer, name:string, impact_factor:integer)
```

Some sorts for the relational schema as `integer`, `pointer` and `string` are predefined and equipped with default interpretations. The identifier `key` can be used as a prefix to sorted field names to specify the primary (compound) key of the schema.

Signature morphisms map sorts, relation symbols and field names in a compatible way, such that primary keys are preserved. A model consists of a carrier set for each sort, where some sorts have predefined carrier sets, and an n-ary relation for each relation symbol with n fields. Model reduction is like that of many-sorted first-order logic. A sentence is a link (integrity constraint) between two field names of two relation symbols. For example, the link

```
paper[published_in] -> journal[id] one_to_many
```

requires that the field `published_in` of any paper coincides with the `id` of at least one journal (the many-one character of this relationship is expressed by the keyword `one_to_many`). Other possible relationships are `one_to_one` and `many_to_many`. Sentence translation is just renaming of relation symbols and of sorts. A link `r[f] -> s[g] t` is satisfied in case of `t = one_to_many` if for each element in `r[f]` there are zero or more occurrences of this element in `s[g]`, but for each element in `s[g]` there is at most one occurrence of an element in `r[f]`. For `t = one_to_one` in both cases only one occurrence is allowed, and for `many_to_many` there is no restriction on the number of occurrences.

Here is an example relational scheme for bibliographies, following [164]:

```
logic RelScheme
spec Biblio_RS =
  Tables
    person(key id:integer, name:string)
    author_of(person, paper:integer)
    paper(key id:integer,title:string,
          published_in:integer)
    journal(key id:integer,name:string,
          impact_factor:float)
  Relationships
    author_of[person]       -> person[id]  one_to_many
    author_of[paper]        -> paper[id]   one_to_many
    paper[published_in]     -> journal[id] one_to_many
```

$\dashv$

Example. **Description Logics.** Signatures of the description logic $\mathcal{ALC}$ consist of a set $\mathcal{A}$ of atomic concepts, a set $\mathcal{R}$ of roles and a set $\mathcal{I}$ of individual constants, 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

$$C ::= \mathcal{A} \,|\, \top \,|\, \bot \,|\, C_1 \sqcup C_2 \,|\, C_1 \sqcap C_2 \,|\, \neg C \,|\, \forall R.C \,|\, \exists R.C$$

These kind of sentences are also called TBox sentences. Sentences can also be ABox sentences, which are membership assertions of individuals in concepts (written $a : C$ for $a \in \mathcal{I}$) or pairs of individuals in roles (written $R(a, b)$ for $a, b \in \mathcal{I}, R \in \mathcal{R}$). Sentence translation and reduct is defined similarly as in $\mathbf{FOL}^=$. Satisfaction is the standard satisfaction of description logics.

$\mathcal{ALC}^{\mathsf{ms}}$ is the many-sorted variant of $\mathcal{ALC}$. $\mathcal{ALCO}$ is obtained from $\mathcal{ALC}$ by adding nominals, i.e. concepts of the form $\{a\}$, where $a \in I$. Other logics, like sub-Boolean $\mathcal{EL}$, $\mathcal{ALCO}$ or $\mathcal{SHOIN}$, are treated similarly. See [115] for a formalisation as an institution.

The (sub-Boolean) description logic $\mathcal{EL}$ has the same sentences as $\mathcal{ALC}$ but restricts the concept language of $\mathcal{ALC}$ as follows:

$$C ::= B \,|\, \top \,|\, C_1 \sqcap C_2 \,|\, \exists R.C$$

The logic $\mathcal{SROIQ}$ [90], which is the logical core of the Web Ontology Language $\mathcal{OWL}$-DL 2.0[21] extends $\mathcal{ALC}$ with the following constructs: (i) complex role boxes (denoted by $\mathcal{SR}$): these can contain: complex role inclusions such as $R \circ S \sqsubseteq S$ as well as simple role hierarchies such as $R \sqsubseteq S$, assertions for symmetric, transitive, reflexive, asymmetric and disjoint roles (called RBox sentences), as well as the construct $\exists R.\mathsf{Self}$ (collecting the set of 'R-reflexive points'); (ii) nominals (denoted by $\mathcal{O}$); (iii) inverse roles (denoted by $\mathcal{I}$); qualified and unqualified number restrictions ($\mathcal{Q}$). For details on the

[21]See also `http://www.w3.org/TR/owl2-overview/`

rather complex grammatical restrictions for $\mathcal{SROIQ}$ (e.g. regular role inclusions, simple roles) compare [90], and see the example given below. $\mathcal{SROIQ}$ can be straightforwardly rendered as an institutions following the previous examples, but compare also [115].

An $\mathcal{OWL}$ bibliographical ontology could look as follows:

$$\begin{array}{lcl} Researcher & \sqsubseteq & \exists name.Thing \\ Article & \sqsubseteq & \exists authorThing \sqcap \exists title.Thing \\ Journal & \sqsubseteq & \exists name.Thing \sqcap \exists hasArticle.Thing \sqcap \exists impactFactor.Thing \end{array}$$

Apart from some exceptions[22], description logics can be seen as fragments of first-order logic via the standard translation [6] that translates both the syntax and semantics of various DLs into untyped first-order logic. It is thus not surprising that parts of the first-order mereology can also be expressed in DLs. In fact, parthood relations take on a rather prominent role in ontologies targeted towards the medical or biological domains. For instance, the Galen ontology covers large parts of anatomy, where parthood relationships are essential, e.g. if a foot is a part of a leg, and the leg is a proper part of the body, then the foot will also be a proper part of the body (see below). Unfortunately, even $\mathcal{SROIQ}$ can only partially capture mereology. Whilst *proper parthood* can be captured as a symmetric, transitive, and asymmetric relation, $\mathcal{SROIQ}$ cannot express antisymmetry, thus *parthood* simpliciter can not adequately be modelled. Moreover, whilst some of the relationships between parthood and proper parthood can be captured by, e.g., complex role inclusions, we quickly run into the expressive limitations of a language such as $\mathcal{SROIQ}$. Kazakov [93] gives the following illustrative example:

1. $\mathsf{isProperPartOf} \sqsubseteq \mathsf{isPartOf}$
2. $\mathsf{isPartOf} \circ \mathsf{isPartOf} \sqsubseteq \mathsf{isPartOf}$
3. $\mathsf{isPartOf} \circ \mathsf{isProperPartOf} \sqsubseteq \mathsf{isProperPartOf}$

Clearly, semantically these are valid role inclusions: (1) says that proper parthood is a sub-relation of parthood, (2) says that parthood is transitive, and (3) says that if x is a part of y and y is a proper part of z then x is a proper part of z. However, as [93] shows, whilst the role inclusions generated by these axioms still define regular languages[23], the syntactic form of these three role inclusions are in conflict with the restrictions imposed on the logic $\mathcal{SROIQ}$.

On the other hand, large medical ontologies, such as SNOMED CT and the NCI thesaurus, largely codify only taxonomical information, with some existential relational structure. These ontologies can be encoded in rather weak DLs such as $\mathcal{EL}$ and allow effective reasoning and query support despite their enormous size [175, 120]. $\dashv$

[22] For instance, adding transitive closure of roles or fixpoints to DLs makes them decidable fragments of second-order logic [26].

[23] This is an important part in the decidability proof for languages such as $\mathcal{SROIQ}$ [90].

2.1.2. Modal and Paraconsistent Logics.

Example. **Modal Logics.** The modal logic **K** has signatures as classical propositional logic, consisting of propositional variables. Sentences are built as in propositional logic, but add a unary modal operator, $\Box$. Models are standard Kripke structures and satisfaction is standard modal satisfaction. Other modal logics are handled by varying the construction of sentences and the model classes accordingly. For instance, in the modal logic $\mathbf{S4_u}$, i.e., Lewis's modal system **S4** with the universal modality added, sentences are now built using propositional variables and two unary modal operators, $\Box$ and $\blacksquare$. Models are again Kripke structures, but based on reflexive and transitive relations. Satisfaction is standard modal satisfaction, but where $\Box$ is interpreted by the transitive reflexive relation, and $\blacksquare$ by universal quantification over worlds.

The logic $\mathbf{S4_u}$ is also complete with respect to a semantics based on topological spaces as the intended interpretation. Here, the propositional variables are interpreted as subsets of a topological space, the necessity operator $\Box$ is interpreted as the interior operator $\boldsymbol{I}$, the possibility operator $\Diamond$ as the closure operator $\boldsymbol{C}$, and the universal quantifier $\blacksquare$ quantifies over all points of the topological space [176, 21, 168].

In this interpretation, it is easily seen that $\mathbf{S4_u}$ can encode the Region Connection Calculus RCC8 [159] heavily being used in qualitative spatial representation and reasoning.[24] Figure 2 displays the 8 basic relations of RCC8, which are mutually exclusive and exhaustive in describing the possible overlap and touching relationships between two (well-behaved[25]) regions in space.

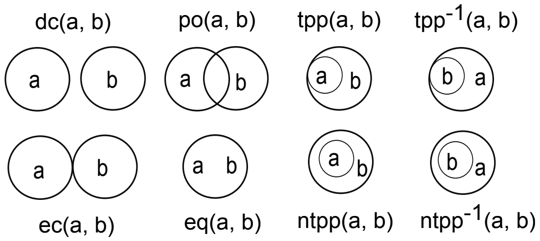

Figure 2. The RCC8 relations

For instance, the relation of partial overlap $po(a, b)$ between regions a and b is defined in $\mathbf{S4_u}$ as follows:

$$\blacklozenge(\boldsymbol{I}a \wedge \boldsymbol{I}b) \wedge \blacklozenge(\boldsymbol{I}a \wedge \neg b) \wedge \blacklozenge(\boldsymbol{I}b \wedge \neg a)$$

[24] And we will give an example later on how this kind of reasoning can be combined with ontological reasoning, see Sec. 3.3.3.

[25] This is typically taken to mean regular-closed subsets of a topological space, i.e. regions X such that $X = \boldsymbol{CI}X$.

saying that there is a point in the intersection of the interiors of a and b, a point in the interior of a not belonging to b, and a point in the interior of b not belonging to a, and similarly for the other relations.[26]

Moving on to combining first-order quantification with modality, the standard and probably simplest formulation of first-order modal logic **QS5** (due to Kripke) has signatures similar to $\mathbf{FOL}^{=}$, including variables and predicate symbols. Sentences follow the grammar for $\mathbf{FOL}^{=}$-sentences using Booleans, quantifiers, and identity, while adding the $\Box$ operator, but leaving out constants and function symbols. Additionally, predicate symbols may be marked as either flexible or rigid. Models are constant-domain first-order Kripke structures, with the usual first-order modal satisfaction. Combining modality with quantification, however, introduces many subtle syntactic and semantic difficulties, see [54, 30, 100] for overviews.

Modality quickly enters the picture when thinking about parthood, most obviously perhaps when analysing temporal parts and thinking about the timeline as a modal dimension [172]. But, more generally, the notions of *essentialism* ('is X a part of Y of necessity?') and *dependence* ('is the existence of an object X, of necessity, dependent on the existence of some other object, Y'? [171]) directly involve modal notions in the analysis of parthood, mostly in combination with first-order quantification (for an overview compare [171]; see also [80]). We here give some of these axioms that are also part of the Dolce ontology, formalised in the modal first-order logic **QS5** introduced above.

Intuitively, rephrasing [129], the Dolce ontology distinguishes between qualities (e.g., the colour of a specific rose), and its 'value' (e.g., a particular shade of red). The latter is called a *quale*, and describes the position of an individual quality within a certain *conceptual space* [59] (called *quality space* in [129]). We refer the reader to [129] for a full discussion and the relevant axiomatisation of the relations $ql_T(t,x)$ (temporal quale) and $ql_S(s,x,t)$ (spatial quale).

The following axioms are part of the Dolce axiomatisation of the theory of *dependence*, re-using the first-order axioms for parthood and proper parthood given earlier, and illustrate the use of modal operators in foundational ontologies.

(Being Present at t)

$$PRE(x,t) \quad :\longleftrightarrow \quad \exists t'.(ql_T(t',x) \wedge P(t,t'))$$

(Being Present in s at t)

$$PRE(x,s,t) \quad :\longleftrightarrow \quad PRE(x,t) \wedge \exists s'.(ql_S(s',x,t) \wedge P(s,s'))$$

(Specific Spatial Dependence)

$$SD_S(x,y) \quad :\longleftrightarrow \quad \Box(\exists t,s.(PRE(x,s,t)) \wedge \forall t',s'.(PRE(x,s',t') \rightarrow PRE(y,s',t')))$$

(Partial Specific Spatial Dependence)

$$PSD_S(x,y) \quad :\longleftrightarrow \quad \Box(\exists t,s.(PRE(x,s,t)) \wedge \forall t',s'.(PRE(x,s',t') \rightarrow \exists s''.(PP(s'',s') \wedge PRE(y,s'',t'))))$$

[26]See [119] for modal logics that explicitly introduce modal operators for the eight RCC8 relations.

The first axiom says that x is present at time t if and only if there is a time t' of which t is a part, and such that x is a temporal quale at t'. The second says that x is present at time t at location s if and only if x is present at t, and there is a location s' of which s is a part and such that x is a spatial quale at s' and t. The third and fourth axiom now also introduce an **S5** modality. x is *specifically spatially dependent* on y if and only if, necessarily (in all possible worlds), there is a time t and a location s such that x is present at time t and location s, and for any time t' and location s', if x if present at t', s' then y is present at t', s'. More informally, this axiom says that, of necessity, y is present at a certain 'space-time location' whenever x is present at that location, thus x can not be present without y being present. Similarly, the fourth one, finally, says the following: x is *partially specifically spatially dependent* on y if and only if, necessarily, there is t and s such that x is present at t, s and, for any t', s' at which x is present there is a location s'' which is a proper part of s' such that y is present at s'', t'. $\dashv$

Example. **Paraconsistent Logics.** In all logics discussed so far, reasoning was *classical* in the sense that all these logics adhere to the classical *principle of explosion*, also known as *ex falsum quod libet.* This principle, which might be rendered as $\bot \models \phi$, for any ϕ, simply maintains that an inconsistency renders the consequence relation $\models$ explosive in the sense that any arbitrary formula ϕ follows.

Paraconsistent logic, which in the most general sense might now be defined as any logic L with a non-explosive consequence relation, has recently gained quite some attention in the ontology engineering community. This attention is particularly directed towards the problems of measuring the *degrees* of inconsistency or incoherence in an ontology [123], and to reasoning [124, 151] and query answering [180, 182] over inconsistent ontologies, or over finite families of ontologies that are jointly inconsistent. A typical scenario where dealing with inconsistency becomes essential is when combination or alignment procedures are applied and result in conflicting information.

Paraconsistent logic might be applied on at least two levels: (i) an ontological domain is considered 'paraconsistent in itself'. Examples might be taken from the branch of paraconsistent logic called *dialetheism*, e.g., from the work, among others, of Graham Priest and Richard Routley, who argued that there can be metaphysically or logically *true inconsistencies* [156]. In this case, clearly, such formal inconsistencies could be taken as genuine ontological axioms of, e.g., a foundational ontology that is grounded in the philosophy of dialetheism. Another example for this case would be ontologies that accept the existence of vague objects [52] (e.g. 'a mountain with vague boundaries') or impossible objects [152] (e.g. 'a round square'). Such objects require a representation formalism that allows to assign (classically) contradictory properties to objects. (ii) the more pragmatic case, and certainly the one that is of more importance for mainstream ontological engineering, is the use of various forms of paraconsistent reasoning for dealing with inconsistent ontologies whilst coping with the problem of logical explosion. As mentioned above, this concerns in the first place the (non-explosive) retrieval of information from classically inconsistent ontologies.

We next sketch how a paraconsistent version of the description logic $\mathcal{ALC}$ based on the four-valued semantics of Belnap [19, 18], namely $\mathcal{ALC}\mathbf{4}$ [124], can be rendered as an institution[27]. The general idea of these logics is to use all subsets of $\{T, F\}$ ($T =$ 'true', $F =$ 'false') as possible truth values, where the truth value $\{T, F\}$ is understood as *overdetermination*, namely being 'both true and false', and $\varnothing$ as *underdetermination*, namely being 'neither true nor false'. A key insight now is that these truth values form a lattice, called $A4$, under a lattice-ordering understood as 'approximates the information in', see Fig. 3 and [18] for details.

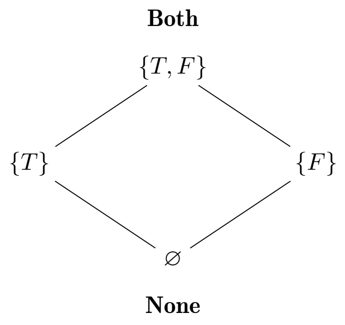

FIGURE 3. The lattice $A4$ of truth values in Belnap's logic

The motivation that Belnap gives for this kind of logical set-up fits particularly well in our argument for logical pluralism [18]:

> [...] it is our impression that hardly any of what individual practitioners of many-valued logic have done is directly concerned with developing logics to use as practical tools for inference. Hence the peculiarity of our task, which is to suggest that a certain four-valued logic ought to be used in certain circumstances as an actual guide to reasoning. Our suggestion for the utility of a four-valued logic is a local one. It is not the Big Claim that we all ought always to use this logic ([...]), but the Small Claim that there are circumstances in which someone—not you—ought to abandon the familiar two-valued logic and use another instead.

The signatures of the description logic $\mathcal{ALC}\mathbf{4}$ are the same as for $\mathcal{ALC}$, and the same holds true for signature morphisms. Moreover, the grammar for complex concepts is the same as for $\mathcal{ALC}$, as well as the shape of Abox sentences.

[27]This logic was first introduced explicitly in [124], but 4-valued semantics have been studied before for terminological logics, e.g. [153, 174, 150].

However, the sentences of $\mathcal{ALC}\mathbf{4}$ now consist of three different kinds of subsumption relations between concepts (i.e. the Tbox), corresponding to the three implication connectives present in four-valued logic. Subsumptions can now be of the forms:

$$(1) \quad C \mapsto D \qquad (2) \quad C \sqsubset D \qquad (3) \quad C \rightarrow D$$

where the sentences of type (1) are called *material inclusion axioms*, (2) are *internal inclusion axioms*, and (3) *strong inclusion axioms*. Sentence translation is defined similarly as in $\mathcal{ALC}$.

The models of $\mathcal{ALC}\mathbf{4}$[28] are again the same single-sorted first-order structures $I = (\Delta^I, .^I)$ used in $\mathcal{ALC}$, with the same interpretation of individual names being mapped to elements of the domain, and roles being mapped to binary relations. However, concepts are interpreted differently. Every atomic concept A is assigned a pair $A^I = \langle A^I_+, A^I_- \rangle$, where $A^I_+, A^I_- \subseteq \Delta^I$. Model reduct is defined similarly as in $\mathcal{ALC}$.

Intuitively, to rephrase [125], A^I_+ is the set of elements known to belong to the extension of a concept A (under $.^I$), while A^I_- is the set of elements known to be *not* contained in the extension of A. In accordance with 4-valued semantics, A^I_+ and A^I_- need not be disjoint, nor mutually complemental with respect to the domain. In other words, if $a \in A^I_+ \cap A^I_-$ we can think of a as both belonging and not belonging to A, whilst if $a \notin A^I_+ \cup A^I_-$, then we are ignorant w.r.t. the question whether a belongs to A, just as illustrated in Fig. 3.

Satisfaction, of course, is defined differently to standard $\mathcal{ALC}$. Rather than giving the full semantics we give the truth conditions for conjunction, negation, existential restriction, and internal inclusion only to illustrate that the semantics easily fits into the framework of institution theory. Let $.^I$ be a four-valued $\mathcal{ALC}$ interpretation with domain Δ^I. As explained above, atomic concepts A are interpreted as pairs $\langle A^I_+, A^I_- \rangle$ with $A^I_{\{+,-\}} \subseteq \Delta^I$. This mapping is inductively extended as follows. First, we set $\top^I = \langle \Delta^I, \varnothing \rangle$ and $\bot^I = \langle \varnothing, \Delta^I \rangle$. Next we set for conjunction, negation and existential restriction:

conj.	$(C \sqcap D)^I$	$=$	$\langle C_+ \cap D_+, C_- \cup D_- \rangle$
neg.	$(\neg C)^I$	$=$	$\langle C_-, C_+ \rangle$
restr.	$(\exists R.C)^I$	$=$	$\langle \{x \mid \exists y.(x,y) \in R^I \text{ and } y \in C^I_+\}, \{x \mid \forall y.(x,y) \in R^I \text{ implies } y \in C^I_-\} \rangle$

The truth conditions for inclusion statements are now as follows. Note that the satisfaction condition for institutions can be shown in a straightforward way.

material inclusion	$I \models C \mapsto D$	$\Longleftrightarrow$	$\Delta^I \setminus C_- \subseteq D_+$
internal inclusion	$I \models C \sqsubset D$	$\Longleftrightarrow$	$C_+ \subseteq D_+$
strong inclusion	$I \models C \rightarrow D$	$\Longleftrightarrow$	$C_+ \subseteq D_+$ and $D_- \subseteq C_-$

[28]We here present the simplified version of the semantics of $\mathcal{ALC}\mathbf{4}$ as given in [122] (since $\mathcal{ALC}$ has no negation on roles we can assume classical semantics for these); this paper also shows that the semantics can be adapted to more expressive DLs up to $\mathcal{SROIQ}$.

These three inclusions have thus different formal properties and are each applicable only in specific cases. This analysis will typically have to be performed by a domain expert. For instance, strong inclusion respects the deduction theorem and contraposition reasoning. In a paraconsistent context, this inclusion might be preferred for cases where a 'universal truth' needs to be formalised, such as $\mathsf{Human} \mapsto \mathsf{Mortal}$, or indeed $\exists \mathit{isProperPartof}.X \mapsto \exists \mathit{isPartof}.X$. Internal inclusion, on the other hand, propagates contradictory information forward, but not backward as it does not allow for contraposition reasoning. As [124] argue, it could be characterized as a 'brave way' of handling inconsistency: if it is important to infer the consequent even if the antecedent may be contradictory, internal inclusion can be used. An obvious example are safety-critical systems where the consequent of an inclusion should yield a safety check regardless of inconsistent information in a certain situation. An example, taken from [124] which contains more extensive discussion and examples, would be $\mathsf{OilLeakage} \sqsubset \mathsf{RobotMalfunction}$.

An important feature of Belnap's semantics is the fact that reasoning over four-valued models can be reduced to classical reasoning, thus making it possible to use existing reasoning algorithms for paraconsistent entailment. This applies both to Belnap's original logic [162], as well as to the various DL variants [122].

$\dashv$

2.1.3. Higher-Order Logics.

Example. **Higher-Order Logics.** While there are many different first-order logics, there tend to be even more different higher-order logics. Concerning untyped variants, Common Logic can be seen as kind of simulation of untyped higher-order logic within a first-order framework; however, it does not feature λ-abstraction. Concerning typed variants, [28] presents an institution for a higher-order logic extending Church's type theory [36] with polymorphism; this is basically the higher-order logic used in modern interactive theorem provers like Isabelle/HOL [148]. A feature of Isabelle/HOL that is not covered by this logic is type classes; the institution of HasCasl [165] is a polymorphic higher-order logic featuring type classes, as well as partial functions and subsorting; moreover, it is intuitionistic.

We continue the example of mereology phrased in monadic second-order logic. David Lewis has investigated a mereological reconstruction of set-theory [114]. Here, the parthood relation is used to identify the singleton sets as the *smallest parts* of any non-empty set, and it is argued that the re-construction of set theory using plural quantification and mereology is ontologically 'more innocent' than the standard iterative conception of sets, resulting in an axiomatisation resembling the Peano system for arithmetic.

We give a part of this reformulation of set-theory based on an axiomatisation by [161] and [85] given in monadic second-order logic. First, recall the definition of second-order fusion given on page 20:

$$Fus(X, a) :\longleftrightarrow \forall y.(Ov(y, a) \leftrightarrow \exists z.(X(z) \wedge Ov(z, a))$$

This definition may be extended by axioms that use quantification over subsets (or unary relations) and is used to define a theory 'Mereology plus Singletons' which has only two basic relations, namely the parthood relation (i.e. the subset relation), and a singleton relation $singl(x, y)$ saying that y is a singleton of x. First, assume the first-order axioms for the parthood relation given on page 18. On top of these we have the following comprehension schema, where $\bigwedge, \bigvee$ are the universal/existential monadic second-order quantifiers and $\forall, \exists$ the usual first-order quantifiers. For any formula ϕ we assume as axioms:

$$\textbf{Compr}(\phi): \quad (\exists x.\phi(x) \to \bigvee P.\ \forall x.(P(x) \leftrightarrow \phi(x))$$

The next axiom says that, for any unary relation (set) P, if P has a member, then the fusion y of P exists.

$$\bigwedge P.\ (\exists x.P(x) \to \exists y.(Fus(P, y))$$

Finally, we can enforce the uniqueness of the fusion as follows.

$$\bigwedge P.\ \forall x, y.(Fus(P, x) \wedge Fus(P, y) \to x = y)$$

From these axioms, it can now already be shown, using comprehension, that there exists a greatest entity, denoted by W, containing all entities as a part. Moreover we may prove the existence and uniqueness of the mereological sum, of the intersection and the relative complement—see [85] for details. ⊣

Little work has been devoted to the general problem of *translation* between ontologies formulated in different logical languages and/or vocabularies. One such approach is given in [49], who discuss translations between $\mathcal{OWL}$ ontologies. They use so-called bridging axioms (formulated in first-order) to relate the meaning of terms in different ontologies,[29] and present an algorithm to find such translations.

In the following, we introduce the fundamentals of what is a very general solution with strong theoretical foundations to the problem of theory and logic translation. Based on institution theory, this completely heterogeneous framework allows to specify translations (so-called comorphisms) between any ontologies that are formulated in logics which can be rendered as institutions.

2.2. Structured Ontologies

As mentioned in the introduction, in ontology design and engineering, several needs for structuring mechanisms arise: firstly, the size of ontologies calls for methods of combining and re-using ontology components, or modules. Secondly, adopting logical pluralism calls for a method of heterogeneous ontology integration; a problem that also is practically important, e.g. when aligning a domain ontology written in a lightweight logic with a foundational ontology written in a more expressive logic. Both problems can be solved with means of the theory of institutions and formal structuring techniques from algebraic specification theory.

[29]Not to be confused with the 'bridge axioms' in DDL [27].

The essential advantage of the theory of institutions is the possibility of providing structuring operations and module concepts *independently of the underlying logical system.* Hence, in the sequel, let us fix some arbitrary institution $I = (\mathbf{Sign}, \mathbf{Sen}, \mathbf{Mod}, \models)$. The basic structuring operation for ontologies is surely that of *importing* other ontologies. The notion of *development graph* captures this, and also renaming of symbols.

Definition 2.2. A **development graph** is an acyclic, directed graph[30] $\mathcal{DG} = \langle \mathcal{N}, \mathcal{L} \rangle$. Here, $\mathcal{N}$ is a set of nodes. Each node $N \in \mathcal{N}$ is labelled with a pair (Σ^N, Ψ^N) such that Σ^N is a signature and $\Psi^N \subseteq \mathbf{Sen}(\Sigma^N)$ is the set of **local axioms** of N. $\mathcal{L}$ is a set of directed links, so-called (global[31]) **definition links** $K \xrightarrow{\sigma} N)$, annotated with a signature morphism $\sigma : \Sigma^K \to \Sigma^N$. There are also **hiding definition links** $K \xrightarrow[h]{\sigma} N)$, annotated with a signature morphism $\sigma : \Sigma^N \to \Sigma^K$ *going against the direction of the link.*

Given a node $N \in \mathcal{N}$, its associated class $\mathbf{Mod}_{\mathcal{DG}}(N)$ of models (or N-models for short) is inductively defined to consist of those σ^N-models M for which

- M satisfies the local axioms Ψ^N, and
- for each $K \xrightarrow{\sigma} N \in \mathcal{DG}$, $M{\restriction_\sigma}$ is an K-model, and
- for each $K \xrightarrow[h]{\sigma} N \in \mathcal{DG}$, M has a σ-expansion M' (i.e. $M'{\restriction_\sigma}= M$) that is a K-model.

$\dashv$

This model-based semantics can be complemented by a theory-based semantics: Given a node $N \in \mathcal{N}$, its associated theory $\mathbf{Th}_{\mathcal{DG}}(N)$ is inductively defined to consist of

- all the local axioms Ψ^N, and
- for each $K \xrightarrow{\sigma} N \in \mathcal{DG}$, all of $\mathbf{Th}_{\mathcal{DG}}(K)$ translated by σ.

Note that the theory of a node only partially captures its semantics; $\mathbf{Mod}_{\mathcal{DG}}(N)$ is always a subset of $\mathbf{Mod}(\mathbf{Th}_{\mathcal{DG}}(N))$, which (in the presence of hiding) can be proper.

Complementary to definition links, which *define* the theories of related nodes, we also allow **theorem links** with the help of which we are able to *postulate* relations between different theories. A (global) theorem link is an edge $K \overset{\sigma}{\dashrightarrow} N$, where $\sigma\colon \Sigma^K \longrightarrow \Sigma^N$. $\mathcal{DG}$ implies a theorem link $K \overset{\sigma}{\dashrightarrow} N$ (denoted $\mathcal{DG} \models K \overset{\sigma}{\dashrightarrow} N$) iff for all $M \in \mathbf{Mod}_{\mathcal{DG}}(N)$, $M{\restriction_\sigma} \in \mathbf{Mod}_{\mathcal{DG}}(K)$.

[30]In [107, 108], we have identified a structured ontology with a diagram in the sense of category theory. Actually, we often use "graph" and "diagram" interchangeably, though technically they are defined is slightly different ways.

[31]There are also local definition and theorem links, which are not needed here.

A global definition (or also theorem) link $K \xrightarrow{\sigma} N$ can be strengthened to a **conservative extension link** (denoted as $K \xrightarrow[cons]{\sigma} N$); it holds if every K-model has a σ-expansion to an N-model. Such annotations can be seen as another kind of proof obligations. Definitional and monomorphic extensions are introduced in a similar way, annotated with *def* (*mono*). For definitional extensions, the σ-expansion has to be unique. An extension is monomorphic, if it is conservative and moreover, any isomorphism $h : A\restriction_\sigma \to B\restriction_\sigma$ between reducts of N-models has a σ-expansion to an isomorphism between A and B. In particular, this implies that any K-model has a unique σ-expansion up to isomorphism.[32]

Many languages for structuring, modularity and alignment of ontologies can be mapped into this formalism of development graphs.

Example ($\mathcal{OWL}$ ontologies as development graphs). The only explicit structuring mechanism of $\mathcal{OWL}$ ontologies is the `imports` construct. Assume there is a set of $\mathcal{OWL}$ ontologies with some import relations among them, such that the graph of import relations is acyclic. Each ontology O in this set leads to a node N^O in the development graph, such that N^O is labelled with a pair $(\Sigma^{N^O}, \Psi^{N^O})$, where Σ^{N^O} is the local signature of the ontology O) and Ψ^{N^O} is the set of its axioms (assertions), i.e. ABox, TBox, and RBox statements.

Moreover, for every import statement importing ontology P into ontology O, there is a directed definition link $N^P \xrightarrow{\sigma} N^O$. Each such link is annotated with a signature morphism $\sigma : \Sigma^{N^P} \to \Sigma^{N^O}$. According to the $\mathcal{OWL}$ semantics, σ is always the signature inclusion map, i.e. sentences are not changed when translated along σ.

$\mathcal{OWL}$ also allows cyclic imports. These can be mapped to development graphs as follows. Besides the nodes for the individual ontologies, the development graph also contains one node for each strongly connected component[33] of the ontology import graph; it contains no local axioms. If ontology O imports ontology P and P is not in the strongly connected component of O, this leads to a definition link $N^{C(P)} \longrightarrow N^O$, where $C(P)$ is the strongly connected component of P, and $N^{C(P)}$ is the corresponding node. Moreover, for each ontology O, there is a definition link $N^{C(O)} \longrightarrow N^O$. As an example, suppose that ontologies P_1 and P_2 import each other, and so do O_1 and O_2,

[32]The latter condition is used to define monomorphicity in [38]. For transportable [1] reduct functors, both conditions are equivalent.

[33]By 'strongly connected component' we mean the usual graph-theoretic notion, i.e. a directed graph G is strongly connected if there is a path from each vertex in the graph to every other vertex. The strongly connected components of G are its maximal strongly connected subgraphs. If each strongly connected component is contracted to a single vertex, the resulting graph is a directed acyclic graph.

and moreover, O_1 imports P_1. Then we get the following development graph:

$$\begin{tikzcd}
P_1 \arrow[dr] & & P_2 \arrow[dl] \\
& C_1 \arrow[dl] & \\
O_1 \arrow[dr] & & O_2 \arrow[dl] \\
& C_2 &
\end{tikzcd}$$

where C_1 is the strongly connected component for P_1 and P_2, while C_2 is that of O_1 and O_2. Note that while the resulting development graph deviates from the original import graph, it is much better suited for both semantic analysis and proof management.

Note that plain $\mathcal{OWL}$ ontologies do not make use of theorem links, and in particular of definitional or conservative extension links. We will discuss their use in more detail when embedding $\mathcal{OWL}$ ontologies in the heterogeneous HetCasl environment introduced below.

2.3. Heterogeneous Ontologies

In Section 1, we brought forward the argument that since ontologies are being written in many different formalisms, like relation schemata, description logics, first-order logic, and modal logics, we should adopt the position of the logical pluralist in ontology design and treat the notion of *heterogeneity* as a first-class citizen in ontology design methodology.

Similarly, [164] have argued convincingly that in such a set-up, ontology-based semantic integration and semantic interoperability is highly desirable, and that a framework that successfully captures semantic integration despite the different existing treatments of semantics should based on institutions linked by suitable translations.

This heterogeneous semantic integration relies on some given graph of logics and logic translations, which we will formalise as institutions and so-called institution comorphisms, see [68, 143]:

Definition 2.3 (Institution Comorphism). Given two institutions I and J with $I = (\mathbf{Sign}, \mathbf{Mod}, \mathbf{Sen}, \models)$ and $J = \big(\mathbf{Sign}', \mathbf{Mod}', \mathbf{Sen}', \models'\big)$, an **institution comorphism** from I to J consists of a functor $\Phi : \mathbf{Sign} \longrightarrow \mathbf{Sign}'$, and natural transformations $\beta : \mathbf{Mod} \Longrightarrow \mathbf{Mod}' \circ \Phi$ and $\alpha : \mathbf{Sen} \Longrightarrow \mathbf{Sen}' \circ \Phi$, such that the *satisfaction condition*

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

holds.

Here, $\Phi(\Sigma)$ is the translation of signature Σ from institution I to institution J, $\alpha_\Sigma(\varphi)$ is the translation of the Σ-sentence φ to a $\Phi(\Sigma)$-sentence, and $\beta_\Sigma(M')$ is the translation (or perhaps better: reduction) of the $\Phi(\Sigma)$-model M' to a Σ-model.

As an example, consider the well-known translation of $\mathcal{OWL}$ into untyped first-order logic, mapping concepts to unary and roles to binary predicates. It can easily be organised as an institution comorphism.

A **subinstitution** [132] is an institution comorphism with Φ an embedding of categories, α_Σ injective and β_Σ an isomorphism for each Σ. For example, propositional logic, and via the above described comorphism, also $\mathcal{OWL}$, are subinstitutions of untyped first-order logic.

The so-called **Grothendieck institution** is a technical device for giving a semantics to heterogeneous theories involving several institution (see [45, 135]). The Grothendieck institution is basically a flattening, or disjoint union, of a logic graph. Fix an arbitrary graph of institutions and institution comorphisms. A signature in the Grothendieck institution over this graph consists of a pair (L, Σ) where L is a logic (formalised as an institution) and Σ is a signature in the logic L. Similarly, a Grothendieck signature morphism $(\rho, \sigma) : (L_1, \Sigma_1) \rightarrow (L_2, \Sigma_2)$ consists of a logic translation (formalised as institution comorphism) $\rho = (\Phi, \alpha, \beta)\colon L_1 \longrightarrow L_2$ plus an L_2-signature morphism $\sigma\colon \Phi(\Sigma_1) \longrightarrow \Sigma_2$. Sentences, models and satisfaction in the Grothendieck institution are defined in a componentwise manner.
From [135] we know:

Lemma 2.4. *A Grothendieck signature morphism can be split into a comorphism (i.e. Grothendieck signature morphism (ρ, id)) followed by a homogeneous signature morphism (i.e. Grothendieck signature morphism (id, σ)).*

We now arrive at the following:

Definition 2.5. An **abstract structured heterogeneous ontology** (w.r.t. some logic graph) is a node O in a development graph $\mathcal{DG}$ in the corresponding Grothendieck institution. We sometimes also identify O with its theory $\mathbf{Th}_{\mathcal{DG}}(O)$; however, note that then the structuring is lost. $\dashv$

In order to understand this definition better, let us discuss it in a bit more detail. An abstract structured heterogeneous ontology consists of individual ontologies sitting at the nodes of the development graph, such that for each ontology, the logic it is formulated in is recorded in the node. The import relations among the ontologies are given by the links of the graph. Typically, most links will be homogeneous, that is, they are decorated with Grothendieck signature morphisms of form (id, σ) that do not change the logic (id is the identity institution comorphism). Some of the links will run between ontologies written in different logics, hence they are decorated with Grothendieck signature morphisms of form (ρ, id), where ρ is an institution comorphism. The general format (ρ, σ) will typically occur only when composing homogeneous and heterogeneous links.

To be able to write down such heterogeneous ontologies in a concise manner, we use the language HetCasl. It is a heterogeneous extension of the Common Algebraic Specification Language Casl [24, 38]. Casl is an expressive specification language that has been designed by CoFI (`http://www.cofi.info`), an open international initiative. Its aim is to supersede many existing algebraic specification languages and provide a standard. Casl consists of several layers, including basic (unstructured) specifications, structured specifications and architectural specifications (the latter are used to prescribe the modular structure of implementations), as well as specification libraries. A crucial feature of CASL is that all of its layers except from that of basic specifications are *institution independent*.

Fig. 4 shows a simple subset of the HETCASL syntax. Due to the historical origins of CASL, namely formal methods for software development, logical theories are called *specifications*; for our purposes, we can regard the expression 'specification' synonymous with 'ontology'.

```
SPEC ::= BASIC-SPEC
       | SPEC then SPEC
       | SPEC then %implies SPEC
       | SPEC then %cons SPEC
       | SPEC then %mono SPEC
       | SPEC then %def SPEC
       | SPEC with SYMBOL-MAP
       | SPEC with logic ID
       | SPEC hide SYMBOL-LIST
       | ID

DEFINITION ::= logic ID
             | spec ID = SPEC end
             | view ID : SPEC to SPEC = SYMBOL-MAP end
             | view ID : SPEC to SPEC = with logic ID end

LIBRARY = DEFINITION*
```

FIGURE 4. Syntax of a simple subset of the heterogeneous specification language. `BASIC-SPEC` and `SYMBOL-MAP` have a logic specific syntax, while `ID` stands for some form of identifiers.

A a basic specification `BASIC-SPEC` is an unstructured presentation of a signature and list of axioms in some institution, using syntax specific to the institution. It leads to a node with local axioms in a development graph. An extension `SPEC1 then SPEC2` imports `SPEC1` into `SPEC2`. It leads to a global definition link from the node for `SPEC1` to that for `SPEC2`, decorated with an inclusion signature morphism. Extensions thus may add new signature elements and axioms to existing specifications. They can be declared to be implied by the first specification, or to be conservative, monomorphic or definitional, written `SPEC1 then %implies SPEC2`, `SPEC1 then %cons SPEC2`, `SPEC1 then %mono SPEC2` and `SPEC1 then %def SPEC2`, respectively. The latter three annotations lead to corresponding annotations at the definition link from `SPEC1` to `SPEC2`, while the former annotation lead to a theorem link in the opposite direction. Renamings, written `SPEC with SYMBOL-MAP`, rename a specification along a signature morphism (given by some symbol map with institution-specific syntax); again, this leads to a definition link in the development graph. Similarly, a specification can be translated along an institution comorphism, written `SPEC with logic ID`. Finally, with the notation `SPEC hide SYMBOL-LIST`, some symbols can be hidden from a specification, which amounts to restricting it to an export interface. This leads to a hiding definition link in the development graph (the signature of which is computed from the symbol list).

Such specifications can be used in global definitions, which are then collected into libraries. A definition `spec ID = SPEC end` stores specification `SPEC` under name `ID` for later concise reference under this name. The current logic can be changed with `logic ID`; this fixes the institution of the following specifications until that keyword occurs again. With `view ID : SPEC1 to SPEC2 = SYMBOL-MAP end`, a so-called view (also known as theory morphism or interpretation of theories) from `SPEC1` into `SPEC2` is introduced and named `ID`; its signature morphism is given by the symbol map. This generates a theorem link between the nodes representing `SPEC1` and `SPEC2` in the development graph. Finally, views of form `view ID : SPEC to SPEC = with logic ID end` are heterogeneous; here, `ID` is the name of the institution comorphism. Again, a theorem link is generated.

Details of the translation to development graphs, as well as a proof calculus, can be found in [138]. The $\mathcal{OWL}$ example introduced on Page 22 looks as follows in HetCasl notation (using Manchester syntax [89] for $\mathcal{OWL}$):

```
logic OWL
spec Biblio_OWL =
  Class: Researcher
    SubclassOf: name some Thing
  Class: Article
    SubclassOf: author some Thing, title some Thing
  Class: Journal
    SubclassOf: name some Thing, hasArticle some Thing,
                impactFactor some Thing
end
```

Also, the extended examples in Section 4 provide a look-and-feel of HetCasl specifications. Of course, abstract structured heterogeneous ontologies can be formulated in different notations, and HetCasl is only one of them. Another option would be an extension of $\mathcal{OWL}$'s structuring mechanisms by keywords dealing with heterogeneity (compare [105] for a discussion). We can also use the HetCasl constructs to present homogeneous structured ontologies e.g. in $\mathcal{OWL}$, let us call the thus extended language hOWL.

The notion of conservative extension is a rather important technical concept to define a notion of ontological module (see e.g. [118, 40]). The definition of conservative extension link above uses the model theoretic definition of conservativity, roughly stated as: O_2 is a conservative extension of O_1 if every O_1-model can be expanded to an O_2-model. Note that this implies (and is strictly stronger than) proof-theoretic (or better consequence-theoretic) conservativity: all sentences in the signature of O_1 that are provable in O_2 are in fact already provable in O_1. For further discussion and for the closely related notions of interpolation and amalgamation, and the interaction with colimits see e.g. [107, 108]. In [155], the interaction of conservativity with ontology query languages is studied.

Note, however, that the use of conservativity as a structuring mechanism has a rather different flavour to its use as an approach to *creating* modules. The latter notion is not so much about writing down ontologies in a structured way (which is our main concern here), but more about taking a given complex ontology and cutting slices out of it, according to the needs of a particular application scenario which determines a certain, often small set of concepts (and relations) from the original ontology. Note that these slices are typically not 'natural' *building blocks* of the overall ontology; to the contrary, they will often involve parts from different such building blocks.

The concept of definitional extension generalises the well-known concept of extension by definitions to an arbitrary institution. Recall that O_2 is a definitional extension of O_1 if any O_1-model has a unique expansion to an O_2-model. Note that this means that model reduct is a bijection between O_1-models and O_2-models. Intuitively spoken, O_2 adds neither additional constraints nor additional freedom of interpretation to O_1, but rather the new symbols in O_2 are uniquely defined in terms of the symbols in O_1. While usually, definitional extensions are required to be explicit, the notion used here also covers implicit definitions. (Note that for first-order logic, by the Beth definability theorem, every implicit definition can be made explicit. See [46] for conditions under which this generalises to other institutions and [177] for results concerning description logics.)

Sometimes, the notion of definitional extension is too strong for capturing definitions (even explicit ones). For example, in many-sorted first-order logic, new sorts can be defined in terms of old ones. However, such a definition specifies a model expansion only uniquely up to isomorphism, and hence the notion of monomorphic extensions fits here.

3. Refinement, Integration, and Connection

The magnitude and diversity of existing ontologies call for the need to relate and combine given ontologies. Two ontologies can be related in several different ways. Focusing on the structural aspects, we see the following three main possibilities:

1. Two ontologies are related by specifying a mapping between them that 'translates' one into the other. We call this **refinement**, studied in Section 3.1.
2. Two ontologies are related by mapping (typically embedding) them into a third *existing* reference ontology. We call this **integration** (of two ontologies into a third). We study this in Section 3.2.
3. Two ontologies are related by specifying some interface ontology, which is typically mapped into the two given ontologies. The interface can then be used to *generate* an overall ontology. We call this **connection**, which is studied in Section 3.3.

Integrations and connections are essentially *symmetric* combination techniques in the sense that the order in which component ontologies participate in the overall combination is irrelevant. The difference between the two lies in the way ‘local’ signatures are mapped into the overall signature. In contrast to this, a heterogeneous refinement is an *asymmetric* technique, stating that all axioms of a ‘coarser’ ontology are also true in a ‘finer’ (refined) one.

3.1. Refinements, Sub-Ontologies, and Equivalence

Refinements are well-known in specification theory [5]; here, we generalise this notion for ontological purposes. Refinements correspond to some form of logical consequence: roughly, a refinement expresses that the target ontology is logically implied by the source theory of the refinement. Refinements are related to interpretations of theories in logic; the term refinement stresses the fact that the target ontology may have a stronger axiomatization and thus may make finer distinctions than the source ontology.

3.1.1. Heterogeneous Refinements. We start with an easy but preliminary definition:

Definition 3.1. Given two ontologies O_1 and O_2 in the same logic, O_2 is called a **standard refinement** of O_1 if there is a theorem link $O_1 \xrightarrow{\sigma} O_2$ that follows from the underlying development graph.

This definition is standard in specification theory. Moreover, there is an easy generalisation to the heterogeneous case: just allow a heterogeneous theorem link in the above definition. However, this notion is too narrow in some cases. In particular, we mostly have chosen signature morphisms in a way that predicates are mapped to predicates, constants are mapped to constants, while a more general notion of *derived* signature morphism [163] would allow mapping predicates to formulas and constants to terms. Since we do not want to enter into the specific details of any institution here, we abstractly capture a derived signature morphism as an ordinary signature morphism into a monomorphic extension. The monomorphic extension would then provide definitions of new predicates (and sorts), constants, etc. in terms of formulas, terms, etc. We hence arrive at:

Definition 3.2. Given two ontologies O_1 and O_2, O_2 is called a **refinement** of O_1 if there is a monomorphic extension O_2' of O_2 and a theorem link $O_1 \xrightarrow{\sigma} O_2'$ that follows from the underlying development graph. If the theorem link is moreover conservative, the refinement is called **conservative** as well.

$$\begin{tikzcd}
 & O_2' & \\
O_1 \arrow[ur, dotted, "\sigma"] & & O_2 \arrow[ul, "\theta"', "mono"]
\end{tikzcd}$$

$\dashv$

In HetCasl, this concept can be expressed by writing a `view` into a monomorphic extension. The standard definition of refinement is recovered by taking θ to be the identity. See Section 4 for reasoning about refinement.

(Conservative) refinements are closely related to (faithful) interpretations of theories for untyped first-order logic in the sense of [50, 73]. Indeed, the main difference is that interpretations of theories may relativise the universe of discourse to a predicate; a construction which is difficult to generalise to the institution independent level. In many-sorted first-order logic, this difficulty does not arise: here, monomorphic extensions may introduce new sorts which are defined using predicates on old sorts.

Note that our definition is very general: the monomorphic extension can also change the logic (as long as it does this in a monomorphic way, e.g. by moving to a super-logic). An example where this is useful is given in Fig. 17, where source and target of the refinement live in different logics. These two logics might be incomparable as concerns expressive power. Consider, as a simple example, two different description logics with specialised expressive means. Then the logic of the source ontology might, say, be able to express transitive roles, while the logic of the target ontology does not, but the latter might have inverse roles which the former might not have. In such a case, the monomorphic extension of the target can be used to reach a common super-logic of both the source and the target logic. Note that it is always possible to pre-compose the refinement signature morphism with an institution comorphism moving the source ontology into a different logic; hence, a monomorphic extension of the source is not needed.
Despite the extra generality, some useful properties of standard refinements still hold:

Proposition 3.3. *For a heterogeneous refinement*

$$\begin{tikzcd} O_1 \arrow[r, dotted, "\sigma"] & O_2' & O_2 \arrow[l, "\theta", "mono"'] \end{tikzcd}$$

1. *any O_2-model can be translated to an O_1-model;*
2. *logical consequence is preserved along refinement:* $O_1 \models \varphi$ *implies* $O_2 \models \theta^{-1}(\sigma(\varphi))$*;*
3. *for conservative refinements, any O_1-model can be translated to an O_2-model;*
4. *the target of a conservative refinement has at least as many non-isomorphic models as the source.*

Proof. 1. Since θ is monomorphic, any O_2-model has a θ-expansion to an O_2'-model. The latter can be reduced via σ to an O_1-model.
2. Let $\psi \in \theta^{-1}(\sigma(\varphi))$, that is, $\theta(\psi) = \sigma(\varphi)$. By the definition of theorem link, $O_1 \models \varphi$ implies $O_2' \models \sigma(\varphi)$, hence $O_2' \models \theta(\psi)$. Since monomorphic extensions are in particular conservative, $O_2 \models \psi$.
3. We define a mapping from O_1-models to O_2-models as follows: given an O_1-model A_1, let A' be some σ-expansion to O' existing by conservativity (since this is not unique, we need to make some arbitrary choice here). Then $A_2 := A'{\restriction}_\theta$ is the desired O_2-model.
4. The map defined under 3 is injective on isomorphism classes: Given O_1-models A_1 and B_1 and their σ-expansions A' and B', if $A_2 \cong B_2$, i.e.

$A' \restriction_\theta \cong B' \restriction_\theta$, since $\theta : O_2 \to O_2'$ is monomorphic, $A' \cong B'$, and hence also $A' \restriction_\sigma \cong B' \restriction_\sigma$, which is $A_1 \cong B_1$.

$\dashv$

Example (Refining Description Logic Parthood). Recall the example of the parthood relation being axiomatised in ontology languages of varying expressivity, from propositional, to description logic, first- and second-order, and finally modal logic. Clearly, from an intuitive point of view we would like, e.g. the first-order theory of parthood to heterogeneously refine the description logic formalisation. That understanding the relationship between different such formalisation is important has been realised already e.g. in [25]: they introduce and differentiate first-order theories of parthood, componenthood, and containment relations, and then study how description logics can capture or approximate the first-order properties. In particular, [175] has used complex role-inclusion axioms available in a logic such as $\mathcal{SROIQ}$ to properly capture parthood relations in medical ontologies (rather than 'simulating' their semantics, see [166, 167]).

Establishing a heterogeneous refinement is a precise formulation of this endeavour. Recall the axiomatisations of *parthood* and *proper parthood* given in DL (page 22) and first-order logic (page 18). First, note that whilst proper parthood is a *defined* predicate PP in the first-order axiomatisation, the roles *isProperPartof* and *isPartof* are independently axiomatised in DL because of lack of quantification machinery and the impossibility to completely capture the parthood relation in the DL $\mathcal{SROIQ}$. To establish the heterogeneous refinement, we in fact do not need a monomorphic extension in this case; the signature morphism σ simply maps σ : *isPartof* $\mapsto P$ and σ : *isProperPartof* $\mapsto PP$, where the comorphism is based on the standard translation from DL to first-order logic.

This refinement is non-trivial in the sense that the translated DL axioms are not part of the first-order axiomatisation; rather, it has to be established, for instance, that the first-order axioms together with the definition of proper parthood entail the translation of the complex role inclusions. As an example, consider the following complex role inclusion

$$\mathit{isPartOf} \circ \mathit{isProperPartOf} \sqsubseteq \mathit{isProperPartOf}$$

the translation of which to single-sorted first-order logic along the standard translation is

$$\forall x, y, z \,.\, \big(P(x,y) \wedge PP(y,z) \to PP(x,z)\big)$$

This is readily proved using the first-order definition of proper parthood and the transitivity axiom for parthood.[34] Note that the theorem link in

[34]A proof of this in the Hets system using the Spass prover takes less than a second presenting a proof with 16 derivation steps.

this heterogeneous refinement is not conservative, more precisely not proof-theoretically conservative (and therefore also not model-theoretically conservative), i.e. the first-order axiomatisation proves new facts about the translated symbols *isPartOf*, *isProperPartOf*. The arguably simplest example for this might be the *antisymmetry* axiom

$$\forall x, y \,.\, \big(P(x,y) \wedge P(y,x) \rightarrow x = y\big)$$

Indeed, for a relation P, the class of P-irreflexive and antisymmetric models can not be finitely axiomatised in $\mathcal{SROIQ}$ [90, 94]. ⊣

Let us come to a more general discussion of refinements related to the ontology Dolce, a foundational ontology developed by the Laboratory for Applied Ontology (LOA) and originally specified in first-order logic. The general complexity of the Dolce ontology stems from the fact that it combines several (non-trivial) formalised ontological theories into one theory, viz. the theories of essence and identity, parts and wholes (mereology), dependence, composition and constitution, as well as properties and qualities. Several versions of Dolce have been designed that are expressible in description logics of varying expressive power, such as Dolce-Lite, which is an $\mathcal{OWL}$-DL ontology.[35]

Firstly, we might want to establish whether a certain domain ontology, O, written in $\mathcal{OWL}$-DL, is consistent with respect to the knowledge represented in this foundational ontology, that is, we want to check whether O refines the abstract knowledge given in Dolce. To establish this, Dolce's axioms, appropriately translated, have to be entailed by O. If we do this w.r.t. Dolce-Lite, we are trying to establish a homogeneous refinement.

Secondly, given that there are different versions of Dolce itself, the less expressive versions of Dolce should not only inherit some of the general modelling principles of the 'full' Dolce version, but, ideally, the full Dolce version should be a heterogeneous refinement of these less expressive ones. However, since the existing DL versions are essentially hand-crafted approximations, it should not be too surprising that the actual logical relationships between these versions of different expressivity are more complex than straightforward refinements. Whilst for the part of the $\mathcal{OWL}$-DL version whose signature has a direct correspondent in the **FOL** version (e.g. the corresponding **FOL** axioms are simply obtained by the standard translation) a direct refinement can be easily established, this is *not* the case for the parts that are 'properly approximated'. An example is given by several predicates that are temporalised in Dolce, and where the temporal aspect is simply deleted in the $\mathcal{OWL}$-DL version, another example cases where ternary predicates have to be approximated by unary and binary predicates available in $\mathcal{OWL}$ (see also [110]). Note that, generally, finding the largest $\mathcal{OWL}$ approximation of a first-order theory is non-trivial.

[35] See `http://wiki.loa-cnr.it/index.php/LoaWiki:Ontologies`

We expect that, when the present framework is properly applied to Dolce, the relation between DOLCE and domain ontologies will be as follows:

$$\begin{tikzcd}
\text{Dolce-Lite} \arrow[r, dotted] \arrow[d] & \text{Dolce} \arrow[d] \\
\text{domain ontology (DL)} & \text{domain ontology (FOL)}
\end{tikzcd}$$

We now come to composition of refinements. We need a preparatory notion:

Definition 3.4. An institution is *semi-exact* [47], if **Sign** has pushouts, and moreover, the model functor takes any pushout

$$\begin{tikzcd}
\Sigma \arrow[r] \arrow[d] & \Sigma_1 \arrow[d] \\
\Sigma_2 \arrow[r] & \Sigma_R
\end{tikzcd}$$

in **Sign** to a pullback of categories

$$\begin{tikzcd}
\mathbf{Mod}(\Sigma) & \mathbf{Mod}(\Sigma_1) \arrow[l] \\
\mathbf{Mod}(\Sigma_2) \arrow[u] & \mathbf{Mod}(\Sigma_R) \arrow[l] \arrow[u]
\end{tikzcd}$$

of categories. Explicitly, this means that any pair $(M_1, M_2) \in \mathbf{Mod}(\Sigma_1) \times \mathbf{Mod}(\Sigma_2)$ that is *compatible* in the sense that M_1 and M_2 reduce to the same Σ-model can be *amalgamated* to a unique Σ_R-model M (i.e., there exists a $M \in \mathbf{Mod}(\Sigma_R)$ that reduces to M_1 and M_2, respectively), and similarly for model morphisms.

All institutions introduced so far except from HasCasl are semi-exact. See [46, 136] for conditions under which Grothendieck institutions are semi-exact.

Proposition 3.5. *In semi-exact institutions, heterogeneous refinements compose.*

Proof. The composition is as follows, where the rhombus is a pushout:

$$\begin{tikzcd}
 & & & O' & & & \\
 & O'_2 \arrow[urr] & & & & O'_3 \arrow[ull, "mono"'] & \\
O_1 \arrow[ur, dotted, "\sigma_1"] & & & O_2 \arrow[ull, "\theta_1", "mono"'] \arrow[urr, dotted, "\sigma_2"] & & & O_3 \arrow[ul, "\theta_2", "mono"']
\end{tikzcd}$$

By semi-exactness, monomorphicity lifts along pushouts. ⊣

The notion of heterogeneous refinement now also yields a general definition of heterogeneous sub-ontology (see also [105]).

Definition 3.6. We call an ontology O_1 a **(heterogeneous) sub-ontology** of O_2 if O_2 is a (heterogeneous) refinement of O_1 such that

- the monomorphic extension is trivial (i.e. the identity),
- the signature morphism part of the theorem link is a monomorphism (the category theoretic generalisation of an injection), and
- the institution comorphism part of the theorem link is a subinstitution.

$\dashv$

The standard notion of sub-ontology, understood as a subset of the axioms [83, 92], is recovered in the homogeneous case.

3.1.2. Ontology Equivalence. Now, we can take this analysis one step further: when should we say that two ontologies (or ontology modules) O_1 and O_2 are *the same*, or, more precisely, 'equivalent' modulo their *syntactic expression* in a specific logic, with axioms phrased in a particular way, etc.? To make this precise, we have to explicate a notion of *(heterogeneous) equivalence of ontologies.*[36] In principle, there are several possibilities to define such an equivalence. Firstly, adapt the notion from [73] to our setting:

Definition 3.7. Two ontologies O_1 and O_2 are called *weakly equivalent*, if they can be conservatively refined into each other.

Secondly, we recall the definition of equivalence from [42, 142] (actually [42] uses the term *synonymity*, while in [142], the notion is generalised to the heterogeneous setting). Intuitively, two ontologies are equivalent if they can be interpreted in each other. However, often this is not directly possible, but new concepts need to be defined in one or both ontologies in order to be able to map the concepts of the other ontology. Hence, we arrive at the following (replacing definitional extensions of [42, 142], since this is more suitable in case of many-sorted logics):

Definition 3.8. Two ontologies O_1 and O_2 are called *pre-equivalent*, written $O_1 \cong O_2$, if there is a common monomorphic extension O of O_1 and O_2. Equivalence (called derived equivalence in [142]) of ontologies is defined to be transitive closure of pre-equivalence.

Proposition 3.9 ([142]). *Derived equivalence is an equivalence relation. In semi-exact institutions, pre-equivalence is transitive, and hence the notions of equivalence and pre-equivalence coincide.*

Via Grothendieck institutions, these notions automatically apply to heterogeneous ontologies as well.

Example (Equivalent mathematical theories). The theory of groups with an inverse operation is equivalent to the theory of groups with an axiom stating

[36]The closely related problems of proof-theoretic reductions [53] and of equivalence of mathematical theories [99, 154] have of course been previously studied in the literature, but the notion of equivalence allows a broader spectrum of interpretations when applied to ontologies. Ontology equivalence w.r.t. query languages is considered in [96].

the existence of inverse elements. The theory of Boolean algebras is equivalent to that of Boolean rings. Less intuitively, the theory of reflexive relations is equivalent to the theory of irreflexive relations. The theory of two binary relations R and Q, axiomatised as

$$\begin{aligned}&\forall x.R(x,x)\\ &\forall x.\neg Q(x,x)\\ &\forall x,y.R(x,y)\longleftrightarrow(Q(x,y)\vee x=y)\end{aligned}$$

is a monomorphic (even definitional) extension of both the theory of reflexive relations and that of irreflexive relations. The explanation for this somewhat counter-intuitive equivalence is that for both reflexive and irreflexive relations, the only information (or freedom of interpretation) contained in a model is whether two *distinct* elements are related. ⊣

Example (Equivalent Ontologies). Recall the example given on page 39 where we showed that the first-order axiomatisation of basic mereology (given on page 18) is a heterogeneous refinement of a DL ontology (given on page 22) which captures some connections between *parthood* and *proper parthood.* We have noted that the theorem link in this refinement is not conservative because the first-order ontology implies antisymmetry of parthood which is not expressible in DL—therefore these two ontologies are not equivalent. To obtain an equivalent first-order ontology, we need to leave out the *definition* of proper parthood through parthood, and instead weaken the axiomatisation as follows:

$$\begin{array}{llcl}\textbf{(Proper Part)} & \forall x,y\in X.PP(x,y) & \longrightarrow & P(x,y)\\ \textbf{(Transitivity)} & \forall x,y,z\in X.P(x,y)\wedge P(y,z) & \longrightarrow & P(x,z)\\ \textbf{(Propagation)} & \forall x,y,z\in X.P(x,y)\wedge PP(y,z) & \longrightarrow & PP(x,z)\end{array}$$

This is still a rather straightforward example of an ontology equivalence as it can easily be established by considering the standard translation from DL to first-order logic. It should be clear, however, that more involved examples can be easily given. ⊣

The following is obvious:

Proposition 3.10. *Ontology pre-equivalence implies weak equivalence. In semi-exact institutions, ontology equivalence implies weak equivalence.*

It is an open question whether pre-equivalence is strictly stronger than weak equivalence.

Proposition 3.11. *Weak equivalence implies (but is strictly stronger than) refinement in both directions.*

Proof. The implication is obvious. To prove that it is strict, consider the untyped first-order theory $\texttt{BinRel}_n$, which introduces a binary relation symbol P and an axiom stating that there are at most n individuals. Moreover, consider its extension $\texttt{Reflexive}_n$, which adds reflexivity: $\forall x.P(x,x)$.

Then `Reflexive`$_n$ obviously refines `BinRel`$_n$. The converse refinement uses a definitional extension of `BinRel`$_n$ introducing the reflexive closure $RefCl_P$ of P axiomatised as

$$\forall x, y. RefCl_P(x, y) \longleftrightarrow P(x, y) \vee x = y.$$

Then `Reflexive`$_n$ refines to this by mapping P to $RefCl_P$. However, `BinRel`$_n$ and `Reflexive`$_n$ admit different numbers of (non-isomorphic) models. Hence, by Prop. 3.12, they are not weakly equivalent. ⊣

Proposition 3.12. *(Weakly) equivalent ontologies have the same number of non-isomorphic models.*

Proof. For weakly equivalent ontologies, apply Prop. 3.3. By Prop. 3.10 and 3.11, this carries over to pre-equivalence, and since equality of number of non-isomorphic models is transitive, it also carries over to equivalence. ⊣

3.2. Integration and Reference Ontologies

Informally, an *integration* of two ontologies O_1, O_2 into a third ontology $\mathcal{O}$ is any operation by which O_1, O_2 are 're-interpreted' from the (global) point of view of $\mathcal{O}$. In the approach of [164], two ontologies O_1 and O_2 are aligned by mapping them into a common reference ontology $\mathcal{O}$ as follows: theories O_1 and O_2 are said to be **semantically integrated** with respect to a theory $\mathcal{O}$ if

1. there exist two *consequence-preserving sentence translations*: $\alpha_1 : O_1 \longrightarrow \mathcal{O}$, and $\alpha_2 : O_2 \longrightarrow \mathcal{O}$;
2. there exist *structure reducts*: $\beta_1 : \mathbf{Mod}(\mathcal{O}) \longrightarrow \mathbf{Mod}(O_1)$, and $\beta_2 : \mathbf{Mod}(\mathcal{O}) \longrightarrow \mathbf{Mod}(O_2)$; and
3. $\mathcal{O}$ is consistent.

Here, $\mathbf{Mod}(S)$ denotes the class of all models of S.

$$\begin{tikzcd} & \mathcal{O} & \\ O_1 \arrow[ur, "\alpha_1"'] & & O_2 \arrow[ul, "\alpha_2"] \end{tikzcd}$$

FIGURE 5. Integration into reference ontology

Example. [From 164, abridged] Suppose that O_1 is a relational scheme. It contains `author_of(person,paper)` and `person(id,name)` with a relationship from `person` to `id`. O_2 is a description logic theory. It contains $\mathsf{Article} \sqsubseteq \exists \mathsf{author}.\top \sqcap \exists \mathsf{title}.\top$, etc. (In Sect. 2.1.1, we have provided more details.)

The reference ontology $\mathcal{O}$ is a first-order theory. It contains, among others:

$$\begin{aligned} \forall x.(Working\_Person(x) &\rightarrow (Tangible\_Thing(x) \wedge \\ &\qquad \exists y.(String(y) \wedge Name(x, y)))) \\ \forall x.(Researcher(x) &\rightarrow Working\_Person(x)) \end{aligned}$$

The sentence maps α_1, α_2 can be given as follows:

$$\begin{aligned}
\alpha_1(\texttt{person}(\texttt{p},\texttt{n})) &= \mathit{Researcher}(p) \wedge \mathit{String}(n) \wedge \\
&\quad \mathit{Name}(p,n) \\
\alpha_1(\texttt{author\_of}(\texttt{p},\texttt{a})) &= \mathit{Researcher}(p) \wedge \mathit{Article}(a) \wedge \\
&\quad \mathit{Author}(a,p) \wedge \exists j.(\mathit{Journal}(j) \\
&\quad \wedge \mathit{Has\_Article}(j,a)) \\
\alpha_2(\mathsf{Article}(x)) &= \mathit{Publication}(x)
\end{aligned}$$

$\dashv$

We see some problems with this approach: allowing arbitrary sentence maps α_i is simply too liberal. For example, α_i could map every sentence to *true*.[37] It seems more reasonable to use *signature morphisms* and their induced sentence translation maps instead. This approach, however, is less flexible in one aspect: with the approach of [164] (using first-order logic), a predicate symbol p may be mapped to a formula φ. However, this is usually better captured by using *derived signature morphisms* (see [163]), which here are just signature morphisms into a monomorphic extension (e.g. an extension by the definition $q(x) \Leftrightarrow \varphi$, where q is a new predicate symbol that can be used as the image of p). Since we have already integrated this into our notion of refinement above, we arrive at:

Definition 3.13. Given ontologies O_1, O_2, and an ontology $\mathcal{O}$, in institutions I_1, I_2 and I, respectively, we say that $\mathcal{O}$ **heterogeneously (conservatively) integrates** O_1 and O_2 if there are (conservative) refinements from both O_1 and O_2 to $\mathcal{O}$.

In Section 4.3, we cast the above example into this setting and discuss it in more detail.

In some cases, there may simply be no suitable common reference ontology at hand. In these cases, the common super-ontology should be suitably *constructed* from O_1 and O_2, identifying certain concepts, while keeping others distinct, leading to the concept of V-*alignment* discussed in the next section. Moreover, the example is reformulated as a heterogeneous refinement on Page 58.

3.3. Heterogeneous Connection

Intuitively, the difference between 'integrations' and 'connections' is that in the former we combine two ontologies O_1 and O_2 using a typically large and previously-known reference ontology $\mathcal{O}$. The models of $\mathcal{O}$ are typically much richer than those of O_1 and O_2. By contrast, *connection* of two ontologies is done in such a way that the respective theories, signatures, and models are kept disjoint, and a (usually small and flexible) *bridge theory* formulated (in a bridge language) over a signature that *goes across* the sort structure of

[37][164] suggest to solve this problem by a possible restriction to conservative translations; however, even then the translation mapping every theorem in O_i to *true* and every non-theorem to *false* still is a valid but useless example.

the components is used to *link together* the two ontologies. Using the general approach of colimits, an overall connection ontology can be *automatically computed* from the bridge theory and the involved ontologies. Moreover, the models of this overall ontology are obtained as amalgamations of models of the individual ontologies—no new structure is added (expect from new definitions, which however can always be interpreted uniquely).

3.3.1. Connection through Alignments.

V-Alignments. [183] address the problem of alignment without a common reference ontology. Given ontologies O_1 and O_2, an **interface** (for O_1, O_2)

$$\langle \Sigma, \sigma_1 \colon \Sigma \longrightarrow \mathsf{Sig}(O_1), \sigma_2 \colon \Sigma \longrightarrow \mathsf{Sig}(O_2) \rangle$$

specifies that (using informal but suggestive notation)

- concepts $\sigma_1(c)$ in O_1 and $\sigma_2(c)$ in O_2 are identified for each concept c in Σ, regardless of whether the concepts have the same name or not, and
- concepts in $O_1 \setminus \sigma(\Sigma_1)$ and $O_2 \setminus \sigma(\Sigma_2)$ are kept distinct, again regardless of whether they have the same name or not.

The resulting common ontology $\mathcal{O}$ is not given a priori, but rather it is computed from the aligned ontologies via the interface. This computation is a pushout in the sense of category theory, which in this case is just a disjoint union with identification of specific parts (namely those given through $\langle \Sigma, \sigma_1, \sigma_2 \rangle$).

V-alignments can thus deal with basic alignment problems, such as *synonymy* (identifying different symbols with the same meaning) and *homonymy* (separating (accidentally) identical symbols with different meaning)—see Figure 6.

$$\begin{tikzcd}
 & & \{\mathsf{Woman}, \mathsf{River\_Bank}, \mathsf{Financial\_Bank}, \mathsf{Human\_Being}\} & & \\
 & & \mathcal{O} \arrow[u, phantom, sloped, "\subseteq"] & & \\
O_1 \arrow[urr, dashed] & & & & O_2 \arrow[ull, dashed] \\
\{\mathsf{Woman}, \mathsf{Bank}, \mathsf{Person}\} \arrow[u, phantom, sloped, "\subseteq"] & & & & \{\mathsf{Woman}, \mathsf{Bank}, \mathsf{Human}\} \arrow[u, phantom, sloped, "\subseteq"] \\
 & & \Sigma \arrow[uull, "\sigma_1"'] \arrow[uurr, "\sigma_2"] & & \\
 & & \{\mathsf{Woman}, \mathsf{Person}\} \arrow[u, phantom, sloped, "="] & &
\end{tikzcd}$$

FIGURE 6. V-alignment: integration through interface (dashed arrows mean definition links automatically computed via colimits)

Example. In Figure 6, the interface $\langle \Sigma, \sigma_1, \sigma_2 \rangle$ specifies that the two instances of the concept Woman as well as Person and Human are to be identified. This yields two concepts Woman and $\mathsf{Human\_Being}$ in the push-out ontology $\mathcal{O}$ obtained along the dashed arrows. It also determines that the two instances of Bank are to be understood as homonyms, and thus generates two new distinct concepts. ⊣

However, notion such as *polysemy* are typically understood to relate terms that have a different, but *related* meaning, and can thus not be dealt with by simply identifying symbols or keeping them apart. This problem can be solved, however, by considering $\mathcal{E}$-connections a general form of alignment (see [109] and Section 3.3.3 below). Similarly, [183] themselves raise the criticism that V-Alignments do not cover the case where a concept Woman in O_1 is aligned with a concept Person in O_2: here, the resulting ontology should turn Woman into a subconcept of Person. This is not directly possible with the pushout approach.

W-**Alignments.** In order to solve this problem of V-Alignments, [183] introduce W-Alignments. They consist of two V-Alignments, using an intermediate bridge ontology B. The latter can be used to specify subconcept relationships like Woman $\sqsubseteq$ Person as mentioned above.

$$\begin{tikzcd}
\{\mathsf{Woman}\} & & \{\mathsf{Woman} \sqsubseteq \mathsf{Person}\} & & \{\mathsf{Person}\} \\
\curlywedge & & \curlywedge & & \curlywedge \\
O_1 & & B & & O_2 \\
& \Sigma_1 \arrow[ul] \arrow[ur] & & \Sigma_2 \arrow[ul] \arrow[ur] & \\
& \shortparallel & & \shortparallel & \\
& \{\mathsf{Woman}\} & & \{\mathsf{Person}\} &
\end{tikzcd}$$

FIGURE 7. W-alignment: integration through bridge ontology

[183] list the behaviour of compositions as a weak point of this approach. However, we see as the main weak point the rather loose coupling of O_1 and O_2; indeed, the bridge ontology is something like a super-ontology of a sub-ontology and hence can be anything. A tighter coupling can be achieved with refinements. In [109], we have shown that various kinds of alignments can be analysed as certain 'shapes' of diagrams that can be represented and reasoned with in HETS.

M-**Alignments.** A natural generalisation of V-Alignments is to form the V using arbitrary refinements. Recall that these are theory morphisms into monomorphic extensions; we relax this here. Given two ontologies O_1 and O_2, let $O_1^\sharp$ and $O_2^\sharp$ be (typically conservative) extensions of O_1 and O_2, respectively, taking into account the possible requirements to (1) define new symbols (in order to emulate a derived theory morphism), and (2) introduce new subconcept relationships, such as Woman $\sqsubseteq$ Person, as discussed above. We thus arrive at the concept of an M-alignment, see Fig. 8.

$\mathcal{E}$-connections as a kind of extended (and heterogeneous) M-alignment will be discussed in Section 3.3.3. Compare also Section 4.2.1 below.

$$\begin{tikzcd}
 & & \{\mathsf{Woman}, \mathsf{Person}, \mathsf{River\_Bank}, \mathsf{Financial\_Bank}\} \arrow[d, phantom, "\rotatebox{90}{$\succcurlyeq$}"] & & \\
 & \{\mathsf{Woman} \sqsubseteq \mathsf{Person}\} \arrow[d, phantom, "\rotatebox{90}{$\succcurlyeq$}"] & \mathcal{O} & \{\mathsf{Person}, \mathsf{Bank}\} \arrow[d, phantom, "\rotatebox{90}{$\succcurlyeq$}"] & \\
 & O_1^\sharp \arrow[ur, dashed] & & O_2^\sharp \arrow[ul, dashed] & \\
O_1 \arrow[ur] \arrow[d, phantom, "\curlyvee"] & & \Sigma \arrow[ull] \arrow[urr] \arrow[d, phantom, "\rotatebox{90}{$=$}"] & & O_2 \arrow[ul] \arrow[d, phantom, "\curlyvee"] \\
\{\mathsf{Woman}, \mathsf{Bank}\} & & \{\mathsf{Person}\} & & \{\mathsf{Person}, \mathsf{Bank}\}
\end{tikzcd}$$

FIGURE 8. M-alignment: integration through bridge along extensions

$$\begin{tikzcd}
 & & \mathcal{O} & & \\
O_1 \arrow[urr, dashed] & & B \arrow[u, dashed] & & O_2 \arrow[ull, dashed] \\
 & \Sigma_1 \arrow[ul] \arrow[ur] & & \Sigma_2 \arrow[ul] \arrow[ur] &
\end{tikzcd}$$

FIGURE 9. Connection through interface and colimit

3.3.2. Connection through Interface and Colimit. The general idea of combination through an interface by computing a colimit is shown in Fig 9. Here, Σ_1 is a subsignature of ontology O_1, Σ_2 a subsignature of ontology O_2, B an interface formalised in a bridge logic such as $\mathbf{FOL}^{\mathsf{ms}=}$, and $\mathcal{O}$ the colimit ontology computed from the diagram.

The example given in Section 3.2 for an integration into a reference ontology can be reformulated in this setting by taking O_1 to be the relational scheme formalisation, O_2 the description logic knowledge base, and B the necessary first-order axioms to achieve the desired reconciliation. The 'reference ontology' is now obtained as a pushout. The complete specification for this scenario will be given in Section 4.4.1.

3.3.3. $\mathcal{E}$-Connections and DDL. Heterogeneous knowledge representation was a major motivation also for the design of 'modular ontology languages', such as distributed description logics (DDLs, [27]) and $\mathcal{E}$-connections [111, 106]. We here concentrate on the latter. $\mathcal{E}$-connections were originally conceived as a versatile and computationally well-behaved technique for combining logics, but were subsequently quickly adopted as a framework for the combination of ontologies in the Semantic Web [41].

We here show how the combination of ontologies via such modular languages can be re-formulated as structured heterogeneous ontologies, and indicate how this idea can be generalised to the institutional level.

The general idea behind this combination method is that the interpretation domains of the connected logics are interpreted by disjoint (or sorted) vocabulary and interconnected by means of *link relations*. The language of the $\mathcal{E}$-connection is then the union of the original languages enriched with operators capable of talking about the link relations.

$\mathcal{E}$-connections, just as description logics, offer an appealing compromise between expressive power and computational complexity: although powerful enough to express many interesting concepts, the coupling between the combined logics is sufficiently loose for proving general results about the transfer of decidability: if the connected logics are decidable, then their connection will also be decidable.

We first sketch the formal definitions for the 2-dimensional case and then outline a general institutional reformulation of $\mathcal{E}$-connections. The reader is referred to [106] for involved examples and technical results on the computational properties of various specific $\mathcal{E}$-connections.

To formulate a 2-dimensional $\mathcal{E}$-connection between two ontologies O_1 and O_2 formulated e.g. in two different DLs DL_1 and DL_2 (here, an ontology is a set of axioms in the respective DL), we assume that the **signatures** $\mathcal{L}_1 = \mathsf{Sig}(\mathrm{DL}_1)$ and $\mathcal{L}_2 = \mathsf{Sig}(DL_2)$ of the two DLs, i.e. their sets of atomic concepts, roles, and object names, are pairwise disjoint.

To form a connection $\mathcal{C}^{\mathcal{E}}(\mathrm{DL}_1, \mathrm{DL}_2)$, fix a non-empty set $\mathcal{E} = \{E_j \mid j \in J\}$ of binary relation symbols. The **basic $\mathcal{E}$-connection**, then, has as signature the disjoint union of $\mathcal{L}_1, \mathcal{L}_2$ and $\mathcal{E}$; its concept language is two-sorted with sorts s_1 and s_2 and defined by simultaneous induction as follows.

- (i) If C is a concept in DL_1, then C is of sort s_1; (ii) if D is of sort s_2 then $\langle E_j \rangle^1 D$ is of sort s_1; (iii) s_1 is closed under the concept-forming operations of DL_1.
- (i) If D is a concept in DL_2, then D is of sort s_2; (ii) if C is of sort s_1 then $\langle E_j \rangle^2 C$ is of sort s_2; (iii) s_2 is closed under the concept-forming operations of DL_2.

Here, the $\mathcal{E}$-connection-operators $\langle E_j \rangle^1$ and $\langle E_j \rangle^2$ are new concept-formation operators, interpreting the added link relations. The formal semantics is as follows: the class of models of $\mathcal{C}^{\mathcal{E}}(\mathrm{DL}_1, \mathrm{DL}_2)$ comprises all structures of the form

$$\mathfrak{M} = \left\langle \mathfrak{W}_1, \mathfrak{W}_2, \mathcal{E}^{\mathfrak{M}} = (E_j^{\mathfrak{M}})_{j \in J} \right\rangle,$$

where $\mathfrak{W}_i = (W_i, .^{\mathfrak{W}_i})$ is an interpretation for DL_i for $i \in \{1, 2\}$ and $E_j^{\mathfrak{M}} \subseteq W_1 \times W_2$ for each $j \in J$.

Given concepts C_i of ontology DL_i, for $i = 1, 2$, denoting subsets of W_i, the semantics of the basic $\mathcal{E}$-connection operators is

$$\begin{aligned} (\langle E_j \rangle^1 C_2)^{\mathfrak{M}} &= \{x \in W_1 \mid \exists y \in C_2^{\mathfrak{M}} \ (x, y) \in E_j^{\mathfrak{M}}\} \\ (\langle E_j \rangle^2 C_1)^{\mathfrak{M}} &= \{x \in W_2 \mid \exists y \in C_1^{\mathfrak{M}} \ (x, y) \in E_j^{\mathfrak{M}}\} \end{aligned}$$

It remains to clarify what the **sentences** of a basic $\mathcal{E}$-connection are. These just follow the same grammar as the component logics (in the case of DLs concept subsumptions, ABox and RBox statements), but respect the enriched concept language, with the obvious semantics interpreted in the local models. Moreover, we have ABox-like sentences for the link relations such as

$$\mathfrak{M} \models (a, b) : E_j \iff {E_j}^{\mathfrak{M}}(a^{\mathfrak{M}}, b^{\mathfrak{M}}).$$

Fig. 10 displays the connection of two ontologies by means of a single link relation E. Here, the concept $\langle E \rangle^1 (\{a\})$ of O_1 'corresponds' to the nominal

$\{a\}$ of ontology O_2: it collects the set of all those points in O_1 that 'can be seen' from a (in O_2) along the relation E.

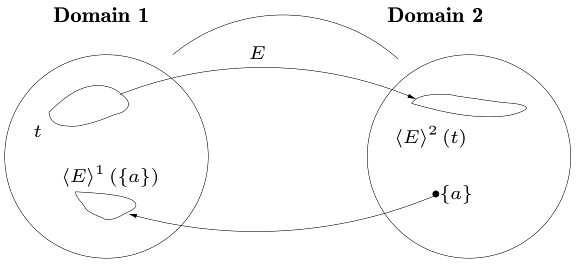

FIGURE 10. A two-dimensional connection.

Example (Connecting two ontologies). Suppose two ontologies O_1 and O_2, formulated in different DLs DL_1 and DL_2, contain the concept Window. Now, ontology O_1 might formalise functionalities of objects found in buildings, while ontology O_2 might be about the properties of materials of such objects. The intended relation between the two instances of Window might now be one of *polysemy* (meaning variation), i.e., Window in O_1 involves 'something with views that can be open or closed':

$$\mathsf{Window} \sqsubseteq \exists \mathit{has\_state}.(\mathsf{Open} \sqcup \mathsf{Closed}) \sqcap \exists \mathit{offers}.\mathsf{Views},$$

while the meaning of Window in O_2 might be 'something that is bulletproof glass':

$$\mathsf{Window} \equiv \mathsf{Glass} \sqcap \exists \mathit{has\_feature}.\mathsf{Bulletproof}.$$

A systematic integration of these two ontologies could now require a mapping of objects in O_1 to the material they are made from, using a link relation '*consists_of*'. A concept of the form $\langle \mathit{consists\_of} \rangle^1 \mathsf{C}$ then collects all objects of O_1 that are made from something in C, while a concept $\langle \mathit{consists\_of} \rangle^2 \mathsf{D}$ collects the materials in O_2 some object in D consists of. A sensible alignment between the two instances of Window, introducing disjoint vocabulary Window_1 and Window_2, could now be formalised in $\mathcal{E}$-connections as:

$$\begin{aligned} \langle \mathit{consists\_of} \rangle^2 \mathsf{Window}_1 &\sqsubseteq \exists \mathit{has\_feature}.\mathsf{Transparent} \\ \langle \mathit{consists\_of} \rangle^1 \mathsf{Window}_2 &\sqsubseteq \mathsf{Window}_1 \sqcap \exists \mathit{provides\_security}.\mathsf{Inhabitant} \end{aligned}$$

assuming that windows in O_1 might also be made of plastic, etc. $\dashv$

Although this example is heterogeneous in the sense that two different description logics are involved, both dimensions in this $\mathcal{E}$-connection still conform to the same semantic paradigm. More interesting heterogeneous $\mathcal{E}$-connections are obtained when mixing logics with different 'reasoning modes', e.g. when combining conceptual with spatial reasoning, as for instance necessary in modelling architectural design as sketched in the next example.

Example (Modelling Architectural Design). The use of $\mathcal{E}$-connections for modelling architectural design involving both conceptual and spatial dimensions has been investigated in [23, 86]. Extending the previous example, let us suppose that we have a third dimension, a knowledge base formalised in the Region Connection Calculus RCC8 (as encoded in the modal logic $\mathbf{S4_u}$) that we have introduced on Page 23. The following constraint is taken from [86] and illustrates the kind of modelling that can be performed in this setup.

> [. . .] sensors have to cover certain regions around doors. These are functional regions that are defined by the doors and instantiated in the qualitative layer. The region of the sensor range has to be an inverse proper part of this functional region.

Here, 'door' and 'sensor' are taken as concepts door, sensor that live in ontology O_1 introduced above. Moreover, we introduce two new relations bridging ontology O_1 and the RCC8 domain, namely *has_functional_space* that relates the instances of door with their functional space, i.e. regions in RCC8, and *has_range_space*, again giving the regions covered by the sensors, see Fig. 11.

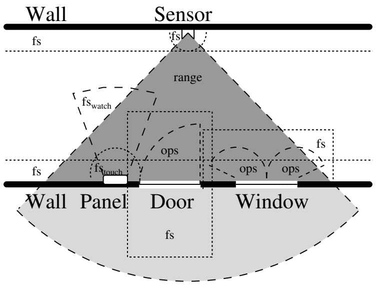

FIGURE 11. Spatial extensions of ontology terms in $\mathbb{R}^2$

Here, models for an $\mathcal{E}$-connection of ontology O_1 and RCC8 are of the form

$$\mathfrak{M} = \left\langle \mathfrak{W}_1, \mathfrak{W}_2, \mathcal{E}^{\mathfrak{M}} = \left(\mathit{has\_range\_space}^{\mathfrak{M}}, \mathit{has\_functional\_space}^{\mathfrak{M}}\right)\right\rangle,$$

where $\mathfrak{W}_1$ interprets ontology O_1, $\mathfrak{W}_2$ interprets RCC8, and the link relations are interpreted as subsets of the cartesian products of the domains of $\mathfrak{W}_1, \mathfrak{W}_2$. The constraint can now be formalised thus:[38]

$$PP^{-1}(\langle \mathit{has\_range\_space}\rangle^3 \mathsf{sensor}, \langle \mathit{has\_functional\_space}\rangle^3 \mathsf{door})$$

Here, e.g. $\langle \mathit{has\_range\_space}\rangle^3$ sensor defines a region by collecting, for a given model $\mathfrak{M}$ of the $\mathcal{E}$-connection, all points in the RCC8 model that are 'connected' by the role *has_range_space* to an element of the concept sensor. ⊣

[38] PP^{-1} here is the abbreviation for the union of tPP^{-1} and $ntPP^{-1}$.

$\mathcal{E}$-connections can be considered as many-sorted heterogeneous theories: component ontologies can be formulated in different logics, but have to be built from many-sorted vocabulary, and link relations are interpreted as relations connecting the sorts of the component logics.

The main difference between distributed description logics (DDLs) [27] and various $\mathcal{E}$-connections now lies in the expressivity of the 'link language' $\mathcal{L}$ connecting the different ontologies. While the basic link language of DDL is a certain sub-Boolean fragment of many sorted $\mathcal{ALC}$, the basic link language of $\mathcal{E}$-connections is $\mathcal{ALCI}^{\mathsf{ms}}$.[39]

Such many-sorted theories can easily be represented in a diagram as shown in Fig. 12. Here, we first (conservatively) obtain a disjoint union $\mathcal{S}_1^{\mathsf{m}} \uplus \mathcal{S}_2^{\mathsf{m}}$ as a pushout, where the component ontologies have been turned into sorted variants (using an institution comorphism from the single-sorted to the many-sorted logic), and the empty interface guarantees that no symbols are shared at this point. An $\mathcal{E}$-connection knowledge base (KB) in language $\mathcal{C}^{\mathcal{E}}(\mathcal{S}_1^{\mathsf{m}}, \mathcal{S}_2^{\mathsf{m}})$ is then obtained as a (typically not conservative) theory extension.

$$\begin{tikzcd}
 & \mathcal{C}^{\mathcal{E}}(\mathcal{S}_1^m, \mathcal{S}_2^m) & \\
 & \mathcal{S}_1^m \uplus \mathcal{S}_2^m \arrow[u] & \\
\mathcal{S}_1^m \arrow[ur, dashed] & & \mathcal{S}_2^m \arrow[ul, dashed] \\
 & \varnothing \arrow[ul] \arrow[ur] & \\
\mathcal{S}_1 \arrow[uu] & & \mathcal{S}_2 \arrow[uu]
\end{tikzcd}$$

FIGURE 12. $\mathcal{E}$-connections as a structured heterogeneous theory

The idea to 'connect' logics can be elegantly generalised to the institutional level (compare [7] who note that their 'connections' are an instance of a more general co-comma construction). Without giving the details of such a generalisation, it should be clear from the above that our Grothendieck institution approach is general enough to formally capture such connections: intuitively, we need to formalise the idea that an abstract connection of two logics $\mathcal{S}_1$ and $\mathcal{S}_2$ is obtained by defining a bridge language $\mathcal{L}(\mathcal{E})$, where the elements of $\mathcal{E}$ *go across* the sort-structure of the respective logics, and where theory extensions are defined over a new language defined from the disjoint union of the original languages together with $\mathcal{L}(\mathcal{E})$, containing certain expressive means applied (inductively) to the vocabulary of $\mathcal{E}$.

Note that this generalises the $\mathcal{E}$-connections of [111], the DDLs of [27], as well as the connections of Baader and Ghilardi [7] in two important respects: first, the institutional level generalises the term-based abstract description

[39]But can of course be weakened to $\mathcal{ALC}^{\mathsf{ms}}$ or sub-Boolean DL, or indeed strengthened to more expressive many-sorted DLs involving e.g. number restrictions or Boolean operators on links, see [101, 106] for details.

languages (ADS) that are an abstraction of modal and description logics, and the rather general definition of bridge theory similarly abstracts from the languages previously employed for linking that were similarly inspired by modal logic operators.

While there are no implemented, specialised algorithms available deciding satisfiability in $\mathcal{E}$-connections (except limited support in some versions of the Pellet system [41]), semi-decidable reasoning for more expressive $\mathcal{E}$-connections is provided by Hets through suitable translation by a comorphisms in a supported logic. For instance, an $\mathcal{E}$-connection of a description logic ontology with an RCC8 knowledge base can be translated into many-sorted first-order logic, providing semi-decidable reasoning using, e.g., the Spass prover.

Given that $\mathcal{E}$-connections have a relatively intuitive and simple semantics (compared e.g. to some multi-dimensional logics [57] or fibrings [56]), they remain quite popular as a modelling paradigm for heterogeneous combinations of ontologies with other formalisms. Apart from being applied in Semantic Web related research [69, 41], they have been employed to model heterogeneous combinations of linguistic ontologies and spatial calculi [88], to model architectural design [86], or to ambient environments [23].

[88], for instance, analyse the problem of relating an ontology encoding the linguistic spatial semantics of natural language utterances as represented in the linguistic ontology GUM [11, 10, 12] with spatial calculi, using the example of the double-cross calculus DCC [55] for projective relations (orientations). The general relation between GUM and DCC is a loose coupling as can be adequately modelled by an $\mathcal{E}$-connection. However, two entirely independent layers need to be added for a ‘complete’ formal representation of a spatial configuration: domain knowledge including naïve physics information is added in a KB $\mathcal{D}$, while contextual information (such as intrinsic orientations, reference system, etc.) is added by a KB $\mathcal{O}$. Both these layers of information are typically formalised in different (heterogeneous) logics. The overall integration is obtained via a pushout operation, as shown in the upper part of Fig. 12, taking $\mathcal{S}_1 = \text{GUM}$ and $\mathcal{S}_2 = \text{DCC}$. [87] take this kind of modelling approach a step further by defining a variant of $\mathcal{E}$-connections, called $\mathcal{S}$-connections, which introduces notions of similarity both to the component logics as well as to the link relations, based on work on similarity [170] and distance logics [102]. Here, ‘local similarity’ compares objects within one domain, whilst comparing objects across domains leads to similarity measures that are motivated by and based on counterpart-theoretic semantics [100].

4. Modes of Reasoning over Ontologies vis-à-vis Structuring

Ontologies afford and require various modes of reasoning. These range from more classical reasoning tasks, such as consistency checks, deduction and computing a classification, to less traditional operations on ontologies such as conservativity checks, colimit computation, or theory interpretation, related to such problems as module construction and gluing together of and separating vocabulary.

The focus of this paper has been on structuring aspects for ontologies, and various operations to combine ontologies together according to these structuring techniques. From a bird's eye perspective, a main aspect of this line of research is that the semantics of the 'combined system' depends functionally only on the semantics of the participating components and the semantics of the structuring/combination operations. In contrast to this, the large field of multi-dimensional logics [128, 57], which is not in the scope of this paper, designs special purpose formalisms by combining the syntax and semantics of component logics into a new logical formalism. Relevant examples from the ontology engineering point of view, among many others, are for instance temporal description logics [121], description logics incorporating a notion of similarity [169, 170], or logics integrating DLs with action formalisms [8].

The distinction between these two approaches to combining ontologies is not necessarily a sharp one. For instance, $\mathcal{E}$-connections can be seen both as a many-dimensional formalism as well as the result of a two-step structuring operation (see Sec. 3.3.3). However, as opposed to typical multi-dimensional logics, $\mathcal{E}$-connections still leave untouched the semantics (model classes) of their components.

Relatively little work has been devoted to the study of combining different logical reasoning techniques into one system. Baumgartner [13] discusses first-order reasoning applied to ontologies, but not in combination with e.g. DL reasoning. Similarly, Voronkov [181] has used first-order reasoners to establish inconsistencies in ontologies.

The structured approach to ontology design and reasoning that we have developed in this paper allows for a richer interaction of reasoning methods. In the following, we will sketch some of these reasoning modes, show how they relate to various structuring aspects, and detail the support provided for them in the Heterogeneous Tool Set Hets (see [140, 141]). We begin by giving a brief description of the Hets system, and then look at specific examples of refinements, integrations, alignments and connections. Hets, as well as a library of examples including many of those presented in this paper, can be obtained freely at `http://www.dfki.de/sks/hets`.

4.1. Reasoning Support in Hets

Hets supports a multitude of logics, given as institutions, that can be used in formal specification.

From the list of logics introduced in Section 2.1, the following are supported by Hets: Propositional logic, untyped first-order logic (in an extended variant called softly typed first-order logic, SoftFOL), many-sorted first-order logic $\mathbf{FOL}^{\mathsf{ms=}}$ (in an extended variant called Casl), Common Logic, Relational Schemes, the description logic $\mathcal{SROIQ}(D)$ underlying $\mathcal{OWL}$ 2 DL (with several concrete syntaxes, including Manchester Syntax [89]), modal

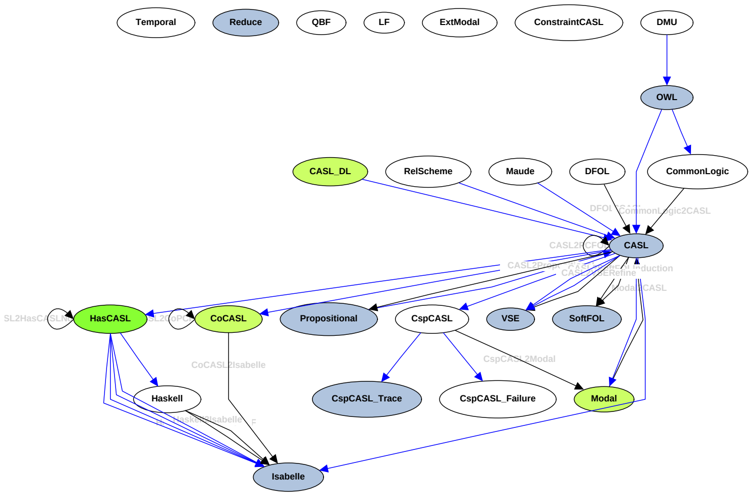

FIGURE 13. HETS's logic graph

first-order logic (called ModalCASL) and the higher-order logics Isabelle/HOL and HASCASL. HETS's logic graph is depicted in Fig. 13. The nodes represent the logics currently available in HETS, while the arrows represent the comorphisms between them.

Recall that there are several ways available of creating proof obligations in HETCASL; in particular, they can be created via extensions `SPEC1 then %implies SPEC2` and via views. Both techniques result in theorem links in the development graph. Moreover, with the keyword `%implied`, a formula can be marked to be a proof goal, which is actually marked in the corresponding development graph node. Such annotations can be quite useful for an ontology designer as a control mechanism to keep track of *desired consequences*: in case such a proof obligation fails, a design error has been made.

A heterogeneous proof calculus (see [38]) for development graphs, as they have been defined in Section 2.2, can be used for shifting proof obligations expressed by theorem links into local proof goals for nodes. This calculus has been implemented in HETS. In many cases the development graph calculus can be applied automatically to a structured specification to yield the desired results.

Finally, in order to discharge local proof goals for development graph nodes, specific theorem provers need to be called. For several of these logics proof support is directly available, as shown in Table 1. If there is no prover available for a logic directly, a prover can be "borrowed" from another logic, if a (composition of) comorphisms is available that maps the logic of the proof goal into a prover-supported logic.

Logic	Connected Prover(s)
Propositional	zChaff, MiniSat, TruthTables
$\mathcal{OWL}$	Pellet, Fact++
SoftFOL	Spass, Darwin, Vampire, EKrHyper
VSE	VSE
Isabelle	Isabelle

Table 1. Supported provers

After having discussed the theory and proof-support for heterogeneous ontologies in some detail, we now illustrate how to define an ontology heterogeneously from three parts formalised in different languages.

Example (A simple heterogeneous ontology). Firstly, consider a basic specification written in $\mathcal{OWL}$, given in Fig. 14 on the left hand side, describing Tigers being carnivores and cats of prey. Secondly, consider another basic

```
logic OWL
spec Predators =
     Class: Carnivore

     Class: Tiger
         SubclassOf: Carnivore,
                     CatsOfPrey
end
```

```
logic CASL
spec Prey =
     Prey_Animals
then %implies
     forall a,b : Thing
     . Hare(a)  /\ Mouse (b) => not isTastier (b,a)
end
```

Figure 14. Predators and their prey

specification in Casl describing their prey given in Fig. 14 on the right hand side, and reusing another Casl ontology `Prey_Animals` given in Fig 15 on the left.

```
spec Prey_Animals =
     sort Thing
     pred Hare      : Thing
     pred Mouse     : Thing
     pred isTastier : Thing * Thing
     forall a,b :Thing
     . isTastier (a,b) =>
       not isTastier (b,a)
     . Hare(a)  /\ Mouse (b) =>
       isTastier (a,b)
end
```

```
logic CASL
spec Animals =
     Predators and {Prey with Hare |-> Lepus}
then
     pred prefers : Thing * Thing * Thing
     forall a,b,c : Thing
     . Tiger(a) /\ isTastier (b,c) <=> prefers (a,b,c)
then %implies
     forall a,b,c : Thing
     . Tiger(a) /\ Lepus(b) /\ Mouse (c) => prefers (a,b,c)
end
```

Figure 15. Prey Animals in Casl

The keyword `%implies` here introduces a proof obligation, namely a theorem link expressing that the part after `%implies` logically follows from the parts before. This particular proof obligation follows from the asymmetry of `isTastier` which is specified in Fig. 15. Such annotations can be quite useful for an ontology designer as a control mechanism to keep track of *desired consequences*: in case such a proof obligation fails, a design error has been made. Thirdly, consider the specification in Fig. 15 on the right. This is the union of the above ones and adds a new 3-ary predicate that cannot

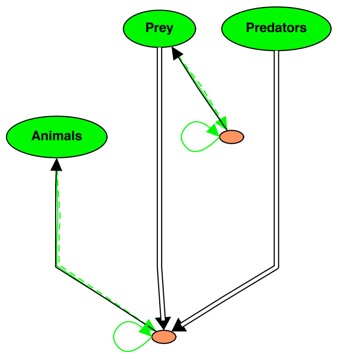

FIGURE 16. Development Graph of the Heterogeneous specification

directly be expressed in $\mathcal{SROIQ}(D)$. In the process of uniting the specifications, `Predators` is mapped along the comorphism from $\mathcal{OWL}$ to CASL. Further, `Hare` is being renamed to `Lepus`. The development graph of this heterogeneous specification is displayed in HETS as shown in Fig. 16. Please note that the solid black arrows depict definition links (the double-lined arrows indicate *heterogeneous* definition links), whilst the light green arrows are theorem links (and the dotted green arrows *local* theorem links introduced by the development graph calculus [138]). The unnamed nodes, which contain the proof obligations, can now be proved by running a theorem-prover on them, i.e., on their local theories. This way, we do not have to deal with axioms that are introduced later and that are not important for this theory. Please note that the theory of the unnamed node with a theorem link from `Prey` is formalised in a DL, thus allowing it to be proved by a DL reasoner. With this approach, many 'conjectures' can already be proven in a smaller, 'local' environment. Further, this approach helps the designer of an ontology to find inconsistencies: if the overall ontology turns out to be inconsistent, it is possible to check the consistency of the theories of all nodes in the development graph that contribute to the overall specification. If one of them turns out to be inconsistent, it might already be possible to fix the inconsistency in this smaller, local theory. Note that this 'scales down' the search space for finding inconsistencies in a way that is independent from the techniques developed in [92]. ⊣

The development graph calculus integrated into HETS can be used for all the integration, connection, and refinement techniques for ontologies discussed in this paper: the verification of semantic integrations as well as refinements are directly supported via the development graph calculus and the connected provers. In the case colimit computation is needed, as in W-alignments, this can be calculated via HETS' built-in colimit feature described in [37], that also allows the approximation of heterogeneous colimits.

4.2. Reasoning About Refinements

A refinement between two ontologies is another special case of an interpretation of theories, again written as views in HetCasl and visualised as theorem links in development graphs. We will now describe two heterogeneous refinements involving the bibliographic ontologies introduced in Section 2.1.

4.2.1. From Relational Scheme to Ontology. Recall the example (taken from [164]) of a semantic integration into a reference ontology given in Section 3.2. The kind of integration required here can be dealt with much more elegantly as a heterogeneous refinement. Consider the $\mathcal{OWL}$ specification given in Fig. 18. Recall that in Section 2.1 we have introduced `Biblio_OWL`, an on-

$$\begin{tikzcd}
\texttt{Biblio\_OWL}' & \texttt{Biblio\_RS}' \arrow[l, dotted] \\
\texttt{Biblio\_OWL} \arrow[u, "def"] & \texttt{Biblio\_RS} \arrow[u, "def"']
\end{tikzcd}$$

Figure 17. A view from RelScheme to $\mathcal{OWL}$.

tology about bibliographical information, written in a concrete syntax close to $\mathcal{OWL}$-DL Manchester Syntax [89], and `Biblio_RS`, the scheme of a relational database intended to capture similar knowledge. Assume we want to show that the ontology is a refinement of the database schema, as illustrated in Fig. 17. A view `Biblio_RS_in_OWL` is used for this purpose, stating that

```
logic CASL
view Biblio_RS_in_OWL : Biblio_RS to
  { Biblio_OWL with logic OWL -> CASL
    then %def
      preds
        journal(j,n,f:Thing) <=>
          Journal(j) /\ name(j,n) /\ impactFactor(j,f);
        paper(a,t,j:Thing) <=>
          Article(a) /\ Journal(j) /\ hasArticle(j,a) /\
          title(a,t);
        author_of(p,a:Thing) <=>
          Researcher(p) /\ Article(a) /\ author(p,a);
        person(p,n:Thing) <=> Researcher(p) /\ name(p,n)
  } = logic RelationalScheme -> CASL
end
```

Figure 18. Heterogeneous specification in HetCasl.

the ontology satisfies the relational scheme axioms (referential integrity constraints). Of course, this is not possible literally, but rather the ontology is mapped to first-order logic (Casl) and then definitionally (look at the `%def`) extended to `Biblio_OWL`′ with a definition of the database tables in terms of the ontology classes and properties. Also, `Biblio_RS` is translated to first-order logic, yielding `Biblio_RS`$'$, and the view expresses a theory morphism from `Biblio_RS`$'$ to `Biblio_OWL`$'$.

The involved signature and theory morphisms live in the Grothendieck institution. Thus, we can avoid the use of arbitrary formula maps α_i as in Fig. 5 and [164], and instead rely entirely on (Grothendieck) signature morphisms. Actually, the above view is *not* provable— the ontologies do not match here. The view can, however, be proved after an appropriate revision of the ontology `Biblio_OWL`: an inverse of the role `hasArticle` needs to be introduced and used to restrict the class `Article` in the following way:

```
ObjectProperty: hasArticle
InverseOf: hasJournal

Class: Article
SubclassOf: author some Thing, title some string,
            hasJournal some Journal
```

4.2.2. From Ontology to Relational Scheme. Dually, assume we are given an ontology that is supposed to logically describe a database scheme.

$$\begin{tikzcd}
\texttt{Biblio\_OWL}' \arrow[r, dotted] & \texttt{Biblio\_RS}' \\
\texttt{Biblio\_OWL} \arrow[u, "def"] & \texttt{Biblio\_RS} \arrow[u, "def"']
\end{tikzcd}$$

FIGURE 19. A view from $\mathcal{OWL}$ to RELSCHEME.

We again use `Biblio_RS` and `Biblio_OWL`, as given in Fig. 18. We now have a heterogeneous refinement from `Biblio_OWL` to `Biblio_RS`, as depicted in Fig. 19, analogous to that of the previous section.

The view `Biblio_OWL_in_RS` is written in many-sorted first-order logic, using the notation of CASL (see [38]).

```
logic CASL
view Biblio_OWL_in_RS : Biblio_OWL to
 { Biblio_RS with logic RelScheme -> CASL
  then %def
   preds Researcher(x:pointer)            <=>
           (exists n:string; a:pointer.
            person(x,n) /\ author_of(x,a));
         Article(x:pointer)               <=>
            (exists t:string; j:integer.paper(x,t,j));
         Journal(x:pointer)               <=>
            (exists n:string; i:float.journal(x,n,i));
         name(x:pointer;n:string)         <=>
           person(x,n);
         hasArticle(x,j:pointer)          <=>
            (exists n,t:string; i:integer .
             journal(x,n,i) /\ paper(j,t,x));
         hasJournal(j,x:pointer)          <=>
            (exists n,t:string; i:integer .
             journal(x,n,i) /\ paper(j,t,x));
         author(a,p:pointer)              <=>
            (exists t,n:string; p:pointer .
             paper(a,t,p) /\ person(p,n));
         title(a:pointer; t:string)        <=>
            (exists p:pointer . paper(a,t,p));
         impact_factor(j:pointer;f:integer) <=>
            (exists n:string . journal(j,n,f));
  }
end
```

Such a specification can be parsed and analysed by the tool HETS, which displays it as a development graph as in Fig. 20.

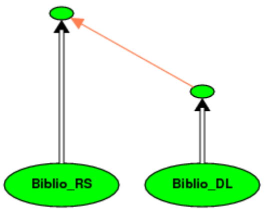

FIGURE 20. Development graph in HETS.

In this picture, red arrows are theorem links, and outlined thick black arrows are definition links. The picture shows the situation shown in the diagram of Fig. 19.

To prove this view, the development graph calculus has to be applied first. Then a prover needs to be invoked on the upper left node, that by now has become red: a node with proof obligations. The proof obligations can e.g. be proved with the SPASS theorem prover.

Related Work on Databases. Integrating DL reasoning and database reasoning is not trivial due partly to open world vs. closed world semantics, and is a very active area of research, e.g. using database technology to provide DL query answering. See e.g. [32, 144, 98] for recent work in the area. In [32], for instance, an approach to link ontologies and relational database systems is presented where ontologies are used for the conceptual layer and databases for the data layer. A DL with limited expressivity is defined which allows efficient reasoning with such a system. Similarly, [144] give an analysis of the formalisms of description logics and databases and study the problem of closed-world description logics semantics vs. open world database semantics. They argue that $\mathcal{A}$Boxes correspond to incomplete databases, introduce a language for integrity constraints for description logics, and investigate several methods of constraint satisfaction in this context.

Rather than contributing to such algorithmic problems, the example provided above shows how the structuring technique of refinements can be applied to specify consistency constraints between a database schema and an ontology. Our approach has a more general setting than the work mentioned above since we bring together several logics and reasoning tools, and the notion of (heterogeneous) refinement is universally applicable, independently of the specifics of the chosen formalisms, where institution comorphisms are used to establish corresponding logic translations. The example discussed using databases and ontologies heterogeneously just involves $\mathcal{T}$Boxes however. When dealing with $\mathcal{A}$Boxes, we would need to add an additional layer to the heterogeneous specifications, addressing the problems discussed in [144].

4.3. Reasoning About Integrations

Following the discussion in Section 3.2, integrations of ontologies into a common reference ontologies are expressed via two refinements into monomorphic extensions, written as views in HetCasl and visualised as theorem links in development graphs. In our example we use our known ontologies `Biblio_OWL` and `Biblio_RS` and integrate them into the first-order ontology about bibliographies from [164].

```
logic CASL

spec BiblioR =
     sorts Thing, DATA
     preds Nothing, Thing : Thing
     preds Researcher, Publication, Composite_Publication : Thing
     preds Working_Person, Tangible_Thing : Thing
     preds Journal, Article, Proceedings, Paper : Thing
     pred String, Integer : DATA
     pred Name, Title, Impact_Factor : Thing * DATA
     preds Has_Publication, Author, Has_Article, Has_Paper : Thing * Thing
     forall s,t:Thing
     . not (Nothing(t))
     . Thing(t)
     . Working_Person(t) => Tangible_Thing(t) /\
     exists y:DATA.(String(y) /\ Name(t,y))
     . Researcher(t)     => Working_Person(t)
     . Composite_Publication(t) => Tangible_Thing(t) /\
                                    exists y:DATA. (String(y) /\
   Name(t,y)) /\
                                    exists z:Thing. (Publication(z) /\
     Has_Publication(t,z))
     . Journal(t) => Composite_Publication(t) /\
                      exists y:Thing. (Article(y) /\ Has_Article(t,y)) /\
     exists i:DATA . (Integer(i) /\
      Impact_Factor(t,i))
     . Has_Article(s,t) => Has_Publication(s,t)
     . Proceedings(t) => Composite_Publication(t) /\
                          exists y:Thing. (Paper(y) /\ Has_Paper(t,y))
     . Has_Paper(s,t) => Has_Publication(s,t)
     . Publication(t) => Tangible_Thing(t) /\
                          exists y:Thing. (Researcher(y) /\ Author(t,y)) /\
  exists z:DATA. (String(z) /\ Title(t,z))
     . Article(t) => Publication(t)
     . Paper(t)   => Publication(t)
end

view OWL2R : Biblio_OWL to
     {BiblioR
      then %def
       preds
        researcher (p : Thing) <=> Researcher(p);
        name (p:Thing; n:DATA) <=> String(n) /\ Name(p,n);
        article(a:Thing) <=> Publication(a);
        author(a:Thing; p:Thing) <=> Researcher(p) /\ Article(a) /\
                                     Author(a,p) /\
                                     exists j:Thing . (Journal(j) /\
       Has_Article(j,a));
        hasArticle(j:Thing; a:Thing) <=> Journal(j) /\ Has_Article(j,a) /\
                                         Publication(a) /\ exists n:DATA .
                                         (String(n) /\ Name(j, n)) /\ exists
                                         i:DATA . (Integer(i) /\ Impact_Factor(j,i));
        hasJournal(a:Thing; j:Thing) <=> Journal(j) /\ Has_Article(j,a) /\
                                         Publication(a) /\ exists n:DATA .
                                         (String(n) /\ Name(j, n)) /\ exists
                                         i:DATA . (Integer(i) /\ Impact_Factor(j,i));
        impactFactor(j:Thing; i:DATA) <=> Journal(j) /\ Integer(i) /\
                                             Impact_Factor(j,i) /\ exists n:DATA .
                                             (String(n) /\ Name(j,n)) /\ exists
                                             i:DATA . (Integer(i) /\ Impact_Factor(j,i));
        journal(j:Thing) <=> Journal(j);
        title(a:Thing;t:DATA) <=> String(t) /\ Article(a) /\ Title(a,t) /\
                                   exists p:Thing . (Researcher(p) /\
                                                    author(a,p)) /\
                                   exists j:Thing . (Journal(j) /\
                                                     Has_Article(a,j))
     }
     end

view RS2R  : {Biblio_RS  with pointer |-> Thing, integer |-> DATA, string |-> DATA
      hide author_of_paper, author_of_person, journal_id, journal_impact_factor,
      journal_name, paper_id, paper_published_in, paper_title,
      person_id, person_name} to
     {BiblioR
      then %def
       preds
        person(p:Thing; n:DATA) <=> Researcher(p) /\ Name(p,n) /\ String(n);
        author_of(p:Thing; a:Thing) <=> Author(a,p) /\ Article(a) /\ Researcher(p);
        paper(a:Thing; n:DATA; j:Thing) <=> Article(a) /\ String(n) /\ Journal(j) /\
                                              Title(a,n) /\ Has_Article(j,a);
        journal(j:Thing; n:DATA; i:DATA) <=> Journal(j) /\ String(n) /\ Integer(i) /\
                                             Name(j,n) /\ Impact_Factor(j,i);
     }
```

The mappings from `Biblio_OWL` and `Biblio_RS` to `BiblioR` respectively are implemented via logical equivalences. HETS displays the development graph of this specification as shown in Fig. 21.

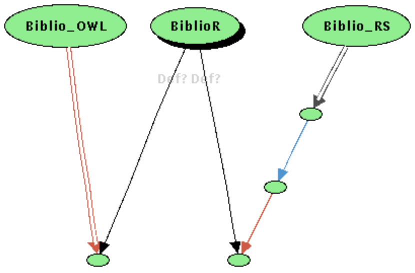

FIGURE 21. Development graph in HETS.

Unnamed nodes in the development graph correspond to definitional extensions. Note that after translating `Biblio_RS` to first-order logic, we need to hide some auxiliary operations that are generated by the comorphism. HETS's proof calculus for the development graph can be applied to such a graph as shown in Fig. 21 shifting the proof obligation between nodes into the nodes at their arrows. A prover for the specific logic of those nodes (in our case first-order logic) can be applied to those nodes to prove the correctness of the integration.

4.4. Reasoning About Connections

HETS also offers an algorithmic method for computing colimits of theories in various logics, based on an implementation for computing colimits of arbitrary sets, which is further applied to sets of signature symbols, like sorts, operation and predicate symbols (the latter two divided according to profiles). As a general strategy, names are kept identical to their original as far as possible (see the example below). If this is not possible, the common origin of symbols is indicated by a (shared) number appended to their name.

Example. Considering the V-alignment introduced in Example 3.3.1, Figure 22 presents the HETS concept graphs of the theories combining it, as well as the one of the push-out ontology obtained with HETS (the top one).

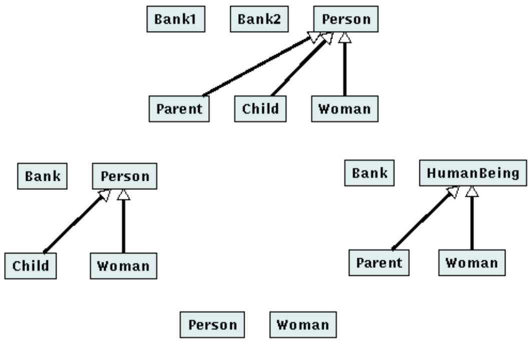

FIGURE 22. Colimit of a V-alignment in HETS.

⊣

The construction of colimits for heterogeneous diagrams is considerably more difficult. We refer the reader to [137, 37] for a detailed analysis of sufficient conditions for obtaining colimits of heterogeneous theories, and for a discussion of weaker notions that are useful in cases where heterogeneous colimits do not exist.

4.4.1. Integration through Interface and Colimit. In this section, an example specification for a W-alignment is given. We assume that the ontology `Biblio_OWL` is split into its signature `Biblio_OWL_Sign`, of which `Biblio_OWL` is an extension. Likewise, `Biblio_RS_Sign` is the signature of `Biblio_RS`. The following specification provides a first-order bridge between the signatures of the two ontologies, without relying on their axioms. The integration with the axioms is obtained via a colimit.

```
logic CASL
spec Interface =
   {Biblio_RS_Sign with logic RelScheme -> CASL}
and
   {Biblio_OWL_Sign with logic OWL -> CASL
```

```
    with Thing |-> pointer}
then
   forall a,j,p,x:pointer;n,t:string;f:integer
  . journal(j,n,f) <=> Journal(j) /\ name(j,n)
    /\ impactFactor(j,f)
  . paper(a,t,j) <=> Article(a) /\ Journal(j)
    /\ hasArticle(j,a) /\ title(a,t)
  . author_of(p,a) <=> Researcher(p) /\ Article(a)
    /\ author(p,a)
  . person(p,n) <=> Researcher(p) /\ name(p,n)
  . Researcher(x) <=> (exists q:pointer;m:string .
    person(x,m) /\ author(x,q))
  . Article(x) <=> (exists q:pointer;m:string .
    paper(x,m,q))
  . Journal(x) <=> (exists m:string;i:integer .
    journal(x,n,i))
end
```

The picture in Figure 23 shows the development graph of the above specification in HETS.

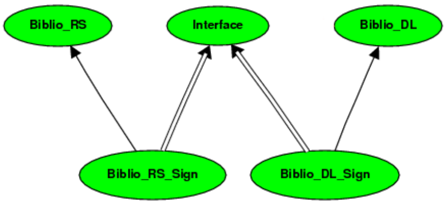

FIGURE 23. W-alignment in HETS.

In this figure, the outlined thick black arrows are heterogeneous definition links, while the normal black arrows are non-heterogeneous definition links. The picture shows the situation depicted in the diagram of Fig. 7. Its colimit can be automatically computed by HETS.

In the previous sections, we have analysed in detail a specific example heterogeneous ontology from the literature, consisting of a bibliographical database and a related ontology. We have shown that these ontologies can be heterogeneously combined in different ways:

1. via *refinement*, which provides the strongest relation between the ontologies: namely, that each bibliographical database satisfying the given relational scheme's integrity constraints also gives rise to a model of the ontology, and vice versa (after some appropriate extension of the ontology).
2. via *integration* in a common reference ontology, which is known beforehand, and
3. via *connection* through a bridge theory, which operates directly on the involved ontologies and allows the *automatic* construction of a combined ontology via colimits.

It is not surprising that the strong combination via refinement is only possible after extending the ontology in a suitable way, leading to a better matching between the relational scheme and the ontology. Certainly, these general patterns of ontology combination can also be found in other examples, also involving completely different application domains (and not necessarily being connected with databases).

5. Discussion and Outlook

In the introduction, we formulated three questions—from the ontology design and reasoning perspective—, and informally sketched our answers to these questions. The core of this paper developed a framework for ontology design that provides the theoretical underpinnings and gave technical argument for these answers. Briefly, they can be summarised as follows:

- **Is there a universal ontology?** No. Different application domains require different levels of expressivity and abstraction in an ontology. While 'universal' attempts to capture ontological or commonsense knowledge such as the Cyc ontology have produced impressive results, they are difficult to adapt to specific purposes and application areas. The need for diversity is nowadays acknowledged even in the area of foundational ontologies that provide very high-level ontological vocabulary and axiomatisation, such as BFO, GFO, and Dolce—they are directed towards different application areas such as Biology, Medicine, or general Information Systems, and no easy integration is conceivable as they disagree on a high conceptual level. Moreover, real world applications of ontologies often require the heterogeneous combination of ontologies.
- **Is there a universal ontology language?** There is no 'one true ontology language' that will fit all purposes. The choice of an ontology language is directly linked to a corresponding logic and a restriction to possible reasoning support, from very fast lightweight DL reasoning to (semi-decidable and semi-automatic) first-order and higher-order reasoning; this concerns in particular a trade-off between axiomatic expressiveness and dealing efficiently with large amounts of data (as e.g. in biomedical ontologies).
- **Is there a universal ontology reasoning?** In the highly populated world of heterogeneous ontologies, not only the languages may vary from module to module, but also the way the modules interact with each other and the way the 'flow of information' [91] is controlled. This necessitates proper, semantically well-founded foundations for various kinds of inter-ontology links. Moreover, such a heterogeneous set-up obviously affects the various reasoning modes that can and need to be supported. This includes combination of conceptual reasoning with, e.g., temporal, spatial, or epistemic information, as well as dealing with problems such as inconsistency tolerance employing paraconsistent inference, non-monotonic inference dealing with changing facts that need to be accommodated, or fuzzy and approximate reasoning in cases where 'precise' reasoning is either too expensive or just undesirable.

Taking these general answers to basic methodological questions concerning ontological engineering and reasoning as our starting point, we have presented the fundamentals of a framework for ontology design that rests on two main principles, namely an endorsement of (onto-)logical pluralism for ontology design in the spirit of Carnap's logical tolerance, and an adoption of Goguen's institutional semantics as a backbone for heterogeneous structuring and combination—this approach we have termed **Carnapian Goguenism**.

The abstract framework for the study of structured heterogeneous ontologies that we have introduced allows a systematic analysis of conceptual and algorithmic problems in heterogeneous environments that were previously considered rather disparate. Essentially all logics used in ontology design today can be rendered as an institution, and the heterogeneous structuring devices that we have described allow ontology designers to build their ontologies accordingly in a modular, heterogeneous and structured fashion, splitting the overall design up into several meaningful modules and joining them together (in various ways) to make up the overall ontology. Here, the heterogeneous approach allows to define various parts of an ontology in different logics, depending on the needed expressivity and desired reasoning support.

Furthermore, the approach allows in particular a description and classification, in an abstract way, of most ontology combination and integration approaches known from the literature, namely alignments, integrations into reference ontologies, loose combinations of ontologies such as DDLs and $\mathcal{E}$-connections, as well as refinements. The notion of heterogeneous refinement has in particular been employed to define a general notion of sub-ontology, where the super-ontology might not only be 'larger' or axiomatised differently, but indeed might be formalised in a different logic. This provides a powerful, syntax-agnostic approach to the comparison of ontologies across different institutions. Indeed, it can also be employed to define notions of equivalence, or synonymity, between ontologies formulated in different logics. Unlike related approaches like Common Logic [39], our approach provides *explicit* structuring mechanisms, and logic translations are treated as first-class citizens. Moreover, as we have discussed in Section 2.1 (Example 2.1.1), due to the highly modular and extensible architecture of HETS, Common Logic was added as a new logic to the HETS logic graph with relative ease.

Indeed, the structured reasoning support that our approach allows has already been used for answering questions that 'standard' automated reasoning can not tackle: the consistency of the first-order version of the foundational ontology DOLCE (reformulated as a HETCASL specification) can be verified by model-checking a view into a finite specification of a model for DOLCE: here, the structuring techniques supported by HETS allow for a modular construction of models for large first-order ontologies such as DOLCE [104] (called architectural specifications).

A general challenge that will need to be addressed in the future is the integration of various ontology tools, editors and reasoners, as well as their interoperability with large ontology repositories that contain ontologies in various formalisms. To mention just a few, the COLORE repository maintains Common Logic ontologies[40], the ORATE repository[41] mostly lightweight ontologies related to assistive ontologies, BioPortal[42] ontologies for the biomedical communities, and the TONES ontology repository[43] $\mathcal{OWL}$ ontologies with varying expressivity intended mostly for tool developers.

Our relativistic approach to ontology design, and the theoretical framework outlined in this paper suggest a picture that, we believe, will eventually supersede monolithic ontology design as well as earlier 'universal' approaches to capturing ontological knowledge, such as the Cyc effort. Similar to the World Wide Web itself, **hyperontologies**[44] are envisaged to be highly modular, richly structured and interlinked resources of formalised ontological knowledge. Some of the key desiderata for hyperontologies will be semantic versioning control and evolution, and ways to retrieve and extract ontological knowledge based not only on formal logical principles, but also steered by application area, level of abstraction and granularity, and intended reasoning scenario. This extends the Semantic Web vision in at least the following two aspects: (1) (various) links between ontologies become first-class citizens and are themselves endowed with rich semantics, and (2) different formalisms can be heterogeneously connected.

Concrete future work in this direction includes an integration of HETS with the PROTÉGÉ system[45], incorporating a tool for the discovery of theory morphisms (as developed in [149]) into the Heterogeneous Tool Set, as well as adding various modularisation algorithms as developed e.g. in [97, 40, 43]. Moreover, we are experimenting with combining statistics-based matching algorithms for ontologies with logic-based modularisation techniques and category-theory-based automatic alignment construction, applied to large repositories of ontologies. Such techniques give a first handle on important operations on hyperontologies, such as to allow for the extraction of thematically related ontology modules from across several ontologies, the (semi)-automatic structuring of ontologies, and the discovery of ontology overlaps modulo alignment mappings.

Acknowledgements

Work on this paper has been supported by the DFG-funded collaborative research center SFB/TR 8 'Spatial Cognition' and by the German Federal Ministry of Education and Research (Project 01 IW 07002 FormalSafe).

[40] See `http://stl.mie.utoronto.ca/colore/`

[41] See `http://ontologies.informatik.uni-bremen.de/`

[42] See `http://bioportal.bioontology.org/`

[43] See `http://owl.cs.manchester.ac.uk/repository/`

[44] The concept of a hyperontology is also central in the European OASIS project [16, 9], see `http://www.oasis-project.eu/docs/OFFICIAL_DELIVERABLES/SP1/D1.2.1/OASIS-D121.pdf`.

[45] See `http://protege.stanford.edu/`

We thank John Bateman, Stefano Borgo, Mihai Codescu, Michael Grüninger, Joana Hois, Immanuel Normann, and Lutz Schröder for fruitful discussions.

References

[1] Adámek, J., Herrlich, H., and Strecker, G. *Abstract and Concrete Categories.* Wiley, New York, 1990. Freely available at `http://www.math.uni-bremen.de/~dmb/acc.pdf`.

[2] Alagić, S., and Bernstein, P. A. A Model Theory for Generic Schema Management. In *Proc. of DBPL-01* (2002), vol. 2397 of *LNCS*, Springer, pp. 228–246.

[3] Artale, A., and Franconi, E. A survey of temporal extensions of description logics. *Annals of Mathematics and Artificial Intelligence 30*, 1-4 (2000), 171–210.

[4] Artale, A., Kontchakov, R., Lutz, C., Wolter, F., and Zakharyaschev, M. Temporalising tractable description logics. In *Proc. of the 14th Int. Symposium on Temporal Representation and Reasoning (TIME)* (Washington, DC, USA, 2007), IEEE, pp. 11–22.

[5] Astesiano, E., Kreowski, H.-J., and Krieg-Brückner, B. *Algebraic Foundations of Systems Specification.* Springer, 1999.

[6] Baader, F., Calvanese, D., McGuinness, D., Nardi, D., and Patel-Schneider, P. F., Eds. *The Description Logic Handbook.* Cambridge University Press, 2003.

[7] Baader, F., and Ghilardi, S. Connecting Many-Sorted Theories. *The Journal of Symbolic Logic 72*, 2 (2007), 535–583.

[8] Baader, F., Lutz, C., Milicic, M., Sattler, U., and Wolter, F. Integrating Description Logics and Action Formalisms: First Results. In *Proceedings of the Twentieth National Conference on Artificial Intelligence (AAAI-05)* (Pittsburgh, PA, USA, 2005).

[9] Bateman, J., Castro, A., Normann, I., Pera, O., Garcia, L., and Villaveces, J.-M. OASIS common hyper-ontological framework (COF). Deliverable D1.2.1, EU Project OASIS, 2010.

[10] Bateman, J., Hois, J., Ross, R., and Tenbrink, T. A Linguistic Ontology of Space for Natural Language Processing. *Artificial Intelligence* (2010). To appear.

[11] Bateman, J., Tenbrink, T., and Farrar, S. The Role of Conceptual and Linguistic Ontologies in Discourse. *Discourse Processes 44*, 3 (2007), 175–213.

[12] Bateman, J. A. Ontological diversity: the case from space. In *Formal Ontology in Information Systems - Proceedings of the Sixth International Conference (FOIS 2010)*, A. Galton and R. Mizoguchi, Eds., vol. 209. IOS Press, 2010.

[13] Baumgartner, P., and Suchanek, F. M. Automated reasoning support for first-order ontologies. In *Principles and Practice of Semantic Web Reasoning 4th International Workshop (PPSWR 2006),*

Revised Selected Papers (2006), J. Alferes, J. Bailey, W. May, and U. Schwertel, Eds., vol. 4187 of *LNAI*, Springer.

[14] Beall, J., and Restall, G. Defending Logical Pluralism. In *Logical Consequences: Rival Approaches. Proceedings of the 1999 Conference of the Society of Exact Philosophy*, B. Brown and J. Woods, Eds. Stanmore, Hermes, 2001.

[15] Beall, J., and Restall, G. *Logical Pluralism.* Clarendon Press, Oxford, 2006.

[16] Bekiaris, E., and Bonfiglio, S. The OASIS Concept. In *Universal Access in Human-Computer Interaction. Addressing Diversity* (2009), C. Stephanidis, Ed., vol. 5614 of *Lecture Notes in Computer Science*, Springer, pp. 202–209.

[17] Belnap, N. Under Carnap's Lamp: Flat Pre-semantics. *Studia Logica 80*, 1 (2005), 1–28.

[18] Belnap, N. D. How a computer should think. In *Contemporary Aspects of Philosophy*, G. Ryle, Ed. Oriel Press, 1977.

[19] Belnap, N. D. A useful four-valued logic. In *Modern Uses of Multiple-Valued Logics*, J. Dunn and G. Epstein, Eds. Reidel, Dordrecht, 1977, pp. 8–37.

[20] Bench-Capon, T. J. M., and Malcolm, G. Formalising Ontologies and Their Relations. In *Proc. of DEXA-99* (1999), vol. 1677 of *LNCS*, Springer, pp. 250–259.

[21] Bennett, B. Modal logics for qualitative spatial reasoning. *Journal of the Interest Group on Pure and Applied Logic 4* (1996), 23–45.

[22] Béziau, J.-Y., Ed. *Logica Universalis: Towards a General Theory of Logic.* Birkhäuser Verlag, Basel, 2005.

[23] Bhatt, M., Dylla, F., and Hois, J. Spatio-Terminological Inference for the Design of Ambient Environments. In *Conference on Spatial Information Theory (COSIT'09)* (2009), K. S. Hornsby, C. Claramunt, M. Denis, and G. Ligozat, Eds., Springer-Verlag, pp. 371–391.

[24] Bidoit, M., and Mosses, P. D. Casl *User Manual.* LNCS Vol. 2900 (IFIP Series). Springer, 2004.

[25] Bittner, T., and Donnelly, M. Computational ontologies of parthood, componenthood, and containment. In *IJCAI* (2005), L. P. Kaelbling and A. Saffiotti, Eds., Professional Book Center, pp. 382–387.

[26] Borgida, A. On the Relative Expressiveness of Description Logics and Predicate Logics. *Artificial Intelligence 82*, 1–2 (1996), 353–367.

[27] Borgida, A., and Serafini, L. Distributed Description Logics: Assimilating Information from Peer Sources. *Journal of Data Semantics 1* (2003), 153–184.

[28] Borzyszkowski, T. Higher-order logic and theorem proving for structured specifications. In *WADT* (1999), D. Bert, C. Choppy, and P. D. Mosses, Eds., vol. 1827 of *Lecture Notes in Computer Science*, Springer, pp. 401–418.

[29] Brachman, R. J. On the Epistemological Status of Semantic Networks. In *Associative Networks: Representation and Use of Knowledge by Computers*, N. V. Findler, Ed. Academic Press, 1979.

[30] Braüner, T., and Ghilardi, S. First-Order Modal Logic. In *Handbook of Modal Logic*, J. v. Benthem, P. Blackburn, and F. Wolter, Eds. Elsevier, Amsterdam, 2006.

[31] Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., and Rosati, R. Epistemic first-order queries over description logic knowledge bases. In *Proc. of the 2006 Description Logic Workshop (DL 2006)* (2006), vol. 189 of *CEUR Electronic Workshop Proceedings, http://ceur-ws.org/Vol-189/*.

[32] Calvanese, D., Giacomo, G. D., Lembo, D., Lenzerini, M., Poggi, A., and Rosati, R. Ontology-based database access. In *Proc. of SEBD* (2007), pp. 324–331.

[33] Carnap, R. *Logische Syntax der Sprache.* Kegan Paul, 1934. English translation 1937, *The Logical Syntax of Language.*

[34] Carnap, R. Empiricism, Semantics, and Ontology. *Revue Internationale de Philosophie 4* (1950), 20–40.

[35] Carnap, R. Intellectual Autobiography. In *The Philosophy of Rudolf Carnap*, P. A. Schilpp, Ed., vol. 11 of *The Library of Living Philosophers.* Open Court, La Salle, IL, 1963.

[36] Church, A. A Formulation of the Simple Theory of Types. *Journal of Symbolic Logic 5*, 1 (1940), 56–69.

[37] Codescu, M., and Mossakowski, T. Heterogeneous colimits. In *MoVaH'08 Workshop on Modeling, Validation and Heterogeneity* (2008), F. Boulanger, C. Gaston, and P.-Y. Schobbens, Eds.

[38] CoFI (The Common Framework Initiative). Casl *Reference Manual.* LNCS Vol. 2960 (IFIP Series). Springer, 2004. Available at `http://www.cofi.info`.

[39] Common Logic Working Group. Common Logic: Abstract syntax and semantics. Tech. rep., 2003.

[40] Cuenca Grau, B., Horrocks, I., Kazakov, Y., and Sattler, U. Modular Reuse of Ontologies: Theory and Practice. *Journal of Artificial Intelligence Research (JAIR) 31* (2008), 273–318.

[41] Cuenca Grau, B., Parsia, B., and Sirin, E. Ontology Integration Using $\mathcal{E}$-connections. In *Ontology Modularization*, H. Stuckenschmidt and S. Spaccapietra, Eds. Springer, 2009. To Appear.

[42] de Bouvère, K. Logical Synonymity. *Indagationes Mathematicae 27* (1965), 622–629.

[43] Del Vescovo, C., Parsia, B., Sattler, U., and Schneider, T. The modular structure of an ontology: an empirical study. In *Modular Ontologies—Proceedings of the Fourth International Workshop (WoMO 2010)* (Toronto, Canada, 2010), O. Kutz, J. Hois, J. Bao, and B. Cuenca Grau, Eds., vol. 210 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, pp. 11–24.

[44] Delugach, H. S. Towards Conceptual Structures Interoperability Using Common Logic. In *Proc. of the Third Conceptual Structures Tool Interoperability Workshop, held at the 16th International Conference on Conceptual Structures (ICCS 2008), July 7, 2008, UTM (Université Toulouse Le Mirail), Toulouse, France)* (2008), M. Croitoru, R. Jäschke, and S. Rudolph, Eds.

[45] Diaconescu, R. Grothendieck institutions. *Applied Categorical Structures 10* (2002), 383–402.

[46] Diaconescu, R. *Institution-independent Model Theory.* Studies in Universal Logic. Birkhäuser, 2008.

[47] Diaconescu, R., Goguen, J., and Stefaneas, P. Logical Support for Modularisation. In *Papers presented at the second annual Workshop on Logical environments, Edinburgh, Scotland*, G. Huet and G. Plotkin, Eds. Cambridge University Press, New York, 1993, pp. 83–130.

[48] Donini, F. M., Lenzerini, M., Nardi, D., Nutt, W., and Schaerf, A. An epistemic operator for description logics. *Artif. Intell. 100*, 1-2 (1998), 225–274.

[49] Dou, D., and McDermot, D. Towards theory translation. In *Declarative Agent Languages and Technologies IV* (2007), Springer.

[50] Enderton, H. B. *A Mathematical Introduction to Logic.* Academic Press, 1972.

[51] Euzenat, J., and Shvaiko, P. *Ontology Matching.* Springer, Heidelberg, 2007.

[52] Evans, M. Can there be vague objects? *Analysis 38* (1978), 208. reprinted in his *Collected Papers*, Oxford, Clarendon Press 1985.

[53] Feferman, S. Hilbert's program relativized: Proof-theoretical and foundational reductions. *The Journal of Symbolic Logic 53*, 2 (1988), 364–384.

[54] Fitting, M., and Mendelsohn, R. L. *First–Order Modal Logic.* Kluwer Academic Publishers, Dordrecht, 1998.

[55] Freksa, C. Using orientation information for qualitative spatial reasoning. In *Theories and methods of spatio-temporal reasoning in geographic space*, vol. 639 of *LNCS.* Springer, 1992, pp. 162–178.

[56] Gabbay, D. *Fibring Logics*, vol. 38 of *Oxford Logic Guides.* Clarendon Press, Oxford, 1999.

[57] Gabbay, D., Kurucz, A., Wolter, F., and Zakharyaschev, M. *Many-Dimensional Modal Logics: Theory and Applications.* No. 148 in Studies in Logic and the Foundations ofMathematics. Elsevier Science Publishers, Amsterdam, 2003.

[58] Gangemi, A., Guarino, N., Masolo, C., Oltramari, A., and Schneider, L. Sweetening Ontologies with dolce. In *Proc. of EKAW 2002* (2002), LNCS Vol. 2473, Springer, pp. 166–181.

[59] Gärdenfors, P. *Conceptual Spaces - The Geometry of Thought.* Bradford Books. MIT Press, 2000.

[60] Gardner, M. *Logic Machines and Diagrams.* McGraw-Hill, 1958.

[61] Genesereth, M. R., and Nilsson, N. J. *Logical Foundations of Artificial Intelligence.* Morgan Kaufmann, Los Altos, 1987.

[62] Goguen, J. A. A Categorical Manifesto. *Mathematical Structures in Computer Science 1* (1991), 49–67.

[63] Goguen, J. A. Ontology, Society, and Ontotheology. In *Formal Ontology in Information Systems: Proceedings of the Third International Conference (FOIS-2004)* (2004), A. C. Varzi and L. Vieu, Eds., Frontiers in Artificial Intelligence and Applications, IOS Press, pp. 95–105.

[64] Goguen, J. A. Data, Schema, Ontology and Logic Integration. *Logic Journal of the IGPL 13*, 6 (2005), 685–715.

[65] Goguen, J. A. Information Integration in Institutions. In *Jon Barwise Memorial Volume*, L. Moss, Ed. Indiana University Press, 2006. To appear.

[66] Goguen, J. A., and Burstall, R. M. Introducing institutions. In *Proc. Logics of Programming Workshop* (1984), E. Clarke and D. Kozen, Eds., vol. 164 of *LNCS*, Springer, pp. 221–256.

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

[68] Goguen, J. A., and Roşu, G. Institution morphisms. *Formal aspects of computing 13* (2002), 274–307.

[69] Grau, B., Parsia, B., Sirin, E., and Kalyanpur, A. Automatic Partitioning of OWL Ontologies Using $\mathcal{E}$-connections. In *Proc. of Description Logic Workshop (DL)* (2005).

[70] Grau, B. C., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., and Sattler, U. OWL 2: The next step for OWL. *Web Semantics: Science, Services and Agents on the World Wide Web 6*, 4 (2008), 309–322. Semantic Web Challenge 2006/2007.

[71] Grenon, P., Smith, B., and Goldberg, L. Biodynamic Ontology: Applying BFO in the Biomedical Domain. In *Ontologies in Medicine*, D. M. Pisanelli, Ed. IOS Press, Amsterdam, 2004, pp. 20–38.

[72] Gruber, T. R. Toward Principles for the Design of Ontologies Used for Knowledge Sharing. *International Journal of Human-Computer Studies 43*, 4-5 (1995), 907–928.

[73] Grüninger, M., Hahmann, T., Hashemi, A., and Ong, D. Ontology Verification with Repositories. In *Formal Ontology in Information Systems—Proceedings of the Sixth International Conference (FOIS-2010)* (2010), A. Galton and R. Mizoguchi, Eds., vol. 209 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, pp. 317–330.

[74] Guarino, N. The ontological level. In *Philosophy and the Cognitive Sciences* (1994), R. Casati, B. Smith, and G. White, Eds., Hölder-Pichler-Tempsky, pp. 443–456. Proc. of the 16th Wittgenstein Symposium, Kirchberg, Austria, Vienna, August 1993.

[75] Guarino, N. Formal Ontology and Information Systems. In *Formal Ontology in Information Systems, Proc. of FOIS-98, Trento, Italy, June*

6–8 (Amsterdam, 1998), N. Guarino, Ed., IOS Press, pp. 3–15.

[76] Guarino, N. The Ontological Level: Revisiting 30 Years of Knowledge Representation. In *Conceptual Modelling: Foundations and Applications. Essays in Honor of John Mylopoulos*, A. Borgida, V. Chaudhri, P. Giorgini, and E. Yu, Eds. Springer Verlag, 2009, pp. 52–67.

[77] Guarino, N., and Giaretta, P. Ontologies and Knowledge Bases: Towards a Terminological Clarification. In *Towards Very Large Knowledge Bases: Knowledge Building and Knowledge Sharing*, N. Mars, Ed. IOS Press, Amsterdam, 1995, pp. 25–32.

[78] Guarino, N., and Welty, C. Evaluating ontological decisions with OntoClean. *Commun. ACM 45*, 2 (2002), 61–65.

[79] Guerra, S. Composition of Default Specifications. *J. Log. Comput. 11*, 4 (2001), 559–578.

[80] Guizzardi, G. Modal Aspects of Object Types and Part-Whole Relations and the *de re/de dicto* Distinction. In *Advanced Information Systems Engineering, 19th International Conference (CAiSE-07)* (2007), J. Krogstie, A. L. Opdahl, and G. Sindre, Eds., vol. 4495 of *Lecture Notes in Computer Science*, Springer, pp. 5–20.

[81] Haack, S. *Philosophy of Logics.* Cambridge University Press, 1978.

[82] Haack, S. *Deviant Logic, Fuzzy Logic: Beyond the Formalism.* Cambridge University Press, 1996.

[83] Haase, P., van Harmelen, F., Huang, Z., Stuckenschmidt, H., and Sure, Y. A framework for handling inconsistency in changing ontologies. In *Proc. of the 4th International Semantic Web Conference (ISWC-05)* (2005), vol. 3729 of *LNCS*, Springer, pp. 353–367.

[84] Heller, B., and Herre, H. Ontological Categories in GOL. *Axiomathes 14*, 1–3 (2004), 57–76.

[85] Herre, H. The Ontology of Mereological Systems. In *Theory and Applications of Ontology - Volume 1: Philosophical Perspectives*, R. Poli, J. Seibt, M. Healy, and A. Kameas, Eds. Springer, 2010.

[86] Hois, J., Bhatt, M., and Kutz, O. Modular Ontologies for Architectural Design. In *Proc. of the 4th Workshop on Formal Ontologies Meet Industry, FOMI-09, Vicenza, Italy* (2009), vol. 198 of *Frontiers in Artificial Intelligence and Applications*, IOS Press.

[87] Hois, J., and Kutz, O. Counterparts in Language and Space—Similarity and $\mathcal{S}$-Connection. In *Formal Ontology in Information Systems (FOIS 2008)* (2008), C. Eschenbach and M. Grüninger, Eds., IOS Press, pp. 266–279.

[88] Hois, J., and Kutz, O. Natural Language meets Spatial Calculi. In *Spatial Cognition VI. Learning, Reasoning, and Talking about Space. 6th International Conference on Spatial Cognition* (2008), C. Freksa, N. S. Newcombe, P. Gärdenfors, and S. Wölfl, Eds., LNCS, Springer, pp. 266–282.

[89] Horridge, M., Drummond, N., Goodwin, J., Rector, A., Stevens, R., and Wang, H. H. The Manchester OWL Syntax. In

OWL: Experiences and Directions (2006).

[90] Horrocks, I., and Kutz, O., and Sattler, U. The Even More Irresistible $\mathcal{SROIQ}$. In *Proc. of the 10th Int. Conf. on Principles of Knowledge Representation and Reasoning (KR2006)* (June 2006), AAAI Press, pp. 57–67.

[91] Kalfoglou, Y., and Schorlemmer, M. The Information Flow Approach to Ontology-Based Semantic Alignment. In *Theory and Applications of Ontology: Computer Applications*, R. Poli, M. Healy, and A. Kameas, Eds. Springer, 2010.

[92] Kalyanpur, A., Parsia, B., Horridge, M., and Sirin, E. Finding all Justifications of OWL DL Entailments. In *Proc. of ISWC/ASWC2007* (2007), LNCS Vol. 4825, Springer, pp. 267–280.

[93] Kazakov, Y. An Extension of Regularity Conditions for Complex Role Inclusion Axioms. In *Proc. of DL-09* (2009), B. C. Grau, I. Horrocks, B. Motik, and U. Sattler, Eds., vol. 477 of *CEUR Workshop Proceedings*, CEUR-WS.org.

[94] Keet, C. M., and Artale, A. Representing and Reasoning over a Taxonomy of Part-Whole Relations. *Applied Ontology 3*, 1-2 (2008), 91–110.

[95] Klinov, P., and Mazlack, L. J. On possible applications of rough mereology to handling granularity in ontological knowledge. In *Proceedings of the 22nd National Conference on Artificial Intelligence (AAAI-07)* (2007), AAAI Press, pp. 1876–1877.

[96] Konev, B., Lutz, C., Walther, D., and Wolter, F. Formal properties of modularization. In *Ontology Modularization*, H. Stuckenschmidt and S. Spaccapietra, Eds. Springer, 2008.

[97] Konev, B., Lutz, C., Walther, D., and Wolter, F. Semantic Modularity and Module Extraction in Description Logics. In *18th European Conf. on Artificial Intelligence (ECAI-08)* (2008).

[98] Kontchakov, R., Lutz, C., Toman, D., Wolter, F., and Zakharyaschev, M. The Combined Approach to Query Answering in DL-Lite. In *Proceedings of the 12th International Conference on Principles of Knowledge Representation and Reasoning (KR2010)* (2010), F. Lin and U. Sattler, Eds., AAAI Press.

[99] Kotas, J., and Pieczkowski, A. Allgemeine logische und mathematische Theorien. *Zeitschrift für Mathematische Logik und Grundlagen der Mathematik (now 'Mathematical Logic Quarterly') 16*, 6 (1970), 353–376.

[100] Kracht, M., and Kutz, O. Logically Possible Worlds and Counterpart Semantics for Modal Logic. In *Philosophy of Logic, Handbook of the Philosophy of Science*, D. Jacquette, Ed., vol. 5. Elsevier, Amsterdam, 2007, pp. 943–996.

[101] Kutz, O. *$\mathcal{E}$-Connections and Logics of Distance.* PhD thesis, The University of Liverpool, 2004.

[102] Kutz, O. Notes on Logics of Metric Spaces. *Studia Logica 85*, 1 (2007),

75–104.

[103] Kutz, O., Lücke, D., and Mossakowski, T. Heterogeneously Structured Ontologies—Integration, Connection, and Refinement. In *Advances in Ontologies. Proceedings of the Knowledge Representation Ontology Workshop (KROW 2008)* (Sydney, Australia, 2008), T. Meyer and M. A. Orgun, Eds., vol. 90 of *CRPIT*, ACS, pp. 41–50.

[104] Kutz, O., Lücke, D., and Mossakowski, T. Modular Construction of Models—Towards a Consistency Proof for the Foundational Ontology Dolce. In *1st Int. Workshop on Computer Science as Logic-Related* (ICTAC 2008, Istanbul, Turkey, 2008).

[105] Kutz, O., Lücke, D., Mossakowski, T., and Normann, I. The OWL in the Casl—Designing Ontologies Across Logics. In *OWL: Experiences and Directions, 5th International Workshop (OWLED-08)* (co-located with ISWC-08, Karlsruhe, Germany, October 26–27, 2008), C. Dolbear, A. Ruttenberg, and U. Sattler, Eds., CEUR-WS, Vol-432.

[106] Kutz, O., Lutz, C., Wolter, F., and Zakharyaschev, M. $\mathcal{E}$-Connections of Abstract Description Systems. *Artificial Intelligence 156*, 1 (2004), 1–73.

[107] Kutz, O., and Mossakowski, T. Modules in Transition: Conservativity, Composition, and Colimits. In *2nd Int. Workshop on Modular Ontologies (WoMO-07)* (2007). K-CAP, Whistler BC, Canada.

[108] Kutz, O., and Mossakowski, T. Conservativity in Structured Ontologies. In *18th European Conf. on Artificial Intelligence (ECAI-08)* (Patras, Greece, 2008), IOS Press.

[109] Kutz, O., Mossakowski, T., and Codescu, M. Shapes of Alignments: Construction, Combination, and Computation. In *Proc of the 1st Workshop on Ontologies: Reasoning and Modularity (WORM-08)* (ESWC, Tenerife, Spain, 2008), U. Sattler and A. Tamilin, Eds., CEUR-WS, Vol-348.

[110] Kutz, O., and Normann, I. Context Discovery via Theory Interpretation. In *Proc. of the IJCAI Workshop on Automated Reasoning about Context and Ontology Evolution, ARCOE-09, Pasadena, California* (2009).

[111] Kutz, O., Wolter, F., and Zakharyaschev, M. Connecting abstract description systems. In *Proc. of the 8th Conference on Principles of Knowledge Representation and Reasoning (KR-02)* (2002), Morgan Kaufmann, pp. 215–226.

[112] Leibniz, G. W. *Sämtliche Schriften und Briefe—VI Sektion: Philosophische Schriften, Band IV, 1680-1692.* Akademie Verlag, Berlin, 2001.

[113] Lenat, D. B., and Guha, R. V. *Building Large Knowledge-Based Systems: Representation and Inference in the Cyc Project.* Addison-Wesley, Reading (Massachusetts), 1990.

[114] Lewis, D. *Parts of Classes.* Basil Blackwell, Oxford, 1991. With an appendix by J. P. Burgess, A. P. Hazen, and D. Lewis.

[115] Lucanu, D., Li, Y.-F., and Dong, J. S. Semantic Web Languages—Towards an Institutional Perspective. In *Algebra, Meaning, and Computation, Essays Dedicated to Joseph A. Goguen on the Occasion of His 65th Birthday* (2006), K. Futatsugi, J.-P. Jouannaud, and J. Meseguer, Eds., vol. 4060 of *Lecture Notes in Computer Science*, Springer, pp. 99–123.

[116] Lukasiewicz, T. Expressive probabilistic description logics. *Artificial Intelligence 172*, 6-7 (2008), 852–883.

[117] Lutz, C., and Schröder, L. Probabilistic Description Logics for Subjective Uncertainty. In *Proceedings of the 12th International Conference on Principles of Knowledge Representation and Reasoning (KR2010)* (2010), F. Lin and U. Sattler, Eds., AAAI Press.

[118] Lutz, C., Walther, D., and Wolter, F. Conservative Extensions in Expressive Description Logics. In *Proceedings of IJCAI-07* (2007), AAAI Press, pp. 453–458.

[119] Lutz, C., and Wolter, F. Modal Logics of Topological Relations. *Logical Methods in Computer Science 2*, 2 (2006).

[120] Lutz, C., and Wolter, F. Mathematical Logic for Life Science Ontologies. In *WoLLIC* (2009), H. Ono, M. Kanazawa, and R. J. G. B. de Queiroz, Eds., vol. 5514 of *Lecture Notes in Computer Science*, Springer, pp. 37–47.

[121] Lutz, C., Wolter, F., and Zakharyaschev, M. Temporal description logics: A survey. In *Proceedings of the Fourteenth International Symposium on Temporal Representation and Reasoning* (2008), IEEE Computer Society Press.

[122] Ma, Y., and Hitzler, P. Paraconsistent Reasoning for OWL 2. In *RR '09: Proceedings of the 3rd International Conference on Web Reasoning and Rule Systems* (Berlin, Heidelberg, 2009), Springer-Verlag, pp. 197–211.

[123] Ma, Y., and Hitzler, P. Distance-based Measures of Inconsistency and Incoherency for Description Logics. In *Proceedings of the 23rd International Workshop on Description Logics (DL-2010)* (Waterloo, Canada, 2010, 2010), V. Haarslev, D. Toman, and G. Weddell, Eds., vol. 573, CEUR Workshop Proceedings, pp. 475–485.

[124] Ma, Y., Hitzler, P., and Lin, Z. Algorithms for Paraconsistent Reasoning with OWL. In *ESWC* (2007), E. Franconi, M. Kifer, and W. May, Eds., vol. 4519 of *Lecture Notes in Computer Science*, Springer, pp. 399–413.

[125] Ma, Y., Hitzler, P., and Lin, Z. Paraconsistent resolution for four-valued description logics. In *Proceedings of the 2007 International Workshop on Description Logics (DL-2007), Brixen-Bressanone, Italy, June 2007* (Juni 2007), vol. Vol-250 of *CEUR Workshop Proceedings*, pp. 395–402.

[126] Mac Lane, S. *Categories for the Working Mathematician*, 2nd ed. Springer, Berlin, 1998.

[127] Madhavan, J., Bernstein, P., Domingos, P., and Halevy, A. Representing and reasoning about mappings between domain models. In *Proc. of AAAI 2002* (Edmonton, Canada, 2002).

[128] Marx, M., and Venema, Y. *Multi-dimensional modal logic.* Kluwer Academic Publishers, Dordrecht, 1997.

[129] Masolo, C., Borgo, S., Gangemi, A., Guarino, N., and Oltramari, A. WonderWeb Deliverable D18: Ontology Library. Tech. rep., ISTC-CNR, 2003.

[130] Masters, J. Structured Knowledge Source Integration and its applications to information fusion. In *Proceedings of the Fifth International Conference on Information Fusion (FUSION 2002)* (Annapolis, MD, 2002), IEEE.

[131] McCorduck, P. *Machines Who Think: A Personal Inquiry into the History and Prospects of Artificial Intelligence.* Peters, Wellesley, 2004. 2nd rev. ed.

[132] Meseguer, J. General logics. In *Logic Colloquium 87.* North Holland, 1989, pp. 275–329.

[133] Meseguer, J., and Martí-Oliet, N. From abstract data types to logical frameworks. In *Selected papers from the 10th Workshop on Specification of Abstract Data Types Joint with the 5th COMPASS Workshop on Recent Trends in Data Type Specification* (London, UK, 1995), Springer-Verlag, pp. 48–80.

[134] Minsky, M. A framework for representing knowledge. In *The Psychology of Computer Vision*, P. Winston, Ed. McGraw-Hill, 1975.

[135] Mossakowski, T. Comorphism-based Grothendieck logics. In *Mathematical Foundations of Computer Science*, vol. 2420 of *LNCS.* Springer, 2002, pp. 593–604.

[136] Mossakowski, T. Comorphism-based Grothendieck logics. In *Mathematical Foundations of Computer Science*, K. Diks and W. Rytter, Eds., vol. 2420 of *LNCS.* Springer, 2002, pp. 593–604.

[137] Mossakowski, T. Institutional 2-cells and Grothendieck institutions. In *Algebra, Meaning and Computation. Essays Dedicated to Joseph A. Goguen* (2006), K. Futatsugi, J.-P. Jouannaud, and J. Meseguer, Eds., LNCS 4060, Springer, pp. 124–149.

[138] Mossakowski, T., Autexier, S., and Hutter, D. Development graphs—proof management for structured specifications. *Journal of Logic and Algebraic Programming 67*, 1–2 (2006), 114–145.

[139] Mossakowski, T., Haxthausen, A., Sannella, D., and Tarlecki, A. CASL: The Common Algebraic Specification Language. In *Logics of Formal Specification Languages*, M. H. D. Bjorner, Ed., Monographs in Theoretical Computer Science. Springer-Verlag Heidelberg, 2008, ch. 3, pp. 241–298.

[140] Mossakowski, T., Maeder, C., and Lüttich, K. The Heterogeneous Tool Set. In *TACAS 2007* (2007), O. Grumberg and M. Huth, Eds., vol. 4424 of *LNCS*, Springer, pp. 519–522.

[141] Mossakowski, T., Maeder, C., and Lüttich, K. The Heterogeneous Tool Set. In *VERIFY 2007*, B. Beckert, Ed., vol. 259. CEUR-WS, 2007.

[142] Mossakowski, T., and Tarlecki, A. Heterogeneous logical environments for distributed specifications. In *WADT 2008* (2009), A. Corradini and U. Montanari, Eds., vol. 5486 of *Lecture Notes in Computer Science*, Springer, pp. 266–289.

[143] Mossakowski, T., Tarlecki, A., and Diaconescu, R. What is a logic translation? *Logica Universalis 3*, 1 (2009), 95–124. Winner of the Universal Logic 2007 Contest.

[144] Motik, B., Horrocks, I., and Sattler, U. Bridging the Gap Between OWL and Relational Databases. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web 7*, 2 (2009), 74–89.

[145] Newell, A., Shaw, J. C., and Simon, H. A. Report on a general problem-solving program. In *Proceedings of the International Conference on Information Processing (IFIP)* (1959), pp. 256–264.

[146] Newell, A., and Simon, H. A. Computer Science as Empirical Inquiry: Symbols and Search. *Communications of the ACM 19*, 3 (1976), 113–126.

[147] Niles, I., and Pease, A. Towards a Standard Upper Ontology. In *FOIS-01: Proc. of the International Conference on Formal Ontology in Information Systems* (New York, NY, USA, 2001), ACM, pp. 2–9.

[148] Nipkow, T., Paulson, L. C., and Wenzel, M. *Isabelle/HOL—A Proof Assistant for Higher-Order Logic*, vol. 2283 of *LNCS*. Springer, 2002.

[149] Normann, I. *Automated Theory Interpretation.* PhD thesis, Department of Computer Science, Jacobs University, Bremen, 2009.

[150] Odintsov, S. P., and Wansing, H. Inconsistency-tolerant Description Logic. Motivation and Basic Systems. In *Trends in Logic. 50 Years of Studia Logica* (Dordrecht, 2003), V. Hendricks and J. Malinowski, Eds., no. 21 in Trends in Logic, Kluwer Academic Publishers, pp. 301–335.

[151] Odintsov, S. P., and Wansing, H. Inconsistency-tolerant Description Logic. Part II: A tableau algorithm for $\mathcal{CALC}^C$. *Journal of Applied Logic 6*, 3 (2008), 343–360.

[152] Parsons, T. *Nonexistent Objects.* Yale University Press, New Haven and London, 1980.

[153] Patel-Schneider, P. F. A Four-Valued Semantics for Terminological Logics. *Artifical Intelligence 38*, 3 (1989), 319–351.

[154] Pieczkowski, A. Über Theorien im erweiterten Sinne. *Studia Logica 33*, 4 (1974), 317–331.

[155] Pokrywczyński, D., and Malcolm, G. Towards a Functional Approach to Modular Ontologies using Institutions. In *Modular Ontologies—Proceedings of the Fourth International Workshop (WoMO*

2010) (Toronto, Canada, 2010), O. Kutz, J. Hois, J. Bao, and B. Cuenca Grau, Eds., vol. 210 of *Frontiers in Artificial Intelligence and Applications*, IOS Press, pp. 53–66.

[156] Priest, G. *In Contradiction: A Study of the Transconsistent*, vol. 39 of *Nijhoff International Philosophy Series.* Dordrecht, Martinus Nijhoff, The Hague, 1987.

[157] Priest, G. Logic: One or Many? In *Logical Consequences*, B. Brown and J. Woods, Eds. Hermes, 2001.

[158] Priest, G. Logical Pluralism Hollandaise. *The Australasian Journal of Logic 6* (2008), 210–214.

[159] Randell, D. A., Cui, Z., and Cohn, A. G. A Spatial Logic Based on Regions and Connection. In *Proceedings of the 3rd International Conference on the Principles of Knowledge Representation and Reasoning (KR'92)* (1992), Morgan Kaufmann, Los Altos, pp. 165–176.

[160] Restall, G. Carnap's Tolerance, Language Change and Logical Pluralism. *Journal of Philosophy 99* (2002), 426–443.

[161] Ridder, L. *Mereologie—Ein Beitrag zur Ontologie und Erkenntnistheorie*, vol. 83 of *Philosophische Abhandlungen.* Vittorio Klostermann, Frankfurt am Main, 2002.

[162] Rodrigues, O., and Russo, A. A Translation Method for Belnap Logic. Research Report Doc 98/7, Imperial College London, September 1998.

[163] Sannella, D., and Burstall, R. Structured theories in LCF. In *Proc. 8th Colloq. on Trees in Algebra and Programming* (1983), vol. 159 of *Lecture Notes in Computer Science*, Springer, pp. 377–391.

[164] Schorlemmer, M., and Kalfoglou, Y. Institutionalising Ontology-Based Semantic Integration. *Journal of Applied Ontology 3*, 3 (2008).

[165] Schröder, L., and Mossakowski, T. HasCASL: Integrated Higher-Order Specification and Program Development. *Theoretical Computer Science 410*, 12-13 (2009), 1217–1260.

[166] Schulz, S., Romacker, M., and Hahn, U. Part-whole reasoning in medical ontologies revisited—introducing SEP triplets into classification-based description logics. *Proc. AMIA Symposium* (1998), 830–834.

[167] Seidenberg, J., and Rector, A. L. Representing Transitive Propagation in OWL. In *Proc. of ER 2006, 25th International Conference on Conceptual Modeling, Tucson, AZ, USA, November 6–9* (2006), D. W. Embley, A. Olivé, and S. Ram, Eds., vol. 4215 of *LNCS*, Springer, pp. 255–266.

[168] Shehtman, V. "Everywhere" and "Here". *Journal of Applied Non-Classical Logic 9* (1999).

[169] Sheremet, M., Tishkovsky, D., Wolter, F., and Zakharyaschev, M. A Logic for Concepts and Similarity. *J. of Logic and Computation 17*, 3 (2007), 415–452.

[170] Sheremet, M., Wolter, F., and Zakharyaschev, M. A modal logic framework for reasoning about comparative distances and topology. *Annals of Pure and Applied Logic 161*, 4 (2010), 534–559.

[171] Simons, P. *Parts: A Study in Ontology.* Clarendon Press, Oxford, 1987.

[172] Simons, P. On Being Spread Out in Time: Temporal Parts and the Problem of Change. In *Existence and Explanation* (Dordrecht, 1991), W. S. et al., Ed., Kluwer Academic Publishers.

[173] Sioutos, N., de Coronado, S., Haber, M. W., Hartel, F. W., Shaiu, W.-L., and Wright, L. W. NCI Thesaurus: A semantic model integrating cancer-related clinical and molecular information. *Journal of Biomedical Informatics 40*, 1 (2007), 30–43.

[174] Straccia, U. A Sequent Calculus for Reasoning in Four-Valued Description Logics. In *Proc. of TABLEAUX-97: Int. Conference on Automated Reasoning with Analytic Tableaux and Related Methods, Pont-à-Mousson, France, May 13–16* (1997), D. Galmiche, Ed., vol. 1227 of *LNCS*, Springer, pp. 343–357.

[175] Suntisrivaraporn, B., Baader, F., Schulz, S., and Spackman, K. Replacing SEP-Triplets in SNOMED CT Using Tractable Description Logic Operators. In *AIME '07: Proceedings of the 11th conference on Artificial Intelligence in Medicine* (Berlin, Heidelberg, 2007), Springer-Verlag, pp. 287–291.

[176] Tarski, A. Der Aussagenkalkül und die Topologie. *Fundamenta Mathematicae 31* (1938), 103–134.

[177] ten Cate, B., Conradie, W., Marx, M., and Venema, Y. Definitorially Complete Description Logics. In *Proceedings of KR 2006* (2006), P. Doherty, J. Mylopoulos, and C. Welty, Eds., AAAI Press, pp. 79–89.

[178] van Benthem, J. Logical dynamics meets logical pluralism? *The Australasian Journal of Logic 6* (2008), 182–209.

[179] Venn, J. *Symbolic Logic.* The MacMillan Company, London, 1881.

[180] Villadsen, J. Paraconsistent Query Answering Systems. In *FQAS '02: Proceedings of the 5th International Conference on Flexible Query Answering Systems* (London, UK, 2002), Springer-Verlag, pp. 370–384.

[181] Voronkov, A. Inconsistencies in Ontologies. In *JELIA-06* (2006), p. 19.

[182] Zhou, L. ., Huang, H., Qi, G., Ma, Y., Huang, Z., and Qu, Y. Paraconsistent Query Answering Over DL-Lite Ontologies. In *Proceedings of the Third Chinese Semantic Web Symposium (CSWS-09)* (2009).

[183] Zimmermann, A., Krötzsch, M., Euzenat, J., and Hitzler, P. Formalizing Ontology Alignment and its Operations with Category Theory. In *Proc. of FOIS-06* (2006), pp. 277–288.

Oliver Kutz
SFB/TR 8 Spatial Cognition, University of Bremen, Cartesium, Enrique-Schmidt Strasse, 28359, Bremen, Germany
e-mail: `okutz@informatik.uni-bremen.de`

Till Mossakowski,
DFKI GmbH and SFB/TR 8 Spatial Cognition, University of Bremen, Cartesium, Enrique-Schmidt Strasse, 28359, Bremen, Germany
e-mail: `Till.Mossakowski@dfki.de`

Dominik Lücke
SFB/TR 8 Spatial Cognition, University of Bremen, Cartesium, Enrique-Schmidt Strasse, 28359, Bremen, Germany
e-mail: `luecke@informatik.uni-bremen.de`