D3.1

Core KR infrastructure and concept repository

Authors	Oliver Kutz, Fabian Neuhaus, Till Mossakowski, Mihai Codescu, John Bateman
Reviewers	Kai-Uwe Kühnberger

Grant agreement no.	611553
Project acronym	COINVENT - Concept Invention Theory
Date	October 1, 2014
Distribution	PU

The project COINVENT acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open Grant number 611553.

Abstract

This deliverable specifies the core KR infrastructure that is needed for formalising conceptual blending. This includes a description of those language constructs of the Distributed Ontology Language DOL that are needed to formally specify blending diagrams as introduced by Joseph Goguen. Moreover, we discuss the `Ontohub.org` platform and its repository node Conceptportal, which support computationally the generation of new concepts based on the specification of DOL blending diagrams.
Keyword list: **Knowledge Representation**, **Reasoning**, **DOL**, **Conceptual Blending**, **Ontohub**, **Conceptportal**

Executive Summary

This report is the deliverable “Core KR infrastructure and concept repository” of the COINVENT project. It summarises (a) the features of a newly developed language, DOL, which supports the formal specification of conceptual blending via blending diagrams, and (b) the implementation of a repository Conceptportal within the `ontohub.org` platform, which provides computational support for DOL and infrastructure support for conceptual blending experiments.

Conceptual blending has been employed very successfully to understand the process of concept invention, studied particularly within cognitive psychology and linguistics. However, despite this influential research, within computational creativity little effort has been devoted to fully formalise these ideas and to make them amenable to computational techniques. Unlike other combination techniques, *blending* aims at creatively generating (new) concepts on the basis of input theories whose domains are thematically distinct but whose specifications share structural similarity based on a relation of analogy, identified in a generic space, called the base ontology. We here introduce the basic formalisation of conceptual blending, as sketched by the late Joseph Goguen, and discuss some of its variations. We illustrate the vast array of conceptual blends that may be covered by this approach and discuss the theoretical and conceptual challenges that ensue.

In order to make possible and formally represent conceptual blending within and across different domains, it is essential to have a knowledge representation framework available that is capable of dealing with a heterogeneous set of logical languages, ranging in expressivity as well as other logical characteristics. To provide a comprehensive knowledge representation and reasoning infrastructure, we developed the Distributed Ontology, Modelling and Specification Language (DOL), which serves as a common overarching specification environment across the COINVENT project. DOL provides the means to identify and structure ontologies, specify blending diagrams, and formulate requirements and evaluation criteria for blendoids. We show how the DOL language can be used to declaratively specify blending diagrams of various shapes, and discuss in detail the workflow and creative act of generating and evaluating new, blended concepts.

Computational support for DOL has been integrated into Conceptportal, a repository at `http://conceptportal.org` for the collection of micro concepts; it is used throughout COINVENT for blending experiments. Conceptportal is an instance of Ontohub, our ontology repository technology. OntoHub supports the ontology development and maintenance along the whole ontology lifecycle. It allows for publishing ontologies and retrieve existing ontologies. Further, OntoHub supports ontology development (including ontology versioning, branching, and merging) and evaluation. This report discusses also the basic architecture of Ontohub.

Contents

1 Concept Invention via Blending

One broad area of phenomena that is often brought into connection with issues of creativity and the emergence of new ideas concerns notions such as metaphor, blending, category mistakes, similes, analogies and the like.[1] In each of these, seemingly inconsistent material is combined in a manner that results in a productive growth of information instead of simple logical contradiction. Approaches to treat this phenomenon are varied but commonly come to the conclusion that more or less well developed notions of 'structure' are crucial for bringing the growth of information about — e.g., 'implication complexes' for metaphor [12], 'conceptual spaces'[2] for blending [24], 'structure mapping' in analogy [31], and so on. On the one hand, the less structure that is available, the less productive the combinations appear to be; on the other, the presence of structure raises the challenge of how such formal commitments can be productively 'overridden' or rearranged in order to avoid contradiction.

In our ongoing work on ontology and its formal underpinnings, we have been led to a very similar set of questions. By 'ontology' we here refer to the now rather standard notion of a formal specification of a shared understanding of the entities, relations and general properties holding in some domain of interest, cf. [40, 37]. Achieving adequate treatments in various domains has demonstrated to us the need for heterogeneous ontological specifications that are capable of capturing distinct perspectives on the phenomena being modelled. In an architectural context, for example, it is beneficial to maintain distinct perspectives on structural integrity, spatial distribution, movement patterns by the occupants of a building ('flow'), navigation networks (possibly varying according to 'normal' and 'emergency' conditions), 'visibility' patterns (both for users and for sensors in the case of security) and many more [10] — each of these perspectives can be modelled well by employing ontological engineering techniques but there is no guarantee that they are simply compatible. Our work on natural language dialogue systems involving spatial language comes to the same conclusion [3], while similar concerns are already well known in Geographic Information Science [29, 53]. To support this fundamental 'multi-perspectivalism' we have therefore been developing an entire toolset of more sophisticated combination methods [55], leading to the formal definition of the notion of a 'hyperontology' in [59].

The similarities apparent between the goals of heterogeneous ontology 'alignment' and the creative combination of thematically distinct information spaces can be built on quite concretely by treating such information spaces explicitly in terms of ontological specifications. This allows us to link directly with previous work by exploring the application of techniques for combining distinct perspectives that are now becoming available. For example, much work on creativity has been pursued in the context of Fauconnier and Turner's 2003 account of conceptual blending [24], in which the blending of two thematically rather different *conceptual spaces* yields a new conceptual space with emergent structure, selectively combining parts of the given spaces whilst respecting common structural properties. The 'imaginative' aspect of blending is summarised as follows in [104]:

> [...] the two inputs have different (and often clashing) organising frames, and the blend has an organising frame that receives projections from each of those organising

[1] This report is based on the publications [84, 62, 99, 54, 76, 17, 75, 98, 83, 78].

[2] The usage of the term 'conceptual space' in blending theory is not to be confused with the usage established by [30].

> frames. The blend also has emergent structure on its own that cannot be found in any of the inputs. Sharp differences between the organising frames of the inputs offer the possibility of rich clashes. Far from blocking the construction of the network, such clashes offer challenges to the imagination. The resulting blends can turn out to be highly imaginative.

We see the almost unlimited space of possibilities supported by 'ontological blending' for combining existing ontologies to create new ontologies with emergent structure as offering substantial benefits not only for ontological engineering — where conceptual blending can be built on to provide a structural and logic-based approach to 'creative' ontological engineering — but also for conceptual blending and related frameworks themselves — by providing a far more general and nevertheless computational, formalised foundation. Re-considering some of the classic problems in conceptual blending in terms of ontological modelling and ontological blending opens up an exciting direction for future research.

This endeavour primarily raises the following two challenges: (1) when combining the terminologies of two ontologies, the shared semantic structure is of particular importance to steer possible combinations — this shared semantic structure leads to the notion of a base ontology, which is closely related not only to the notion of 'tertium comparationis' found in classical rhetoric and poetics, but also to more recent cognitive theories of metaphor (see, e.g., [49]); (2) having established a shared semantic structure, there typically remains a considerable number of possibilities that can capitalise on this information in the combination process — here, structural optimality principles as well as ontology evaluation techniques can take on a central role in selecting 'interesting' blends.

There is still much to explore concerning the relationships between the principles governing ontological blending and the principles explored to date for blending phenomena in language or poetry or, indeed, the rather strict principles ruling blending in mathematics, in particular in the way formal inconsistencies are dealt with. For instance, whilst blending in poetry might be particularly inventive or imaginative when the structure of the basic categories found in the input spaces is almost completely ignored, in areas such as mathematics a rather strict adherence to sort structure is important in order to generate meaningful blends.[3] The use that we might typically make of ontological blending is situated somewhere in the middle: re-arrangement and new combination of basic categories can be quite interesting, but has to be finely controlled through corresponding interfaces, often regulated by or related to choices found in foundational or upper ontologies so that basic categorial relationships are maintained.

For all such cases, however, we can consider the formal mechanisms that support specific blends that we explore with respect to their potential relevance and value for understanding 'blending' phenomena in general. This will be one of the purposes of this report. We will summarise some of the progress that has been made towards adopting the fruitful idea of conceptual blending in a theoretically well-understood and computationally supported formal model for concept invention, focusing in particular on ontology languages. Here we elaborate on ideas first introduced in [47], with detailed technical definitions given in [58]. More specifically, we:

- briefly characterise the kinds of creativity that have been considered hitherto in the areas of

[3]For instance when creating the theory of transfinite cardinals by blending the perfective aspect of counting up to any fixed finite number with the imperfective aspect of 'endless counting' [88].

blends, metaphors and related operations where structured mappings or analogies are relied upon;

- sketch the logical analysis of conceptual blending in terms of blending diagrams and colimits, as originally proposed by Joseph Goguen, and give an abstract rendering of ontological blendoids capturing the basic intuitions of conceptual blending in the ontological setting;
- sketch a formal meta-language, namely the distributed ontology language DOL, that is capable of declaratively specifying blending diagrams in a variety of ontology languages. This provides a structured approach to ontology languages and blending and combines the simplicity and good tool support for languages such as OWL[4] with the more complex blending facilities of OBJ3 [34] or Haskell [52]; DOL also facilitates the specification of a range of variations of the basic blending technique;
- discuss the capabilities of the Ontohub/HETS ecosystem with regard to collaboratively managing, creating, and displaying blended concepts, ontological theories, and entire blending diagrams; this includes an investigation of the evaluation problem in blending, together with a discussion of structural optimality principles and current automated reasoning support.

We close with a discussion of open problems and future work.

2 An Ocean of Blends

In this section, we briefly characterise the rather diverse phenomena that may be subject to beneficial formalisations in terms of ontological blending. The starting point is the obvious one of conceptual blending, which we use as a prototypical case of emergent organisation throughout this document. As noted above, conceptual blending in the spirit of [24] operates by combining two input 'conceptual spaces', construed as rather minimal descriptions of some thematic domains, in a manner that creates new 'imaginative' configurations. A classic example for this is the blending of the concepts *house* and *boat*, yielding as most straightforward blends the concepts of a *houseboat* and a *boathouse*, but also an *amphibious vehicle* [36]; we return to this example below. This case shows well how it is necessary to maintain aspects of the structural semantics of the spaces that are blended in order to do justice to the meanings of the created terms: the houseboat stops neither being a vehicle on water nor being a place of residence, for example.

Very similar processes appear to be operating in cases of metaphor [12, 51]. Here a semantically structured 'source' is used so that facets of the semantics of the source are selected for appropriate take up by a semantically structured 'target'. This can operate on a small scale, analogously to the *house* and the *boat*, as for example in metaphors such as that evident in the 1940s film title "Wolf of New York" or the recent "The Wolf of Wall Street" (2013), where certain conventionalised properties of the wolf as animal (the source) are transferred to the people referred to by the titles (the target). Structure is essential here since the transfer is very specific: a reading of the metaphor in which 'four-leggedness' or 'furry' is transferred is in the given contexts most unlikely. Only particular relations and relational values are effected. Metaphors can also operate on much broader scales, as in considerations of metaphors as contributions to creative scientific

[4]With 'OWL' we refer to OWL 2 DL, see `http://www.w3.org/TR/owl2-overview/`

theory construction, as in the well known transfer of a 'sun-and-planet' conceptual model to models of the atom [74, 41, 42] (see Sec. 3.1 below). Structural transfer of this kind has consequently been suggested to play a substantial role for persuasive text creation as such. Hart, for example, discusses the use of phrases such as 'limitless flow of immigration', 'flood of asylum seekers' and so on as ideologically-loaded constructions that need to be unpacked during critical discourse analysis [44].

Metaphors also bring with them some particular formal features of their own – for example, they are typically seen as *directed* in contrast to blends and have been related to models of embodiment via accounts of *image schemas* [50]. Image schemas suggest how multimodal patterns of experience can be linked to increasingly abstract conceptualisations: abstract thought is then seen as a metaphorical construction on top of concrete experience. The use of the word 'flood' in the above example can then be expected to bring about a physical component in its reception where feelings of force, damage and lack of control are activated; this makes it clear that much more than 'flowery language' might be involved in such phrasings and their selection.

A related consideration is the proposal for internalised spatial representations for supporting reasoning and more abstract conceptualisations (such as time) as well as externalised spatial representations for diagrammatic reasoning. In the former case, it is common to work within blended spaces where time and spatial extent appear to have 'collapsed', giving rise to language use such as "keep going straight until the church" or "turn left before the tower" and so on. Blends of this kind are so familiar that they may be considered to be *entrenched* in the cognitive linguistic sense of having become part of the semantics of the respective terms and shared by the language community [24, 49].

Blends may also be multiple in that once established, for example in a text, further conceptual spaces might be added as an argument progresses. These may progressively add details to a developing emergent space (or, alternatively, lead to a space which strains the credibility of a reader or hearer too far resulting in a charge of 'mixing metaphors'). In the right-wing immigration example above from Hart, the texts do in fact continue with phrases such as 'Britain is full up', 'no matter how open or closed its immigration policy', and 'our first step will be to shut the door'. This builds on the previous blend of immigration-as-flood by (i) combining 'Britain' with a 'container' (which can then be full) that is itself (ii) combined with a 'building' or 'room' that has 'doors' that can be closed, and (iii) those doors can in turn also be 'policies' (which can be open or closed) [44, 102]. There need in principle be no end to this creative extension and combination of concepts. This aspect of iteration of blending is also explored in the area of conceptual mathematics as explored in [63], where is it argued that abstract mathematical concepts such as modern number systems, algebra, or set theory, are created through a succession of conceptual metaphors and blends, grounded in embodied concepts and image schemas. The structure of such blends and blending patterns in general are discussed more formally in Section 4.4.

There is also now increasing discussion of the potential role of blending or similar mechanisms when considering the creative use of combinations of information from different semiotic modes, e.g., drawing relations between verbal information, visual and gestural information [28]. In such cases representations or entities in one mode of presentation are made to take on properties or behaviours in another. The general applicability of an ontological approach to semiotic blending of this kind is argued in [4]. Again, there are many examples of such creativity in action. Consider for example the extract from an advertisement discussed by [106] and shown in Fig. 1. Here an open-

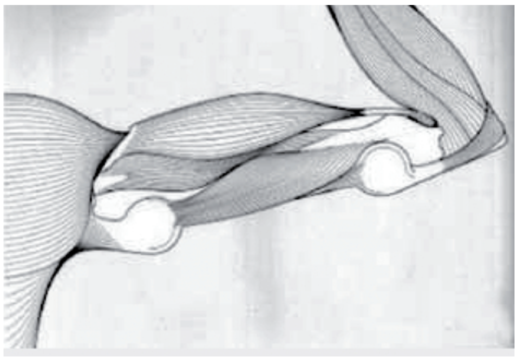
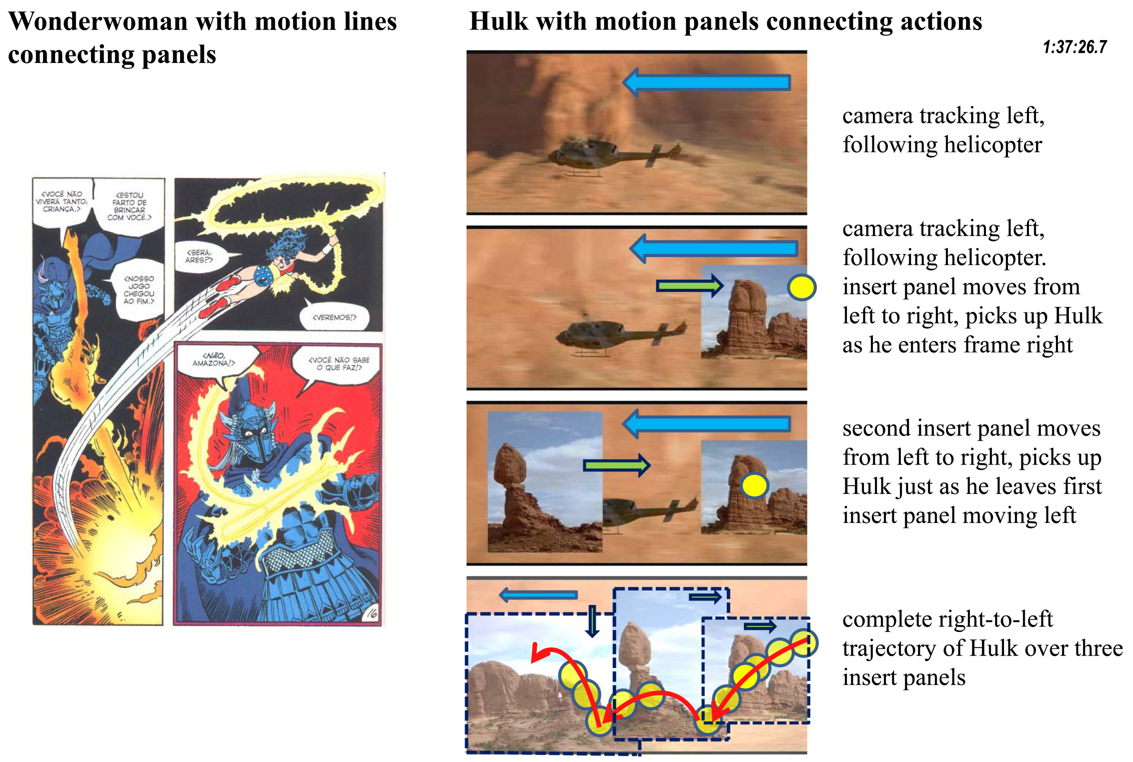


Figure 1: Visual blending of a car and an anatomical representation used for advertising discussed by [106].

ended set of potential further inferences, all supporting the general intention of the advertisement, is opened up by virtue of the blend. There are also commonly discussed combinations such as the use of space for time in comics and visual narrative – moving across the space of a comic's panel, typically in Western comics therefore from left to right, often correlates with a progression in time [72, 95] – as well as blends for dramatic or emotional effect, such as when typography is shaped visually for affective purposes [21, 12].

A particularly creative and novel example of semiotic blending across media can be seen in the following example. In this case the film director Ang Lee works with the *dynamic* possibilities of the film medium to enlist graphic resources for expressing movement developed within the *static* medium of comics. The result is an interesting and highly explorative expansion of the creative potential of what can be done with film. An illustration is shown in Fig. 2. On the left for comparison is a now quite traditional static rendition of movement from a comic – in this particular case showing 'continuity' of movement across panels. In contrast, on the right we see a short sequence of stills taken from a chase scene in Lee's film *Hulk* (2003), where the main character is trying to escape from pursuers in a helicopter. In this case, the escape trajectory is shown in a sequence of dynamically inserted 'panels' that move across the screen to the point where they can pick up the character's movement. This blending of properties in Lee's film does much more than 're-create' a visual effect analogous to comics as sometimes suggested in analyses of this film. Lee's appropriation of framing and movement techniques within an already dynamic medium appears instead to provide a resource that considerably heightens continuity for narrative effect. A more detailed discussion of the consequences of this appropriation for interpretation and reception is given in [5].

We are just beginning to be able to explore extensions of meaning-making potential of these kinds. Indeed, although there are now many examples in the literature of such creative meaning growth in action, deep questions remain concerning how precisely this may be modelled. In particular, following simpler operations of 'alignment' of structures across spaces (e.g., by graph matching [27, 109]), it is by no means clear how the results that are achieved can function as productively as they evidently do. This relates also to Fauconnier's suggestion that it is actually what is done with the result of blending, termed *elaboration* (or 'running the blend'), that is the most significant stage of the entire blending process. Elaboration "consists in cognitive work performed within the blend, according to its own emergent logic" [25, 151]. This makes it evident


Wonderwoman with motion lines connecting panels
<VOCÊ NÃO VIVERÁ TANTO, CRIANÇA.>
<ESTOU FARTO DE BRINCAR COM VOCÊ.>
<NOSSO JOGO CHEGOU AO FIM.>
<SERÁ, ARES?>
<VEREMOS!>
<NÃO, AMAZONA!>
<VOCÊ NÃO SABE O QUE FAZ!>
Hulk with motion panels connecting actions
1:37:26.7
camera tracking left, following helicopter
camera tracking left, following helicopter. insert panel moves from left to right, picks up Hulk as he enters frame right
second insert panel moves from left to right, picks up Hulk just as he leaves first insert panel moving left
complete right-to-left trajectory of Hulk over three insert panels

Pérez, G., Potter, G. and Wein, L. (2008), *Biblioteca DC – Mulher-Maravilha/Wonder Woman,* Brasil, São Paulo: Panini. p. 140.

Lee, Ang (2003), *Hulk*, based on Marvel characters by Stan Lee and Jack Kirby, Screenplay: John Turman, Michael France and James Schamus, Story: James Schamus, USA: Universal Pictures, Marvel Enterprises, Valhalla Motion Pictures, Good Machine.

Figure 2: Blending expressive resources from comics and film.

that something more is required in the formalisation than a straightforward recording or noting of a structural alignment: a new blended theory should also be 'logically productive', with new and surprising entailments which may well be quite specific to the blend. This is therefore another motivation for the rather more formal and ontologically-driven approach to this kind of creative meaning creation that we now present.

3 Blending Computationalised

There have now been several approaches moving towards effective computational treatments of blending, metaphor and related constructs such as analogy, cf. e.g. [109, 93, 102, 64, 108, 70]. Here we follow the research direction of *algebraic semiotics* established by Goguen. In this approach certain structural aspects of semiotic systems are logically formalised in terms of algebraic theories, sign systems, and their mappings [32]. Sign systems are theories 'with extra structure' connected by a particular class of mappings, which Goguen terms 'semiotic morphisms', which preserve that extra structure to a greater or lesser degree. In [36], algebraic semiotics has been applied to user interface design and blending. Algebraic semiotics does not claim to provide a comprehensive formal theory of blending—indeed, Goguen and Harrell admit that many aspects of blending, in particular concerning the meaning of the involved notions, as well as the optimality principles for blending, cannot be captured formally. However, the structural aspects *can* be formalised and provide insights into the space of possible blends.

Goguen defines sign systems as algebraic theories that can be formulated by using the algebraic specification language OBJ3 [34]. One special case of such a sign system is a *conceptual space*: it consists only of constants and relations, one sort, and axioms that define that certain relations hold on certain instances.

We now relate such spaces to a general formalisation of ontologies as we understand them and as introduced above. Since we will focus on standard ontology languages, namely OWL and first-order logic, we use these to replace the logical language OBJ3 used by Goguen and Malcolm. However, as some structural aspects are necessary in the ontology language to support blending, we augment these standard ontology languages with structuring mechanisms known from algebraic specification theory [56]. Such mechanisms are now included in the DOL language specification discussed below in Section 4. This allows us to translate most parts of Goguen's theory to these augmented ontology languages. Goguen's main insight has been that sign systems and conceptual spaces can be related via *morphisms*, and that blending is comparable to *colimit* construction. In particular, the blending of two concepts is often a *pushout* (also called a *blendoid* in this context). Some basic definitions we then need are the following.[5]

Non-logical symbols are grouped into **signatures**, which for our purposes can be regarded as collections of typed symbols (e.g. concept names, relation names). **Signature morphisms** are maps between signatures that preserve (at least) types of symbols (i.e. map concept names to concept names, relations to relations, etc.). A **theory** or **ontology** pairs a signature with a set of sentences over that signature, and a **theory morphism** (or **interpretation**) between two theories is just a signature morphism between the underlying signatures that preserves logical consequence, that is, $\rho : T_1 \to T_2$ is a theory morphism if $T_2 \models \rho(T_1)$, i.e. all the translations of sentences of T_1 along ρ follow from T_2. This construction is completely logic independent. Signature and theory morphisms are an essential ingredient for describing conceptual blending in a logical way.

We can now give a general definition of ontological blending capturing the basic intuition that a blend of input ontologies shall partially preserve the structure imposed by base ontologies, but otherwise be an almost arbitrary extension or fragment of the disjoint union of the input ontologies with appropriately identified base space terms.

For the following definition, a variant of which we first introduced in [58], a diagram consists of a set of ontologies (the nodes of the diagram) and a set of morphisms between them (the arrows of the diagram). The **colimit** of a diagram is similar to a disjoint union of its ontologies, with some identifications of shared parts as specified by the morphisms in the diagram. We refrain from presenting the category-theoretic definition here (which can be found in [1]), but will explain (the action of) the colimit operation in the examples in Section 4.3. In the following definition, we use $|D|$ to denote the set of all nodes in a diagram.

Definition 1 (Ontological Base Diagram) *An **ontological base diagram** is a diagram D for which a distinguished set $\mathcal{B} = \{B_i \mid i \in I\} \subset |D|$ of nodes are called **base ontologies**, and where a second distinguished set of nodes $\mathcal{I} = \{I_j \mid j \in J\} \subset |D|$ are called **input ontologies**, and where the theory morphisms $\mu_{ij} : B_i \to I_j$ from base ontologies to input ontologies are called the **base morphisms**.*

[5]Note that these definitions apply not only to OWL, but also to many other logics. Indeed, they apply to *any* logic formalised as an *institution* [33].

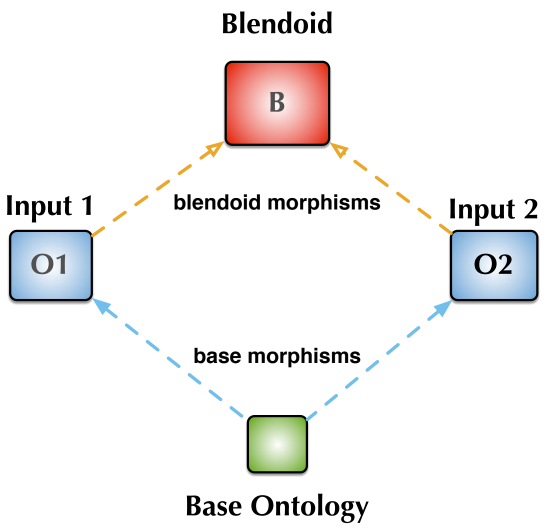


Figure 3: The basic integration network for blending: concepts in the base ontology are first refined to concepts in the input ontologies and then selectively blended into the blendoid.

If there are exactly two inputs I_1, I_2, and precisely one base $B \in \mathcal{B}$ and two base morphisms $\mu_k : B \to I_k$, $k = 1,2$, the diagram D is called ***classical*** *and has the shape of a 'V'. In this case, $\mathcal{B}$ is also called the* ***tertium comparationis****.*

Fig. 3 illustrates the basic, classical case of an ontological blending diagram. The lower part of the diagram shows the base space (tertium), i.e. the common generalisation of the two input spaces, which is connected to these via total (theory) morphisms, the base morphisms. The newly invented concept is at the top of this diagram, and is computed from the base diagram via a colimit. More precisely, any consistent subset of the colimit of the base diagram may be seen as a newly invented concept, a **blendoid**.[6] Note that, in general, ontological blending can deal with more than one base and two input ontologies, and in particular, the sets of input and base nodes need not exhaust the nodes participating in a base diagram. We will further discuss this and give some examples in Section 4.4.

3.1 Computing the Tertium Comparationis

To find candidates for base ontologies that could serve for the generation of ontological blendoids, much more shared semantic structure is required than the surface similarities that statistical term alignment approaches rely on [22]. The common structural properties of the input ontologies that are encoded in the base ontology are typically of a more abstract nature. The standard example here relies on *image schemata*, such as the notion of a *container* (see e.g. [52]). Thus, in particular, foundational ontologies can support such selections. In analogical reasoning, 'structure' is (partially) mapped from a source domain to a target domain [27, 101]. Therefore, intuitively the operation of computing a base ontology can thus be seen as a bi-directional search for analogy

[6] A technically more precise definition of this notion is given in [58]. Note also that our usage of the term 'blendoid' does not coincide with the (non-primary) blendoids defined in [36].

or generalisation into a base ontology together with the corresponding mappings. Providing efficient means for finding a number of suitable such candidate generalisations is essential to making the entire blending process computationally feasible. Consider the example of blending 'house' with 'boat' discussed in detail in Section 4.3: even after fixing the base ontology itself, guessing the right mappings into the input ontologies means guessing within a space of approximately 1.4 Billion signature morphisms. Three promising candidates for finding generalisations are:

(1) **Ontology intersection:** [87] has studied the automatisation of theory interpretation search for formalised mathematics, implemented as part of the Heterogeneous Tool Set (HETS, see below). [60] applied these ideas to ontologies by using the ontologies' axiomatisations for finding their shared structure. Accidental naming of concept and role names is deliberately ignored and such names are treated as arbitrary symbols (i.e., any concept may be matched with any other). By computing mutual theory interpretations between the inputs, the method allows the computation of a base ontology as an *intersection* of the input ontologies together with corresponding theory morphisms. While this approach can be efficiently applied to ontologies with non-trivial axiomatisations, lightweight ontologies are less applicable, e.g., 'intersecting' a smaller taxonomy with a larger one clearly results in a huge number of possible taxonomy matches [60]. In this case, the following techniques are more appropriate.

(2) **Structure-based ontology matching:** matching and alignment approaches are often restricted to find simple correspondences between atomic entities of the ontology vocabulary. In contrast, work such as [97, 110] focuses on defining a number of *complex correspondence patterns* that can be used together with standard alignments in order to relate complex expressions between two input ontologies. For instance, the 'Class by Attribute Type Pattern' may be employed to claim the equivalence of the atomic concept $\mathsf{PositiveReviewedPaper}$ in ontology O_1 with the complex concept $\exists\mathsf{hasEvaluation.Positive}$ of O_2. Such an equivalence can be taken as an axiom of the base ontology; note, however, that it could typically not be found by intersecting the input ontologies. Giving such a library of design patterns may be seen as a variation of the idea of using image schemata.

(3) **Analogical Reasoning:** *Heuristic-driven theory projection* is a logic-based technique for analogical reasoning that can be employed for the task of computing a common generalisation of input theories. [101] establish an analogical relation between a source theory and a target theory (both first-order) by computing a common generalisation (called 'structural description'). They implement this by using anti-unification [95]. A typical example is to find a generalisation (base ontology) formalising the structural commonalities between the Rutherford atomic model and a model of the solar system. This process may be assisted by a background knowledge base (in the ontological setting, a related domain or foundational ontology). Indeed, this idea has been further developed in [71].

3.2 Selecting the Blendoids: Optimality Principles

Having a common base ontology (computed or given) with appropriate base morphism, there is typically still a large number of possible blendoids whenever some kind of partiality is allowed. For example, even in the rather simple case of combining House and Boat, allowing for blendoids which only partially maintain structure (called *non-primary* blendoids in [36]), i.e., where any subset of the axioms may be propagated to the resulting blendoid, the number of possible blendoids is in the magnitude of 1000. Clearly, from an ontological viewpoint, the overwhelming majority of these candidates will be rather meaningless. A ranking therefore needs to be applied on the basis of specific ontological principles. In conceptual blending theory, a number of **optimality principles** are given in an informal and heuristic style [24]. While they provide useful guidelines for evaluating natural language blends, they do not suggest a direct algorithmic implementation, as also analysed in [36] who in their prototypical implementation only covered certain structural, logical criteria. However, the importance of designing computational versions of optimality principles has been realised early on, and one such attempt may be found in the work of [94], who proposed an implementation of the eight optimality principles presented in [23] based on quantitative metrics for their more lightweight logical formalisation of blending. Such metrics, though, are not directly applicable to more expressive languages such as OWL or first-order logic.

Moreover, the standard blending theory of [24] does not assign types, which might make sense in the case of linguistic blends where type information is often ignored. A typical example of a type mismatch in language is the operation of *personification*, e.g., turning a boat into an 'inhabitant' of the 'boathouse'. However, in the case of blending in mathematics or ontology, this loss of information is often rather unacceptable: on the contrary, a fine-grained control of type or sort information may be of the utmost importance.

Optimality principles for ontological blending will be of two kinds:

(1) purely **structural/logical principles**: these will extend and refine the criteria as given in [36], namely *degree of commutativity* of the blend diagram, *type casting* (preservation of taxonomical structure), *degree of partiality* (of signature morphisms), and *degree of axiom preservation*. In the context of OWL, typing needs to be replaced with preservation of specific axioms encoding the taxonomy.

(2) **heuristic principles**: these include introducing preference orders on morphisms (an idea that [32] labelled $3/2$ pushouts) reflecting their 'quality', e.g. measured in terms of degree of type violation; specific ontological principles, e.g. adherence to the OntoClean methodology [39] and ontological modelling principles, or general ontology evaluation techniques such as competency questions and fidelity requirements, as further discussed in Section 5.3.

4 Blending with the Distributed Ontology Language DOL

The distributed ontology language DOL is a formal language for specifying both ontologies, base diagrams, and their blends. DOL is a metalanguage in the sense that it enables the reuse of existing ontologies (written in some ontology language like OWL or Common Logic) as building blocks for new ontologies and, further, allows the specification of intended relationships between ontologies. One important feature of DOL is the ability to combine ontologies that are written in different languages without changing their semantics. DOL is going to be submitted as response to the Object Management Group's (OMG) Ontology, Model and Specification Integration and Interoperability (OntoIOp) Request For Proposal[7]. DOL is supported by the Heterogeneous Tool Set HETS [81] and the Ontohub platform [76] discussed below.

In this section, we introduce DOL only informally. A formal specification of the language and

[7]http://www.omg.org/cgi-bin/doc?ad/2013-12-02

its model-theoretic semantics can be found in [78, 80].

4.1 Foundations of DOL

The large variety of logical languages in use can be captured at an abstract level using the concept of *institutions* [33]. This allows us to develop results independently of the particularities of a logical system and to use the notions of institution and logical language interchangeably throughout the rest of this report. The main idea is to collect the non-logical symbols of the language in signatures and to assign to each signature the set of sentences that can be formed with its symbols. For each signature, we provide means for extracting the symbols it consists of, together with their kind. Signature morphisms are mappings between signatures. We do not assume any details except that signature morphisms can be composed and that there are identity morphisms; this amounts to a category of signatures. Readers unfamiliar with category theory may replace this with a partial order (signature morphisms are then just inclusions). See [79] for details of this simplified foundation.

Institutions also provide a model theory, which introduces semantics for the language and gives a satisfaction relation between the models and the sentences of a signature. The only restriction imposed is the satisfaction condition, which captures the idea that truth is invariant under change of notation (and enlargement of context) along signature morphisms. This relies on two further components of institutions: the translation of sentences along signature morphisms, and the reduction of models against signature morphisms (generalising the notion of model reduct known from logic).

It is also possible to complement an institution with a proof theory, introducing a derivability relation between sentences, formalised as an *entailment system* [73]. In particular, this can be done for all logics that have so far been in use in DOL.

To sum up, an institution provides notions of signature and signature morphism (formally, this is given by a category **Sign**), and for each signature Σ in **Sign**, a set of sentences $Sen(\Sigma)$, a class of models $Mod(\Sigma)$ and a binary satisfaction relation $\models_\Sigma$ between models and sentences. Furthermore, given a signature morphism $\sigma\colon \Sigma_1 \to \Sigma_2$, an institution provides sentence translation along σ, written $\sigma(\varphi)$, and model reduct against σ, written $M|_\sigma$, in a way that satisfaction remains invariant:

$$M'|_\sigma \models_{\Sigma_1} \varphi \text{ iff } M' \models_{\Sigma_2} \sigma(\varphi)$$

for each $\varphi \in Sen(\Sigma_1)$ and $M' \in Mod(\Sigma_2)$.

DOL and HETS support a variety of different logics; the most important and currently most frequently used logics for conceptual blending within COINVENT are the following:

OWL 2 is the Web Ontology Language recommended by the World Wide Web Consortium (W3C, `http://www.w3.org`); see [90]. It is used for knowledge representation on the Semantic Web [9]. HETS supports OWL 2 DL and the provers Fact++ and Pellet.

FOL/TPTP is an untyped first-order logic with equality,[8] underlying the interchange language TPTP [103], see `http://www.tptp.org`. HETS offers several automated theorem proving

[8]FOL/TPTP is called SoftFOL in the HETS implementation. SoftFOL extends first-order logic with equality with a softly typed logic used by SPASS; however, in this paper we will only use the sublanguage corresponding to FOL.

(ATP) systems for TPTP, namely SPASS [112], Vampire [96], Eprover [100], Darwin [6], E-KRHyper [92], and MathServe Broker (which chooses an appropriate ATP upon a classification of the FOL problem) [113].

CFOL is many-sorted first-order logic with so-called sort generation constraints, expressing that each value of a given sort is the interpretation of some term involving certain functions (called constructors). This is equivalent to an induction principle and allows the axiomatisation of lists and other datatypes, using the usual Peano-style axioms (such an axiomatisation is called a *free type*). CFOL is a sublogic of the Common Algebraic Specification Language CASL, see [82, 11]. Proof support for CFOL is available through a simple induction scheme in connection with automated first-order provers like SPASS [67], or via a comorphism to HOL. A connection to the induction prover KIV [2] is under development.

HOL is typed higher-order logic [15]. HETS actually supports several variants of HOL, among them THF0 (the higher-order version of TPTP [8]), with automated provers LEO-II [7], Satallax [16] and an automated interface to Isabelle [86], as well as the logic of Isabelle, with an interactive interface.

HETS supports the input languages of these logics directly. Adding a new logic can be done by writing a number of Haskell data types and functions, providing abstract syntax, parser, static analysis and prover interfaces for the logic. It is also possible to integrate logics (as described in [18]) by specifying them in a logical framework like LF [43].

We next informally sketch how to describe OWL within the framework of institutions:

OWL **as institution:** OWL *signatures consist of sets of atomic classes, individuals and properties.* OWL *signature morphisms map classes to classes, individuals to individuals, and properties to properties. For an* OWL *signature* Σ*, sentences are subsumption relations between classes or properties, membership assertions of individuals in classes and pairs of individuals in properties, complex role inclusions, and some more. Sentence translation along a signature morphism simply replaces non-logical symbols with their image along the morphism. The kinds of symbols are class, individual, object property and data property, respectively, and the set of symbols of a signature is the union of its sets of classes, individuals and properties. Models are (unsorted) first-order structures that interpret concepts as unary and properties as binary predicates, and individuals as elements of the universe of the structure, and satisfaction is the standard satisfaction of description logics. This gives us an institution for* OWL.

In this framework, a basic ontology O over an institution I is a pair (Σ, E) where Σ is a signature and E is a set of Σ-sentences. Given a basic ontology O, we denote by $\mathsf{Sig}(O)$ the signature of the ontology. An ontology morphism $\sigma : (\Sigma_1, E_1) \to (\Sigma_2, E_2)$ is a signature morphism $\sigma : \Sigma_1 \to \Sigma_2$ such that $\sigma(E_1)$ is a logical consequence of E_2.

Several notions of *translations* between institutions can be introduced. The most frequently used variant are *institution comorphisms* [35]. A comorphism from institution L_1 to institution L_2 maps L_1-signatures to L_2-signatures along a functor Φ and Σ-sentences in L_1 to $\Phi(\Sigma)$-sentences in L_2, for each L_1-signature Σ, while $\Phi(\Sigma)$-models are mapped to Σ-models. Again, a satisfaction condition has to be fulfilled. For *institution morphisms*, the directions of the translation of sentences and models are reversed. See [35] for full details.

Comorphism and related morphism will play a major role in applying conceptual blending techniques within the context of hetereogenous knowledge representation languages.

4.2 Features of DOL

An essential novelty introduced in DOL is that a user can specify the ontological base diagram as a DOL theory, from which the colimit and other blendoids can then be computed.[9] This is a crucial task, as the computed colimit ontology depends on the dependencies between symbols that are stored in the diagram. Ontohub, a DOL-enabled repository discussed further in Section 5, is able to use the specification of a base diagram to automatically generate the colimit ontology. In the next section, we illustrate the specification of base diagrams in DOL and the computation of the resulting blendoids by blending house and boat to houseboat and boathouse.

For the purpose of ontology blending the following features of DOL are relevant:

(a) a *basic ontology* O written inline, in a conforming ontology language and serialisation. The semantics is inherited from the ontology language. O can also be an ontology fragment, which means that some of the symbols or axioms may refer to symbols declared outside O (i.e. in an imported ontology). This is mainly used for extensions and equivalences. Here are two sample ontologies in OWL (using Manchester syntax) and Common Logic (using CLIF):

```
Class: Woman EquivalentTo: Person and Female
ObjectProperty: hasParent

(cl-module PreOrder
  (forall (x) (le x x))
  (forall (x y z) (if (and (le x y) (le y z)) (le x z))))
```

(b) an ontology qualified with the ontology *language* that is used to express it (written **language** l : O, where l identifies a language). Similarly, qualifications can also be by *logic* (written **logic** l : O), and/or *serialisation* (written **syntax** s : O).[10]

(c) an IRI reference to an ontology existing on the Web[11], possibly abbreviated using prefixes.[12] For example:

```
%prefix(
  co-ode: <http://owl.cs.manchester.ac.uk/co-ode-files/ontologies/> )%
http://owl.cs.manchester.ac.uk/co-ode-files/ontologies/pizza.owl
co-ode:pizza.owl
```

(d) an *extension* of an ontology by new symbols and axioms, written O_1 **then** O_2, where O_2 is an ontology (fragment) in a conforming ontology language. The resulting signature is that of O_1, augmented with the symbols in O_2. A model of an extension ontology is a model of this signature, that satisfies the axioms on O_2 and is (when appropriately reduced) a model of O_1.

[9]While OBJ3 already provides the possibility to write down theory morphisms, only DOL provides means to collect them into a formally defined diagram; see the **distribtued ontology** construct below.

[10]Some of the following listings omit obvious qualifications for readability.

[11]Note that not all ontologies can be downloaded by dereferencing their IRIs. Implementing a catalogue mechanism in DOL-aware applications might remedy this problem.

[12]Some of the following listings abbreviate IRIs using prefixes but omit the prefix bindings for readability.

An extension can optionally be marked as conservative (%mcons or %ccons after the "**then**"). The semantics is that each O_1-model must have at least one expansion to the whole extension O_1 **then** O_2 (for %mcons) resp. that each logical consequence of O_1 **then** O_2 is already one of O_1 if it is over the signature of O_1 (for %ccons). In case that O_2 does not introduce any new symbols, the keyword %implied can be used instead of %ccons or %mcons; the extension then merely states intended logical consequences. The keyword %def stands for definitional extensions. This is similar to %mcons, but the model expansion must always exist uniquely. The following OWL ontology is an example for the latter:

```
  Class Person
  Class Female
then %def
  Class: Woman EquivalentTo: Person and Female
```

(e) a *union* of two self-contained ontologies (not fragments), written O_1 **and** O_2. Models of this union are those models that are (perhaps after appropriate reduction) models of both O_1 and O_2. For example, the class of commutative monoids can be expressed as

```
algebra:Monoid and algebra:Commutative
```

Forming a union of ontologies is a particularly common operation in the RDF logic, where it is known as merging graphs, see Section 0.3 of [45]; however, the RDF language provides no explicit syntax for this operation. When multiple RDF ontologies ("graphs") contain statements about the same symbol ("resource"), i.e., syntactically, triples having the same subject, the effect is that in the merged graph the resource will have all properties that have previously been stated about it separately. Different kinds of properties, e.g. multilingual labels, geodata, or outgoing links to external graphs, are often maintained in different RDF graphs, which are then merged; consider the following excerpt:

```
{ :UniBremen rdfs:label "Université de Brême"@fr .} and
{ :UniBremen geo:lat "53.108612"^^xsd:float . } and
{ :UniBremen owl:sameAs[13]
    <http://dbpedia.org/page/University_of_Bremen> . }
```

(f) a *translation* of an ontology to a different signature (written O **with** σ, where σ is a signature morphism) or into some ontology language (written O **with translation** ρ, where ρ is an institution comorphism). For example, we can combine an OWL ontology with a first-order axiom (formulated in Common Logic) as follows:

```
ObjectProperty: isProperPartOf
  Characteristics: Asymmetric
  SubPropertyOf: isPartOf
with translation trans:SROIQtoCL
then
        (forall (x y)
          (if
                (and (isProperPartOf x y) (isProperPartOf y z))
                (isProperPartOf x z)))
```

Note that OWL can express transitivity, but not together with asymmetry.

[13]While is borrowed from the *vocabulary* of OWL, it is commonly used in the RDF logic to link to resources in external graphs, which should be treated as if their IRI were the same as the subject's IRI.

(g) a *reduction* of an ontology to a smaller signature Σ is written O **reveal** Σ. Alternatively, it can be written O **hide** Σ, where Σ is the set of symbols to be hidden (i.e. this is equivalent to O **reveal** $\mathsf{Sig}(O) \setminus \Sigma$). The effect is an existential quantification over all hidden symbols. For example, when specifying a group in sorted first-order logic, using the CASL language,

```
sort Elem
ops 0: Elem; __+__: Elem * Elem -> Elem; inv: Elem -> Elem
forall x,y,z . 0 + x       = x
             . x + (y + z) = (x + y) + z
             . x + inv(x)  = 0
reveal Elem, 0, __+__
```

revealing everything except the inverse operation inv results in a specification of the class of all monoids that can be extended with an inverse operation, i.e. the class of all groups with inverse left implicit.

Here is an example of hiding:

```
ontology Pizza = %% a simplified remake of the Pizza ontology [48]
  Individual: TomatoTopping
  Individual: MozzarellaTopping DifferentFrom: TomatoTopping
  ObjectProperty: hasTopping
  Class: VegetarianTopping
    EquivalentTo: { TomatoTopping, MozzarellaTopping, ... }
  Class: VegetarianPizza SubClassOf: some hasTopping VegetarianTopping
  ...
end

ontology Pizza_hide_VegetarianTopping =
  Pizza hide VegetarianTopping
end
```

A reduction to a less expressive logic is written O **hide along** μ, where μ is an institution morphism. This is a common operation in TBox/ABox settings, where an ontology in an expressive language provides the terminology (TBox) used in assertions (ABox) stated in a logic that is less expressive but scales to larger data sets; OWL DL (whose logic is $\mathcal{SROIQ}$) vs. RDF is a typical language combination:

```
ontology TBoxABox =
  Pizza hide along trans:SROIQtoRDF
  then language lang:RDF syntax ser:RDF/Turtle : {
    :myPizza :hasTopping
      [ a :TomatoTopping ], [ a :MozzarellaTopping ] .
  }
```

(h) an *interpolation* of an ontology, either in a subsignature or a sublogic, optionally with respect to a logic L (written O **keep in** Σ **with** L, where Σ is a signature or a logic and L is a logic)[14]. The effect is that sentences not expressible in Σ are weakened or removed, but the resulting theory still has the same L-consequences. The "**with** L" is optional, it defaults to the logic of O. Technically, this is a uniform interpolant [111, 69]. In case that Σ is a sublogic, this is also called approximation [68]. For example, we can interpolate the first-order DOLCE mereology in OWL:[15]

[14] It is also possible to specify a signature and a logic simultaneously: O **keep in** Σ, L_1 **with** L_2

[15] Interpolants need not always exist, and even if they do, tools might only be able to approximate them.

```
DOLCE_Mereology keep in log:OWL
```

Dually, O **forget** Σ **with** L interpolates O with the signature $\mathsf{Sig}(O) \setminus \Sigma$, i.e. Σ specifies the symbols that need to be left out. Cf. the notion of forgetting in [111, 69]. For example,

```
  Pizza forget VegetarianTopping
```

This has a theory-level semantics; i.e., it yields a theory in the reduced signature (without `VegetarianTopping`). By contrast `Pizza hide VegetarianTopping` has a model-level semantics.

(i) a module *extracted* from an ontology, written O **extract** Σ. Here, Σ is a restriction signature, which needs to be a subsignature of $\mathsf{Sig}(O)$. The extracted module is a subontology of O with signature larger than (or equal to) Σ, such that O is a conservative extension of the extracted module. Dually, O **remove** Σ extracts w.r.t. the signature $\mathsf{Sig}(O) \setminus \Sigma$.[16]

```
Pizza remove
  VegetarianTopping
```

Table 1 illustrates some of the connections between (g)–(i). We have three ways of removing the class `VegetarianTopping` from the ontology `Pizza`: (1) using hiding, we keep the model class of `Pizza`, but just remove the interpretation of `VegetarianTopping` from each model. Note that the resulting ontology has

```
VegetarianPizza SubClassOf:
  Annotations: dol:iri (*)
  some hasTopping { TomatoTopping, MozzarellaTopping, ... }
```

as a logical consequence. This is also a consequence of the corresponding uniform interpolant

```
  Pizza forget VegetarianTopping
```

which captures the theory of `Pizza hide VegetarianTopping`. Note that there is a subtle difference between (model-theoretic) hiding and (consequence-theoretic) forgetting: a model satisfying the *theory* of O **hide** Σ might itself not be a model of O **hide** Σ. In examples involving "**with** L", the uniform interpolant can be weaker than the hiding, because it is only required to have the same logical consequences in some language L, and a formula like (*) might not be a formula of L. Finally, an extracted module does not contain (*), because it only selects a subontology, and `Pizza` does not contain (*).

Note that while forget/keep and hide/reveal both work w.r.t. smaller signatures and sublogics, remove/extract does not work for sublogics. This is because remove/extract must always respect the conservative extension property, which may not be possible when projecting to a sublogic. And if conservativity cannot be guaranteed, then forget/keep can be used in any case.

(j) a *minimisation* of an ontology imposes a closed-world assumption on part of the ontology. It forces the non-logical symbols declared in O to be interpreted in a minimal way. This is written **minimize {** O **}**. Symbols declared before the minimised part are considered to be fixed for the minimisation (that is, we minimise among all models with the same reduct). Symbols

[16]Note that the resulting module can still contain symbols from Σ, because the resulting signature may be enlarged.

	remove/extract	forget/keep	hide/reveal
semantic background	conservative extension	uniform interpolation	model reduct
relation to original	subtheory	interpretable	interpretable
approach	theory level	theory level	model level
type of ontology	flattenable	flattenable	elusive
signature of result	$\geq \Sigma$	$= \Sigma$	$= \Sigma$
change of logic	not possible	possible	possible

Table 1: Extract – Forget – Hide

declared after the minimisation can be varied. This is borrowed from circumscription [65, 13]. Alternatively, the non-logical symbols to be minimised and to be varied can be explicitly declared: O **minimize** Σ_1 **vars** Σ_2. For example, in the following OWL theory, B2 is a block that is not abnormal, because it is not specified to be abnormal, and hence it is also on the table.

```
  Class: Block
  Individual: B1 Types: Block
  Individual: B2 Types: Block DifferentFrom: B1
then minimize {
        Class: Abnormal
        Individual: B1 Types: Abnormal }
then
  Class: OnTable
  Class: BlockNotAbnormal EquivalentTo:
    Block and not Abnormal SubClassOf: OnTable
then %implied
  Individual: B2 Types: OnTable
```

Dually to minimisations, there are also maximisations.

(k) an ontology *bridge*, written O_1 **bridge with translation** t O_2, where t is a logic translation. The semantics is that of O_1 **with translation** t **then** O_2. Typically, t will translate a language like OWL to some language for distributed description logic or $\mathcal{E}$-connections [14, 57, 20], and O_2 introduces some axioms involving the relations (introduced by t) between ontologies in O_1. For example,

```
Publications_Combined
bridge with translation trans:MS-OWL2DDL
  %% implicitly added by translation trans:MS-OWL2DDL:
  %% binary relation providing the bridge
  1:Publication $\xrightarrow{\sqsubseteq}$ 2:Publication
  1:PhdThesis $\xrightarrow{\sqsubseteq}$ 2:Thesis
  1:InBook $\xrightarrow{\sqsubseteq}$ 2:BookArticle
  1:Article $\xrightarrow{\sqsubseteq}$ 2:Article
  1:Article $\xrightarrow{\sqsupseteq}$ 2:Article
end
```

(l) a *distributed ontology*: the syntax for specifying distributed ontologies (= diagrams of ontologies and morphisms) in DOL is

```
distributed ontology D = D_1,...,D_m,O_1,...,O_n,M_1,...,M_p,A_1,...,A_k
```

where D_i are distributed ontologies, O_i are ontologies, M_i are morphisms and A_i are alignments. The user specifies a distributed ontology D formed with existing distributed ontologies D_i, extended with ontologies O_i and the morphisms M_i and the diagrams of the alignments A_i (full details regarding alignments is given in [17]).
Models of distributed ontologies are familis of models for the involved individual ontologies that are compatible along the morphisms in the distributed ontology.

(m) a *combination* of ontologies: DOL also provides means for combining a distributed ontology into a new ontology, such that the symbols related in the distributed ontology are identified. The syntax of combinations is `ontology O = combine D`, where D is a distributed ontology, named or specified as above. The semantics of a combination O is the class of models of the colimit ontology of the distributed ontology specified in the combination. Under rather mild technical assumptions, this model class captures exactly the models of the distributed ontology.

The simplest example of a combination is a disjoint union (we here translate OWL ontologies into many-sorted OWL in order to be able to distinguish between different universes of individuals):

```
ontology Publications1 =
  Class: Publication
  Class: Article SubClassOf: Publication
  Class: InBook SubClassOf: Publication
  Class: Thesis  SubClassOf: Publication
  ...

ontology Publications2 =
  Class: Thing
  Class: Article SubClassOf: Thing
  Class: BookArticle SubClassOf: Thing
  Class: Publication SubClassOf: Thing
  Class: Thesis  SubClassOf: Thing
  ...

ontology Publications_Combined =
combine
  1 : Publications1 with translation trans:OWL2MS-OWL,
  2 : Publications2 with translation trans:OWL2MS-OWL
  %% implicitly: Article ↦ 1:Article ...
  %%             Article ↦ 2:Article ...
end
```

(This example will be continued using bridges below.) If links or alignments are present, the semantics of a combination is a quotient of a disjoint union (aligned symbols are identified). Technically, this is a colimit, see [114, 19]. An example for this is given along with the examples for alignments below.

```
    Class: Artifact
    Class: Capability
    ObjectProperty: has_function
        Range: Capability
    ObjectProperty: executes
        Range: Capability
    ObjectProperty: is_located_on
    Class: Person
    Class: Plot
    ObjectProperty: is_inhabited_by
        Domain: House
        Range:  Person
    Class: ServeAsResidence
        SubClassOf: Capability
    Class: ArtifactThatExecutesResidenceFunction
        EquivalentTo: Artifact that executes
                      some ServeAsResidence
        SubClassOf: is_inhabited_by some Person
    Class: House
        SubClassOf: Artifact
            that is_located_on some Plot
            and has_function some
                          ServeAsResidence
```

Figure 4: Ontology `House`

4.3 The classic House+Boat Blend

The main inputs for the blendings consist of two ontologies, one for HOUSE and the other for BOAT. We adapt them from [36] but give a stronger axiomatisation to make them more realistic and ontologically sound. Fig. 4 shows the ontology for HOUSE in OWL Manchester Syntax. The ontology is a fragment introducing several concepts necessary for understanding the basic meaning of the term 'house', including that it is an artefact that has the capability of serving as a residence for people and is generally located on a plot of land. The precise formalisation is not criterial at this point; any adequate ontological description of 'house' would, however, needs to provide similar distinctions.[17]

As discussed above, finding candidate base ontologies and base morphisms is a non-trivial task. For the purpose of this example, we create them manually. The purpose of the example is to show how the DOL specifications naturally allow us to express these kinds of 're-mappings' of relations and entities that are required when considering blends in general. The base ontologies used for the two blends discussed here are both quite simple, they mostly introduce shared concepts and contain only weak axiomatisations. The second base ontology only differs from the first by replacing the class `Agent` by `Person` and two additional classes, namely `Object` and `Site`.

```
ontology base1 =
    Class: Artifact  [...]  Class: Agent
```

[17] In the examples, note that concepts such as 'ArtifactThatExecutesResidenceFunction' are auxiliary symbols that are needed because of limitation of the Manchester Syntax being used, which does not allow the use of complex concepts on the left-hand side of subsumption statements. The ontology for BOAT is axiomatised similarly, it can be found at `http://www.ontohub.org/conceptportal`.

```
    end

ontology base2 =
    Class: Artifact [...]  Class: Person
    Class: Object       Class: Site
    end
```

The blending of boat and house to houseboat is achieved by turning the boat into a habitat and moving the house from a plot of land to a body of water. This can be represented by two interpretations `boat_habitable` and `house_floating`.

```
interpretation boat_habitable : base2 to Boat =
    Object ↦ Boat,
    Site ↦ BodyOfWater

interpretation house_floating : base2 to House =
    Object ↦ House,
    Site ↦ Plot
```

The base ontologies and the interpretations above provide the necessary ingredients for a blending of BOAT and HOUSE to HOUSEBOAT.

In our example, houseboat can be defined by the colimit based on the interpretations. To make the result easier to read, some of the classes are renamed:

```
ontology house_boat =
    combine boat_habitable, house_floating
    with Object ↦ HouseBoat, Site ↦ BodyOfWater
```

This captures formally the informal description of the house+boat blend as often given in examples of blending diagrams. Our specification then allows us to go further and derive both consequences of this and other blends. Here Ontohub is able to compute the colimit, which combines both the BOAT and HOUSE ontologies along the morphism. The colimit inherits the axioms of the input ontologies and the base with appropriate identifications of symbols. Here we just show the declaration of the blended class `Houseboat`:

```
Class: HouseBoat
  SubClassOf: Artifact
    and has_function some MeansOfTransportation
    and has_function some Floating
    and is_navigated_by some Agent
  SubClassOf: Artifact
    and is_located_on some BodyOfWater
    and has_function some ServeAsResidence
```

In the case of blending BOAT and HOUSE to BOATHOUSE, the crucial part in this blend is to view a boat as a kind of "person" that lives in a house. The two ontologies `House` and `Boat` presented above can be blended by selecting a base, which here provides (among others) a class `Agent`, and two interpretations, mapping `Agent` to `Boat` and `Person`, respectively. Therefore, the second base ontology only differs from the first by replacing the class `Agent` by `Person` and two additional classes, namely `Object` and `Site`.

```
ontology base1 =
    Class: Artifact  [...]  Class: Agent
     end
```

In this way, we let a boat play the role of a person (that inhabits a house).[18]

```
interpretation boat_personification :
    base1 to Boat =
    Agent ↦ Boat

interpretation house_import :
    base1 to House =
    Agent ↦ Person

ontology boat_house =
 combine boat_personification , house_import
 with Agent ↦ Boat , House ↦ BoatHouse
```

As before, Ontohub is able to compute the colimit. As above, we present here only the relevant declarations of the blended concept.

```
Class: BoatHouse
  SubClassOf: Artifact
    and is_located_on some Plot
    and has_function some ServeAsResidence
Class: ArtifactThatExecutesResidenceFunction
       EquivalentTo: Artifact
           and executes some ServeAsResidence
       SubClassOf: is_inhabited_by some Boat
```

Figure 5 shows the representation of the ontologies and their relations in Ontohub.

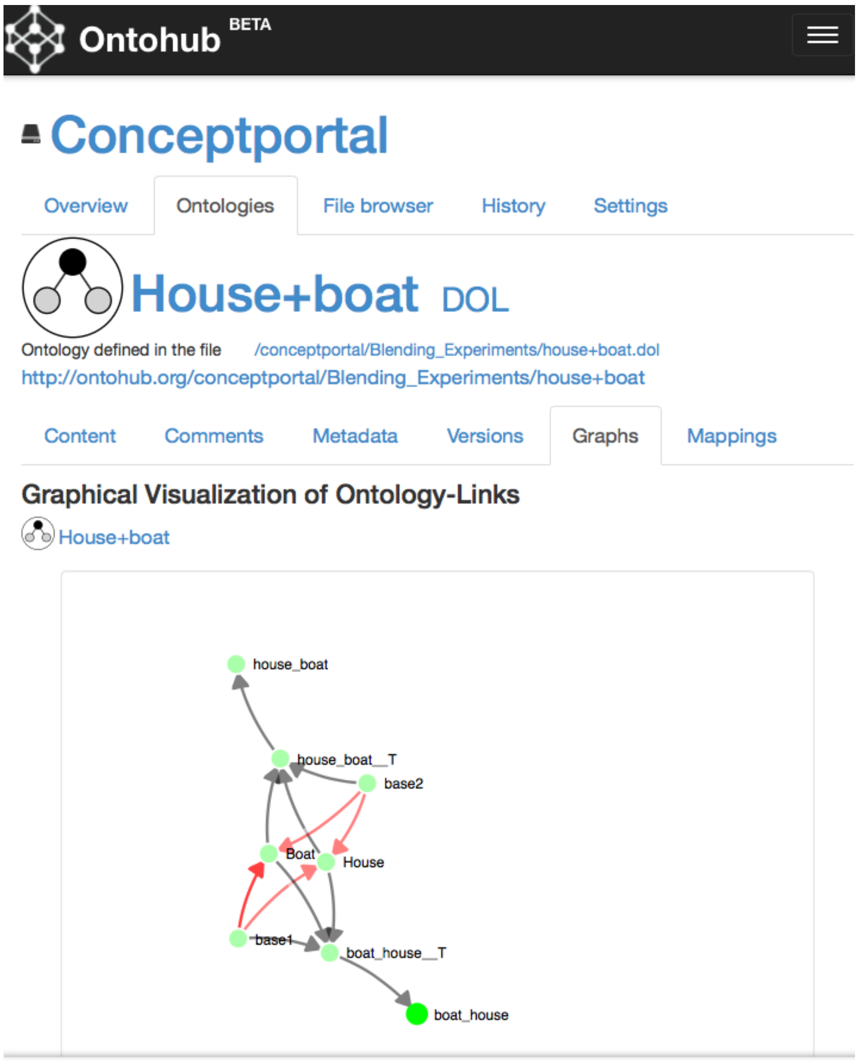
Of course, the possibilities for blending the two concepts do not stop here. For example, we could map the agent in the base ontology to person in the boat ontology. This can be achieved by first defining an additional interpretation and by blending all three interpretations.

```
interpretation boat_import :
    base1 to Boat =
        Agent ↦ Person

ontology boat_house =
 combine boat_personification , house_import ,  boat_import
 with Agent ↦ Boat , House ↦ BoatHouse
```

The resulting blendoid is consistent, but it contains some strange consequences. For example, in the blendoid boats are driven by boats. However, if we are interested both in hosting boats and a hub for autonomous vehicles, this would count as an interesting result. In general, whether such more creative aspects of blendoids are desirable or not will depend on the context of the blending. We will address this issue in the section on evaluation below. It should be noted, however, that an ontologically cleaner axiomatisation of the input spaces makes blending in fact easier — this is because it reveals more clearly the type structure of the inputs, whose modification can then be more elegantly controlled via the base morphisms.

[18]Compared to [36], the advantage of our formulation is that no projections (“retracts”) from a supersort to a subsort are needed. Instead, we can carefully select which parts of the theory of houses and their inhabitants are instantiated with boats.



Ontology defined in the file /conceptportal/Blending_Experiments/house+boat.dol
http://ontohub.org/conceptportal/Blending_Experiments/house+boat

Graphical Visualization of Ontology-Links

House+boat

Figure 5: Blendoid representation and colimit computation via HETS/Ontohub: the screenshot of Ontohub shows Conceptportal, which contains the double-blend of house and boat. In the graph the dots represent the ontologies: the input spaces (House, Boat), the two bases, and the computed blendoids (boat_house, house_boat). (The ontologies boat_house_T and house_boat_T are generated by Onthub as an intermediate step before the terms in the signature are renamed.) The arrows denote the relationships between the ontologies (interpretations, blending, and renaming).

4.4 Variations: Blends of Blends and Partiality

We have discussed a more sophisticated version of the classic HOUSE + BOAT blending in order to illustrate some of the fine detail in the workings of formalised blending in the Goguen tradition, here based on the DOL language. However, the basic blending diagram only covers the most basic situation, that of an 'atomic blend' using basic concepts and one base space. The real power of blending, however, is only unleashed when blends are iterated and when partiality is allowed.

[63] give a detailed and powerful analysis of this in the field of conceptual mathematics. A

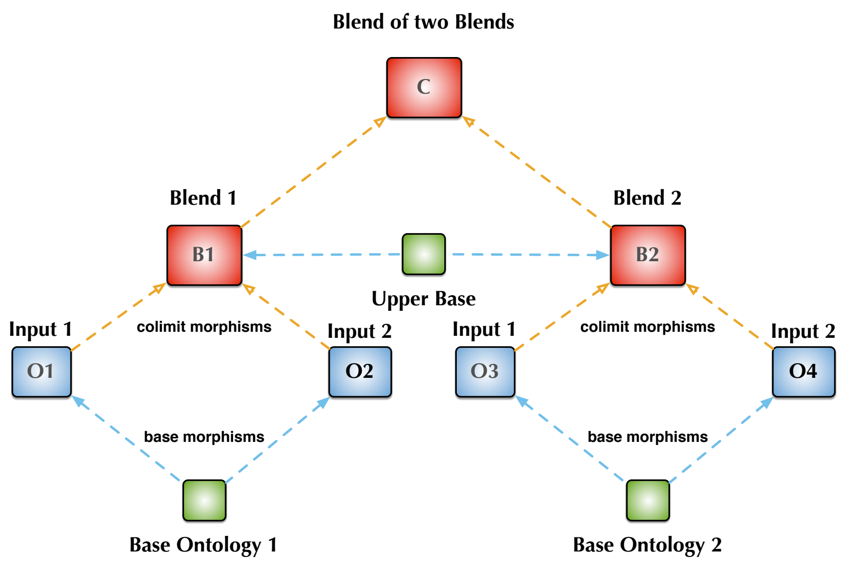


Figure 6: Blending two basic blends into a third.

basic claim they make is that the most sophisticated mathematical concepts have been created, over time, through a tower of blended concepts, generating more and more abstract notions. A basic case is that of arithmetic, where several metaphors, image schemas, and analogies are successively blended into modern number systems such as rationals, reals, or complex numbers, including ‘arithmetic as object collection’, ‘object construction’, the ‘measuring stick metaphor’ and ‘arithmetic as motion along a path’ (see [63] and [41] for further details and [26] for a conceptual blend of the complex numbers along these lines). A detailed formal re-construction of such iterated blends is a challenging task, both conceptually and on a technical level. Figure 6 shows the basic diagrammatical structure of such iterated blends.

Iteration of blends, however, is not the only variation of the basic blendoid structure. Figure 7 shows two triple blends, both have three input spaces, but the one on the left has one base, the one on the right has two base spaces. For instance, we might have 3 inputs that are simultaneously aligned with a basic image schema in the base (left), or we have three ontologies that pairwise interpret different metaphors, e.g. ‘arithmetic as object collection’ and ‘arithmetic as motion along a path’.

Note that on a purely technical level, such complex diagrams can always be reduced to a succession of squares, possibly by duplicating some nodes or adding trivial ones[19]—however, such a reduction loses the direct connection between the diagrammatic representation and the cognitive-conceptual processes that are being formalised here. In a similar vein, Def. 1 introducing the notion of an ontological base diagram in Section 3 easily generalises to the case of partial base morphisms, i.e. where only parts of the signature of an ontology are mapped. Such partial morphisms can be coded as spans of two (total) theory morphisms $B_i \leftarrow dom(\mu_{ij}) \rightarrow I_j$, where the first morphism is the embedding of the domain (actually, the larger $dom(\mu_{ij})$ is, the more defined

[19]A well-known theorem of category theory states that every finite colimit can be expressed by pushouts and initial objects.

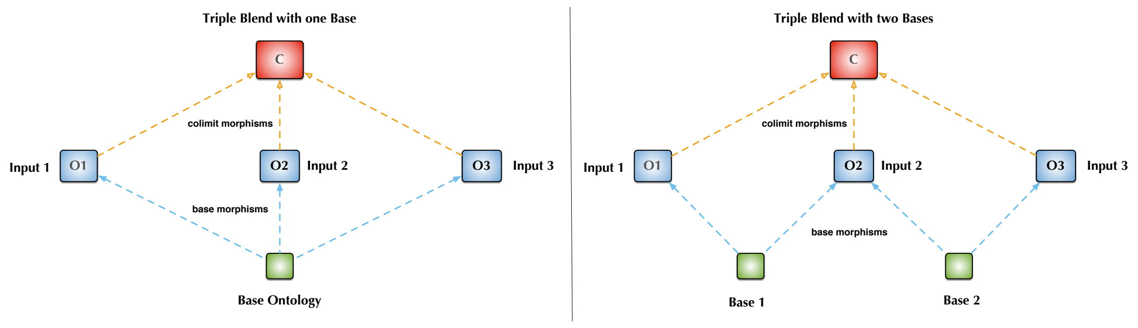


Figure 7: Blending three input spaces using one respectively two base ontologies.

is the partial morphism), and the second action represents the action of the partial morphism.[20] Similarly, arbitrary relations can be coded as spans $B_i \leftarrow R \rightarrow I_j$. Here, $R \subseteq B_i \times I_j$ is a relation, and the arrows are the projections to the first and second component. However, such complexities can be hidden from a user by allowing partial morphisms to be used directly in the specification of a blending diagram, and by letting a tool handle the simulation through total morphisms as discussed above.

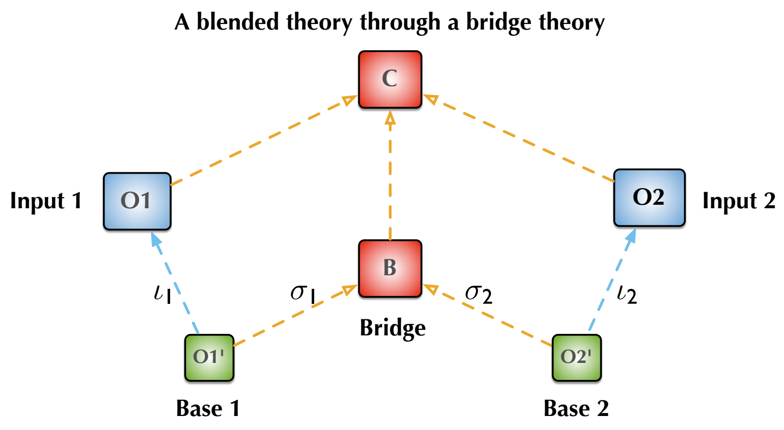


Figure 8: Blending two input spaces through two bases and a bridge theory, deviating from the Goguen construction.

Finally, a more severe deviation from the basic blending diagram is shown in Fig. 8. Here, we interpret Base1 into Input1, Base2 into Input2, and connect the two bases by a bridge theory. For example, the bridge might introduce a higher-level connection between two image schemas which is then used to create the blended theory. An example of this might be where we have image schemas involved with time and with space and combine these first for the definition of a theory in which time and space are linked (as in our navigation examples above or even in the linking between time and space discussed for comics and visual narrative).

Technically, such diagrams are closely related to alignments [114, 17], and to distributed mod-

[20]In this case, the base diagram becomes a bit more complex; in particular, there are minimal nodes $dom(\mu_{ij})$ which have only auxiliary purposes and do not belong to the base.

elling languages [57], which also inspird the DOLterm "distributed ontology" for such diagrams. Concerning the formalisation of conceptual blending, these techniques and diagram patterns will be of particular importance to tackle a computational treatment and formal representation of so-called *generic integration templates* (GIT), i.e. the idea of studying general blending templates, first introduced and discussed in detail by [91], with more examples to be found in [105]. We have begun to study reasoning support for such bridge theories in [83].

5 Blending in Ontohub: The Conceptportal repository

In this section we will discuss the computational and representational support for formalised blending via the Ontohub.org platform as well as the evaluation problem.

5.1 Representation and Computation

To begin, combinations (or, alternatively, the underlying colimits) can be computed directly by the web platform Ontohub. Ontohub is a repository engine for managing distributed heterogeneous ontologies. Ontohub supports a wide range of formal logical and ontology languages and allows for complex inter-theory (concept) mappings and relationships with formal semantics, as well as ontology alignments and blending. Ontohub understands various input languages, among them OWL and DOL.

We describe the basic design and features of Ontohub in general, and outline the extended feature-set that we pursue to add to Ontohub for conceptportal.org — a specialised repository for blending experiments within the distributed Ontohub architecture.

The back-end of Ontohub is the Heterogeneous Tool Set HETS, which is used by Ontohub for parsing, static analysis and proof management of ontologies. HETS can also compute colimits of both OWL and first-order logic diagrams and even approximations of colimits in the case where the input ontologies live in different ontology languages [19].

Computation of colimits in HETS is based on HETS' general colimit algorithm for diagrams of sets and functions (note that signatures in most cases are structured sets, and signature morphisms structure preserving functions.) Such a colimit of sets and functions is computed by taking the disjoint union of all sets, and quotienting it by the equivalence relation generated by the diagram, which more precisely is obtained by the rule that given any element x of an involved set, any images of x under the involved functions are identified. The quotient is computed by selecting a representative of each equivalence class.

A difficulty that arises is that we have to make a choice of these representatives, and therefore of names for the symbols in the colimit, since a symbol is often not identically mapped in the base diagram of the blendoid. The convention in HETS is that, in case of ambiguity, from among all symbols of the equivalence class, that name of the symbol is chosen which is the most frequently occurring one. In any case, the user has control over the namespace because the symbols in the colimit can later be renamed. We can see this for our boathouse example above, where `Agent` appears most often in the diagram and therefore the symbol has been correspondingly renamed.

5.2 Architecture of Ontohub

The current architecture of Ontohub is shown in Fig. 9. The front-end providing the web interface is implemented in Ruby on Rails. Tomcat/Solr is used for efficient indexing and searching. The database backend is PostgreSQL, but any database supported by Rails could be used.

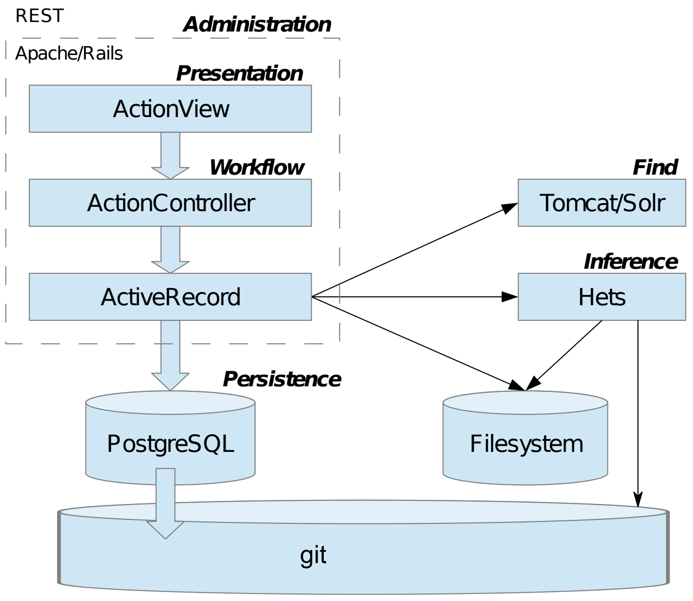


Figure 9: Current architecture of Ontohub

The parsing and inference backend is the Heterogeneous Tool Set (Hets [77, 81]). Hets supports a large number of basic ontology languages and logics, and is capable of describing the structural outline of an ontology from the perspective of DOL, which is not committed to one particular logic. This structural information is stored in the Ontohub database and exposed to human users via a web interface and to machine clients as RDF linked data [46]. Beyond basic ontologies, Ontohub supports linking ontologies, across ontology languages, and creating distributed ontologies as sets of basic ontologies and links among them, as can be seen from the left half of the diagram in Fig. 10, which closely corresponds to the abstract syntax of DOL.

Note that the Ontohub database schema takes advantage of another useful abstraction: Same as basic ontologies, we treat distributed ontologies as ontologies. The entities of distributed ontologies are ontologies (basic, or, in complex scenarios, again distributed), and their sentences are links.

The Open Ontology Repository (OOR) initiative aims at "promot[ing] the global use and sharing of ontologies by (i) establishing a hosted registry-repository; (ii) enabling and facilitating open, federated, collaborative ontology repositories, and (iii) establishing best practices for expressing interoperable ontology and taxonomy work in registry-repositories, where an ontology repository is a facility where ontologies and related information artifacts can be stored, retrieved and managed" [89]. OOR aims at supporting multiple ontology languages, including OWL and Common Logic. OOR is a long-term initiative, which has not resulted in a complete implementa-

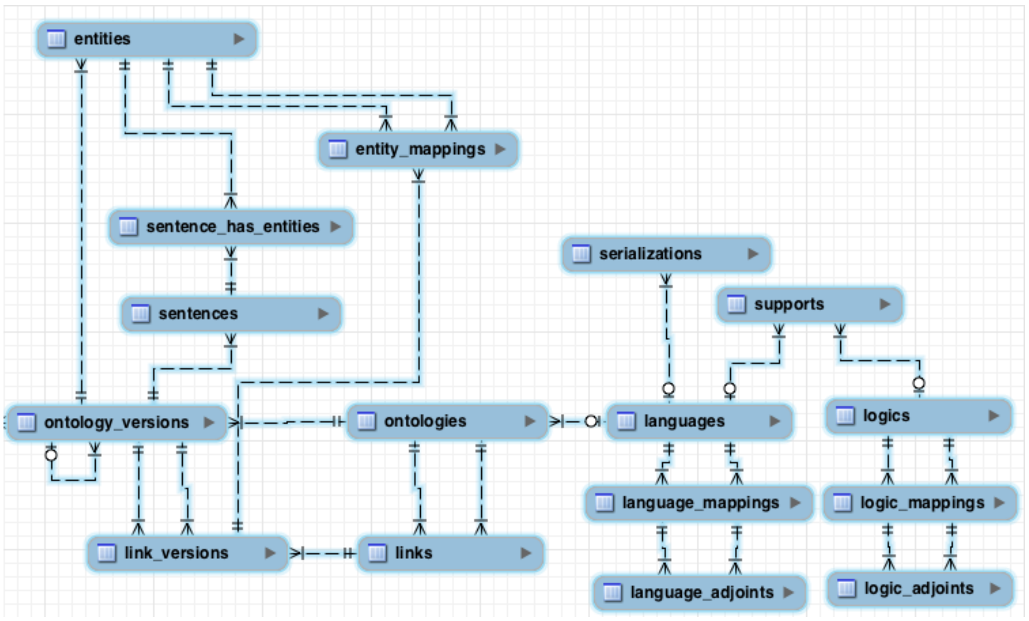


Figure 10: Subset of the Ontohub database schema (entity-relationship diagram using crow's foot notation); left side: ontologies; right side: OntoIOp registry

tion so far[21], but established requirements and designed an architecture, see Fig. 11.[22]

The key feature of the OOR architecture is the decoupling into decentralized services, which are ontologically described (thus arriving at Semantic Web services). With Ontohub, we are moving towards this architecture, while keeping a running and usable system. Fig. 12 depicts the new Ontohub architecture, which will be realized as a set of decoupled RESTful services[23], while Ontohub is still at the center of the architecture.

A *federation* API allows the data exchange with among Ontohub and also BioPortal instances. We therefore have generalized the OWL-based BioPortal API to arbitrary ontology languages, e.g. by abstracting classes and object properties to symbols of various kinds. *Parsing and static analysis* is a service of its own, returning the symbols and sentences of an ontology in XML format. Hets can do this for a large variety of ontology languages, while the OWL API does scale better for very large OWL ontologies. That is, some enhanced services may be provided for a restricted of ontology languages. This is also the case for *presentation*: while Ontohub has a language-independent presentation, WebProtégé provides an enhanced presentation for OWL ontologies. We plan to add enhanced presentation layers for other languages as well (e.g. following the Sigma/SUMO environment for first-order logic). We have already integrated OOPS! as an ontology *evaluation* service (for OWL only), and from the OOPS! API, we have derived a generalized API for use with other evaluation services.

Local inference is done by encapsulating standard batch-processing reasoners (Pellet, Fact,

[21]The main implementation used by OOR is BioPortal, which however does not follow the OOR principles very much.

[22]See `http://tinyurl.com/OOR-Requirement` and `http://tinyurl.com/OOR-Candidate3`, respectively

[23]See `http://tinyurl.com/onto-arch` for detailed API specifications, however not linked to the LoLa ontology yet. OOR already provides an ontologically enriched API.

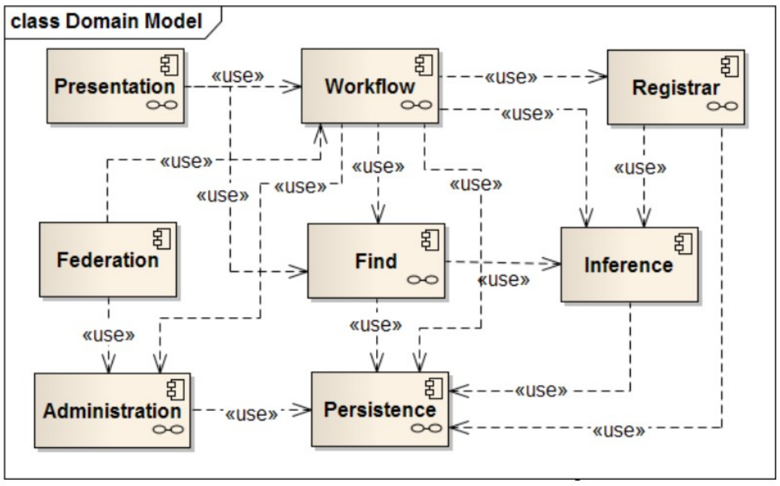


Figure 11: Architecture of the Open Ontology Repository (OOR)

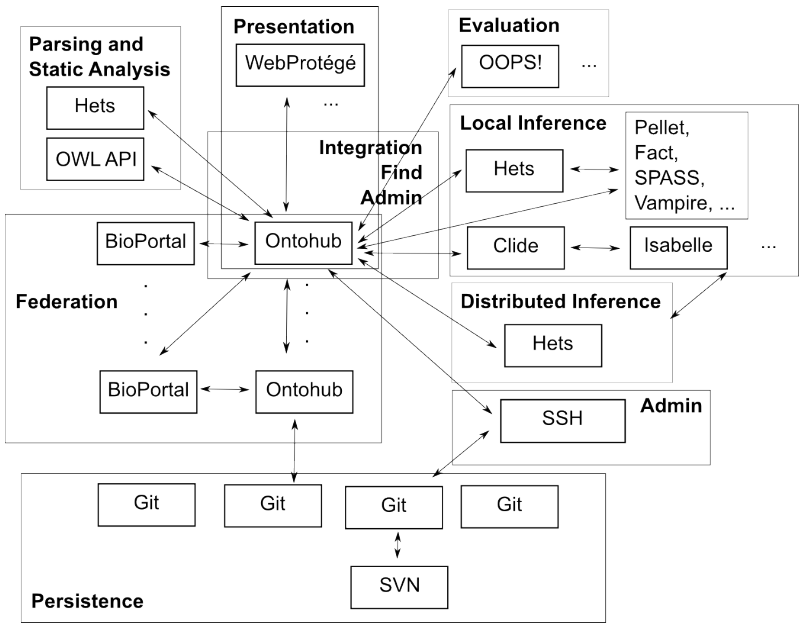


Figure 12: Ontohub in a network of web services

SPASS, Vampire etc.) into a RESTful API, as well as through Hets (which has been interfaced with 15 different reasoners). The integration of interactive provers bears many challenges; a first step is the integration of Isabelle via the web interface Clide [66] developed by colleagues in Bremen, which currently equipped with an API for this purpose. *Distributed inference* is done via Hets. For example, if an interpretation between two ontologies shall be proved, Hets computes what this means in terms of local inferences, and propagates suitable proof obligations to individual ontologies.

Finally, the *persistence* layer is based on Git (via git-svn, also Subversion repositories can be used). Git provides version control and branching of versions. We have equipped Git with a web interface[24], such that ontology versions can be directly edited and committed. Moreover, users can also use a Git repository on their local machine, and commits will be immediately available in Ontohub.

5.3 Evaluating the Blending Space

Optimality principles (see Section 3.2), in particular structural ones, can be used to rank candidate blendoids on-the-fly during the ontology blending process. However, even if they improve on existing logical and heuristic methods, optimality principles will only narrow down the potential candidates and not tell us whether the result is a 'successful' blend of the ontologies. For example, assume that we had optimality principles that would show that from the roughly 1000 candidate blendoids of *House* and *Boat* that Goguen computed, only two candidates $\mathfrak{B}_{hb}$ and $\mathfrak{B}_{bh}$ are optimal. Is either $\mathfrak{B}_{hb}$ or $\mathfrak{B}_{bh}$ any good? And, if so, which of them should we use? To answer these question, it seems natural to apply ontology evaluation techniques.

Ontologies are human-intelligible and machine-interpretable representations of some portions and aspects of a domain that are used as part of information systems. To be more specific, an ontology is a logical theory written in some knowledge representation language, which is associated with some intended interpretation. The intended interpretation is partially captured in the choice of symbols and natural language text (often in the form of annotations or comments). The evaluation of an ontology covers both the logical theory and the intended interpretation, their relationship to each other, and how they relate to the requirements that are derived from the intended use within a given information system. Therefore, ontology evaluation is concerned not only with formal properties of logical theories (e.g., logical consistency), but, among other aspects, with the *fidelity* of an ontology; that is whether the formal theory accurately represents the intended domain [85]. For example, if $\mathfrak{B}_{hb}$ is an excellent representation of the concept *houseboat*, then $\mathfrak{B}_{hb}$ provides a poor representation of the concept *boathouse*. Thus, any evaluation of the blend $\mathfrak{B}_{hb}$ depends on what domain $\mathfrak{B}_{hb}$ is intended to represent.

Given these considerations, $\mathfrak{B}_{hb}$ and $\mathfrak{B}_{bh}$ are not ontologies, they are logical theories that are the result of the blending of two logical theories that are part of ontologies. This is illustrated by the following thought-experiment: let's assume the theory $\mathfrak{B}_{hb}$ captures the concept *houseboat* very well, and that $\mathfrak{B}_{hb}$ is not the result of some automatic blending process, but was intentionally developed by an ontology engineer. In case that the ontology engineer intended to develop an ontology of houseboats $\mathfrak{B}_{hb}$, he would have done very well. However, if the engineer intended to develop an ontology of boathouses, then $\mathfrak{B}_{hb}$ would be a poor outcome. In other words, the

[24]See `https://github.com/eugenk/bringit`

ontology consisting of $\mathfrak{B}_{hb}$ and the intention *houseboat* would have high fidelity, but the ontology consisting of $\mathfrak{B}_{hb}$ and the intention *boathouse* would have low fidelity. Thus, the evaluation of the theory $\mathfrak{B}_{hb}$ is dependent on the domain it is supposed to represent.

The lesson from this thought experiment is that the evaluation of the results of ontology blending is dependent on the intended goal and, more generally, on the requirements that one expects the outcome of the blending process to meet. One way to capture these requirement is similar to competency questions, which are widely used in ontology engineering [38]. Competency questions are usually initially captured in natural language; they specify examples for questions that an ontology needs to be able to answer in a given scenario. By formalising the competency questions one can use automatic theorem provers to evaluate whether the ontology meets the intended interpretation.

The requirements that are used to select between the different blends fall, roughly, into two categories: *ontological constraints* and *consequence requirements*. Ontological constraints prevent the blends from becoming 'too creative' by narrowing the space for conceptual blending. E.g., it may be desirable to ensure that the `is_inhabited_by` relationship is asymmetric and that `is_navigated_by` is irreflexive. To achieve that any blendoid can be checked for logical consistency with the following ontology:

```
ontology OntologicalConstraints =
  ObjectProperty: is_inhabited_by
    Characteristics: Asymmetric
  ObjectProperty: is_navigated_by
    Characteristics: Irreflexive
```

Given these requirements, any blendoid that involves a house that lives in itself, or any boat navigated by itself (see the blendoid `boat_house1` above) would be discarded.

Consequence requirements specify the kind of characteristics the blendoid is supposed to have. E.g., assume the purpose of the conceptual blending is to find alternative housing arrangements, because high land prices make newly built houses unaffordable. In this case, the requirement could be 'a residence that is not located on a plot of land', which can be expressed in OWL as follows:

```
ontology ConsequenceRequirements =
[...]
  Class PlotFreeResidence
    EquivalentTo: Residence
       and (is_located_on only (not (Plot)))
```

For the evaluation of a blendoid against requirements (both ontological constraints and consequence requirements) it is often not sufficient to just consider the information that is contained in the blendoid itself. Some background knowledge usually needs to be added in order to evaluate a blendoid.

Background knowledge plays another crucial role in the blending process, which we have not addressed here so far. The basic blending diagram in Figure 3 presents a static view, which describes how two input spaces, a base, and two interpretations give rise to a blendoid. However, any system that attempts to automate conceptual blending will need to perform not one but many blends in order to get a decent result. In this process, the background knowledge and the evaluation of previous blending results can be utilised in the selection of candidate bases and interpretations. Further, the violation of ontological constraints may be symptom of an attempt to blend input spaces that are too rich. In these cases, the result of the evaluation can be used to guide heuristics, which remove information from the original input spaces that may have caused the violation of the ontological constraints. The result is a new, weakened input space, which may be easier to blend. In short, following a proposal by Marco Schorlemmer discussed in detail in [84], we envision an approach where background knowledge and evaluation are driving an iterative blending process, as illustrated in Figure 13.

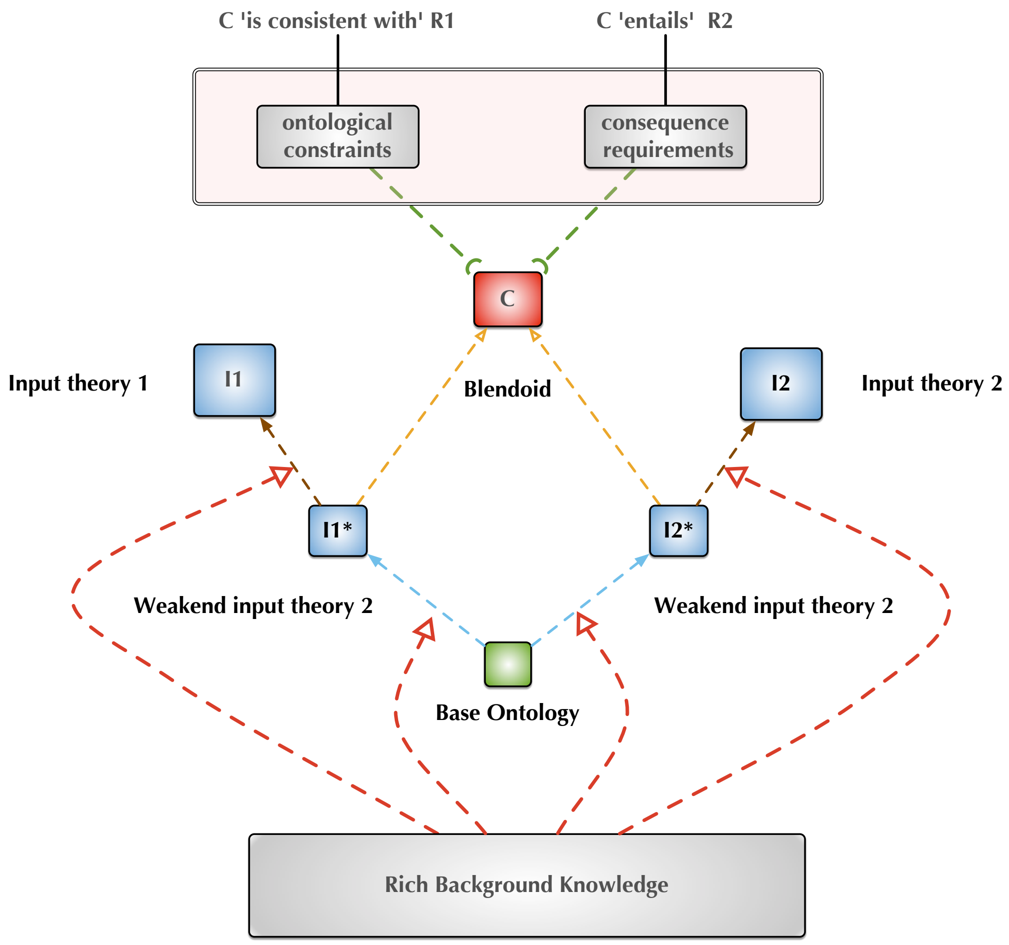


Figure 13: The core Schorlemmer model for computational blending enriched with evaluation and background layers

Ontohub allows to use ontological constraints and consequence requirements to evaluate blended concepts automatically. The requirements are managed as DOL files, which allow to express that a given blendoid (together with some background knowledge) is logically consistent with a set of ontological constraints or that it entails some consequence requirements. The requirements themselves may be stored as regular ontology files (e.g., in OWL Manchester syntax). Ontohub executes the DOL files with the help of integrated automatic theorem provers, and is able to detect whether a blendoid meets the specified requirements. At this time, the evaluation of blendoids for ontological constraints and consequence requirements depends on the use of DOL files. We are planning to integrate this functionality into the GUI of Ontohub to make it more convenient for the user.

6 Conclusions

The work presented in this report follows a research line in which blending processes are primarily controlled through mappings and their properties [31, 27, 107, 93]. By introducing blending techniques to ontology languages, we have provided a method which allows us to combine two or more thematically different ontologies into a newly created ontology, the blendoid, describing a novel concept or domain. The blendoid creatively mixes information from both input ontologies on the basis of structural commonalities of the inputs and selective combination of their axiomatisations.

We have moreover illustrated that the Ontohub/HETS tool ecosystem and the DOL language provide an excellent starting point for developing the theory and practice of ontology blending further [78]. They (1) support various ontology languages and their heterogeneous integration [56]; (2) allow the specification of theory interpretations and other morphisms between ontologies [59]; (3) support the computation of colimits as well as the approximation of colimits in the heterogeneous case [19]; (4) provide (first) solutions for automatically computing a base ontology through ontology intersection [60] and blendoid evaluation using requirements [62, 84].

In particular, we have shown that the blending of ontologies can be declaratively encoded in a DOL theory representing the respective blending diagram—here, primarily employing the homogeneous fragment of DOL just using OWL ontologies. Blendoid ontologies, as well as their components, i.e. input and base ontologies, can be stored, formally related, and checked for consistency within Conceptportal, a repository node within Ontohub dedicated to blending experiments carried out across the COINVENT project [99]. Ontohub moreover gives access to thousands of ontologies from a large number of different scientific and common sense domains, searchable via rich metadata annotation, logics used, formality level, and other dimensions, to provide not only a rich pool of ontologies for blending experiments, but also for the evaluation of newly created concepts.

Of course, constructing a homogeneous blendoid from a basic blending diagram is one of the simplest cases of conceptual blending. As discussed in Section 4.4, on a technical level a blendoid is just like an alignment diagram, except that instead of dealing with synonymy and homonymy relations, and just signature in the base, in the blendoid case we are dealing with selectively merging axioms. Following this intuition, a whole range of more complex alignment and theory combination techniques can be combined with the basic blending ideas of Goguen: this includes constructions such as W-alignments [114, 61, 17], and connections of theories following the $\mathcal{E}$-connection/DDL paradigm [57, 14, 83].

The next important milestone for computational conceptual blending will be to make the step from a *reconstructive* approach, where conceptual blending is illustrated by blending one concept (e.g., houseboat) with the help of some carefully selected input spaces (e.g., a house and a boat) and a hand-crafted base ontology, to a system that *autonomously* selects two (or more) ontologies from a repository in Ontohub and attempts to blend them in a way that meets some given requirements. In [84], we have described the first steps towards designing a computational architecture that performs conceptual blending autonomously and self-evaluates its own creations.

Within the extensive literature on conceptual blending, only few attempts have been made at a (more or less) complete automation of the blending process; notable exceptions include [36], [93], [64], and [109, 108]. To make concept invention via ontological blending more feasible in practice from within Ontohub, a number of further plugins into the architecture and refinements are planned covering in particular the automatic creation of base ontologies together with their mappings, the implementation of filtering blendoids by structural optimality principles and preference orders on morphisms, as well as the addition of more ontologically motivated evaluation techniques as discussed above.

References

[1] J. Adámek, H. Herrlich, and G. Strecker. *Abstract and Concrete Categories*. Wiley, New York, 1990.

[2] Michael Balser, Wolfgang Reif, Gerhard Schellhorn, and Kurt Stenzel. Kiv 3.0 for provably correct systems. In Dieter Hutter, Werner Stephan, Paolo Traverso, and Markus Ullmann, editors, *FM-Trends*, volume 1641 of *Lecture Notes in Computer Science*, pages 330–337. Springer, 1998.

[3] John A. Bateman. Language and Space: a two-level semantic approach based on principles of ontological engineering. *International Journal of Speech Technology*, 13(1):29–48, 2010.

[4] John A. Bateman. The decomposability of semiotic modes. In Kay L. O'Halloran and Bradley A. Smith, editors, *Multimodal Studies: Multiple Approaches and Domains*, Routledge Studies in Multimodality, pages 17–38. Routledge, London, 2011.

[5] John A. Bateman and Francisco O. D. Veloso. The Semiotic Resources of Comics in Movie Adaptation: Ang Lee's *Hulk* (2003) as a case study. *Studies in Comics*, 4(1):137–159, 2013.

[6] Peter Baumgartner, Alexander Fuchs, and Cesare Tinelli. Implementing the Model Evolution Calculus. *Special Issue of the International Journal of Artificial Intelligence Tools (IJAIT)*, 15(1), 2005.

[7] C. Benzmüller, L. C. Paulson, F. Theiss, and A. Fietzke. LEO-II - A Cooperative Automatic Theorem Prover for Classical Higher-Order Logic (System Description). In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *IJCAR*, volume 5195 of *LNCS*, pages 162–170. Springer, 2008.

[8] Christoph Benzmüller, Florian Rabe, and Geoff Sutcliffe. THF0 – the core of the TPTP language for higher-order logic. In Alessandro Armando, Peter Baumgartner, and Gilles Dowek, editors, *IJCAR 2008*, volume 5195 of *Lecture Notes in Computer Science*, pages 491–506. Springer, 2008.

[9] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May 2001.

[10] Mehul Bhatt, Joana Hois, and Oliver Kutz. Ontological modelling of form and function in architectural design. *Applied Ontology*, 7(3):233–267, 2012.

[11] M. Bidoit and P. D. Mosses. CASL *User Manual*, volume 2900 of *LNCS*. Springer, 2004. `www.cofi.info`.

[12] Max Black. More about metaphor. In Andrew Ortony, editor, *Metaphor and Thought*, pages 19–43. Cambridge University Press, Cambridge, 1979.

[13] Piero A. Bonatti, Carsten Lutz, and Frank Wolter. The complexity of circumscription in DLs. *J. Artif. Intell. Res. (JAIR)*, 35:717–773, 2009.

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

[15] T. Borzyszkowski. Higher-order logic and theorem proving for structured specifications. In D. Bert, C. Choppy, and P. D. Mosses, editors, *WADT*, volume 1827 of *LNCS*, pages 401–418. Springer, 1999.

[16] Chad E. Brown. Satallax: An automatic higher-order prover. In Bernhard Gramlich, Dale Miller, and Uli Sattler, editors, *IJCAR*, volume 7364 of *LNCS*, pages 111–117. Springer, 2012.

[17] M. Codescu, T. Mossakowski, and O. Kutz. A Categorical Approach to Ontology Alignment. In *Proc. of the 9th International Workshop on Ontology Matching (OM-2014)*, ISWC-2014, Riva del Garda, Trentino, Italy, 2014. CEUR-WS.

[18] Mihai Codescu, Fulya Horozal, Michael Kohlhase, Till Mossakowski, Florian Rabe, and Kristina Sojakova. Towards logical frameworks in the heterogeneous tool set hets. In Till Mossakowski and Hans-Jörg Kreowski, editors, *WADT 2010*, volume 7137 of *Lecture Notes in Computer Science*, pages 139–159. Springer, 2012.

[19] Mihai Codescu and Till Mossakowski. Heterogeneous colimits. In Frédéric Boulanger, Christophe Gaston, and Pierre-Yves Schobbens, editors, *MoVaH'08 Workshop on Modeling, Validation and Heterogeneity*, pages 131–140. IEEE press, 2008.

[20] Bernardo Cuenca Grau, Bijan Parsia, and Evren Sirin. Ontology Integration Using $\mathcal{E}$-connections. In Heiner Stuckenschmidt, Christine Parent, and Stefano Spaccapietra, editors, *Modular Ontologies—Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *LNCS*. Springer, 2009.

[21] Will Eisner. *Comics and Sequential Art*. Kitchen Sink Press Inc., Princeton, WI, 1992.

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

[23] G. Fauconnier and M. Turner. Conceptual integration networks. *Cognitive Science*, 22(2):133—187, 1998.

[24] G. Fauconnier and M. Turner. *The Way We Think: Conceptual Blending and the Mind's Hidden Complexities*. Basic Books, 2003.

[25] Gilles Fauconnier. *Mappings in Thought and Language*. Cambridge University Press, Cambridge, 1997.

[26] J. Fleuriot, E. Maclean, A. Smaill, and D. Winterstein. Reinventing the Complex Numbers. In Tarek Besold, Kai-Uwe Kühnberger, Marco Schorlemmer, and Alan Smaill, editors, *Computational Creativity, Concept Invention, and General Intelligence, Proc. of the 3rd Int. Workshop C3GI@ECAI-14*, volume 1-2014, Prague, Czech Republic, August 19 2014. Publications of the Institute of Cognitive Science, Osnabrück.

[27] K. Forbus, B. Falkenhainer, and D. Gentner. The structure-mapping engine. *Artificial Intelligence*, 41:1–63, 1989.

[28] Charles J. Forceville and Eduardo Urios-Aparisi, editors. *Multimodal Metaphor*. Mouton de Gruyter, Berlin/New York, 2009.

[29] A. Frank and W. Kuhn. A specification language for interoperable GIS. In M.F. Goodchild, M. Egenhofer, R Fegeas, and C. Kottmann, editors, *Interoperating geographic information systems*, pages 123–132. Kluwer, Norwell, MA, 1999.

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

[31] D. Gentner. Structure mapping: A theoretical framework for analogy. *Cognitive Science*, 7(2):155–170, 1983.

[32] J. A. Goguen. An Introduction to Algebraic Semiotics, with Applications to User Interface Design. In *Computation for Metaphors, Analogy and Agents*, number 1562 in LNCS, pages 242–291. Springer, 1999.

[33] J. A. Goguen and R. M. Burstall. Institutions: Abstract Model Theory for Specification and Programming. *Journal of the Association for Computing Machinery*, 39(1):95–146, 1992. Predecessor in: LNCS 164, 221–256, 1984.

[34] J. A. Goguen and G. Malcolm. *Algebraic Semantics of Imperative Programs*. MIT, 1996.

[35] Joseph Goguen and Grigore Roşu. Institution morphisms. *Formal aspects of computing*, 13:274–307, 2002.

[36] Joseph A. Goguen and D. Fox Harrell. Style: A Computational and Conceptual Blending-Based Approach. In Shlomo Argamon and Shlomo Dubnov, editors, *The Structure of Style: Algorithmic Approaches to Understanding Manner and Meaning*, pages 147–170. Springer, Berlin, 2010.

[37] Tom Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43(5/6):907–928, 1995.

[38] Michael Grüninger and Mark S Fox. The role of competency questions in enterprise engineering. In *Benchmarking—Theory and Practice*, pages 22–31. Springer, 1995.

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

[40] Nicola Guarino. The Ontological Level. In Roberto Casati, Barry Smith, and G. White, editors, *Philosophy and the Cognitive Sciences*, pages 443–456. Hölder-Pichler-Tempsky, Vienna, 1994.

[41] Markus Guhe, Alison Pease, Alan Smaill, Maricarmen Martínez, Martin Schmidt, Helmar Gust, Kai-Uwe Kühnberger, and Ulf Krumnack. A computational account of conceptual blending in basic mathematics. *Cognitive Systems Research*, 12(3–4):249–265, 2011.

[42] Helmar Gust, Kai-Uwe Kühnberger, and Ute Schmid. Metaphors and anti-unification. In *Proc. Twenty-First Workshop on Language Technology: Algebraic Methods in Language Processing, Verona, Italy*, pages 111–123, 2003.

[43] R. Harper, F. Honsell, and G. Plotkin. A framework for defining logics. *Journal of the Association for Computing Machinery*, 40(1):143–184, 1993.

[44] Christopher Hart. Critical discourse analysis and metaphor: toward a theoretical framework. *Critical Discourse Studies*, 5(2):91–106, 2008.

[45] Patrick Hayes. RDF semantics. W3C recommendation, World Wide Web Consortium (W3C), 2004.

[46] Tom Heath and Christian Bizer. *Linked Data: Evolving the Web into a Global Data Space*. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool, San Rafael, CA, 1 edition, 2011.

[47] J. Hois, O. Kutz, T. Mossakowski, and J. Bateman. Towards Ontological Blending. In *Proc. of the The 14th International Conference on Artificial Intelligence: Methodology, Systems, Applications (AIMSA-2010)*, Varna, Bulgaria, September 8th–10th, 2010.

[48] Matthew Horridge. *Protégé OWL Tutorial*.

[49] K. M. Jaszczolt. On Translating 'What Is Said': *Tertium Comparationis* in Contrastive Semantics and Pragmatics. In *Meaning Through Language Contrast Vol. 2*, pages 441–462. J. Benjamins, 2003.

[50] M. Johnson. *The Body in the Mind. The Bodily Basis of Meaning, Imagination, and Reasoning*. The University of Chicago Press, 1987.

[51] Zoltan Kövescses. *Metaphor: A Practical Introduction*. Oxford University Press, Oxford, 2 edition, 2010.

[52] W. Kuhn. Modeling the Semantics of Geographic Categories through Conceptual Integration. In *Proc. of GIScience 2002*, pages 108–118. Springer, 2002.

[53] Werner Kuhn. Semantic Reference Systems. *International Journal of Geographical Information Science*, 17(5):405–409, 2003.

[54] O. Kutz, J. Bateman, F. Neuhaus, T. Mossakowski, and M. Bhatt. E pluribus unum: Formalisation, Use-Cases, and Computational Support for Conceptual Blending. In T. R Besold, M. Schorlemmer, and A. Smaill, editors, *Computational Creativity Research: Towards Creative Machines*, Thinking Machines. Atlantis/Springer, 2014.

[55] O. Kutz, D. Lücke, and T. Mossakowski. Heterogeneously Structured Ontologies—Integration, Connection, and Refinement. In *Proc. KROW 2008*, volume 90 of *CRPIT*, pages 41–50. ACS, 2008.

[56] O. Kutz, D. Lücke, T. Mossakowski, and I. Normann. The OWL in the CASL—Designing Ontologies Across Logics. In *Proc. of OWLED-08*, volume 432. CEUR, 2008.

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

[58] O. Kutz, T. Mossakowski, J. Hois, M. Bhatt, and J. Bateman. Ontological Blending in DOL. In Tarek Besold, Kai-Uwe Kühnberger, Marco Schorlemmer, and Alan Smaill, editors, *Computational Creativity, Concept Invention, and General Intelligence, Proc. of the 1st Int. Workshop C3GI@ECAI*, volume 01-2012, Montpellier, France, August 27 2012. Publications of the Institute of Cognitive Science, Osnabrück.

[59] O. Kutz, T. Mossakowski, and D. Lücke. Carnap, Goguen, and the Hyperontologies: Logical Pluralism and Heterogeneous Structuring in Ontology Design. *Logica Universalis*, 4(2):255–333, 2010. Special Issue on 'Is Logic Universal?'.

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

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

[62] Oliver Kutz, Fabian Neuhaus, Till Mossakowski, and Mihai Codescu. Blending in the Hub—Towards a collaborative concept invention platform. In *Proc. of the 5th International Conference on Computational Creativity*, Ljubljana, Slovenia, June 10–13 2014.

[63] George Lakoff and Rafael E. Núñez. *Where Mathematics Comes From*. Basic Books, 2000.

[64] Boyang Li, Alexander Zook, Nicholas Davis, and Mark O. Riedl. Goal-Driven Conceptual Blending: A Computational Approach for Creativity. In *Proc. of the 2012 International Conference on Computational Creativity*, Dublin, Ireland, 2012.

[65] V. Lifschitz. Circumscription. In *Handbook of Logic in Artificial Intelligence and Logic Programming*, volume 3, pages 297–352. Oxford University Press, 1994.

[66] Christoph Lüth and Martin Ring. A web interface for isabelle: The next generation. In Jacques Carette, James H. Davenport, Wolfgang Windsteiger, Petr Sojka, David Aspinall, and Christoph Lange, editors, *Conferences on Intelligent Computer Mathematics CICM 2013*, Springer LNCS.

[67] Klaus Lüttich and Till Mossakowski. Reasoning support for CASL with automated theorem proving systems. In J. Fiadeiro, editor, *WADT 2006*, number 4409 in LNCS, pages 74–91. Springer, 2007.

[68] Carsten Lutz, Inanç Seylan, and Frank Wolter. An automata-theoretic approach to uniform interpolation and approximation in the description logic el. In Gerhard Brewka, Thomas Eiter, and Sheila A. McIlraith, editors, *KR*. AAAI Press, 2012.

[69] Carsten Lutz and Frank Wolter. Foundations for Uniform Interpolation and Forgetting in Expressive Description Logics. In Toby Walsh, editor, *IJCAI*, pages 989–995. IJCAI/AAAI, 2011.

[70] C. Mamakos, P. Stefaneas, M. Dimarogkona, and J. Ireson-Paine. Polytropos Project: Experiments in Blending. In Tarek Besold, Kai-Uwe Kühnberger, Marco Schorlemmer, and Alan Smaill, editors, *Computational Creativity, Concept Invention, and General Intelligence, Proc. of the 3rd Int. Workshop C3GI@ECAI-14*, volume 1-2014, Prague, Czech Republic, August 19 2014. Publications of the Institute of Cognitive Science, Osnabrück.

[71] M. Martinez, T. R. Besold, A. Abdel-Fattah, K.-U. Kühnberger, H. Gust, M. Schmidt, and U. Krumnack. Towards a Domain-Independent Computational Framework for Theory Blending. In *Proc. of the AAAI Fall 2011 Symposium on Advances in Cognitive Systems*, 2011.

[72] Scott McCloud. *Understanding comics: the invisible art*. HarperPerennial, New York, 1994.

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

[74] Arthur I. Miller. Metaphor and scientific creativity. In Fernand Hallyn, editor, *Metaphor and Analogy in the Sciences*, pages 147–164. Kluwer Academic Publishers, Dordrecht, 2000.

[75] T. Mossakowski, M. Codescu, O. Kutz, C. Lange, and M. Gruninger. Proof Support for Common Logic. In *Proc. of the Workshop on Automated Reasoning for Quantified Non-Classical Logic (ARQNL)*, July 23, Vienna Summer of Logic, 2014.

[76] T. Mossakowski, O. Kutz, and M. Codescu. Ontohub: A semantic repository for heterogeneous ontologies. In *Proc. of the Theory Day in Computer Science (DACS-2014)*, Satellite workshop of ICTAC-2014, University of Bucharest, September 15–16, 2014.

[77] Till Mossakowski. Hets: the Heterogeneous Tool Set.

[78] Till Mossakowski, Oliver Kutz, Mihai Codescu, and Christoph Lange. The Distributed Ontology, Modeling and Specification Language. In Chiara Del Vescovo, Torsten Hahmann, David Pearce, and Dirk Walther, editors, *Proceedings of the 7th International Workshop on Modular Ontologies (WoMO-13)*, volume 1081. CEUR-WS, 2013.

[79] Till Mossakowski, Oliver Kutz, and Christoph Lange. Semantics of the distributed ontology language: Institutes and institutions. In Narciso Martí-Oliet and Miguel Palomino, editors, *Recent Trends in Algebraic Development Techniques, 21th International Workshop, WADT 2012*, volume 7841 of *Lecture Notes in Computer Science*, pages 212–230. Springer, 2013.

[80] Till Mossakowski, Oliver Kutz, Fabian Neuhaus, Mihai Codescu, Christoph Lange, Michael Gruninger, and Maria Keet. The distributed ontology, modeling and specification language. Draft answer to the OMG RFP "OntoIOp", 2014.

[81] Till Mossakowski, Christian Maeder, and Klaus Lüttich. The Heterogeneous Tool Set. In Orna Grumberg and Michael Huth, editors, *TACAS 2007*, volume 4424 of *Lecture Notes in Computer Science*, pages 519–522. Springer-Verlag Heidelberg, 2007.

[82] Peter D. Mosses, editor. CASL *Reference Manual*, volume 2960 of *Lecture Notes in Computer Science*. Springer, 2004.

[83] Claudia Nalon and Oliver Kutz. Towards Resolution-based Reasoning for Connected Logics. *Elsevier's Electronic Notes in Theoretical Computer Science (ENTCS)*, 305:85–102, 2014. Post-proceedings of the 8th Workshop on Logical and Semantic Frameworks (LSFA-13).

[84] Fabian Neuhaus, Oliver Kutz, Mihai Codescu, and Till Mossakowski. Fabricating Monsters is Hard: Towards the Automation of Conceptual Blending. In Tarek Besold, Kai-Uwe Kuehnberger, Marco Schorlemmer, and Alan Smaill, editors, *Computational Creativity, Concept Invention, and General Intelligence, Proc. of the 3rd Int. Workshop C3GI@ECAI-14*, volume 1-2014, Prague, Czech Republic, August 19 2014. Publications of the Institute of Cognitive Science, Osnabrück.

[85] Fabian Neuhaus, Amanda Vizedom, Ken Baclawski, Mike Bennett, Mike Dean, Michael Denny, Michael Grüninger, Ali Hashemi, Terry Longstreth, Leo Obrst, et al. Towards ontology evaluation across the life cycle: The Ontology Summit 2013. *Applied Ontology*, 8(3):179–194, 2013.

[86] T. Nipkow, L. C. Paulson, and M. Wenzel. *Isabelle/HOL — A Proof Assistant for Higher-Order Logic*. Springer Verlag, 2002.

[87] I. Normann. *Automated Theory Interpretation*. PhD thesis, Jacobs University Bremen, 2009.

[88] R. E. Núñez. Creating mathematical infinities: Metaphor, blending, and the beauty of transfinite cardinals. *Journal of Pragmatics*, 37:1717–1741, 2005.

[89] Open Ontology Repository (OOR), 2012.

[90] OWL Working Group. OWL 2 web ontology language: Document overview. W3C recommendation, World Wide Web Consortium (W3C), October 2009.

[91] Christóbal Pagán Cánovas. Erotic Emissions in Greek Poetry: A Generic Integration Network. *Cognitive Semiotics*, 6:7–32, 2010.

[92] Björn Pelzer and Christoph Wernhard. System description: E-KRHyper. In Frank Pfenning, editor, *CADE*, volume 4603 of *Lecture Notes in Computer Science*, pages 508–513. Springer, 2007.

[93] F. C. Pereira. *Creativity and Artificial Intelligence: A Conceptual Blending Approach*, volume 4 of *Applications of Cognitive Linguistics (ACL)*. Mouton de Gruyter, Berlin, December 2007.

[94] Francisco C. Pereira and Amilcar Cardoso. Optimality Principles for Conceptual Blending: A First Computational Approach. *AISB Journal*, 1(4), 2003.

[95] G. D. Plotkin. A note on inductive generalization. *Machine Intelligence*, 5:153–163, 1970.

[96] Alexandre Riazanov and Andrei Voronkov. The design and implementation of VAMPIRE. *AI Communications*, 15(2-3):91–110, 2002.

[97] D. Ritze, C. Meilicke, O. Šváb Zamazal, and H. Stuckenschmidt. A Pattern-based Ontology Matching Approach for Detecting Complex Correspondences. In *OM-09*, volume 551 of *CEUR*, 2009.

[98] M. Schorlemmer, A. Smaill, K.-U. Kühnberger, O. Kutz, S. Colton, E. Cambouropoulos, and A. Pease. COINVENT: Concept Invention Theory - Ontologies and Semantic Web Technologies for Concept Invention. In *5th European Semantic Web Conference on (ESWC 2014), EU Project Networking Session*, May 27, Crete, Greece, 2014.

[99] Marco Schorlemmer, Alan Smaill, Kai-Uwe Kühnberger, Oliver Kutz, Simon Colton, Emilios Cambouropoulos, and Alison Pease. COINVENT: Towards a Computational Concept Invention Theory. In *Proc. of the 5th International Conference on Computational Creativity*, Ljubljana, Slovenia, June 10–13 2014.

[100] S. Schulz. E – A Brainiac Theorem Prover. *Journal of AI Communications*, 15(2/3):111–126, 2002.

[101] A. Schwering, U. Krumnack, K.-U. Kühnberger, and H. Gust. Syntactic Principles of Heuristic-Driven Theory Projection. *Cognitive Systems Research*, 10(3):251–269, 2009.

[102] Angela Schwering, Kai-Uwe Kühnberger, Ulf Krumnack, Helmar Gust, Tonio Wandmacher, Bipin Indurkhya, and Amitash Ojha. A Computational Model for Visual Metaphors: Interpreting Creative Visual Advertisements. In *International Conference on Agents and Artificial Intelligence (ICAART-09)*, pages 339–344, 2009.

[103] Geoff Sutcliffe. The TPTP world – infrastructure for automated reasoning. In Edmund M. Clarke and Andrei Voronkov, editors, *LPAR (Dakar)*, volume 6355 of *LNCS*, pages 1–12. Springer, 2010.

[104] M. Turner. The Way We Imagine. In Ilona Roth, editor, *Imaginative Minds - Proc. of the British Academy*, pages 213–236. OUP, Oxford, 2007.

[105] Mark Turner. *The Origin of Ideas: Blending, Creativity, and the Human Spark*. Oxford University Press, 2014.

[106] Morgot van Mulken, Rob le Pair, and Charles Forceville. The impact of perceived complexity, deviation and comprehension on the appreciation of visual metaphor in advertising across three European countries. *Journal of Pragmatics*, 42:3418–3430, 2010.

[107] T. Veale. Creativity as pastiche: A computational treatment of metaphoric blends, with special reference to cinematic “borrowing”. In *Proc. of Mind II: Computational Models of Creative Cognition*, 1997.

[108] Tony Veale. From Conceptual Mash-ups to “Bad-Ass” Blends: A Robust Computational Model of Conceptual Blending. In *Proc. of the 2012 International Conference on Computational Creativity*, Dublin, Ireland, 2012.

[109] Tony Veale and Diarmuid O'Donoghue. Computation and Blending. *Cognitive Linguistics*, 11(3/4):253–281, 2001.

[110] Brian Walshe. Identifying complex semantic matches. In *The Semantic Web: Research and Applications*, pages 849–853. Springer, 2012.

[111] Zhe Wang, Kewen Wang, Rodney W. Topor, and Jeff Z. Pan. Forgetting for knowledge bases in dl-lite. *Ann. Math. Artif. Intell.*, 58(1–2):117–151, 2010.

[112] C. Weidenbach, U. Brahm, T. Hillenbrand, E. Keen, C. Theobalt, and D. Topic. SPASS version 2.0. In A. Voronkov, editor, *Automated Deduction – CADE-18*, LNCS 2392, pages 275–279, 2002.

[113] Jürgen Zimmer and Serge Autexier. The MathServe System for Semantic Web Reasoning Services. In U. Furbach and N. Shankar, editors, *3rd IJCAR*, LNCS 4130. Springer, 2006.

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