Ontology mapping: the state of the art

YANNIS KALFOGLOU[1] and MARCO SCHORLEMMER[2,3]

[1]*Advanced Knowledge Technologies, Department of Electronics and Computer Science, University of Southampton, UK; e-mail: y.kalfoglou@ecs.soton.ac.uk*
[2]*Advanced Knowledge Technologies, Centre for Intelligent Systems and their Applications, School of Informatics, The University of Edinburgh, UK; e-mail: marco@inf.ed.ac.uk*
[3]*Escola Superior de Tecnologies d'Informació i Comunicació, Universitat Internacional de Catalunya, Spain*

Abstract

Ontology mapping is seen as a solution provider in today's landscape of ontology research. As the number of ontologies that are made publicly available and accessible on the Web increases steadily, so does the need for applications to use them. A single ontology is no longer enough to support the tasks envisaged by a distributed environment like the Semantic Web. Multiple ontologies need to be accessed from several applications. Mapping could provide a common layer from which several ontologies could be accessed and hence could exchange information in semantically sound manners. Developing such mappings has been the focus of a variety of works originating from diverse communities over a number of years. In this article we comprehensively review and present these works. We also provide insights on the pragmatics of ontology mapping and elaborate on a theoretical approach for defining ontology mapping.

Keywords: Ontologies, ontology mapping, ontology merging, ontology integration, ontology alignment.

1 Introduction

Nowadays, the interested practitioner[1] in ontology mapping, is often faced with a knotty problem: there is an enormous amount of diverse work originating from different communities who claim some sort of relevance to ontology mapping. For example, terms and works encountered in the literature which claimed to be relevant include: *alignment, merging, articulation, fusion, integration, morphism*, and so on. Given this diversity, it is difficult to identify the problem areas and comprehend solutions provided. Part of the problem is the lack of a comprehensive survey, a standard terminology, hidden assumptions or undisclosed technical details, and the dearth of evaluation metrics.

This article aims to fill-in some of these gaps, primarily the first one: lack of a comprehensive survey. We scrutinised the literature and critically reviewed works originating from a variety of fields to provide a comprehensive overview of ontology mapping work to date. We also worked on the theoretical grounds for defining ontology mapping, which could act as the glue for better understanding similarities and pinpointing differences in the works reported.

[1]We use a broad definition of the term, and when we refer to practitioners throughout the article, these could range from academics—either students or members of staff—to industrialists—from software engineers to knowledge engineers—or simply interested end users.

The overall goal of this paper is not only to give readers a comprehensive overview of the ontology mapping works to date, but also to provide necessary insights for the practical understanding of the issues involved. As such, we have been critiquing while reporting these works, and not just been descriptive. At the same time though, we objectively review the works with emphasis given on a practitioner's interests, and try to provide answers to the following questions:

- What are the lessons learnt from this work?
- How easily can this work be replicated in similar domains?

Outline. We start by elaborating on the survey style we adopt in Section 2, where we also provide a theoretical definition of the term 'ontology mapping'. As this article is mostly a descriptive exercise and not a normative one, we do not claim that this is the only one. We include it here for the sake of comprehending the issues involved in mapping, especially when these originate from different communities. We continue with the main section of the article, the actual survey, Section 3, which also includes illustrative examples of ontology mapping usage. In Section 5 we discuss the pragmatics for ontology mapping, and we conclude the article in Section 6.

2 Survey style

Current practice in ontology mapping entails a large number of fields ranging from machine learning, concept lattices, and formal theories to heuristics, database schema, and linguistics. Their applications also range significantly, from academic prototypes to large scale industrial applications. Therefore, it was impractical and overwhelming to conduct a marketing-style survey with questionnaires, standardised categories, and multiple participants. In fact, there is an acknowledged dearth of standards and metrics in knowledge engineering which would have made our job even more difficult. The few that are defined, like for example the *CommonKADS* methodology (Schreiber et al. 2000), or the recent *OntoWeb* EU thematic network (OntoWeb 2002), are not fully endorsed by recognised bodies, neither do they specifically mention ontology mapping works.[2]

We therefore scrutinised the literature to identify works that target ontology mapping, or at least are somehow related to it. We deliberately widened the scope of our survey and included works that target integration and merging, originate from other communities (for example, database schemata), and works that are purely theoretical. We aim to give a broad picture of ontology mapping practice today and hence do not restrict our survey to those works that are 'labelled' as ontology mapping tools. As we will show in the sequel, there are many angles at which the problem can be viewed from, and we aim to highlight this diversity. Despite the fact that we quote original works, we also provide critiquing, whenever appropriate, in order to maintain a uniform style, to provide comparative indicators, and to focus on a broader picture of ontology mapping. As such, the reader should expect a certain degree of subjectivity.

[2]The *OntoWeb* deliverable is probably the report which is closest to an ontology mapping survey.

However, this has been kept to a minimum, and we gathered most of our personal judgement in Section 5, where we elaborate on issues that we found important for the interested practitioner.

We should also note what this survey is not about: It is not a comparative review, we do not compare the works reported under any specific framework, simply because such a framework does not exist! Although efforts have been made to provide such a framework (see, for example, (OntoWeb 2002), pp. 35–51), these are far from being standards. Experience from software engineering shows that developing and agreeing on these standards is a lengthy process which takes many years and extensive resources (Moore 1998). This survey also does not make any attempt to provide standardised definitions and scope of ontology mapping. The origin and diversity of works reported makes this task arguably impossible. Only a theoretical approach could help us understand the differences and commonalities. In the next section, we elaborate on such an approach.

2.a Defining ontology mapping

We shall adopt an algebraic approach and present ontologies as logical theories. An ontology is then a pair $O = (S, A)$, where S is the *(ontological) signature*—describing the vocabulary—and A is a set of *(ontological) axioms*—specifying the intended interpretation of the vocabulary in some domain of discourse.

Typically, an ontological signature will be modelled by some mathematical structure. For instance, it could consist of a hierarchy of concept or class symbols modelled as a partial ordered set (poset), together with a set of relations symbols whose arguments are defined over the concepts of the concept hierarchy. The relations themselves might also be structured into a poset. For the purposes of this survey we shall not commit to any particular definition of ontological signature; we refer to the definitions of 'ontology', 'core ontology', or 'ontology signature' in (Kalfoglou and Schorlemmer 2002; Stumme and Maedche 2001; Bench-Capon and Malcolm 1999), respectively, for some examples of what we consider here an ontological signature. In addition to the signature specification, ontological axioms are usually restricted to a particular sort or class of axioms, depending on the kind of ontology.

Ontological signature morphisms. We understand ontology mapping as the task of relating the vocabulary of two ontologies that share the same domain of discourse in such a way that the mathematical structure of ontological signatures and their intended interpretations, as specified by the ontological axioms, are respected. Structure-preserving mappings between mathematical structures are called morphisms; for instance, a function f between two posets that preserves the partial order ($a \leqslant b$ implies $f(a) \leqslant f(b)$) is a morphism of posets. Hence we shall characterise ontology mappings as morphisms of ontological signatures as follows.

A *total ontology mapping* from $O_1 = (S_1, A_1)$ to $O_2 = (S_2, A_2)$ is a morphism $f : S_1 \to S_2$ of ontological signatures, such that, $A_2 \models f(A_1)$, i.e., all interpretations that satisfy O_2's axioms also satisfy O_1's translated axioms. This makes an ontology mapping a *theory morphism* as it is usually defined in the field of algebraic specification (see, for instance, (Meseguer 1989)).

In order to accommodate a weaker notion of ontology mapping we will say that there is a *partial ontology mapping* form $O_1 = (S_1, A_1)$ to $O_2 = (S_2, A_2)$ if there exists a sub-ontology $O'_1 = (S'_1, A'_1)$ ($S'_1 \subseteq S_1$ and $A'_1 \subseteq A_1$) such that there is a total mapping from O'_1 to O_2.

Populated ontologies. Central to several approaches to ontology mapping is the concept of a *populated ontology*. In this case, classes of an ontological signature come equipped with their respective instances. A populated ontology can be characterised by augmenting the signature with a classification relation that defines the classification of instances to the concept symbols in the signature. This brings forth issues about the *correctness* of populated ontologies, namely if the classification of instances respects the structure of the ontological signature. See (Kalfoglou and Schorlemmer 2002) for a use of populated ontologies in the definition of ontology mapping.

Taking into account the population of ontologies when establishing the mapping between ontologies may be useful for relating concepts according to the meaning and use that these concepts are given by particular communities. This idea is theoretically described in (Kent 2000) and (Schorlemmer 2002), for instance, and is fundamental to the information-flow based approaches described in Section 3.f.2.

Ontology morphisms. So far, we have defined ontology mapping only in terms of morphisms of ontological signatures, i.e., by determining which concept and relation symbols of one ontology are mapped to concept and relation symbols of the other. A more ambitious and practically necessary approach would be to take into account how particular ontological axioms are mapped as well. Formally, this would require ontology mappings to be defined in terms of morphisms of ontologies, i.e., signature + axioms, instead of morphisms of signatures only.

Most works on ontology mapping reported here adopt the more restrictive view of ontology mapping as signature morphism. Nevertheless, some of them consider the alignment of logical sentences, and not of signature symbols only (Calvanese et al. 2001b; Madhavan et al. 2002). Thus, we will use the term 'ontology mapping' for mappings as ontological signature morphisms as well as mappings as ontology morphisms.

Ontology alignment, articulation and merging. Ontology mapping only constitutes a fragment of a more ambitious task concerning the alignment, articulation and merging of ontologies. Here we want to clarify our understanding of these concepts within the above theoretical picture. An ontology mapping is a morphism, which usually will consist of a collection of functions assigning the symbols used in one vocabulary to the symbols of the other. But two ontologies may be related in a more general fashion, namely by means of *relations* instead of functions. Hence, we will call *ontology alignment* the task of establishing a collection of binary relations between the vocabularies of two ontologies. Since a binary relation can itself be decomposed into a pair of total functions from a common intermediate source, we may describe the alignment of two ontologies O_1 and O_2 by means of a pair of ontology mappings from an intermediate source ontology O_0 (see Figure 1). We shall call the intermediate ontology O_0,

together with its mappings, the *articulation of two ontologies*. For an example of ontology articulation see (Maedche and Staab 2000; Madhavan et al. 2002; Compatangelo and Meisel 2002).

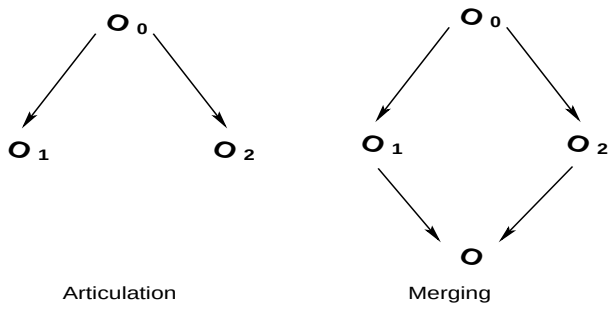

Figure 1: Diagrammatic views of articulation and merging of two ontologies.

Finally, an articulation allows for defining a way in which the fusion or merging of ontologies has to be carried out. The intuitive idea is to construct the minimal *union* of vocabularies S_1 and S_2 and axioms A_1 and A_2 that respects the articulation, i.e., that is defined *modulo* the articulation (see Figure 1). This corresponds to the mathematical *pushout* construct, and is exploited, for instance, in the frameworks described in (Bench-Capon and Malcolm 1999; Kent 2000; Schorlemmer 2002). Again, this 'strong' notion of merging can be relaxed by taking the articulation of two sub-ontologies of O_1 and O_2 respectively, and defining the merged ontology O according to their articulation.

A word on translation and integration. Translation is used by different authors to describe two different things. First, there is the translation between formal languages, for example from Ontolingua to Prolog. This changes the syntactic structure of axioms, but not the vocabulary. This is not of our concern in this survey. Second, there is the actual translation of the vocabulary. This is intimately linked to the issue of ontology mapping. Actually, the difference between mapping and translation is that the former denotes the process of defining a collection of functions that specify which concepts and relations correspond to which other concepts and relation, while the latter is the application of the mapping functions to actually translate the sentences that use the one ontology into the other. This presupposes that the ontologies share the domain in which the respective vocabularies are interpreted. Under integration, on the other hand, we regard the composition of ontologies to build new ones, but whose respective vocabulary are usually not interpreted in the same domain of discourse.

2.b Categorisation of works

We selected the following categories as the most appropriate ones to classify the 35 works we report in this article. These categories are not by any means standard, but merely identify the type of work being reported. In addition, some of them belong to more than one category. In such a case, we include the

cited work in both categories with emphasis given on its primary category. The categories are as follows:

- *Frameworks:* These are mostly a combination of tools, they provide a methodological approach to mapping, and some of them are also based on theoretical work.

- *Methods and tools:* Here we report tools, either stand-alone or embedded in ontology development environments, and methods used in ontology mapping.

- *Translators;* Although these works might be seen as peripheral to ontology mapping, they are mostly used at the early phases of ontology mapping.

- *Mediators:* Likewise, mediators could be seen as peripheral, but they provide some useful insights on algorithmic issues for mapping programs.

- *Techniques:* This is similar to *methods and tools*, but not so elaborated or directly connected with mapping.

- *Experience reports:* We found it useful to include in our survey reports on doing large-scale ontology mapping, as it provides a first-hand experience on issues of scalability and of resources involved.

- *Theoretical frameworks:* This is probably, the most interesting category. We argue that a lot of theoretical work has not been exploited yet by ontology mapping practitioners. This category aims to highlight these works.

- *Surveys:* This is similar to *experience reports* but they are more comparative in style.

- *Examples:* This is our last category and the most illustrative one. It aims to show the diversity of applications of ontology mapping and the variety of case studies that have benefitted from it. We quote examples from a selection of original works which have been reported in previous categories.

3 Ontology mapping survey

3.a Frameworks

We selected the following frameworks from the literature: **Fernández-Breis and Martínez-Béjar**'s (2002) cooperative framework for ontology integration, the **MAFRA** framework for distributed ontologies in the Semantic Web (Maedche and Staab 2000), the **OISs** framework for ontology integration systems (Calvanese et al. 2001b), **Madhavan and colleagues**' framework and language for ontology mapping (Madhavan et al. 2002), the **OntoMapO** framework for integrating upper level ontologies (Kiryakov et al. 2001), and the **IFF** framework for ontology sharing (Kent 2000).

Fernández-Breis and Martínez-Béjar (2002) describe a cooperative framework for integrating ontologies. In particular, they present a system that

> ... could serve as a framework for cooperatively built, integration-derived (i.e., global) ontologies.

Their system is aimed towards ontology integration and is intended for use by normal and expert users. The former are seeking information and provide specific information with regard to their concepts, whereas the latter are integration-derived ontology constructors, in the authors jargon. As the normal users enter information regarding the concepts' attributes, taxonomic relation, and associated terms in the the system, the expert users process this information, and the system helps them to derive the integrated ontology. The algorithm that supports this integration is based on taxonomic features and on detection of synonymous concepts in the two ontologies. It also takes into account the attributes of concepts, and the authors have defined a typology of equality criteria for concepts. For example, when the name-based equality criterion is called upon, both concepts must have the same attributes. An example of its use is included in Section 4.

Maedche and Staab (2000) devised a mapping framework for distributed ontologies in the Semantic Web. The authors argue that mapping existing ontologies will be easier than creating a common ontology, because a smaller community is involved in the process. **MAFRA** is part of a multi-ontology system, and it aims to automatically detect similarities of entities contained in two different department ontologies. Maedche and Staab (2000) argue:

> Both ontologies must be normalized to a uniform representation, in our case RDF(S), thus eliminating syntax differences and making semantic differences between the source and the target ontology more apparent.

This normalisation process is done by a tool, LIFT, which brings DTDs, XML-Schema and relational databases to the structural level of the ontology. Another interesting contribution of the MAFRA framework is the definition of a *semantic bridge*. This is a module that establishes correspondences between entities from the source and target ontology based on similarities found between them. All the information regarding the mapping process is accumulated, and populate an ontology of mapping constructs, the so called *Semantic Bridge Ontology (SBO)*. The SBO is in DAML+OIL format, and the authors argue:

> One of the goals in specifying the semantic bridge ontology was to maintain and exploit the existent constructs and minimize extra constructs, which could maximize as much as possible the acceptance and understanding by general semantic web tools.

In Section 4 we give a brief mapping example taken directly from (Maedche and Staab 2000).

Calvanese and colleagues (2001b) proposed a formal framework for Ontology Integration Systems—**OISs**. The framework provides the basis for ontology integration, which is the main focus of their work. Their view of a formal framework is close to that of Kent (see Section 3.f.2), and it

> ... deals with a situation where we have various local ontologies, developed independently from each other, and we are required to build an integrated, global ontology as a mean for extracting information from the local ones.

Ontologies in their framework are expressed as Description Logic (DL) knowledge bases, and mappings between ontologies are expressed through suitable mechanisms based on queries. Although the framework does not make explicit any of the mechanisms proposed, they are employing the notion of queries, which

> ... allow for mapping a concept in one ontology into a view, i.e., a query, over the other ontologies, which acquires the relevant information by navigating and aggregating several concepts.

They propose two approaches to realise this query/view based mapping: global-centric and local-centric. The global-centric approach is an adaptation of most data integration systems. In such systems, the authors continue, sources are databases, the global ontology is actually a database schema, and the mapping is specified by associating to each relation in the global schema one relational query over the source relations. In contrast, the local-centric approach requires reformulation of the query in terms of the queries to the local sources. The authors provide examples of using both approaches in (Calvanese et al. 2001a) and we recapitulate some of them in Section 4.

Madhavan and colleagues (2002) developed a framework and propose a language for ontology mapping. Their framework enables mapping between models in different representation languages without first translating the models into a common language, the authors claim. The framework uses a *helper model* when it is not possible to map directly between a pair of models, and it also enables representing mappings that are either incomplete or involve loose information. The models represented in their framework are representations of a domain in a formal language, and the mapping between models consists of a set of relationships between expressions over the given models. The expression language used in a mapping varies depending on the languages of the models being mapped. The authors claim that mapping formulae in their language can be fairly expressive, which makes it possible to represent complex relationships between models. They applied their framework in an example case with relational database models. They also define a typology of mapping properties: query answerability, mapping inference, and mapping composition. The authors argue:

> A mapping between two models rarely maps all the concepts in one model to all concepts in the other. Instead, mappings typically loose some information and can be partial or incomplete.

Question answerability is a proposed formalisation of this property. Mapping inference provides a tool for determining types of mappings, namely equivalent mappings and minimal mappings; and mapping composition enables to map between models that are related by intermediate models. Examples of their framework are given in Section 4.

Kiryakov and colleagues (2001) developed a framework for accessing and integrating upper level ontologies. They provide a service that allows a user to import linguistic ontologies onto a Web server, which will then be mapped onto other ontologies. The authors argue for

> ... a uniform representation of the ontologies and the mappings between them, a relatively simple meta-ontology (*OntoMapO*) of property types and relation-types should be defined.

Apart from the **OntoMapO** primitives and design style, which is peripheral to our survey, the authors elaborate on a set of primitives that *OntoMapO* offers for mapping. There are two sets of primitives defined, *InterOntologyRel* and *IntraOntologyRel*, each of which has a number of relations that aim to capture the correspondence of concepts originating from different ontologies (i.e., equivalent, more-specific, meta-concept). A typology of these relations is given in the form of a hierarchy and the authors claim that an initial prototype has been used to map parts of the *CyC* ontology to *EuroWordNet*.

Kent (2000) proposed a framework for ontological structures to support ontology sharing. It is based on the Barwise-Seligman theory of information flow (Barwise and Seligman 1997). Kent argues that **IFF** represents the dynamism and stability of knowledge. The former refers to instance collections, their classification relations, and links between ontologies specified by ontological extension and synonymy (type equivalence); it is formalised with Barwise-Seligman's *local logics* and their structure-preserving transformations—logic infomorphisms. Stability refers to concept/relation symbols and to constraints specified within ontologies; it is formalised with Barwise-Seligman's *regular theories* and their structure-preserving transformations—theory interpretations. IFF represents ontologies as logics; and ontology sharing as a specifiable ontology extension hierarchy. An ontology, Kent continues, has a classification relation between instances and concept/relation symbols, and also has a set of constraints modelling the ontology's semantics. In Kent's proposed framework, a community ontology is the basic unit of ontology sharing; community ontologies share terminology and constraints through a common generic ontology that each extends, and these constraints are consensual agreements within those communities. Constraints in generic ontologies are also consensual agreements but across communities. We further examine Kent's work in section 3.f.2, where we include a discussion on theoretical frameworks.

3.b Methods and tools

In this section we report on the **FCA-Merge** method for ontology merging (Stumme and Maedche 2001), the **IF-Map** method for ontology mapping (Kalfoglou and Schorlemmer 2002), the **SMART**, **PROMPT** and **PROMPTDIFF** tools for the *Protégé* ontology development environment from Noy and Musen, the **Chimeara** tool (McGuinness et al. 2000), the **GLUE** (Doan et al. 2002) and **CAIMAN** (Lacher and Groh 2001) systems, both of which use machine learning, the **ITTalks** web-based system (Prasad et al. 2002), the **ONION** system for resolving heterogeneity in ontologies (Mitra and Wiederhold 2002), and **ConcepTool** for entity-relationship models (Compatangelo and Meisel 2002).

Stumme and Maedche (2001) presented the **FCA-Merge** method for ontology merging. It is based on Ganter and Wille's work on Formal Concept Analysis (Ganter and Wille 1999) and lattice exploration. The authors incorporate natural language techniques in FCA-Merge to derive a lattice of concepts. The lattice is then explored manually by a knowledge engineer who builds the merged ontology with semi-automatic guidance from FCA-Merge. In particular, FCA-Merge works as follows: the input to the method is a set of documents from which concepts and the ontologies to be merged are extracted. These documents should be representative of the domain at question and should be related

to the ontologies. They also have to cover all concepts from both ontologies as well as separating them well enough. These strong assumptions have to be met in order to obtain good results from FCA-Merge. As this method relies heavily on the availability of classified instances in the ontologies to be merged, the authors argue that this will not be the case in most ontologies, the authors opt to extract instances from documents:

> The extraction of instances from text documents circumvents the problem that in most applications there are no objects which are simultaneously instances of the source ontologies, and which could be used as a basis for identifying similar concepts.

In this respect, the first step of FCA-Merge could be viewed as an ontology population mechanism. This initial step could be skipped, though, if there are shared classified instances in both ontologies. Once the instances are extracted, and the concept lattice is derived, Stumme and Maedche use Formal Concept Analysis techniques to generate the formal context for each ontology. They use lexical analysis to perform, among other things, retrieval of domain-specific information:

> It associates single words or composite expressions with a concept from the ontology if a corresponding entry in the domain-specific part of the lexicon exists.

Using this lexical analysis the authors associate complex expressions, like `Hotel Schwarzer Adler` with concept `Hotel`. Next, the two formal contexts are merged to generate a pruned concept lattice. This step involves disambiguation (since the two contexts may contain the same concepts) by means of indexing. The computation of the pruned concept lattice is done by an algorithm, TITANIC, which computes formal contexts via their key sets (or minimal generators). In terms of Formal Concept Analysis, the extents of concepts are not computed (these are the documents that they originate from, and are not needed for generating the merged ontology, the authors say), only the intents are taken into account (sets of concepts from the source ontologies). Finally, Stumme and Maedche do not compute the whole concept lattice,

> ... as it would provide too many too specific concepts. We restrict the computation to those formal concepts which are above at least one formal concept generated by an (ontology) concept of the source ontologies.

Having the pruned concept lattice generated, FCA-Merge enters its last phase, the non-automatic construction of the merged ontology, with human interaction. This construction is semi-automatic as it requires background knowledge about the domain. The engineer has to resolve possible conflicts and duplicates, but there is automatic support from FCA-Merge in terms of a query/answering mechanism, which aims to guide and focus the engineer's attention on specific parts of the construction process. A number of heuristics are incorporated in this phase (like using the key sets of concepts for evidence of class membership), and the `is_a` lattice is derived automatically.

Kalfoglou and Schorlemmer (2002) developed an automatic method for ontology mapping, **IF-Map**, based on the Barwise-Seligman theory of information

flow (Barwise and Seligman 1997). Their method draws on the proven theoretical ground of Barwise and Seligman's channel theory, and provides a systematic and mechanised way for deploying it on a distributed environment to perform ontology mapping among a variety of different ontologies. In Figure 2 we illustrate IF-Map's underpinning framework for establishing mappings between ontologies. These mappings are formalised in terms of *logic infomorphisms*. We elaborate on these in Section 3.f.2.

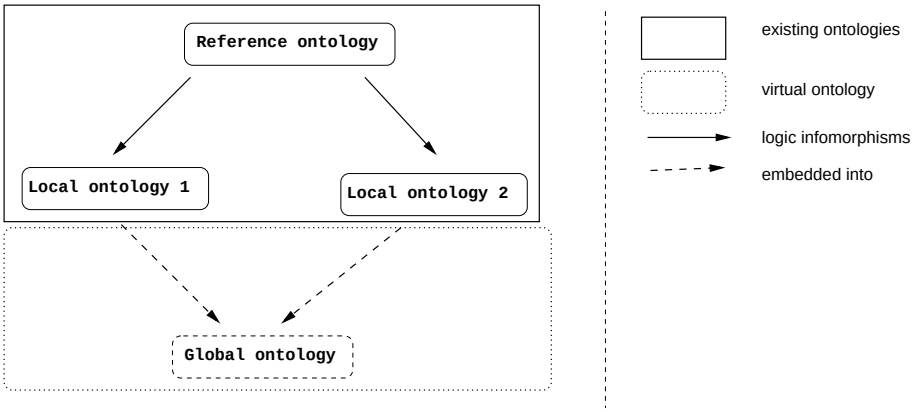

Figure 2: IF-Map scenario for ontology mapping.

Figure 2 clearly resembles Kent's proposed two-step process for ontology sharing (see (Kent 2000) and Section 3.f.2), but it has differences in its implementation. The solid rectangular line surrounding `Reference ontology`, `Local ontology 1` and `Local ontology 2` denotes the existing ontologies. We assume that `Local ontology 1` and `Local ontology 2` are ontologies used by different communities and populated with their instances, while `Reference ontology` is an agreed understanding that favours the sharing of knowledge, and is not supposed to be populated. The dashed rectangular line surrounding `Global ontology` denotes an ontology that does not exist yet, but will be constructed 'on the fly' for the purpose of merging. This is similar to Kent's *virtual ontology of community connections* (Kent 2000). The solid arrow lines linking `Reference ontology` with `Local ontology 1` and `Local ontology 2` denote information flowing between these ontologies and are formalised as *logic infomorphisms*. The dashed arrow lines denote the embedding from `Local ontology 1` and `Local ontology 2` into `Global ontology`. The latter is the sum of the local ontologies *modulo* `Reference ontology` and the generated *logic infomorphisms*.

In Figure 3 we illustrate the process of IF-Map. The authors built a stepwise process that consists of four major steps: (a) ontology harvesting, (b) translation, (c) infomorphism generation, and (d) display of results. In the ontology harvesting step, ontology acquisition is performed. They apply a variety of methods: using existing ontologies, downloading them from ontology libraries (for example, from the Ontolingua (Farquhar et al. 1997) or WebOnto (Domingue 1998) servers), editing them in ontology editors (for example, in Protégé (Grosso et al. 1999)), or harvesting them from the Web. This versatile ontology acquisition step results in a variety of ontology language formats, rang-

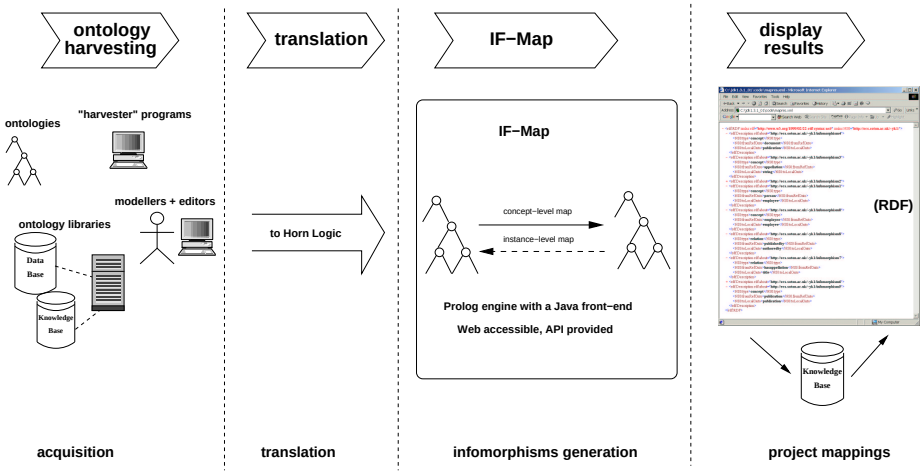

Figure 3: The IF-Map architecture.

ing from KIF (Genesereth and Fikes 1992) and Ontolingua to OCML (Motta 1999), RDF (Lassila and Swick 1999), Prolog, and native Protégé knowledge bases. This introduces the second step in their process, that of translation. The authors argue:

> As we have declaratively specified the IF-Map method in Horn logic and execute it with the aim of a Prolog engine, we partially translate the above formats to Prolog clauses.

Although the translation step is automatic, the authors comment:

> We found it practical to write our own translators. We did that to have a partial translation, customised for the purposes of ontology mapping. Furthermore, as it has been reported in a large-scale experiment with publicly available translators (Corrêa da Silva et al. 2002), the Prolog code produced is not elegant or even executable.

The next step in their process is the main mapping mechanism—the IF-Map method. This step finds *logic infomorphisms*, if any, between the two ontologies under examination and displays them in RDF format. The authors provide a Java front-end to the Prolog-written IF-Map program so that it can be accessed from the Web, and they are in the process of writing a Java API to enable external calls to it from other systems. Finally, they also store the results in a knowledge base for future reference and maintenance reasons.

Noy and Musen have developed a series of tools over the past three years for performing ontology mapping, alignment and versioning. These tools are **SMART** (Noy and Musen 1999), **PROMPT** (Noy and Musen 2000), and **PROMPTDIFF** (Noy and Musen 2002). They are all available as a plug-in for the open-source ontology editor, Protégé-2000 (Grosso et al. 1999). The tools use linguistic similarity matches between concepts for initiating the merging or alignment process, and then use the underlying ontological structures of the Protégé-2000 environment (classes, slots, facets) to inform a set of heuristics

for identifying further matches between the ontologies. The authors distinguish in their work between the notions of merging and alignment, where merging is defined as

> ... the creation of a single coherent ontology and alignment as establishing links between [ontologies] and allowing the aligned ontologies to reuse information from one another.

The SMART tool is an algorithm that

> ... goes beyond class name matches and looks for linguistically similar class names, studies the structure of relations in the vicinity of recently merged concepts, and matches slot names and slot value types...

the authors describe. Some of the tasks for performing merging or alignment, like the initial linguistic similarity matches, can be outsourced and plugged into the PROMPT system by virtue of Protégé-2000's open-source architecture. PROMPT is a (semi-)automatic tool and provides guidance for the engineer throughout the steps performed during merging or alignment:

> Where an automatic decision is not possible, the algorithm guides the user to the places in the ontology where his intervention is necessary, suggests possible actions, and determines the conflicts in the ontology and proposes solutions for these conflicts.

Their latest tool, PROMPTDIFF, is an algorithm which integrates different heuristic matchers for comparing ontology versions. The authors combine these matchers in a fixed-point manner, using the results of one matcher as input for others until the matcher produces no more changes. PROMPTDIFF addresses structure-based comparison of ontologies as its comparisons are based on the ontology structure and not their text serialisation, the authors argue. Their algorithm works on two versions of the same ontology and is based on the empirical evidence that a large fraction of frames remains unchanged and that, if two frames have the same type and have the same or very similar name, one is almost certainly an image of the other. All Protégé-specific tools from Noy and Musen have been empirically evaluated in a number of experiments using the Protégé-2000 ontology editing environment. We present examples of them in Section 4.

McGuinness and colleagues (2000) developed a similar tool for the Ontolingua editor. As in PROMPT, **Chimaera**, is an interactive tool, and the engineer is in charge of making decisions that will affect the merging process. *Chimaera* analyses the ontologies to be merged, and if linguistic matches are found, the merge is done automatically, otherwise the user is prompted for further action. When comparing it with PROMPT, these are quite similar in that they are embedded in ontology editing environments, but they differ in the suggestions they make to their users with regard to the merging steps.

Doan and colleagues (2002) developed a system, **GLUE**, which employs machine learning techniques to find mappings. Given two ontologies, for each concept in one ontology, GLUE finds the most similar concept in the other ontology using probabilistic definitions of several practical similarity measures. The authors claim that this is their difference when comparing their work with

other machine learning approaches, where only a single similarity measure is used. In addition to this, GLUE also

> ... uses multiple learning strategies, each of which exploits a different type of information either in the data instances or in the taxonomic structure of the ontologies...

the authors continue. The similarity measures they employ is the joint probability distribution of the concepts involved, so

> ... instead of committing to a particular definition of similarity, GLUE calculates the joint distribution of the concepts, and lets the application use the joint distribution to compute any suitable similarity measure.

GLUE uses a multi-learning strategy, the authors continue, because there are many different types of information a learner can glean from the training instances in order to make predictions. It can exploit the frequencies of words in the text value of instances, the instance names, the value formats, or the characteristics of value distributions. To cope with this diversity, the authors developed two learners, a content learner and a name learner. The former uses a text classification method, called Naive Bayes learning. The name learner is similar to the content learner but uses the full name of the instance instead of its content. They then developed a meta-learner that combines the predictions of the two learners. It assigns to each one of them a learner weight that indicates how much it trusts its predictions. The authors also used a technique, relaxation labelling, that assigns labels to nodes of a graph, given a set of constraints. This technique is based on the observation that the label of a node is typically influenced by the features of the node's neighbourhood in the graph. The authors applied this technique to map two ontologies' taxonomies, O_1 to O_2, by regarding concepts (nodes) in O_2 as labels, and recasting the problem as finding the best label assignment to concepts (nodes) in O_1, given all knowledge they have about the domain and the two taxonomies. That knowledge can include domain-independent constraints like 'two nodes match if nodes in their neighbourhood also match'—where neighbourhood is defined to be the children, the parents or both—as well as domain-dependent constraints like 'if node Y is a descendant of node X, and Y matches professor, then it is unlikely that X matches assistant-professor'. The system has been empirically evaluated with mapping two university courses catalogues.

Lacher and Groh (2001) present **CAIMAN**, another system which uses machine-learning for ontology mapping. The authors elaborate on a scenario where members of a community would like to keep their own perspective on a community repository. They continue by arguing that

> ... each member in a community of interest organizes her documents according to her own categorization scheme (ontology).

This rather weak account of an ontology justifies, to a certain extent, the use of a user's bookmark folder as a 'personal' ontology. The mapping task is then to align this ontology with the directory structure of *CiteSeer*[3] (also known as *ResearchIndex*). The use of more formal community ontologies is not supported by the authors, who argue:

[3] Accessible at `citeseer.nj.nec.com`.

> Information has to be indexed or categorized in a way that the user can understand and accepts... [This] could be achieved by enforcing a standard community ontology, by which all knowledge in the community is organized. However, due to loose coupling of members in a Community of Interest, this will not be possible.

Their mapping mechanism uses machine learning techniques for text classification, it measures the probability that two concepts are corresponding. For each concept node in the 'personal' ontology, a corresponding node in the community ontology is identified. It is also assumed that repositories both on the user and on the community side may store the actual documents, as well as links to the physical locations of the documents. CAIMAN is then offering two services to its users: *document publication*, which publishes documents that a user has newly assigned to one of the concept class to the corresponding community concept class, and *retrieval of related documents*, which delivers newly added documents from the community repository to the user.

Prasad and colleagues (2002) presented a mapping mechanism which uses text classification techniques as part of their web-based system for automatic notification of information technology talks (**ITTalks**). Their system

> ... combines the recently emerging semantic markup language DAML + OIL, the text-based classification technology (for similarity information collection), and Bayesian reasoning (for resolving uncertainty in similarity comparisons).

They experimented with two hierarchies: the ACM topic ontology and a small ITTalks topic ontology that organises classes of IT related talks in a way that is different from the ACM classification. The text classification technique they use generates scores between concepts in the two ontologies based on their associated exemplar documents. They then use Bayesian subsumption for subsumption checking:

> If a foreign concept is partially matched with a majority of children of a concept, then this concept is a better mapping than (and thus subsumes) its children.

An alternative algorithm for subsumption checking, the authors continue, is to take a Bayesian approach that considers the best mapping being the concept that is the lowest in the hierarchy and the posterior probability greater than 0.5.

Mitra and Wiederhold (2002) developed the ONtology compositION system (**ONION**) which provides an articulation generator for resolving heterogeneity in different ontologies. The authors argue that ontology merging is inefficient:

> A merging approach of creating an unified source is not scalable and is costly... One monolithic information source is not feasible due to unresolvable inconsistencies between them that are irrelevant to the application.

They then argue that semantic heterogeneity can be resolved by using articulation rules which express the relationship between two (or more) concepts belonging to the ontologies. Establishing such rules manually, the authors continue, is a very expensive and laborious task, on the other hand, they also claim

that full automation is not feasible due to inadequacy of today's natural language processing technology. So, they take into account relationships in defining their articulation rules, but these are limited to *subclass_of, part_of, attribute_of, instance_of*, and *value_of*. They also elaborate on a generic relation for heuristic matches:

> *Match* gives a coarse relatedness measure and it is upon to the human expert to then refine it to something more semantic, if such refinement is required by the application.

In their experiments the ontologies used were constructed manually and represent two websites of commercial airlines. The articulation rules were also established manually. However, the authors used a library of heuristic matchers to construct them. Then, a human expert, knowledgeable about the semantics of concepts in both ontologies, validates the suggested matches. Finally, they include a learning component in the system which takes advantage of users' feedback to generate better articulation in the future while articulating similar ontologies. The algorithms used for the actual mapping of concepts are based on linguistic features. We elaborate on these in Section 4.

Compatangelo and Meisel (2002) developed a system, **ConcepTool**, which adopts a description logic approach to formalise a class-centred, enhanced entity relationship model. Their work aims to facilitate knowledge sharing, and *ConcepTool* is an interactive analysis tool that guides the analyst in aligning two ontologies. These are represented as enhanced entity-relationship models augmented with a description logic reasoner. They also use linguistic and heuristic inferences to compare attributes of concepts in both models, and the analyst is prompted with relevant information to resolve conflicts between overlapping concepts. Their approach is similar to MAFRA's framework in that they both define semantic bridges: as the authors argue:

> Overlapping concepts are linked to each other by way of semantic bridges. Each bridge allows the definition of transformation rules to remove the semantic mismatches between these concepts.

The methodology followed when using *ConceptTool* consists of 6 steps: (1) analysis of both schemata to derive taxonomic links, (2) analysis of both schemata to identify overlapping entities, (3) prompt the analyst to define correspondences between overlapping entities, (4) automatic generation of entities in the articulation schema for every couple of corresponding entities, (5) prompt the analyst for defining mapping between attributes of entities, and (6) analysis of the articulated schema. In Section 4 we present an example case of *ConcepTool*'s articulation generation.

3.c Translators

We report on two translator systems: **OntoMoprh**, for symbolic knowledge (Chalupksy 2000), and **W3TRANS**, for integrating heterogeneous data (Abiteboul et al. 2002).

Chalupksy (2000) developed a translation system for symbolic knowledge—**OntoMorph**. It provides a powerful language to represent complex syntactic transformations, and it is integrated within the *PowerLoom* knowledge representation system. The author elaborates on criteria for translator systems:

> Translation needs to go well beyond syntactic transformations and occurs along many dimensions, such as expressiveness or representation languages, modelling conventions, model coverage and granularity, representation paradigms, inference system bias, etc., and any combination thereof.

OntoMorph uses syntactic rewriting via pattern-directed rewrite rules that allow the concise specification of sentence-level transformations based on pattern matching; and semantic rewriting, which modulates syntactic rewriting via (partial) semantic models and logical inference supported by *PowerLoom*. *OntoMoprh* performs knowledge morphing as opposed to translation. To quote Chalupsky:

> A common correctness criterion for translation systems is that they preserve semantics, i.e., the meaning of the source and the translation has to be the same. This is not necessarily desirable for our transformation function T, since it should be perfectly admissible to perform abstractions or semantic shifts as part of the translation. For example, one might want to map an ontology about automobiles onto an ontology of documents describing these automobiles. Since this is different from translation in the usual sense, we prefer to use the term knowledge transformation or morphing.

An interesting technique of *OntoMorph* is *semantic rewriting*. When, for example, someone is interested in conflating all subclasses of truck occurring in some ontology about vehicles into a single truck class, semantic rewriting allows for using taxonomic relationships to check whether a particular class is a subclass of truck. This is achieved through the connection of *OntoMorph* with *PowerLoom*, which accesses the knowledge base to import source sentences representing taxonomic relationships, like subset and superset assertions.

Abiteboul and colleagues (2002) elaborate on a middleware data model and on declarative rules for integrating heterogeneous data. Although their work is more akin to the database world, their techniques for integration could be useful for ontology mapping. In their data model, the authors use a structure which consists of ordered labelled trees. The authors claim:

> This simple model is general enough to capture the essence of formats we are interested in. Even though a mapping from a richer data model to this model may loose some of the original semantics, the data itself is preserved and the integration with other data models is facilitated.

They then define a language for specifying correspondence rules between data elements and bi-directional data translation. These correspondences could serve for other purposes, for example, as an aid for ontology mapping. These ideas have been implemented in a prototype system, **W3TRANS**, which uses the middleware data model and the rule language for specifying the correspondences mentioned above.

3.d Mediators

Two indicative mediator works are reported here: The **rule-based algebra** of Jannink and colleagues (1998) and the **mediation algorithms** of Campbell

and Shapiro (1998).

Jannink and colleagues (1998) developed a **rule-based algebra** for ontology clustering into contexts. They define interfaces that link the extracted contexts to the original ontologies. As changes occur in the contexts, the original ontology remains unchanged, and it is the responsibility of the interface to ensure that the context will fit coherently back into the ontology. Their work aims to encapsulate ontologies in contexts and to compose contexts. As the authors argue:

> Contexts provide guarantees about the knowledge they export, and contain the interfaces feasible over them... [They] are the primary building blocks which our algebra composes into larger structures. The ontology resulting from the mapping between two source ontologies is assumed to be consistent only within its own context.

The authors provide four types of interfaces to contexts: Schema interfaces (templates specifying the set of concepts, types and relationships in the context), source interfaces (access to the input data sources used to answer the query), rule interfaces (return the rule sets used to transform the data from the sources they conform to to the items in the schema), and owner interfaces (contain a time stamp and names of the context owners). Their rule-based algebra defines two classes of mapping primitives, formed from sequences of simpler operations. Each simple operation is in fact a logical rule, belonging to one of instance, class or exception rule. These rules are fired according to structural and lexical properties of the source data, i.e., to position and string matching techniques. We will revisit their work in Section 3.f.1 when we report on algebraic frameworks for ontology mapping.

Campbell and Shapiro (1998) devised a set of **algorithms for ontological mediation**. They define an ontological mediator as:

> An agent capable of reasoning about the ontologies of two communicating agents, or communicants, learning about what W means for S, and looking for an ontological translation (W') that means for L the same thing in the domain that W means for S.

They devised three algorithms, one for exploiting single hierarchical ontological relations (subclass/superclass), one for multiple hierarchical ontological relations (part/whole), and an algorithm that chooses the best candidate concept representing one agent's concept that the other agent believes to be equivalent with its own concept. They evaluated their work with lexical ontologies, like WordNet.

3.e Techniques

The following works use techniques that could be applied in certain phases of ontology mapping. These are *ontology projections* of Borst and colleagues (1997) in the **PhysSys** project, the **semantic values** of Sciore and colleagues (1994), and information integration techniques of Mena and colleagues (1998) in **OBSERVER**.

Borst and colleagues (1997) developed the **PhysSys** ontology set. This is a set of seven ontologies that represents the domain of system dynamics

and expresses different viewpoints of a physical system. Interdependences between these ontologies are formalised as ontology projections and included in the *PhysSys* ontology. Three kinds of projections are demonstrated in their work: *include-and-extend*, *include-and-specialise*, and *include-and-project*. The latter was used to link an ontology developed by the authors of *PhysSys* to an outsourced ontology, the *EngMath*. These projections, though, are not computed automatically but defined manually by the knowledge engineer when designing the ontologies.

Sciore and colleagues (1994) worked on a theory of **semantic values** as a unit of exchange that facilitates semantic interoperability between heterogeneous information systems. In their work, a semantic value is defined to be a piece of data together with its associated context. These can either be stored explicitly or be defined by data environments. The authors also developed an architecture which includes a context mediator, whose job is to identify and construct the semantic values being sent, to determine when the exchange is meaningful, and to convert the semantic values to the form required by the receiver. In their work, contexts are defined as metadata relating data to their properties (such as source, quality, and precision) and represented as sets: Each element of the set is an assignment of a semantic value to a property. The advocated semantic interoperability is based on using conversion functions, which convert a semantic value from one context to another. These functions are stored in conversion libraries. Their architecture also uses ontologies:

> The shared ontology component specifies terminology mappings. These mappings describe naming equivalences... so that references to attributes (e.g., exchange or company name), properties (e.g., currency), and their values (e.g., US dollar) in one information system can be translated to the equivalent names in another.

Ontologies are accessed by the context mediators to check the terminology mappings. Their prototype system has been applied to a relational database model.

Mena and colleagues (1998) developed the Ontology Based System Enhanced with Relationships for Vocabulary hEterogeneity Resolution (**OBSERVER**) in order to access heterogeneous, distributed and independently developed data repositories. Their aim was to tackle the problem of semantic information integration between domain-specific ontologies. They use interontology relationships such as synonyms, hyponyms and hypernyms defined between terms in different ontologies to assist the brokering of information across domain-specific ontologies. Their system is based on a query-expansion strategy where the user poses queries in one ontology's terms and the system tries to expand the query to other ontologies' terms. This is supported by algorithms to manage the relevance of information returned. As far as the mappings are concerned, they use the data structures underlying the domain-specific ontologies and the synonymy, hyponymy and hypernymy relations to inform linguistic matches between concepts.

3.f Theoretical frameworks

We classify the works presented here in three broad categories: *Algebraic* approaches, which comprise the works of Bench-Capon and Malcolm (1999) on

ontology morphisms, and that of Jannink and colleagues (1998) on an **ontology composition algebra**; *Information-flow-based* approaches, which include the works of Kent (2000) on the **Information Flow Framework**, that of Schorlemmer (2002) on **duality in knowledge sharing**, the **IF-Map** method of Kalfoglou and Schorlemmer (2002) based on information-flow theory and populated ontologies, the work of Priss (2001) on **Peircean sign triads**, and **FCA-Merge** (Stumme and Maedche 2001), based on formal concept analysis and lattice exploration; and *Translation* frameworks, with the formative work of Grüninger (1997) on the **TOVE project**. .

3.f.1 Algebraic approaches

Bench-Capon and Malcolm (1999) give a formalisation of ontologies and the relations between them building upon the universal-algebraic tradition, extending the concept of abstract data type (ADT) to that of ontology—specifying classes of entities with attributes that take their values from given ADTs. For that purpose they provide rigorous definitions of *data domain*, *ontology signature*, and *ontology*, and more importantly, they provide definitions of the structure-preserving transformations—morphisms—between them.

Based on this framework, they capture the relation, or mapping, between two ontologies by means of a pair of **ontology morphisms** that share the same domain (source of the morphism). The combination (or merging) of ontologies is then characterised by means of a categorical *pushout construction*, which is widely used by researchers in formal specifications for characterising the combination of separate ADTs or specification modules.

Studying the relations between ontologies by means of ontology morphisms is also central to the IF-Map methodology (Kalfoglou and Schorlemmer 2002), and it bears some resemblance to other definitions of ontology mapping based on *infomorphisms* (Barwise and Seligman 1997), as we shall see when we survey IF-based approaches to ontology mapping and merging further down in this section.

As we reported in Section 3.d, Jannink and colleagues propose an **algebra**, based on category-theoretic constructions, for extracting *contexts* from knowledge sources and combining these contexts (this algebra has been investigated further by Mitra and Wiederhold—see Section 3.b). Although no formal definition of 'context' is given, this is considered to be the unit of encapsulation for well-structured ontologies. The categorical constructions are also used in an informal way, by means of definitions of *informal categories*—the union of concept specifications and instances that represent their extensions—and informal uses of *pullbacks* and *pushouts*. Their framework allows to model the *semantic mismatch* between the source instances and the concept intension, and they give definitions for *false positives* (i.e., missing instances) and *false negatives* (i.e., exceptional instances). They argue:

> Morphisms allow translation from one specification to another when there is no semantic mismatch. Therefore, they are applicable when intensions and extension are not distinguishable, such as in mathematical structure.

On the contrary, we argue that IF-based approaches can overcome this difficulty by incorporating instances and the notions of 'missing instance' or 'exceptional

instance' into the mapping framework, and hence into the potential definitions of ontology morphism.

3.f.2 IF-based approaches

The first attempt to apply the results of recent efforts towards a mathematical theory of information and information flow in order to provide a theoretical framework for describing the mapping and merging of ontologies is probably the **Information Flow Framework (IFF)** (Kent 2000). IFF is based on channel theory (Barwise and Seligman 1997).

Kent exploits the central distinction made in channel theory between *types*—the syntactic elements, like concept and relation names, or logical sentences—and *tokens*—the semantic elements, like particular instances, or logical models—and its organisation by means of *classification tables*, in order to formally describe the stability and dynamism of conceptual knowledge organisation. He assumes two basic principles,

1. that a community with a well-defined ontology owns its collection of instances (it controls updates to the collection; it can enforce soundness; it controls access rights to the collection), and

2. that instances of separate communities are linked through the concepts of a *common generic ontology*,

and then goes on to describe a two-step process that determines the *core ontology of community connections* capturing the organisation of conceptual knowledge across communities (see Figure 4). The process starts from the assumption that the *common generic ontology* is specified as a logical theory and that the several *participating community ontologies* extend the *common generic ontology* according to theory interpretations (in its traditional sense as consequence-preserving mappings, see (Enderton 2001)), and consists of the following steps:

1. A *lifting step* from theories to logics that incorporates instances into the picture (proper instances for the community ontologies, and so called *formal instances* for the generic ontology).

2. A *fusion step* where the logics (theories + instances) of community ontologies are linked through a *core ontology of community connections*, which depends on how instances are linked through the concepts of the common generic ontology (see second principle above).

Kent's framework is purely theoretical, and no method for implementing his two-step process is given. Kent's main objective with IFF is to provide a meta-level foundation for the development of upper ontologies.

Very close in spirit and in the mathematical foundations of IFF, Schorlemmer (2002) studied the intrinsic **duality of channel-theoretic constructions**, and gave a precise formalisation to the notions of *knowledge sharing scenario* and *knowledge sharing system*. He used the categorical constructions of Chu spaces (Gupta 1994; Barr 1996; Pratt 1995) in order to precisely pin down some of the reasons why ontologies turn out to be insufficient in certain knowledge sharing scenarios (Corrêa da Silva et al. 2002). His central argument is that formal analysis of knowledge sharing and ontology mapping has to take a duality between syntactic types (concept names, logical sentences, logical sequents)

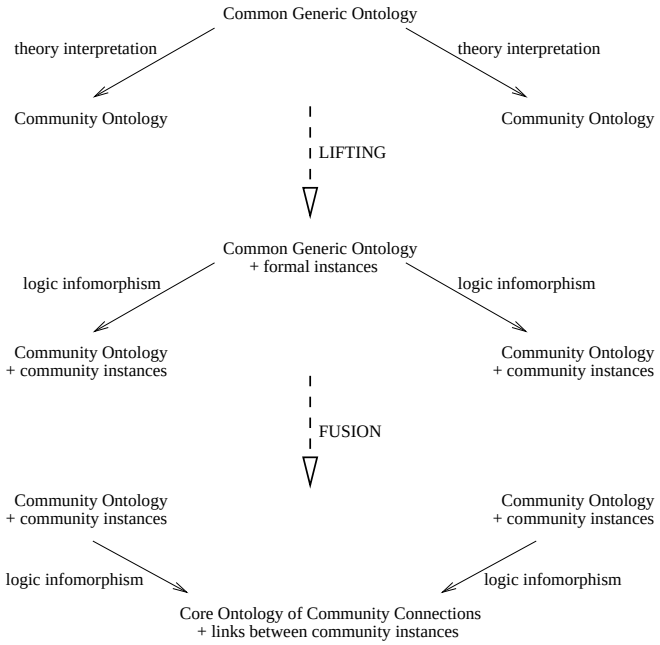

Figure 4: Kent's two-step process for conceptual knowledge organisation.

and particular situations (instances, models, semantics of inference rules) into account. Although no explicit definition of ontology mapping is given, there is an implicit one within the definition of knowledge sharing scenario, namely as a *Chu transform*.

Drawing from the theoretical ideas of Kent's IFF and Schorlemmer's analysis of duality in knowledge sharing scenarios, Kalfoglou and Schorlemmer (2002) propose the **IF-Map** methodology already discussed in Section 3.b. From the theoretical point of view, Kalfoglou and Schorlemmer also adopt an algebraic approach similar to that of Bench-Capon and Malcolm, by providing precise definitions for ontology and ontology morphism in the tradition of algebraic specification. But, based on the knowledge sharing ideas of IFF and Schorlemmer—and the role instances (tokens) play in the reliable flow of information, and hence in knowledge sharing—they give precise definitions of *populated ontologies*, and base their IF-Map methodology on ontology morphisms between populated ontologies, such that these morphisms are coherent with the channel-theoretic framework of Barwise and Seligman.

From a more philosophical perspective, Priss (2001) explores how issues arising in aligning and merging ontologies can be tackled by adopting a Peircean approach based on **sign triads**. Priss argues that the relevant issues concerning information representation and processing among natural and artificial agents are those concerning the *consensual sign triad*, i.e., the relationships between concept entities, context, and sign representations as they are consensually agreed upon for a collectivity of individuals (natural or artificial). Priss suggests that techniques from formal concept analysis (Ganter and Wille 1999) could be used to provide formal representations of context and concepts of a

consensual sign triad. A context would be a *formal context* (i.e., a classification table of objects with respect to their attributes); concepts would be nodes in a *concept lattice*. Alternatively, concepts could also be represented by conceptual graphs (Sowa 1984), Priss claims.

She also claims that the issues arising during the interaction of agents that have different ontologies, and when different representational signs have to be aligned, need to be tackled by establishing a clear separation of signs, concepts, and context, thus determining the consensual sign triads for each agent. Priss suggests that, since the shift between context could be formalised by means of infomorphisms in the Barwise-Seligman information theory, the alignment could then be established through information-flow channels between contexts.

Priss's approach to ontology mapping and merging is, from a philosophical and technical point of view, again very close to those of Kent and of Kalfoglou and Schorlemmer. Although Priss does not tackle the mathematical detail, nor does she discuss any methodology or computer implementation, hers is a first attempt to provide a

> ... semi-formal ontological foundation that facilitates an explicit representation, use and differentiation of representations, conceptual entities and contexts in applications...

based on the deep philosophical ideas concerning the nature of representation, but using modern techniques of information flow and formal concept analysis.

Although Stumme and Maedche's ontology merging methodology **FCA-Merge** (see Section 3.b) is not exactly an 'IF-based' approach, it is nevertheless closely related to these approaches by virtue that formal concept analysis (Ganter and Wille 1999) shares with channel theory the same mathematical foundations. Like in channel-theoretic approaches as those of Kent or Kalfoglou and Schorlemmer, ontologies, and in particular their concept hierarchies, are represented by tables that classify instances to concepts, called *formal contexts* in FCA. Stumme and Maedche do not discuss any formal definition of ontology mapping. They give a formal definition of *core ontology* and determine their relationship to formal concepts. The merging method and algorithm, which assumes that participating communities share the same instances (which are text documents in their particular scenario), is then based on inferring the merged concept hierarchy from the combined table—formal context—representing both ontologies and their shared instances.

3.f.3 Translation frameworks

Within the original efforts of the **TOVE Ontology Project** and the development of the Process Interchange Format (PIF) (Lee et al. 1998), Grüninger (1997) has established a formal framework for studying issues of ontology translation. He formalises several kinds of translations based on the structure of ontologies, assuming that these are specified by structured sets of axioms consisting of foundational theories, object libraries providing the terminological definitions, and templates that determine certain classes of axioms. Translation then depends on which parts of the ontologies are shared and which are not.

Grüninger's work is a logic-based approach, for ontology translation is defined in terms of logical equivalence—theories can be translated if sentences in one theory can be expressed using the definitions of another theory's ontology,

such that they are logically equivalent with respect to their foundational theories. This is a strong definition and is called *strong translation*. Grüninger formalises other, weaker kinds of translations: *Partial translation* is achieved if it can be established either through sub-ontologies, or because one of the ontologies is extendible with new definitions to make strong translation feasible. Strong and partial translation rely on the ontologies sharing the same foundational theories. If this is not the case, one may still establish *weak translation*, where a partial (or strong) translation can be defined after one foundational theory is interpreted into the other (in the usual sense of a *theory interpretation*, see, for instance, (Enderton 2001)).

In order to determine if two application ontologies are sharable, Grüninger proposes to use an *interchange ontology library* that compiles a set of participating ontologies, organised by how their foundational theories and object libraries are structured according to the sub-theory relation between foundational theories and to stratification of definitions between object libraries. For any two such participating ontologies in the library the lexicon of one should not be expressible using the lexicon of the other, which is achieved by defining them by means of a notion of 'lexicon-closure' within the foundation theory hierarchy and the stratification of object libraries. For a given application ontology, one would need to take the *participating ontology* of the library with which it is sharable—intuitively this is the 'image' of the application ontology in the interchange library. The sort of translations between application ontologies that are feasible would then easily be determined, and constructed, from the structure of their corresponding participating ontologies with respect to the library.

Grüninger's work provides the theoretical ground for discussing the various possible sorts of ontologies and their translations, and for establishing necessary conditions for sharability between applications. His aim was not to tackle the issues of ontology mapping as such, but to provide an architecture—the interchange ontology library—in which various forms of translations could and would be described. This approach to translation requires the explicit definition—and eventual construction—of the interchange ontology library in which all ontologies have sound and complete axiomatisations with respect to their intended models.

3.g Experience reports

Two experience reports are cited here: The experiences with **CyC** ontology mapping of Reed and Lenat (2002) and a report from an experiment of ontology reuse at **Boeing** (Uschold et al. 1998).

Reed and Lenat (2002) report on their experiences with mapping the **CyC** ontology to a number of external ontologies. In particular, their report

> ... presents the process by which over the last 15 years several ontologies of varying complexity have been mapped or integrated with CyC... These include: SENSUS, FIPS 10-4, several large (300k-term) pharmaceutical thesauri, large portions of WordNet, MeSH/ SNOMED/ UMLS, and the CIA World Factbook.

Their work has been manual, laborious, but arguably represents the most comprehensive example of ontology mapping today. Their ultimate goal is to enable subject matter experts to directly map, merge or integrate their ontologies with

the aim of interactive clarification-dialog-based tools. Their process defines a well grain-sized typology of the term 'mapping', in *CyC* language, and distinguishes four types of differences when mapping ontologies: terminological (i.e., different names), simple structural (i.e., similar but disjoint), complex structural (i.e., having action predicates vs. reified events), representational differences (i.e., Bayesian probabilistic vs. truth-logic). Their long term objective is to develop dialogue tools that will use natural language parsing, understanding and generation to insulate the subject matter expert from having to read or write in the *CyC* language.

In an experiment of ontology reuse (Uschold et al. 1998), researchers working at **Boeing** were investigating the potential of using an existing ontology for the purpose of specifying and formally developing software for aircraft design. Their work is not directly related with ontology mapping, however, their insights and experiences gathered are interesting indicators for the level of difficulty involved in the process. The ontology used was the `EngMath` ontology (Gruber and Olsen 1994), and the application problem addressed was to enhance the functionality of a software component used to design the layout of an aircraft stiffened panel. Their conclusions were that, despite the effort involved, knowledge reuse was cost-effective, and that it would have taken significantly longer to design the knowledge content of the ontology used from scratch. However, the lack of automated support was an issue, and the authors elaborate on the effort required from the knowledge engineer:

> The process of applying an ontology requires converting the knowledge-level specification which the ontology provides into an implementation. This is time-consuming, and requires careful consideration of the context, intended usage, and idioms of both the source ontology representation language, and the target implementation language as well as the specific task of the current application.

3.h Surveys

The following surveys originate from a number of different communities: Pinto and colleagues (1999) elaborate and compare issues for **ontology integration**, Visser and colleagues (1998) identify a typology of **ontology mismatches**, Rahm and Bernstein (2001) report on **database schema matching**, and Sheth and Larson (1990) survey **federated database systems**.

In their survey, Pinto and colleagues (1999) elaborate on issues concerning **ontology integration**. Their work attempts to offer terminological clarifications of the term 'integration' and how it has been used in different works. To quote the authors:

> We identify three meanings of ontology 'integration': when building a new ontology by reusing (assembling, extending, specialising or adapting) other ontologies already available; when building an ontology by merging several ontologies into a single one that unifies all of them; when building an application using one or more ontologies.

They also conducted a survey for tools that allow integration, ontologies built through integration and methodologies that include integration.

Visser and colleagues (1998) present a typology of **ontology mismatches**. Their work assesses heterogeneity by classifying ontology mismatches. Their intention is to identify a set of heuristics that allow them to determine whether systems can join a cooperative community, or to provide guidance for the design of such systems. In a related work, Visser and Tamma (1999) propose methods that make use of this information to perform ontology clustering. Their underlying methods for clustering use linguistic resources, like WordNet.

Rahm and Bernstein (2001) present a survey on approaches to automatic **database schema matching**. As we elaborate in Section 5, there might be practitioners for whom ontology mapping equates database schema matching. In this respect, Rahm and Bernstein's work is a comprehensive resource which could be used when comparing different approaches to schema matching, when developing a new match algorithm, and when implementing a schema matching component.

In the same spirit, the work of Sheth and Larson (1990) originates from the databases realm, and reviews the field of **federated database systems**. Federated database systems favour partial and controlled data sharing. However, sharing these data is not easy nor an automated task. The problem lies in the semantic heterogeneity of the schemas used, as the authors say:

> Semantic heterogeneity occurs when there is a disagreement about the meaning, interpretation, or intended use of the same or related data... This problem is poorly understood and there is not even an agreement regarding a clear definition of the problem... Detecting semantic heterogeneity is a difficult problem. Typically, DBMS schemas do not provide enough semantics to interpret data consistently. Heterogeneity due to differences in data models also contributes to the difficulty in identification and resolution of semantic heterogeneity. It is also difficult to decouple the heterogeneity due to differences in DBMSs from those resulting from semantic heterogeneity.

Database schemata consist of schema objects and their relationships. Schema objects are typically class definitions (or data structure descriptions, e.g., table definitions in a relational model), and entity types and relationship types in the entity-relationship model. Schema integration, which is arguably the databases world counterpart of ontology mapping, is manual and laborious work. As the authors report:

> The user is responsible for understanding the semantics of the objects in the export schemas and resolving the DBMS and semantic heterogeneity... A user of a loosely coupled FDBS has to be sophisticated to find appropriate export schemas that can provide required data and to define mappings between his or her federated schema and export schemas. Lack of adequate semantics in the component schemas make this task particularly difficult.

Another approach for the database administrator is to write mapping rules to generate the target schema from the source schema. These rules specify how each object in the target schema is derived from objects in the source schema. These rules are typically based on syntactic and structural similarities

of the schemata. The authors also surveyed the types of relationships between attributes in database schemata and they argue:

> Two attributes a_1 and a_2 may be related in one of the three ways: a_1 `is_equivalent_to` a_2, a_1 `includes` a_2, a_1 `is_disjoint_with` a_2. Determining such relationships can be time consuming and tedious... This task cannot be automated, and hence we may need to depend on heuristics to identify a small number of attribute pairs that may be potentially related by a relationship other than `is_disjoint_with`.

4 Examples

Fernández-Breis and Martínez-Béjar: In Figure 5 we illustrate the example used in (Fernández-Breis and Martínez-Béjar 2002). As we reported in Section 3.a, Fernández-Breis and Martínez-Béjar developed an algorithm for integrating ontologies. The algorithm works as follows: it detects synonymous concepts (e.g., `BUILDING`, `SCIENCES_FACULTY` in both ontologies), as well as exploits nodes in the hierarchy that have the same attributes. The upper part of Figure 5 illustrates two university ontologies describing a faculty of sciences, whereas the lower part illustrates the integrated ontology. The concept `PEOPLE` has been converted to `PERSON` since both concepts share the same attributes (`AGE, INCOME`). The algorithm also integrates attributes of the same concepts (`BUILDING` in the integrated ontology has the sum of its predecessors' attributes in the original ontologies).

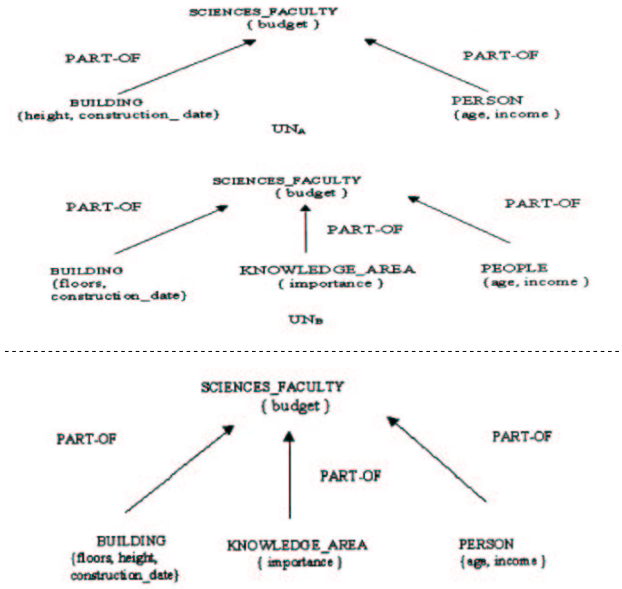

Figure 5: Fernández-Breis and Martínez-Béjar's algorithm at work: Integration of two Faculty of Sciences ontologies.

MAFRA: In Section 3.a we presented the work of Maedche and Staab (2000) on defining semantic bridges to facilitate mapping. In Figure 6 we illustrate MAFRA's framework applied to two small ontologies depicted in UML notation. The ontology on the right-hand side (`o2`) represents individuals using a simple approach by distinguishing only between man and woman; the ontology on the left-hand side (`o1`) enumerates marriages and divorces, events, etc. MAFRA aims to specify mappings between these two using the semantic bridge ontology. The semantic bridges are defined hierarchically and take into account the structure of the ontologies to be mapped. There could be simple semantic bridges, like attribute bridges which are one-to-one correspondences of attributes, like the `o1:Individual:name` and `o2:Individual:name`, as well as complex bridges which take into account structural information. For example, the *SemanticBridgeAlt* at the bottom of Figure 6, is an alternative semantic bridge that was created to map `o1:Individual` to `o2:Man` and `o2:Woman` by establishing two concept bridges, `Individual-Man` and `Individual-Woman`. Once bridges are specified, others can use of this information. For example, attribute bridges rely on the `o1:Individual` to `o2:Individual` bridge to translate the attributes of `o2:Man` and `o1:Woman` inherited from `o2:Individual`.

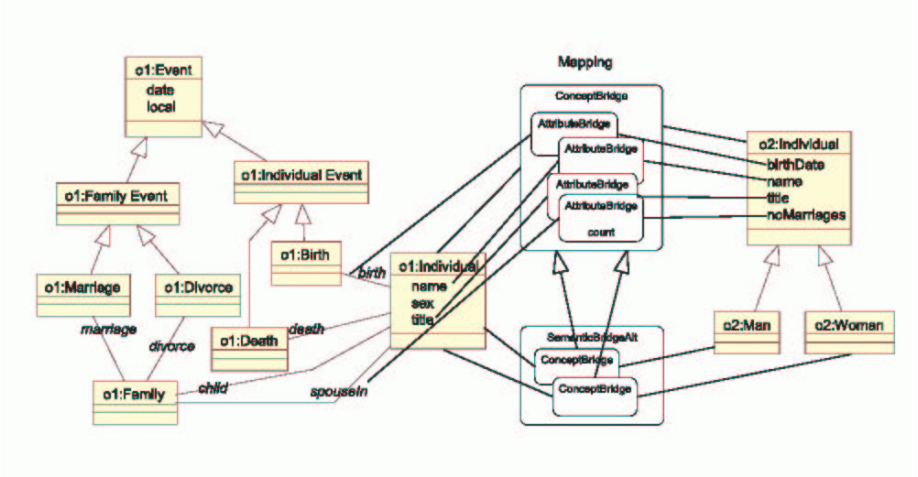

Figure 6: UML representation of MAFRA's semantic bridge based ontology mapping.

OISs: As we mentioned in Section 3.a, OISs framework's mappings are expressed as queries. We briefly present here an example case taken from (Calvanese et al. 2001a): Consider the OIS $O_u = \langle G_u; S_u; M_u \rangle$, where both G_u and the two ontologies S_1 and S_2 forming S_u are simply sets of relations with their extensions. The global ontology G_u contains two binary relations, `WorksFor`, denoting researchers and projects they work for, and `Area`, denoting projects and research areas they belong to. The local ontology S_1 contains a binary relation `InterestedIn` denoting persons and fields they are interested in, and the local ontology S_2 contains a binary relation `GetGrant`, denoting researchers and grants assigned to them, and a binary relation `GrantFor` denoting grants

and projects they refer to. The mapping M_u is formed by the following correspondences:

$\langle V_1; \texttt{InterestedIn}; \textit{complete}\rangle$, with $V_1(r; f) \leftarrow \texttt{WorksFor}(r; p) \wedge \texttt{Area}(p; f)$
$\langle \texttt{WorksFor}; V_2; \textit{sound}\rangle$, with $V_2(r; p) \leftarrow \texttt{GetGrant}(r; g) \wedge \texttt{GrantFor}(g; p)$

In the correspondences given above, V_1 and V_2 are views which represent the best way to characterise the objects which satisfy these views in terms of the concepts in the local ontologies S_1 and S_2. *Sound* and *complete* are characterisations of these correspondences; for their formal specification we point the interested reader to (Calvanese et al. 2001a).

Madhavan and colleagues: In Figure 7 we give an example of Madhavan and colleagues' framework that we mentioned earlier in Section 3.a. That Figure includes two different models of a domain of students. The first model, `MyUniv`, is in DAML+OIL, the second one, `YourUniv`, is a relational schema. The ontology `MyUniv` includes the concepts STUDENT with subclasses ARTS-STD and SCI-STD and COURSE with subclasses ARTS-STD and SCI-CRS. The binary relationship `Taken` represents the courses taken by students, and the relationships `Grade` and `Lives-In` represent properties of students. `Lives-In` is constrainted to have the value "myCity". The schema `YourUniv` includes the tables `student`, `course`, and `enrolled-in`. In addition, the schema includes an integrity constraint specifying that the attribute `address` must contain the string "yourCity". Madhavan and colleagues' framework uses helper models as we mentioned in Section 3.a. One possible mapping between `YourUniv` and `MyUniv` could use a helper model `Univ`, a relational schema with tables `Student`, `Course`, `Arts-Std`, `Sci-Std`, `Arts-Crs`, and `Sci-Crs`. Then the mapping formulae are as follows:

```
Univ.Student(std,ad,gpa) ⊇ MyUniv.STUDENT(std)
                ∧ MyUniv.Lives-In(std,ad) ∧ MyUniv.Grade(std,gpa)
Univ.Student(std,ad,gpa) ⊇ YourUniv.student(std,ad,x,gpa,y)
Univ.Arts-Std(std) ⊇ MyUniv.ARTS-STD(std)
Univ.Arts-Std(std) ⊇ YourUniv.student(std,x,"arts",y,z)
```

The first two formulae map students in the two universities' models to a single student concept in the helper model. The other two formulae map art students and art majors to a single table for arts students.

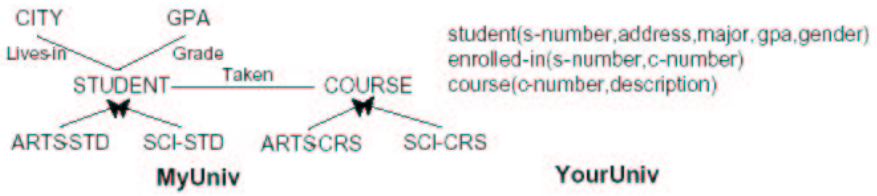

Figure 7: Madhavan and colleagues' models of a student domain.

IF-Map: Kalfoglou and Schorlemmer's IF-Map (Section 3.b) was applied to map AKT's project ontologies (AKT 2001), namely `AKT Reference` to `Southampton`'s and `Edinburgh`'s local ontologies. These local ontologies were populated with a few thousand instances (ranging from 5k to 18k) and a few hundreds of concepts. There were a few axioms defined, and both had relations. The `AKT Reference` ontology was more compact, it had no instances and approximately 65 concepts with 45 relations. There were a few axioms defined as well. In Figure 8 we include a screenshot of a Web-accessible RDF results page for some relations and concepts mapped. In this page, we show a small fraction of the results from mapping concepts and relations from `AKT Reference` to their counterparts in `Southampton`'s ontology. As we can see, apart from mapping concepts, like `AKT Reference`'s *document* to `Southampton`'s *publication*, they also map relations: `AKT Reference`'s *hasappellation* to `Southampton`'s *title*. The arities of these relations and the way local communities are classifying their instances allow this sort of mapping, whereas in other situations this would have been inappropriate, when for example *title* refers to the title of a paper. These mappings were generated automatically.

Figure 8: IF-Map's generated infomorphisms of two CS departments' ontologies in Web-accessible RDF format.

PROMPTDIFF: In Section 3.b we mentioned Noy and Musen's tools for the Protégé ontology editing environment. In Figure 9 we give an example of one of their tools, PROMPTDIFF. As we can see, there are two versions of an ontology of wines. The first one, at the left-hand side of Figure (a), has a class *Wine* with three subclasses *Red wine*, *White wine*, and *Blush wine*. The class *Wine* has a slot *maker* whose values are instances of class *Winery*. The class *Red*

wine has two subclasses, *Chianti* and *Merlot*. The second version, at the middle of Figure 9 (b) has changed the name of the *maker* slot to *produced_by* and the name of the *Blush wine* class to *Rose wine*; there is also a *tannin level* slot to the *Red wine* class; and *Merlot* is also a subclass of *White wine*. At the right-hand side of Figure 9 (c), PROMPTDIFF has found automatically the differences in these two versions of ontology wine. The *map level* rightmost column in that table indicates whether the matching frames are different enough from each other to warrant the user's attention. There are three types of mapping level defined: *unchanged* (nothing has changed), *isomorphic* (images of each other), and *changed* (they are not images of each other). For example, the *Red wine* class has changed: it has a new slot (*tannin level*).

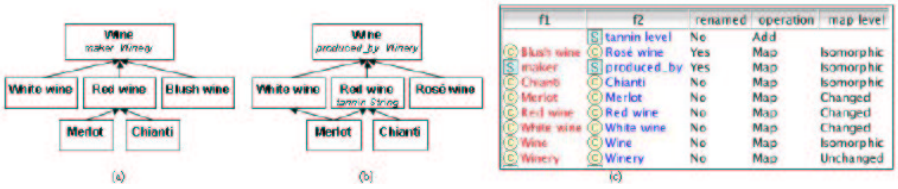

Figure 9: PROMPTDIFF in (c) showing the difference of two wine ontologies, (a) and (b).

ONION: As we mentioned in Section 3.b when we presented Mitra and Wiederhold's system, they use linguistic features to inform their heuristics in order to define articulation rules for mapping. Their linguistic matcher looks at all possible pairs of terms from the two ontologies and assigns a similarity score to each pair. For example, given the strings "Department of Defence" and "Defense Ministry", the `match` function, returns `match(Defence,Defense) = 1.0` and `match(Department,Ministry) = 0.4`. Then, they calculate the similarity between the two strings as: `match("Department of Defence","Defense Ministry") = (1 + 0.4)/2 = 0.7`. The denominator is the number of words in the string with less number of words. The similarity score of two strings is then normalised with respect to the highest generated score in the application. If the generated similarity score is above the threshold, then the two concepts are said to match, and they generate an articulation rule: *(Match "Department of Defence" "Defense Ministry"), 0.7*, the last number gives the confidence measure with which the articulation generator returned this match. Their algorithm, however, is not infallible. If we try to scale up this approach, and take into account Ministries of Foreign Affairs in three countries, USA, UK, and Greece, this linguistic matcher will fail to spot the similarities as we need to take into account the intended semantics, not just the syntax. For example, USA's foreign affairs ministry is called "US Department of State", in the UK it is called "Foreign and Commonwealth Office", and in Greece, "Ministry of Foreign Affairs".

ConcepTool: In Figure 10, we include an example case that Compatangelo and Meisel used in their work (see Section 3.e). The lower half of the figure

shows two entity-relationship schemata, `CARRIER` and `FACTORY`. The upper half shows the articulated schema that has been generated semi-automatically by *ConcepTool*. We will not get into detail when describing the steps followed in generating the articulated schema, but we elaborate on some indicative ones: heuristic lexical analysis is used to spot lexical correlations, e.g., between `PASSENGER-VEHICLE` and `VEHICLE` in the schema `FACTORY`. These satisfy a heuristc rule of having at least 4 characters matched. The underlying description logic reasoner enables formal analysis of the two schemata and highlights that `CARRIER.CARRIER` and `FACTORY.TRANSPORTER` are synonymous. Further linguistic analysis using lexicons, like WordNet, establishes that `CARRIER.LORRY` is a subclass of `FACTORY.TRUCK`. The analyst also plays a vital role in the process as he needs to endorse correspondences between concepts (the dotted lines in the figure). Once the articulated schema is generated, *ConcepTool* detects conflicts or omissions and prompts the analyst to resolve them. For example, entity `CAR` in the articulated schema only contains the attributes which are common to `CARRIER.CAR` and `FACTORY.PASSENGER-VEHICLE`.

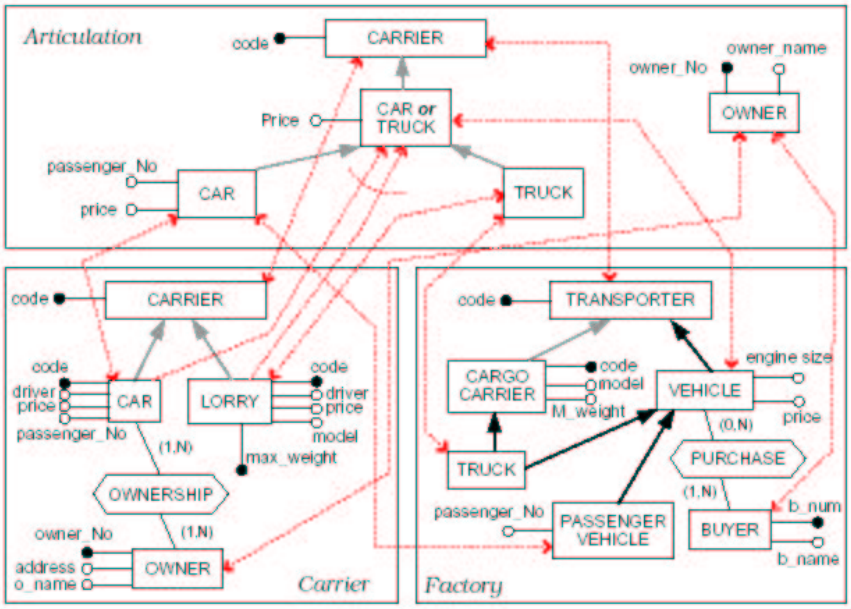

Figure 10: ConcepTool's articulation of two independent modes.

5 Pragmatics

In Sections 3 and 4 we have described and showed examples of 35 works related to ontology mapping. In this section we will elaborate on important topics that emerged when examining these works. We were selective in choosing the topics that we think are prevailing when practitioners are faced with the subtle task of ontology mapping. While the main section of this article aims to act

as a road map of ontology mapping works today, herein, we critically review issues concerned with the relation of ontology mapping and databases schemata integration, the normalisation of ontologies and the creation of formal instances, the role of formal theory in support of ontology mapping, the use of heuristics, the use of articulation and mapping rules, the definition of semantic bridges, and we also discuss the thorny issue of automated ontology mapping.

We start by discussing the relation of ontology mapping and **database schema integration**. In Section 3.h, we reported on the work of Rahm and Bernstein (2001) on database schema matching, and the survey of Sheth and Larson (1990) on federated databases. Database schema matching or integration is regarded by many practitioners as similar to ontology mapping. This follows the ever increasing belief that ontologies are similar to database schemata. Although this statement has many supporters—mainly from a databases background—it also generates a lot of controversy. We are not going to analyse arguments in favour or against the issue of whether a database schema is an ontology, as this is peripheral to our discussion. However, techniques that have been used for database schema matching or integration might be of interest to ontology mapping practitioners. Nevertheless, there are substantial differences which should be taken into account. For example, in a comparative survey, (Noy and Klein 2002) identified a number of areas where ontologies and database schemata are different from the perspective of evolution. These are: (1) Database schema evolution aims to preserve the integrity of data itself, whereas ontology evolution is more complex since ontologies can be seen as data themselves, and a typical query on an ontology could result in elements of the ontology itself. (2) Database schemata do not provide explicit semantics for their data, whereas ontologies are logical systems, and hence the intended semantics is explicitly and formally specified. (3) Database schemata are not sharable or reusable, usually they are defined over a specific database, whereas ontologies are by nature reusable and typically extent other ontologies. (4) Traditionally, database schema development and update is a centralised process, whereas ontology development is more de-centralised and collaborative. (5) Database schema evolution should take into account the effects of each change operation on the data, like addition of a new class; in ontologies, however, the number of knowledge representation primitives is much higher and more complex: Cardinality constraints, inverse properties, transitive properties, disjoint classes, definition of logical axioms, type-checking constraints. (6) Databases make a clear distinction between schema and instance data, whereas in rich knowledge representation languages used for ontology modelling it is difficult to distinguish where the ontology ends and the instances begin.

Another issue which we found in few of the works we surveyed, was the generation of **formal instances** and the **normalisation** of ontologies. Both are techniques which could be used prior to mapping in order to facilitate it. Generating formal instances is imminent for ontologies that are not populated with instances. This is common for upper level ontologies, which are supposed to act as *global* ontologies that are sharable and agreed upon by different communities. Generating these instances is a core issue in the works of Kalfoglou and Schorlemmer (2002), and Madhavan and colleagues (2002). Both use the intended semantics of ontological constructs explicitly given in these ontologies to generate formal instances. In the work of Kalfoglou and Schorlemmer these are classifications that satisfy the semantics of types (concepts) they belong to,

and are generated automatically by using the ontology structure.[4] Having a mechanism to populate ontologies with instances is an important aid for ontology mapping practitioners, as they can explore a different angle in mapping: to focus on the way local communities classify their instances. This is essential when mapping involves a number of ontologies originating from different communities where we should anticipate common concepts to be interpreted differently in local ontologies. Another technique which we found interesting, if not necessary, was that of **normalisation**. In the works of MAFRA (Maedche and Staab 2000) and IF-Map (Kalfoglou and Schorlemmer 2002) these are used to bring different representation formalisms under the same roof. In MAFRA, the authors translate the input ontologies to RDF(S) whereas in IF-Map they are partially translated into Prolog. The aim is similar, namely to work with the same formalism throughout the mapping phase. This is essential for IF-Map where the mapping is completely automated. Their translation style and source languages are different though. However, Madhavan and colleagues (2002) and Chalupsky (2000) argue that their systems can deal with a number of different representation languages without the need to translate them into a common format. We should be cautious, though, when we interpret these claims, particularly in the work of the former: their aim is to construct mapping rules that define mappings between different representation formalisms (ranging from XML to relational databases). Despite of that, the whole process is manual, laborious and presupposes that the knowledge engineer is familiar with the input formalisms, and does thorough inspection of the model semantics and domain to write meaningful mapping rules. In Chalupsky's system a similar goal is achieved by using rewrite rules which are also defined manually.

A similar style of defining these mappings are that of **articulation** rules. We found these in a couple of works mentioned in the survey, (Compatangelo and Meisel 2002) and (Mitra and Wiederhold 2002), and they are similar to the transformation and mapping rules mentioned before. They differ in style though, as articulation rules aim to be more compact and to use the ontology structure, whereas transformation rules are more dependent on the semantics of the language used. As before, these were also constructed manually.

In few of the works we reviewed, we found evidence of ontology mapping maintenance and evolution techniques. That can be achieved by explicitly defining **semantic bridges**. Among those, the work of Maedche and Staab (2000) is probably the most advanced, as it is not only defining a typology of semantic bridges, but the authors provide a reusable ontology of semantic bridges in a format which is compatible with Semantic Web applications (DAML+OIL). Having such an ontology could arguably facilitate maintenance of ontology mappings, support evolution, and enable exchange of semantic bridges among similar domains.

Among the most popular techniques we encountered is that of using **heuristics**. It is not a surprise to everyone who has attempted to do ontology mapping: Heuristics are cheap to develop, easy to deploy, and support automation. However, the main problem with heuristics is that they are easily defeasible. Even well-crafted heuristics for a particular case can fail in similar situations. In Section 4 we showed a small example case involving ONION where a relatively simple and easy to implement heuristic failed to perform in a similar

[4]The whole technique is presented in detail in (Kalfoglou and Schorlemmer 2002).

case. The crux of the problem is in the use of syntactic features, linguistic clues, and structural similarities in input ontologies for designing heuristics. Almost none of the works we encountered used the intended semantics of the concepts to be mapped. This is not surprising either, as these semantics are often not captured in the underlying formalism, and a human expert is needed to give their precise meaning. Several works we reported used this approach, namely, by manually constructing mapping and transformation rules based on these human-interpreted semantics. An alternative was explored in (Kalfoglou and Schorlemmer 2002), where the assumption made was that semantics are controlled by local communities and are reflected in the classification of local instances in accordance to globally agreed types (or concepts). Although there might be misinterpretations of concepts among different communities, the authors of IF-Map aim to capture these as communities classify their instances. However, even in this approach, heuristics are not missing: They are part of the kick-off mechanism for exploring the classification tables and generating automatically infomorphisms among similar concepts.

Last, but certainly not least, the issue that matters most is that of **automation**. It is not extravagant to claim that almost every work we came across failed to produce a fully automated method for ontology mapping. Historical references on works that resemble some of the problems ontology mapping practitioners try to solve today shows that this is inevitable. Sheth and Larson (1990) in their survey argued:

> Complete automation is not feasible as we need more information than currently provided by database schemata, the semantics of data models do not adequately capture the real world, and the absence of structural similarity between schemata or absence of instance data in target applications makes their automatic matching or integration difficult.

Even though ontologies are not the same as databases schemata, the fact that they are more complex makes the problem even trickier. We also have to highlight a hidden assumption in works where the intervention of a human user is highly welcome. The proponents of this approach claim that a human user should be a core part of the system as it can validate and endorse the results, update mapping rules, and inspect the input ontologies and domains. Although we found this effective, it is not practical. These human users have to be domain experts, familiar with the underlying formalisms and technologies and definitely capable of spotting the subtle differences in the semantics of seemingly similar concepts. Furthermore, the advent of the Semantic Web, the proliferation of ontologies nowadays, and agent technology advances, pose hard requirements on the time scales for performing ontology mapping. *It has to be automatic in order to be practical*. So, the majority of works we presented in this article try to reconcile both requirements, automation and high quality mappings, by adopting semi-automatic approaches. However, we should mention that the non-automated part of these approaches remains manual, laborious and still dependent on human experts. In works that full automation is claimed, certain assumptions are made: for example, the authors of IF-Map rely on a set of heuristics to kick-off the method. Although these are ontology-independent, once they fail, a human user has to revise them. Furthermore, full automation in the actual mapping method equals combinatorial explosion, as their method

suffers from exponential growth of the number of possible mappings. The remedy taken to alleviate this situation is that only reasonably-sized fragments of the actual ontologies will be fed into IF-Map. These fragments are identified by the heuristics mentioned above.

6 Conclusions

In this article we presented the state-of-the-art in ontology mapping: 35 works have been reviewed and some of them illustrated through example cases. Many more have been left out of this survey: It was not feasible neither practical to include everything that has been done to date. Rather, we selected indicative examples that characterise a range of related works.

We argue that ontology mapping nowadays faces some of the challenges we were facing ten years ago when the ontology field was at its infancy. We still do not understand completely the issues involved. However, the field evolves fast and attracts the attention of many practitioners among a variety of disciplines, the result being the variety of works we presented in this article. As today we know more about ontologies, how to design, develop, and deploy them. We hope that this article contributes to a better understanding of the emerging field of ontology mapping.

Acknowledgements

This work is supported under the Advanced Knowledge Technologies (AKT) Interdisciplinary Research Collaboration (IRC), which is sponsored by the UK Engineering and Physical Sciences Research Council under grant number GR/N15764/01. The AKT IRC comprises the Universities of Aberdeen, Edinburgh, Sheffield, Southampton and the Open University. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing official policies or endorsements, either expressed or implied, of the EPSRC or any other member of the AKT IRC.

References

S. Abiteboul, S. Cluet, and T. Milo. Correspondence and translation for heterogeneous data. *Theoretical Computer Science*, (275):179–213, March 2002.

Advanced Knowledge Technologies Interdisciplinary Research Collaboration. The AKT Manifesto. Technical Report, 2001. URL: `www.aktors.org/publications/Manifesto.doc`.

M. Barr. The Chu construction. *Theory and Applications of Categories*, 2(2): 17–35, 1996.

J. Barwise and J. Seligman. *Information Flow: The Logic of Distributed Systems.* Cambridge Tracts in Theoretical Computer Science 44. Cambridge University Press, 1997.

T. Bench-Capon and G. Malcolm. Formalising ontologies and their relations. In *Proceedings of the 16th International Conference on Database and Expert Systems Applications (DEXA'99)*, pages 250–259, 1999.

P. Borst, H. Akkermans, and J. Top. Engineering Ontologies. *International Journal of Human-Computer Studies*, 46:365–406, 1997.

D. Calvanese, G. De Giacomo, and M. Lenzerini. A framework for ontology integration. In *Proceedings of the 1st Internationally Semantic Web Working Symposium (SWWS), Stanford, CA, USA*, August 2001a.

D. Calvanese, G. De Giacomo, and M. Lenzerini. Ontology of integration and integration of ontologies. In *Proceedings of the 9th International Conference on Conceptual Structures (ICCS'01), Stanford, CA, USA*, August 2001b.

A.E. Campbell and S.C. Shapiro. Algorithms for Ontological Mediation. Technical Report 98-03, Department of Computer Science and Engineering, State University of New York at Buffalo, January 1998.

H. Chalupksy. OntoMorph: A Translation System for Symbolic Knowledge. In *Proceedings of the 17th International Conference on Knowledge Representation and Reasoning (KR-2000), Colorado, USA*, April 2000.

E. Compatangelo and H. Meisel. Intelligent support to knowledge sharing through the articulation of class schemas. In *Proceedings of the 6th International Conference on Knowledge-Based Intelligent Information & Engineering Systems (KES'02), Crema, Italy*, September 2002.

F. Corrêa da Silva, W. Vasconcelos, D. Robertson, V. Brilhante, A. de Melo, M. Finger, and J. Agustí. On the insufficiency of ontologies: Problems in knowledge sharing and alternative solutions. *Knowledge Based Systems*, 15 (3):147–167, 2002.

A. Doan, J. Madhavan, P. Domingos, and A. Halevy. Learning to map between ontologies on the semantic web. In *Proceedings of the 11th International World Wide Web Conference (WWW 2002), Hawaii, USA*, May 2002.

J. Domingue. Tadzebao and WebOnto: Discussing, Browsing, and Editing Ontologies on the Web. In *Proceedings of the 11th Knowledge Acquisition, Modelling and Management Workshop, KAW'98, Banff, Canada*, April 1998.

H. Enderton. *A Mathematical Introduction to Logic*. Academic Press, 2nd edition, January 2001.

A. Farquhar, R. Fikes, and J. Rice. The Ontolingua Server: A tool for collaborative ontology construction. *International Journal of Human-Computer Studies*, 46(6):707–728, June 1997.

J. Fernández-Breis and R. Martínez-Béjar. A cooperative framework for integrating ontologies. *International Journal of Human-Computer Studies*, 56: 665–720, 2002.

B. Ganter and R. Wille. *Formal Concept Analysis: Mathematical Foundations*. Springer, 1999.

R. Genesereth and R. Fikes. *Knowledge Interchange Format*. Computer Science Dept., Stanford University, 3.0 edition, 1992. Technical Report, Logic-92-1.

W. Grosso, H. Eriksson, R. Fergerson, J. Gennari, S. Tu, and M. Musen. Knowledge Modelling at the Millennium — The design and evolution of Protege2000. In *Proceedings of the 12th Knowledge Acquisition, Modelling, and Management(KAW'99), Banff, Canada*, October 1999.

T. Gruber and G. Olsen. An ontology for engineering mathematics. In J. Doyle, P. Torasso, and E. Sandewall, editors, *Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning, San Mateo, CA, USA.*, pages 258–269, 1994.

M. Grüninger. Ontologies for translation: Notes for refugees from Babel. EIL Technical Report, Enterprise Integration Laboratory (EIL), University of Toronto, Canada, November 1997.

V. Gupta. *Chu Spaces: A Model of Concurrency.* PhD thesis, Stanford University, 1994.

J. Jannink, S. Pichai, D. Verheijen, and G. Wiederhold. Encapsulation and Composition of Ontologies. In *Proceedings of the AAAI'98 Workshop on Information Integration, Madison, WI, USA*, July 1998.

Y. Kalfoglou and M. Schorlemmer. Information-flow-based ontology mapping. In *On the Move to Meaningful Internet Systems 2002: CoopIS, DOA, and ODBASE.* Lecture Notes in Computer Science 2519, pages 1132–1151. Springer, 2002.

R. Kent. The Information Flow Foundation for Conceptual Knowledge Organization. In *Proceedings of the 6th International Conference of the International Society for Knowledge Organization (ISKO), Toronto, Canada*, August 2000.

A. Kiryakov, K. Simov, and M. Dimitrov. OntoMap: Portal for Upper-Level Ontologies. In *Proceedings of the 2nd International Conference on Formal Ontology in Information Systems (FOIS'01), Ogunquit, Maine, USA*, October 2001.

M. Lacher and G. Groh. Facilitating the exchange of explicit knowledge through ontology mappings. In *Proceedings of the 14th International FLAIRS conference, Key West, FL, USA*, May 2001.

O. Lassila and R. Swick. Resource Description Framework (RDF) Model and Syntax Specification. W3C recommendation, W3C, feb 1999.

J. Lee, M. Gruninger, Y. Jin, T. Malone, A. Tate, G. Yost, and other members of the PIF working group. The PIF Process Interchange Format and framework. *Knowledge Engineering Review*, 13(1):91–120, February 1998.

J. Madhavan, P.A. Bernstein, P. Domingos, and A. Halevy. Representing and reasoning about mappings between domain models. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI'02), Edmonton, Alberta, Canada*, August 2002.

A. Maedche and S. Staab. Semi-automatic engineering of ontologies from texts. In *Proceedings of the 12th International Conference on Software Engineering and Knowledge Engineering (SEKE 2000), Chicago, IL, USA*, pages 231–239, July 2000.

D. McGuinness, R. Fikes, J. Rice, and S. Wilder. An Environment for Merging and Testing Large Ontologies. In *Proceedings of the 17th International Conference on Principles of Knowledge Representation and Reasoning (KR-2000), Colorado, USA*, April 2000.

E. Mena, V. Kashyap, A. Illarramendi, and A. Sheth. Domain Specific Ontologies for Semantic Information Brokering on the Global Information Infrastructure. In N. Guarino, editor, *Proceedings of the 1st International Conference on Formal Ontology in Information Systems(FOIS'98), Trento, Italy*, pages 269–283. IOS Press, June 1998.

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

P. Mitra and G. Wiederhold. Resolving terminological heterogeneity in ontologies. In *Proceedings of the ECAI'02 workshop on Ontologies and Semantic Interoperability, Lyon, France*, July 2002.

W.J. Moore. *Software engineering standards: A user's road map.* IEEE Computer Society, 1998.

E. Motta. *Reusable Components for Knowledge Models: Case Studies in Parametric Design Problem Solving*, volume 53 of *Frontiers in Artificial Intelligence and Applications.* IOS Press, 1999.

N.F. Noy and M. Klein. Ontology evolution: Not the same as schema evolution. Smi-2002-0926, University of Stanford, Stanford Medical Informatics, USA, 2002.

N.F. Noy and M. Musen. SMART: Automated Support for Ontology Merging and Alignment. In *Proceedings of the 12th Workshop on Knowledge Acquisition, Modelling and Management (KAW'99), Banff, Canada*, October 1999.

N.F. Noy and M. Musen. PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. In *Proceedings of the 17th National Conference on Artificial Intelligence (AAAI'00), Austin, TX, USA*, July 2000.

N.F. Noy and M. Musen. PROMPTDIFF: A Fixed-Point Algorithm for Comparing Ontology Versions. In *Proceedings of the 18th National Conference on Artificial Intelligence (AAAI'02), Edmonton, Alberta, Canada*, August 2002.

OntoWeb. A survey on ontology tools. EU Thematic network, IST-2000-29243 Deliverable 1.3, OntoWeb — Ontology-based information exchange for knowledge management and electronic commerce, May 2002. URL: `www.ontoweb.org/deliverable.htm`.

S. Pinto, A. Gomez-Perez, and J. Martins. Some Issues on Ontology Integration. In *Proceedings of the IJCAI-99 Workshop on Ontologies and Problem-Solving Methods(KRR5), Stockholm, Sweden*, August 1999.

S. Prasad, Y. Peng, and T. Finin. Using explicit information to map between two ontologies. In *Proceedings of the AAMAS 2002 Wokshop on Ontologies in Agent Systems (OAS'02), Bologna, Italy*, July 2002.

V.R. Pratt. The Stone gamut: A coordination of mathematics. *10th Annual Symposium on Logic in Computer Science*, pages 444–454. IEEE Computer Society Press, 1995.

U. Priss. A Triadic Model of Information Flow. In *Proceedings of the 9th International Conference on Conceptual Structures (ICCS'01), Stanford, CA, USA*, August 2001.

A. Rahm and A. Bernstein. A survey of approaches to automatic schema matching. *The Very Large Databases Journal*, 10(4):334–350, 2001.

S. Reed and D. Lenat. Mapping ontologies into CyC. In *Proceedings of the AAAI'02 workshop on Ontologies and the Semantic Web, Edmonton, Canada*, 2002.

M. Schorlemmer. Duality in Knowledge Sharing. In *Proceedings of the Seventh International Symposium on Artificial Intelligence and Mathematics, Fort Lauderdale, Florida, USA*, January 2002.

G. Schreiber, R. de Hoog, H. Akkermans, A. Anjewierden, N. Shadbolt, and W. Van de Velde. *Knowledge Engineering and Management.* MIT Press, 2000.

E. Sciore, M. Siegel, and A. Rosenthal. Using semantic values to facilitate interoperability among heterogeneous information systems. *ACM Transactions on Database Systems*, 19(3):254–290, 1994.

A. Sheth and J. Larson. Federated database systems for managing distributed, heterogeneous, and autonomous databases. *ACM Computing Surveys*, 22(3): 183–230, September 1990.

J. Sowa. *Conceptual graphs.* Information Processing in Mind and Machine, 1984.

G. Stumme and A. Maedche. Ontology Merging for Federated Ontologies on the Semantic Web. In *Proceedings of the International Workshop for Foundations of Models for Information Integration (FMII-2001), Viterbo, Italy*, September 2001.

M. Uschold, M. Healy, K. Williamson, P. Clark, and S. Woods. Ontology Reuse and Application. In N. Guarino, editor, *Proceedings of the 1st International Conference on Formal Ontology in Information Systems(FOIS'98), Trento, Italy*, pages 179–192. IOS Press, June 1998.

P. Visser and V. Tamma. An experiment with ontology-based agent clustering. In *Proceedings of the IJCAI-99 Workshop on Ontologies and Problem-Solving Methods, Stockholm, Sweden*, August 1999.

P.R.S. Visser, D.M. Jones, T.J.M. Bench-Capon, and M.J.R. Shave. Assessing Heterogeneity by Classifying Ontology Mismatches. In N. Guarino, editor, *Proceedings of 1st International Conference on Formal Ontologies in Information Systems, FOIS'98, Trento, Italy*, pages 148–162. IOS Press, June 1998.