CROSI

Capturing Representing and Operationalising Semantic Integration

A joint research project between

The University of Southampton
School of Electronics and Computer Science
Advanced Knowledge Technologies group

and

Hewlett Packard Laboratories, Bristol, UK

31 October 2005

12th month deliverable: ***final report***

Executive Summary

In October of 2004, the University of Southampton and Hewlett Packard Laboratories at Bristol joined forces to work on a collaborated project, CROSI, in order to investigate semantic integration. The project lasted 12 months and its primary focus was the use of Artificial Intelligence technology, like ontologies, in the Semantic Web environment in order to research issues related to ontology mapping.

A number of new insights and emergent issues were identified by using a practical testbed and a thorough evaluation strategy. The project's main deliverables were: (a) a *comprehensive survey* of state-of-the-art technologies that address semantic integration; (b) engineering artifacts for guiding the potential semantic integration practitioner to develop a semantic integration system: the *Semantic Intensity Spectrum* (SIS) and a *Modular Architecture* for developing semantic integration systems; (c) an ontology mapping prototype system, *CMS (CROSI Mapping System)* which acts as a demonstrator of mapping techniques.

CROSI also shed light and provided fruitful insights for potential extensions of this technology. For example, it was proven that combining different matchers gives us a better alignment than using them independently. But, we also identified areas where more research is needed, like for example, practical and re-usable algorithms for aggregating alignment results.

The project also had a good visibility record and achieved a decent dissemination across various communities (Web community, Knowledge Engineering community), especially when its short life cycle is taken into account. The highlight was our participation in the annual ontology alignment contest where CMS was ranked in the top three systems for alignment and it was one of the two out of seven participating systems that managed to parse correctly and without a flaw the industrial-strength, sizeable ontologies in the medical informatics domain.

In this final project report, we elaborate on the issues mentioned above. This report is also accompanied by a software system and its manual where more technical details regarding the mechanisms used for alignment are explained.

Contents

List of Figures

List of Tables

Chapter 1

Semantic Interoperability and Integration

An important problem acknowledged by a variety of businesses in distributed environments, such as Web and its ambitious extension, the Semantic Web, is that of *semantic heterogeneity*. As systems become more distributed and disparate within and across organisational boundaries and market segments, there is a need to preserve the *meaning of concepts* used in everyday transactions that involve information sharing. In order for these transactions to be successful we need to be able to uncover and expose the semantics of the elements taking part in these transactions. A narrative interpretation of the problem under consideration is that "given two models of the same domain (or mostly overlapping domains), a set of corresponding pairs from respective models should be returned to indicate how these two models can be aligned or merged". Solutions of this kind, are often characterised as *semantic integration* or *semantic interoperability*.

Semantic integration and the technologies that implement it in the Artificial Intelligence (AI) and DataBases (DB) worlds, ontologies and schemata, respectively, are often used to empower solutions to the problem of semantic heterogeneity. To motivate the importance of semantic integration, we briefly present some key application areas where semantic heterogeneity occurs and there is a need for resolving it. This is a non-exhaustive list but merely an indication of the diversity for the application domain of semantic integration.

1. ***Database schema integration***: "Given a set of independently developed schemas, construct a global view." [47]. The schemata often have different structure and the process of integration aims to unify matching elements. Matching is a whole field in its own right and is the core operation of schema integration.

2. ***Data warehouses***: This is a variation of the schema integration where the data sources are integrated into a data warehouse: "A data warehouse is a decision support database that is extracted from a set of data sources. The extraction process requires transforming data from the source format into the warehouse format.". These transformations could be assisted by database schema matching operations.

3. ***E-Commerce***: Trading partners frequently exchange messages that describe business transactions. As each trading partner uses its own message format, this creates the problem of heterogeneity. That is, message formats may differ in their syntax (EDI structured, XML formatted, etc.) or use different message schemata. To enable systems to exchange messages, application developers need to convert messages between the formats required by different trading partners.

4. ***Semantic query processing***: "A user specifies the output of a query (e.g., the SELECT clause in SQL), and the system figures out how to produce that output (e.g., by determining the FROM and WHERE clause on SQL).". The heterogeneity arises when the user specifies the query output in terms which are different from those used in the schema.

5. ***Ontology integration (or merging)***: Given two distinct, and independently developed ontologies, produce a fragment which captures the intersection of the original ontologies. This area is similar to that of schema integration but more difficult in nature due to the rich and complex knowledge representation structures found in ontologies.

6. ***Ontology mapping***: This is a subset of the previous area, mapping ontologies is a step towards integration and it is often the case that mapping ontologies is adequate for most interoperability scenarios on the Semantic Web.

7. ***Semantic Web agents' interoperability***: A pre-requisite for Semantic Web agents to collaborate is their ability to understand and communicate their mental models. These are often model in the form of an ontology and it is likely to be distinct albeit modelling the same universe of discourse. Mapping their ontologies is a major area of interest where automated and scalable solutions are also sought due to the vast number of agents involved in these scenarios.

8. ***Web-based systems interoperability***: interoperability is also a pre-requisite for a number of Web-based systems who serve distinct applications [25]. In particular, in areas where there is diverse and heterogeneous Web-based data acquisition, there is also a need for interoperability.

1.1 Ontologies and Semantic Interoperability

The time has long gone when ontologies were conceived (in their modern computer science incarnation) as a medium for achieving knowledge sharing. The infamous Gruber quote: "a shared conceptualization of a specification" [17] drove much of the research and development in this field throughout the nineties. Ever since these lines were written, ontologies have transformed from a Knowledge Representation (hereafter, KR) experiment in the Artificial Intelligence (hereafter, AI) community, to a mainstream technology that transcends community boundaries and increasingly penetrates the commercial world.

However, the widespread adoption of ontologies in conjunction with the advances of technologies for distributed environments (like the Web and its ambitious extension, the Semantic Web), has brought new challenges for knowledge engineers. The unmoderated proliferation of ontologies in such environments, the increasing need for semantics to be codified in ontologies, and the need to "do business distributively" and in a service-oriented fashion is pushing the original ontology promise of achieving knowledge sharing to its limits. Part of the problem with achieving knowledge sharing between ontologies effectively and efficiently, is the potential mismatches of models that emerge from using different codifications of semantics.

1.2 Mismatches of semantic models

Discovering similarities among semantic models is hard due to the diversity of semantic mismatches. Having examined priori research on the classification of mismatches [12, 43], we propose the following semantic discrepancy hierarchy.

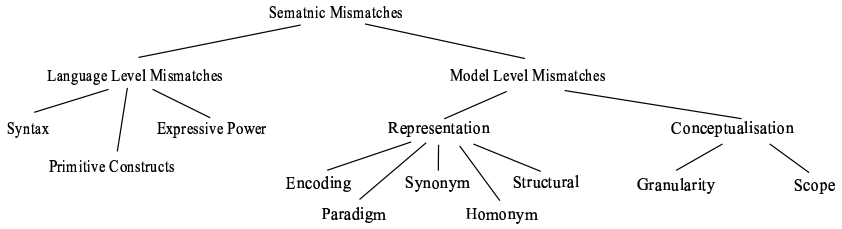


Figure 1.1: Classification of Semantic Mismatches

As illustrated in Figure 1.1, semantic discrepancy might come from both the modelling language and the way knowledge is modelled. While *language level mismatches* can be solved with the help of translation tools, e.g. XPath[1], etc, *model level mismatches* are more difficult to identify and resolve. It might be rooted in different ways of encoding measurement units, different paradigms of representation (e.g. point and internal approaches of representing time), *synonyms*, *homonym*, or different terminological structures. It might also stem from the granularity and the intended scope of the semantic model.

1.3 KR challenges

Before ontologies became popular, knowledge engineers hardly ever had to work with more than one ontology at a time. Even in cases where multiple ontologies were used (see for example, [6]), these were mostly controlled experiments [48] in moderated environments [13]. Nowadays however, the practice is somewhat different. Modern trends in knowledge management dictate that we should expect to work more and more within distributed and open-ended environments like the Web, and its ambitious extension, Semantic Web. That fact alone, has had a significant impact on KR with ontologies:

Firstly, we observe that sourcing ontologies is far easier today than it was in the recent past. Once Semantic Web technologies became more mature (like, for example, the W3C's OWL family[2] of languages or the RDF language[3]), a plethora of ontologies made readily available and accessible via the World Wide Web[4]. Even if the quality or the purpose served by these ontologies is questionable from a strict KR point of view, their impact on the practice is undisputed.

Second, the nature of the environment that most ontologies operate in (Semantic Web, for example), means that it is more likely that we will need more than one ontology to achieve knowledge sharing. It is increasingly unlikely that a single ontology will adequately capture the domain in question and be consensual among all interested parties.

Third, strict knowledge engineering practice is difficult to enforce when dealing with outsourced ontologies. Syntactic compliance with Semantic Web standards (like OWL) is

[1] http://www.w3.org/TR/xpath
[2] http://www.w3.org/2004/OWL/
[3] http://www.w3.org/RDF/
[4] By using, for example, the SWOOGLE tool: http://swoogle.umbc.edu/

not enough to guarantee that our inferences will make sense and automated reasoning will be possible. It is not uncommon to find subtle differences in meaning between any two ontologies even if they represent the same domain and encoded in the same formalism.

Fourth, there a number of reasons that go beyond computational reasoning: (a) *social factors*, like for example the impact that ontology-based codification of knowledge can have in a real world environment (e.g., does it facilitate or complicate human to human knowledge sharing?); (b) *contextual reasoning* (how and when should an ontology take into account and/or represent contextual information?); (c) *social agreements* (are ontologies too formal for enforcing and/or facilitating social agreements between agents (human or artificial?). The answers to these questions are not easy to find neither are they clearly understood. They are the subject of continuous investigations by a variety of communities and are beyond the scope of CROSI.

1.4 Ontology mapping

These challenges should be intertwined with ontology mapping though, since KR plays an important role in the design and deployment of efficient ontology mapping systems. Ontology mapping was always a long standing issue in the research agendas of various communities (databases in the eighties and early nineties, ontologies in mid nineties until today). It is concerned with the task of relating the vocabulary of two ontologies that share the same domain of discourse. A generic definition based on an algebraic definition of an ontology introduced in [28]: "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.". Ontology mapping is only a fragment of a more ambitious task concerning the alignment, articulation and merging of ontologies. Ontology alignment is 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. This ontology together with its mappings, is often called the *articulation* of two ontologies. An articulation allows for defining a way in which the *merging* of ontologies has to be carried out. A systematic and epistemological account of ontology mapping and its related definitions is provided in the surveys of [28] for ontology mapping, [43] for database schema matching, and [40] and [10] for combined views of these two mapping regimes.

1.5 Desiderata for semantic integration systems

By carefully examining existing ontology mapping and DB schema matching approaches [26], we identified the following criteria that one could look at when developing a system for semantic integration: *objectives*, *input*, *output*, *automation*, *extensibility*, *complexity and scalability*.

Objectives: Semantic integration is a core functionality that lends itself to various applications. A number of themes could be placed under the name semantic integration, ranging from federated database systems to distributed ontology development. Hence, it is beneficial identify the objectives that a potential semantic integration is meant to serve. We found it useful to consider the following questions:

Q1 *What does the approach want to achieve?*

Tools and systems might be developed for different purposes of which mapping might be the major (or one of) the goals. For instance, *ConcepTool* [34] is mainly for *ontology articulation* whereas *OMEN* [37] is for *refining existing mappings*.

Q2 *At what stage is the tool or method applicable?*

Schema matching and integration consists of different stages, namely, pre-process, mapping, and post-process stage. Different tools or mechanisms may target a different stage of mapping. It is necessary to define the different stage of a mapping process *a priori* that largely shapes the overall architecture of integration systems.

Input: The characteristics of input to a semantic integration system rely heavily on the nature of the subject problem. For instance,

- it is essential to understand whether the input schemata (or ontologies) are structured or not, as in the latter case tools are needed to extract structured representation from semi-structured data sources and as in MOMIS [5] and Information Manifold [31] systems.
- input schemata (or ontologies) may be represented in different languages, e.g., XML, OWL, etc. When is it necessary to normalise and build an internal model, do schema translation or schema rewriting?
- to what extent, schemata (or ontologies) are similar to each other? Some approaches follow the assumption that the source schemata are inherently similar so that heuristics can be applied when computing similarities. For instance, PROMPTDIFF [39] focuses on finding the differences between two versions of an ontology that are assumed to be overlapping.
- to what extend knowledge other than the schemata themselves (or ontologies) is used? Some approaches rely heavily on domain knowledge, context and/or global ontologies.
- how many schemata (or ontologies) can the tools or mechanisms process at one time? Some approaches claim that they can find mappings among more than one source schemata (or ontologies) at one time, e.g., the Holistic Matching algorithm [20].

Output: The output of a semantic integration system depends on the intended users. Obviously, human users and machines will have different requirements on the representation of the output. Therefore, the following need to be considered:

Mapping representation: the output mapping pairs are for human readers or automated mediators. Certain representations may facilitate further automated processes (post-mapping processes), e.g,. using the Semantic Web Rules Language (SWRL) [21].

Complex vs. Simple mapping: although the ability of discovering complex (n:m) mapping is seldom discussed in ontology mapping, it becomes a *de facto* criterion in database schema matching to demonstrate the capacity of a matching tool or method.

Ranking mechanism: in most cases, exact mapping (i.e, mapping that is specified by a human observer) cannot be identified. Rather, a series of mappings are given with the corresponding confident level subject to a certain ranking mechanism.

Automation: At the early stage of both schema matching and ontology mapping, correspondences are manually crafted by domain experts and/or application engineers [46, 40]. Thus far, fully automated integration is still difficult to achieve and interactions with human and/or external resources are inevitable. To what extent the matching process relies on the input from human observers becomes one of the critical factors. Human intervention may come into the picture at pre-process, similarity-discovery and post-process stages. It is difficult to quantitatively analyze the amount of human effort in the schema matching or ontology mapping process. It is possible, however, to estimate the role played by a human observer, for example, whether human intervention is critical or marginal. Some criteria are enumerated as follows:

1. To what extent, the matching process relies on the existence of external resources. Some approaches rely on a predefined domain ontology, a dictionary or a lexicon to help identifying synonyms, hypernyms (subsumers) and hyponyms (subsumees); some require a set of mappings defined by human experts to initiate further actions. The construction of such resources, if not available already, might require substantial effort.

2. Multi-strategy approaches require a mechanism to screen out and/or compile the best matchings. Such a mechanism might have various forms ranging from interaction with end users to supervised semi-automated methods to adaptive and fully automated ones.

3. Machine learning approaches are popular in schema matching. Training is not trivia in such approaches. Hence, how the training set is collected and how the results are interpreted dictate how much the system is automated.

4. Corpus-based approaches require the construction and validation of a corpus that might be the result of a previous matching circle from other integration systems or manually crafted by human experts.

Extensibility: Different systems normally speak different languages in the sense of input/output format and internal representation. Frequently used formats are Rational, XML, SGML, EER, HTML, RDF, OWL, KRR-specific, to name a few. Furthermore, most mapping (or matching) cases apply a variety of similarity computation techniques. For instance, *linguistic matchers* are employed for names, comments and descriptions; *structural matchers* are hired for comparing hierarchical structures; and logic expressions are analysed for establishing equivalence. Hence, an ideal semantic integration system should adopt a modular design philosophy facilitating the invocation of multiple matchers and aggregation of multiple matching results. We built such an architecture which we describe in detail in chapter 4.

Complexity and Scalability: Semantic Integration is computationally expensive [4, 41, 46]. Thus far, there is no comparative study of the complexity and scalability of different algorithms. This might be due to the fact that most of the systems are addressing the problem of finding the correct correspondences rather than identifying the optimal solving procedure. In many cases, systems are evaluated against toy examples and purposely-built test cases or adopt strong assumptions where empirical analysis is not adequate to demonstrate the computational complexity and scaling features. We advocate the use of formal aspects such as *soundness*, *completeness*, *consistency*, and practical issues such as *scalability* and *complexity*, in combination with empirical evaluation results.

Few of the systems we surveyed ([26]) satisfy one or some of the desiderata listed above. In the next chapter we provide an overview of semantic integration systems' capabilities and characteristics in the form of tables. Full details can be found at the 6th month deliverable, project report [26].

Chapter 2

Semantic Integration Systems

For a comprehensive list and discussion on the different kinds of semantic integration technologies we point the reader to the 6th month report [26]. In this short chapter we simply recapitulate those findings in the form of summary tables with respect to some key issues: a classification regarding a stepwise process in applying semantic integration systems (table 2.1; the systems' objectives and capabilities (table 2.2; criteria for designing and developing semantic integration systems (table 2.3).

2.1 Classification of Semantic Integration Systems

In [26] we identified four phases of semantic integration: (a) pre-integration preparation (a.k.a. normalisation), (b) similarity discovery, (c) similarity representation (also includes reasoning), (d) similarity execution (a.k.a. post-process). The merit of such a representation is to provide the means for comparing different systems and technologies and putting them into context.

In table 2.1 we present such a comparison in a classification table. For instance, IF-Map focuses mainly on how to discover similarities between two ontologies and how to represent similarities, e.g. as RDF triples. On the other hand, FCA-Merge addresses only the similarity discovery.

2.2 Objectives of Semantic Integration Systems

A number of themes could be placed under the name semantic integration, ranging from federated database systems to distributed ontology development. Hence, it is beneficial to identify the objectives that a potential semantic integration is meant to serve.

In table 2.2 we summarise the major objectives for each system we reviewed in [26]. For instance, IF-Map is mainly for ontology mapping while CUPID is for database schema matching. Note that a system might have multiple major objectives.

2.3 Criteria for developing Semantic Integration Systems

We identified the following criteria that one could look into when developing a system for semantic integration: *objectives*, *input*, *output*, *automation*, *extensibility*, *complexity* and *scalability*. We elaborate on each of these criteria in detail in [26] but here we summarise them in table 2.3.

	Pre-Integration Preparation	Similarity Discovery	Similarity Representation	Similarity Execution
Madhavan et. al. system ([26]: p.21)			√	
Information Flow Framework ([26]: p.24)	√		√	√
OIS framwork ([26]: p.26)			√	
DIKE ([26]: p.33)		√		
InfoSleuth ([26]: p.34)		√		
MOMIS ([26]: p.34)		√	√	
SIMS ([26]: p.35)				√
TSIMMIS ([26]: p.35)	√		√	√
Clio ([26]: p.36)		√		√
DELTA ([26]: p.37)		√		
TranScm ([26]: p.41)				√
Breis and Bejar system ([26]: p.42)		√		√
MAFRA ([26]: p.47)		√	√	√
OntoMapO ([26]: p.48)	√		√	
OntoMorph ([26]: p.49)				√
SKAT ([26]: p.50)		√	√	√
Automatch ([26]: p.35)		√		
Autoplex ([26]: p.35)		√		
COMA ([26]: p.36)		√		
CUPID ([26]: p.36)		√	√	
GLUE and iMAP ([26]: p.38)		√		
Holistic Matching ([26]: p.39)		√		
OBSERVER ([26]: p.39)		√		√
OntoBuilder ([26]: p.39)	√	√	√	
SEMINT ([26]: p.40)		√		
W3TRANS ([26]: p.41)			√	
CAIMAN ([26]: p.43)		√		
Chimeara ([26]: p.44)		√		
ConcepTool ([26]: p.44)	√	√	√	√
FCA-Merge ([26]: p.44)		√		
Information Manifold ([26]: p.33)			√	√
IF-Map ([26]: p.45)		√	√	
ITTalks ([26]: p.47)		√		
ONION ([26]: p.48)		√	√	
QOM ([26]: p.49)	√	√		√
SMART, PROMPT, PROMPTDIFF([26]:p.51)		√		
S-Match([26]:p.52)		√		
FALCON ([22])		√	√	
SAMBO ([30])		√	√	
OMEN ([26]: p.48)		√		√

Table 2.1: Classification of systems with respect to four phases of semantic integration.

	Database Schema				Ontology				Information Integration
	Matching	Integration	Rewriting	Translation	Mapping	Translation	Articulation	Integration	
Madhavan *et.al.* system([26]:p.21)					√				
Information Flow Framework([26]:p.24)					√				
OIS framwork([26]:p.26)							√		
DIKE([26]:p.33)		√							
InfoSleuth([26]:p.34)									√
MOMIS([26]:p.34)									√
SIMS([26]:p.35)			√						
TSIMMIS([26]:p.35)			√						
Clio([26]:p.36)			√						
DELTA([26]:p.37)	√								
TranScm([26]:p.41)				√					
Breis & Bejar system([26]:p.42)							√	√	
MAFRA([26]:p.47)					√				
OntoMapO([26]:p.48)					√				
OntoMorph([26]:p.49)						√			
SKAT([26]:p.50)							√		
Autoplex([26]:p.35)		√							
Automatch([26]:p.35)	√								
COMA([26]:p.36)	√								
CUPID([26]:p.36)	√								
GLUE (iMAP)([26]:p.38)	√								
Holistic Matching([26]:p.39)	√								
Information Manifold([26]:p.33)			√						√
OBSERVER([26]:p.39)			√						
OntoBuilder([26]:p.39)								√	
SEMINT([26]:p.40)	√								
W3TRANS([26]:p.41)				√					
CAIMAN([26]:p.43)					√				
Chimeara([26]:p.44)					√			√	
ConcepTool([26]:p.44)							√		
FCA-Merge([26]:p.44)					√				
IF-Map([26]:p.45)					√				
ITTalks([26]:p.47)					√				
OMEN([26]:p.48)					√				
ONION([26]:p.48)							√		
QOM([26]:p.49)					√				
SMART, PROMPT, PROMPTDIFF([26]:p.51)					√			√	
FALCON([22])					√				
SAMBO([30])					√				
S-Match([26]:p.52)					√				

Table 2.2: Objectives of semantic integration systems.

	Input	Output	Cardinality[1]	Rating
DIKE([26]:p.33)	ER	Mapping Table	1:1	yes
MOMIS([26]:p.34)	XML/XMLS/RDF/ relational/object	XML	n:1 (related terms)	N/A
SIMS([26]:p.35)	KL-ONE (Loom)	Query Rewriting Rules	unspecified	N/A
TSIMMIS([26]:p.35)	unspecified	Logic-based Queries	unspecified	N/A
Clio([26]:p.36)	Relational	Relational	unspecified	
DELTA([26]:p.37)	Text	Correspondence	unspecified	yes
TranScm([26]:p.41)	(Semi-)Structured	Rules	unspecified	
Breis and Bejar system([26]:p.42)	Text	Text/Tree-structure	unspecified	N/A
MAFRA([26]:p.47)	XMLS/Relational	DAML+OIL[2]	1:1/1:n/m:1	
OntoMorph([26]:p.49)	KR Languages	Rules	unspecified	no
SKAT([26]:p.50)	Structured	Matching Rules	1:1/1:n/m:1	
Autoplex([26]:p.35)	Relational	Relational	1:1/1:n	yes
Automatch([26]:p.35)	HTML	unspecified	1:1	yes
COMA([26]:p.36)	XML/Relational	Matching Pairs	1:1	yes
CUPID([26]:p.36)	XML/Relational	Matching Pairs	1:1/n:1	yes
GLUE (iMAP)([26]:p.38)	Unspecified (Relational)	unspecified	1:1(complex[3])	yes
Holistic Matching([26]:p.39)	Text/HTML	Mapping N-tuples	m:n:k	yes
OBSERVER([26]:p.39)	Text/HTML/ Relational/others	Lisp-style Query	unspecified	N/A
OntoBuilder([26]:p.39)	HTML Forms[4]	Term dictionary	1:1	
SEMINT([26]:p.40)	Relational	Relational	1:1	
W3TRANS([26]:p.41)	SGML/HTML/OO/ Structured Text	(R-)Correspondence Translation Rules	complex	no
CAIMAN([26]:p.43)	Bookmark Hierarchy	unspecified	unspecified	yes
Chimeara([26]:p.44)	OKBC-compliant files	HTML	unspecified	N/A
ConcepTool([26]:p.44)	EER	EER	unspecified	
FCA-Merge([26]:p.44)	Text documents	Merged Ontology	unspecified	no
Information Manifold([26]:p.33)	(Semi-)Structured	unspecified	complex	yes (sources)
IF-Map([26]:p.45)	unspecified	RDF	1:1/1:n	
ITTalks([26]:p.47)	DAML+OIL	DAML+OIL	semantic	
OMEN([26]:p.48)	RDF-like	Mapping Pairs	unspecified	yes
ONION([26]:p.48)	XML	Articulation Rules	1:1	yes
QOM([26]:p.49)	RDFS	Mapping Pairs	unspecified	
SMART, PROMPT, PROMPTDIFF([26]:p.51)	Protégé Knowledge Model	(Mis-)Matching Pairs	1:1/1:n/m:1	yes
FALCON([22])	OWL	Mapping table	1:1	no
SAMBO([30])	OWL and DAML+OIL	Mapping suggestions	semantic	N/A
S-Match([26]:p.52)	Graph-like Structure	Mapping Matrix	semantic	yes

[1] *Semantic here refers to cases that instead of giving pair-wise mappings, relationships such as "broader than", "subsume", etc. are used.*
[2] *the output of MAFRA is instances of the Semantic Bridge Ontology in DAML+OIL.*
[3] *complex matching refers to the correspondences between combination of attributes in one data source and combination in the other.*
[4] *Ontobuilder is experimented on HTML forms, but, is argued that it is model independent.*

Table 2.3: Semantic integration systems' features.

2.4 Semantic Intensity Spectrum

Semantic integration practitioners belong to, broadly speaking, two major communities: database (DB) and ontology. DB semantic integration is mainly associated with schema integration. It has a long line of research which goes back to more than twenty years when distributed databases caught the attention of practitioners and, at roughly the same time, the notion of *Federated Database* was introduced [46]. An integration mechanism takes as input local schemata and fuse them as either an integrated view over concrete local schemata (referred to as *global-as-view*) or a concrete global schema based on which local views are created (referred to as *local-as-view*). As research of integration evolves from an *ad hoc* effort by individuals to a joined effort by entire communities, a rather disappointing conclusion which seemed that was accepted by most researchers was that "a fully automatic approach to the schema integration problem is not possible." [46].

On the other end of semantic integration practice, distributed ontology development efforts became popular. Fueled by the need to (semantically) integrate the sheer numbers of publicly available ontologies, ontology practitioners face a similar problem: find mechanisms for ontology mapping. Artificial intelligence (AI) researchers argue that ontology mapping is a more complicated problem compared to DB schema matching due to the semantics and diversity inherited in various knowledge representation formalisms [41]. Such a difference is evident in the fact that many early schema matching systems and methods considered, for example, only names of attributes. However, as DB programming languages have been gradually enriched with more semantics, e.g., Enhanced Entity Relationships (EER), Object-Oriented DBs (OODB), these differences become blurred.

We are currently witnessing a shift of both communities towards a converging point [29]. This is evident from the matching and mapping techniques adopted by both communities, which have much in common and a tendency to take into account the underlying semantics.

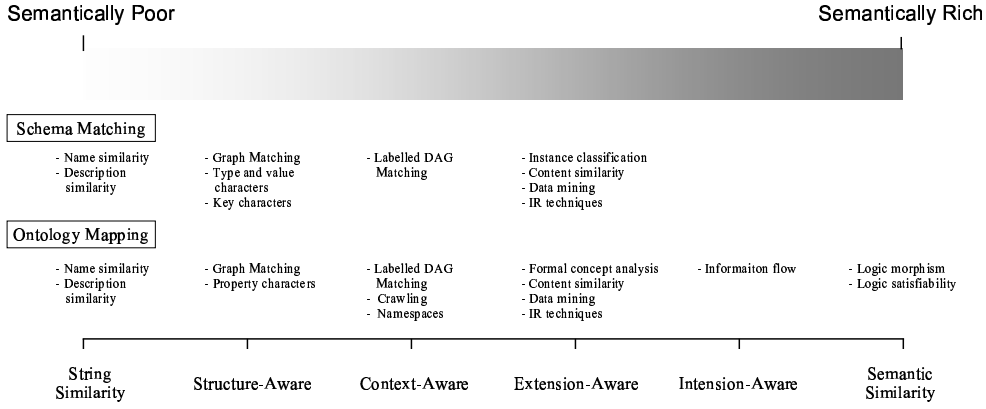


Figure 2.1: Semantic Intensity Spectrum

We observe a common trend for DB and AI semantic integration practitioners: to progress from semantically-poor to semantically-rich solutions, so to speak. We therefore, used this metaphor of semantic richness to classify works from both communities along a *semantic intensity spectrum*. We marked several interim points in the spectrum to address string similarity, structure, context, extension and intension awareness as different layers of semantic intensity (see Figure 2.1).

String similarity, occupying the semantically-poor end of the spectrum, compares na-

mes of elements from different semantic models. A refinement of such techniques enhances the result by also taking into account the lengthy textual descriptions (a.k.a., comments) associated with concepts and properties. These techniques are based on the assumption that concepts and properties names representing semantic similarity will have similar syntactic features. A string matcher usually first normalises the input string of names and/or descriptions via stemming and tokenisation. In the simplest form, the equality of tokens will be obtained and combined to give a score of the equality for the whole string. In a slightly more complicated form, similarity of two strings is computed by evaluating their substrings, edit distance, etc. Nowadays, pure string similarity measures are seldom used in practice, but rather in combination with external resources, like user-defined lexica and/or dictionaries.

Linguistic Similarity, at a position very close to the semantically-poor end, is an example of string similarity measures blended with some sense of semantics. For instance, pronunciation and soundex are taken into account to enhance the similarity purely based on strings. Also, synonyms and hypernyms will be considered based on generic and/or domain-specific thesauri, e.g. WordNet, Dublin Core. In many cases, user-defined name matches are often treated as useful resources. For lengthy descriptions, Information Retrieval (IR) techniques can be applied to compare and score similarities.

As a basic group of matching techniques, linguistics usually are the initial step to suggest a set of raw mappings that other matchers can work with. Many systems invoke linguistic matchers at some stage: PROMPT [38] relies on a linguistic matcher to give initial suggestions of potential mappings which are then refined and updated in later stages; CUPID [33] employees linguistics at the first phase of its matching process when a thesaurus for short forms, acronyms and synonyms matches individual schema elements based on their names, data types, domains, etc.

Structure-aware, refers to approaches that take into account the structural layout of ontologies and schemata. Going beyond matching names (strings), structural similarity considers the entire underlying structure. That is, when comparing ontologies there is a hierarchical, partially ordered lattice where ontology classes are laid out. Similarly, DB schemata use a lattice of connections between tables and classes, not necessarily in a hierarchical fashion though.

In pure structural matching techniques, ontologies and schemata are transformed into trees with labelled nodes, thus matching is equivalent to matching vertices of the source graph with those of the targeted one. Similarity between two such graphs, G_1 and G_2 is computed by finding a subgraph of G_2 that is *isomorphic* to G_1 or vice versa. Although nodes of such graphs are labelled, their linguistic features rarely play a significant role in computing the similarity. Furthermore, labels of edges are normally ignored with the assumption that only one type of relation holds between connected nodes. For instance, suppose we have two fragments of e-Commerce schemata, one describing an arbitrary `Transaction` and the other one a `PurchaseOrder` (see Figure 2.2). Graph's isomorphism then gives us, among other possible mappings: $\{\text{PO} \leftrightarrow \text{PurchaseOrder}, \text{POShipTo} \leftrightarrow \text{Address1}, \text{POBillTo} \leftrightarrow \text{Address2}, \ldots\}$.

Analogous to pure *string similarity* methods, structure matching approaches, such as the one presented in [50] are not common in practice, but they are usually enhanced with other matching techniques. We deliberately use the notion of *structure similarity* in a broad sense in order to accommodate many relevant methods that relate to each other, and which could, and sometimes are used in such a combined fashion.

Typically, algorithms that do structure to structure comparison use the properties found in these structures (transitivity, cardinality, symmetry, etc) as well as their tree

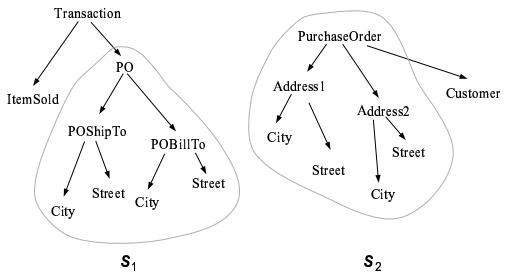


Figure 2.2: Structure Awareness

form similarity (for example, similar branches). Other algorithms use information at the nodes other than label, for example, attributes such as datatype, range and domain, etc., [36]. These are used as if they were labels (strings) with the range of methods discussed above available for comparing them.

Context-aware, in many cases there are a variety of relations among concepts or schema elements which makes it necessary to differentiate distinct types of connections among nodes. This gives rise to a family of matching techniques which are more semantically rich than *structure similarity* ones.

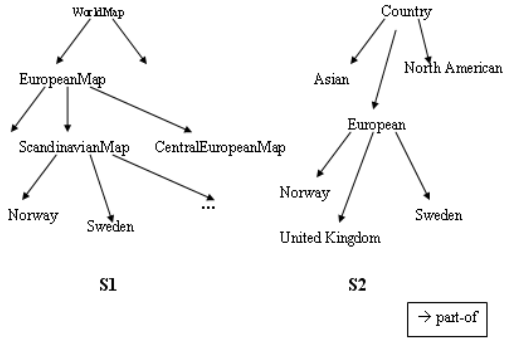
Both DB schema and ontology can be transferred into a labelled directed graph of which nodes could be elements and concepts, and edges, could be attributes and properties, respectively, with the names of attributes and properties as labels. A *context*, defined in graph jargon, is an arbitrary node together with nodes that are connected to it via particular types of edges which at the same time satisfy certain criteria, e.g., a threshold of the length of paths.

Sometimes, *context-aware* approaches group and weigh the edges from and to a node to impose a view of the domain of discourse from the end user perspective. Depending on whether importing external resources is allowed, there are two types of context-awareness.

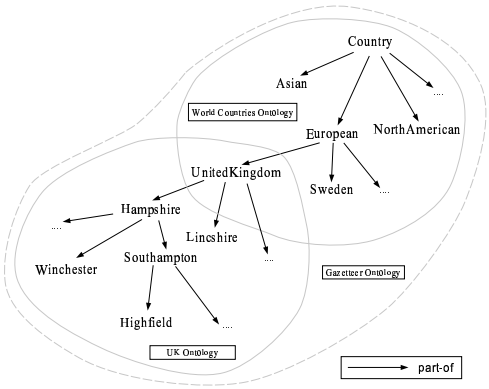
In the simplest form, algorithms that compare nodes from two schemata also traverse downwards several layers along the direction of edges from the node under consideration, or upwards against the direction of edges to the node under consideration. All the visited nodes, together with the information about edges connecting them (taxonomic relationships like part-of, subclass-of, etc.) are evaluated as a whole to infer further mappings between nodes in the context. For instance, in Figure 2.3(a), the issue whether "Norway" in S_1 corresponds to "Norway" in S_2 is evaluated together with the information provided by their ancestors along the part-of relationship path. In this figure, these two nodes do not match, as "Norway" in S_1 refers to a map of this country while "Norway" in S_2 refers to the country itself.

Similarity flooding [35] is an example of a *context-aware* approach. An arbitrary schema S_n is first transformed into a directed labelled graph. The initial mappings between two schemata, S_1 and S_2, are obtained using certain mapping techniques, e.g., a simple string matcher comparing common prefixes and suffixes of literals, and captured to a Pairwise Connectivity Graph (PCG). Nodes of a PCG are elements from $S_1 \times S_2$, denoted as $\mathrm{N}_{S_1 \times S_2}$. An edge labelled $\alpha : (m \times k) \rightarrow (n \times l)$ $(m, n \in S_1$ and $k, l \in S_2)$ of a PCG means that an α edge is present in the original schemata between m and n as well as k and l, i.e. $\alpha : m \rightarrow n$ and $\alpha : k \rightarrow l$.

From a PCG, a similarity propagation graph is induced which assigns to each edge in the PCG a propagation coefficient to indicate the influence between nodes of the PCG. In other words, the weighted edges indicate how well the similarity of a given PCG node



(a) "Norway" appears in different contexts



(b) Co-referencing to "UnitedKingdom"

Figure 2.3: Context Awareness

propagates to its neighbour. The accumulation of similarity is performed until a pre-set threshold is reached or terminated by the user after some maximal number of iterations. A series of filter methods are then adopted to reduce the size of the resultant mapping candidates and select the most plausible ones.

Following the same philosophy—similarity propagation, Palopoli and colleagues [42] integrates multiple ER schemata by using the following principle: similarity of schema elements depends on the similarity of elements in their vicinity (nearby elements influence match more than those farther away). ER schemata are first transformed into graphs with entities, relationships, and attributes as nodes. The similarity coefficient is initialised by standard thesauruses and re-evaluated based on the similarity of nodes in their corresponding vicinities.

With the use of *namespaces*, along comes another type of *context awareness*. As illustrated in Figure 2.3(b), "UnitedKingdom" belongs to both "World Countries Ontology" and "UK Ontology". Articulating these two ontologies summons the resolution of different namespaces that might involve string matchers in certain forms. An example of dealing with co-reference resolution of such namespaces is given in [1].

Extension-aware, when a relatively complete set of instances can be obtained, semantics of a schema or ontology can be reflected through the way that instances are classified. A major assumption made by techniques belonging to this family is that instances with

similar semantics might share features [32], therefore, an understanding of such common features can contribute to an approximate understanding of the semantics.

Formal Concept Analysis (FCA) [14] is a representative of instance-aware approaches. FCA is a field of mathematics emerged in the nineties that builds upon lattice theory and the work of Ganter and Wille on the mathematisation of concept in the eighties. It is mostly suited for analysing instances and properties of entities (concepts) in a domain of interest. FCA consists of formal contexts and concept lattices. A formal context is a triple K=$(\mathcal{O}, \mathcal{P}, \mathcal{S})$, where $\mathcal{O}$ is a set of objects, $\mathcal{P}$ is a set of attributes (or properties), and $\mathcal{S} \subseteq \mathcal{O} \times \mathcal{P}$ is a relation that connects each object o with the attributes satisfied by o.

The intent (set of attributes belonging to an object) and the extent (set of objects having these attributes) are given formal definitions in [14]. A formal concept is a pair $\langle A, B \rangle$ consisting of an extent $A \subseteq O$ and an intent $B \subseteq P$, and these concepts are hierarchically ordered by inclusion of their extents. This partial order induces a complete lattice, the concept lattice of the context. FCA can be applied to semi-structured domains to assist in modelling with instances and properties in hierarchical, partially ordered lattices. This is the main structure most the mapping systems work with. Thus, FCA albeit not directly related to mapping, it is a versatile technology which could be used at the early stages of mapping for structuring a loosely defined domain.

Intension-aware refers to the family of techniques that establish correlations between relations among extent and intent. Such approaches are particularly useful when it is impossible or impractical to obtain a complete set of instances to reflect the semantics.

Barwise and Seligman [3] propose a mathematical theory, *Information Flow*, that aims at establishing the laws that govern the flow of information. It is a general theory that attempts to describe information flow in any kind of a distributed system. It is based on the understanding that information flow results from regularities in a distributed system, and that it is by virtue of regularities among the connections that information of some components of a system carries information of other components. As a notion of a component carrying information about another component, Barwise and Seligman follow the analogy of types and tokens where tokens and its connections carry information. These are classified against types and the theory of information flow aims to capture this aspect of information flow which involves both types and tokens.

When integration is our major concern, the same pattern arises: two communities with different ontologies (or schemata) will be able to share information when they are capable of establishing connections among their tokens in order to infer the relationship among their types. Kalfoglou and Schorlemmer [27] argued for the relation of information flow to a distributed system like the (Semantic) Web, where the regularities of information flowing between its parts can be captured and used to do mapping. The mathematical background of information flow theory ensures that the corresponding types (concepts) respect token (instance) membership to each of the mapped types. Their approach is community-oriented, in the sense that communities on the (Semantic) Web own and control their data (instances) and they use them (i.e., classify them) against ontologies for the purpose of knowledge sharing and reuse. It is precisely this information of classifying your own instances against ontologies that is used as evidence for computing the mapping relation between communities' heterogeneous ontologies. It is evident that *information flow* goes beyond *extension-awareness* towards the tick marked by *intension-aware*.

Semantic Similarity, very close to the semantically-rich end lays the family of *logic satisfiability* approaches which focus on the logic correspondences. Logic constructors play a significant role in expressive formalisms, such as DLs, implying that the discovery of similarity is more like finding logic consequence. The idea behind techniques in this

category is to reduce the matching problem to one that can be solved by resorting to logic satisfiability techniques. Concepts in a hierarchical structure are transformed into well-formed logic formulae (wffs). To compute the relationships between two set of wffs amounts to examine whether (ψ, *wffs1*, *wffs2*) is satisfiable. ψ is the set of relationships normally containing not only equivalence but also "more general than" denoted as $\supseteq$, "less general than" denoted as $\subseteq$, "disjoint with" denoted as $\oplus$, etc.

The major difference among these approaches is on how the wffs are computed with respect to each concept (and/or label of concept). Bouquet and colleagues [7] introduce an algorithm with the notions of *label interpretation* and *contextualization*, called CTX-MATCH. Each concept in a concept hierarchy is associated with a formula based on the WordNet senses of each word in the label of the concept. The senses associated with each label are refined according to the information provided by its ancestors and direct descendants. Matching of two concepts, C_1 and C_2, is then transformed into checking the satisfiability of a formula composed by contextualised senses associated with their labels and the known WordNet relations among senses expressed in logic formulae, e.g. `art#1` $\subseteq_{\text{WordNet}}$ `humanities#1` denotes that, according to WordNet, the first sense of the word "art" is less general than the first sense of the word "humanities" where "art" and "humanities" are words from the labels of C_1 and C_2 respectively.

S-Match [18] goes one step further by distinguishing two different notions of concept, namely the *concept of label* and *the concept of node*. Concept of a label is context insensitive concerning only the WordNet senses of the labels of a concept. On the other hand, concept of a node is context-sensitive, its logic formula is computed as the "intersection of the concepts at labels of all the nodes from the root to the node itself." [18]. The concept of label matrix is constructed containing the relations exist between any two concepts of labels in the two hierarchies of which the matching is to be obtained. Based on such a matrix the concept of node matrix is calculated.

Chapter 3

CROSI System

In the context of this project, we built an architecture that is characterized as a multi-stage and multi-strategy system comprising of four modules, namely, *Feature Generation*, *Feature Selection and Processing*, *Aggregator* and *Evaluator*. In this system, different features of the input data are generated and selected to fire off different sorts of feature matchers. The resultant similarity values are compiled by multiple similarity aggregators running in parallel or consecutive order. The overall similarity is then evaluated to initiate iterations that backtrack to different stages.

3.1 Modular Architecture

Multi-component and multi-strategy approaches are demonstrated by many systems, e.g. COMA [19], GLUE [11], and QoM [12]. Our approach, as illustrated in Figure 3.1, is different in that it allows: 1) multiple matchers: several heterogeneous matchers run independently producing intrinsic, yet different but complementary results; 2) use of existing systems which are treated as standard building blocks each of which is a plug and play component of the overall hybrid mapping system; 3) multiple loops: the overall similarity is evaluated by users or supervised learners to initiate iterations that backtrack to different stages of the process.

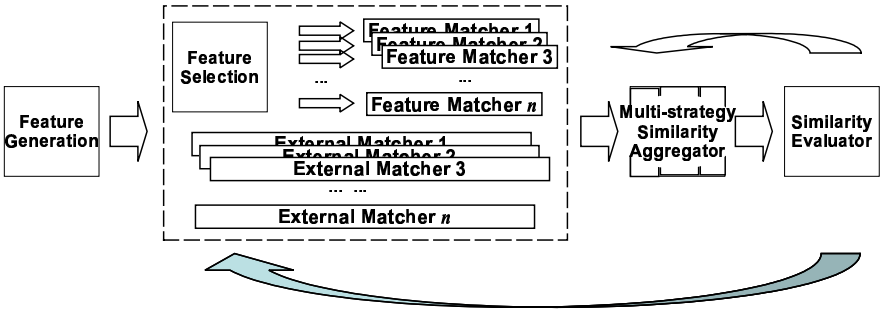


Figure 3.1: A modular architecture.

3.1.1 Challenges for deploying the architecture

There are a number of challenges which we need to consider when building such a system: in ideal situations, each independent matcher considers an identical set of characteristics of the input ontologies and produces homogeneous output for further processes. However,

this is seldom true in practice. There is currently no standard or common agreement on how an ontology mapping system should behave, i.e. no formal specification on what should be the input and how the system should output. If we consider some recent OWL based ontology alignment systems, we see intrinsic diversities: some take only names (URIs) of classes, others take as input the whole taxonomy; some generate as output abstract relationships (e.g. *more general than*, *more specific than*, etc.) while others produce pairwise correspondences with or without confidence values; and some are stand-alone systems when others operate as Web services. Thus, the first and most imminent task is to extract from the input ontologies *features* that suit not only systems that are to be included in the architecture but also future ones. In other words, extracted features should fully characterize the input ontologies no matter which representation language is used.

Equally difficult to build are methods to process and aggregate results from different mapping systems (also refer to as *external matchers*). An unbiased measure is to run in parallel componential matchers each of which produces its own results. The output that might be heterogeneous is then normalized and unified to facilitate accumulation and aggregation with numeric and non-numeric methods.

In the next section we present a mapping system, CMS (CROSI Mapping System), which we built as an instantiation of the proposed modular architecture.

3.2 CMS: Crosi Mapping System

An instantiation of the modular architecture is the CROSI Mapping System (CMS). CMS is a structure matching system that capitalizes on the rich semantics of the OWL constructs found in source ontologies and on its modular architecture that allows the system to consult external linguistic resources. It operationalises the modular architecture described in the previous chapter and employs a multi-strategy system comprising of four modules, namely, *Feature Generation*, *Feature Selection and Processing*, *Aggregator* and *Evaluator*.

3.2.1 Ontology features used for mapping

In CMS, different features of the input data are generated and selected to fire off different sorts of feature matchers. Hence, the first step when deploying CMS was to extract characteristics that can be used to identify similar entities from different ontologies. We summarize the characteristics we extracted in table 3.1.

There are several points that need further explanation. First, in many cases, identifying corresponding instances is considered to be an easier task than identifying corresponding classes. This is because instances are expected to have more grounded variables. Corresponding instances provide a ground on which the number of candidate mapping classes can be narrowed down to a few (as we discovered in our past work with the IF-Map instance-based system [23]). Second, in case of complement classes, let c_s be a class from the source ontology and c_t from the target ontology, if $\texttt{sim}(c_s, c_t) = a$ and $d = \neg c$, we can safely conclude that $\texttt{sim}(d, c_s) = 1 - a$, where $\texttt{sim}/2$ is the similarity function and a, a real number, gives the confident value.

3.2.2 CMS GUI

The resultant similarity values are then compiled by multiple similarity aggregators running in parallel or consecutive order. The overall similarity is then evaluated to initiate

Local features	
class *URIs*	names in many cases convey the intended semantics.
equivalent classes	hints for identifying new mapping candidates.
declared *properties*	both declared and inherited properties contribute to the meaning of a class.
complement classes	complement classes indicates semantic dissimilarity.
property *URIs*	see *classes URI*.
property *domain*	means to refine the semantics of classes
inverse (transitive) property	hints for similar properties and thus indirect hints for similar classes.
functional property	unique identifier for instances
instance *URIs*	see *classes URI*.
instantiated classes	the set of instances convey the semantics of a class.
comments	well documented design rationale is a reliable source for revealing semantics.
Global features	
super and *sub classes*	hints for identifying the location of a class in the taxonomy and thus capture the structural semantics.
sibling classes	hints of how the parent class is defined.
super and *sub properties*	hints for matching properties and thus discovering the semantics of classes
disjoint classes	hints of class dissimilarity
comments	documentation of changes in hierarchy, etc.
version information	the record of modifications and authentication

Table 3.1: Features extracted for ontology mapping.

iterations that backtrack to different stages. We include a screenshot of the Web-based interface of CMS in figure 3.2.

3.2.3 CMS specific techniques

To fit the requirements of different applications, CMS implements a series of mapping techniques, which are regarded as independent components that made up CMS.

Name matchers

Ranging from pure syntactical approaches to more semantically enriched ones, name matchers are categorised as: String (tokenised) distance, Thesaurus, and WordNet hierarchical distance. Levenstein distance is the simplest implementation of string distance. More sophisticated ones are: *Monge-Elkan* distance which optimizes edit-distance functions with well-tuned editing cost and the *Jaro* metric and its variants which computes an accumulated similarity of s and t from the order and number of common characters between s and t. In CMS a thesaurus comes into play in two forms: WordNet and a predefined corpora that are implemented as `WNNameMatcher` and `CorpusNameMatcher`, respectively. To facilitate the use of WordNet, we assume that local names of classes are either nouns or noun

Figure 3.2: The Web-based Interface of CMS.

phrases while local names of properties are phrases starting with verbs followed by either nouns or adjectives. Elements in the retrieved synsets are then compared against each other using either exact string matching or one of the string-distance based algorithms discussed in the previous section. WordNet arranges its entries in hierarchical structures. Hence, the similarity between names can be computed as follows: let w_i and w_j be the corresponding WordNet entries of $name_i$ and $name_j$, w be the least common hypernym of w_i and w_j, r be the root of the underlying WordNet hierarchy, and h_i, h_j, h be the distances between w_i and r, w_j and r, w and r, respectively, the similarity between w_i and w_j is approximated as $2 \times h/h_i + h_j$.

Semantic matchers

in CMS, a semantic flavour is added in two different ways: structure-aware and intension-aware matchers. Structure-awareness refers to the capability of traversing class hierarchies and accumulate similarities along the sub-class (sub-property) relationships. Let c and d be two classes from source and target ontologies, c_i and d_i are their direct parents in respective ontologies, the similarity between c and d is recursively defined as $\texttt{sim}(c,d) = \alpha\texttt{sim}_{local}(c,d) + \beta\texttt{sim}(c_i,d_i)$, where α and β are arbitrary weights and $\texttt{sim}_{\text{local}}/2$ gives the local similarity with regard to c and d which can be computed using one or a combination of techniques discussed above.

Intension-awareness takes into account the definitions of classes. A class c is regarded as a tuple $\langle S, P\rangle$ where S is a set of classes of which c is a subclass and P is a set of properties having c as domain and other classes or concrete data types as range. Hence, finding the semantic similarity between $c = \langle S_c, P_c\rangle$ and $d = \langle S_d, P_d\rangle$ amounts to finding the similarity between S_c and S_d as well as P_c and P_d, i.e. $\texttt{sim}(c,d) = \alpha\texttt{sim}(S_c,S_d) + \beta\texttt{sim}_{property}(P_c,P_d)$, where α and β are arbitrary weights and $\texttt{sim}_{property}/2$ computes the property similarity. More specifically, we differentiate the following situations:

- classes with matching property names, property domains and property ranges: $\mathcal{L}_{p_c} = \mathcal{L}_{p_d}$ and $\texttt{sim}_{set}(\Delta_{p_c}, \Delta_{p_d}) \geq v$ and $\texttt{sim}_{set}(\Phi_{p_c}, \Phi_{p_d}) \geq v$ where $\texttt{sim}_{set}/2$ computes the similarity of two sets of entities and v is a predefined threshold.
- classes with matching property names and property domains but different property ranges: $\mathcal{L}_{p_c} = \mathcal{L}_{p_d}$ and $\texttt{sim}_{set}(\Delta_{p_d}, \Delta_{p_d}) \geq v$, $\texttt{sim}_{set}(\Phi_{p_c}, \Phi_{p_d}) < v$, and
- classes with matching property names but different property domains as well as ranges: $\mathcal{L}_{p_c} = \mathcal{L}_{p_d}$ and $\texttt{sim}_{set}(\Delta_{p_c}, \Delta_{p_d}) < v$ and $\texttt{sim}_{set}(\Phi_{p_c}, \Phi_{p_d}) < v$.

The first situation contributes the most to the similarity of c and d. We regard classes with matching names and exact matching properties, i.e., properties with same name, domain and range, as semantically equivalent classes.

In many cases, matching between Δ_{P_c} and Δ_{P_d} (Φ_{P_c} and Φ_{P_c}, respectively) can only be concluded after traversing several levels upwards or downwards of the class hierarchy. Although not as strong as exact matching of property domains and ranges, matching classes of Δ_{P_c} (Φ_{P_c}) to remote ancestors or descendants of classes of Δ_{P_d} (Φ_{P_d}) provides a hint on how close the different properties are, and thus how similar the two concepts c and d are. Such an idea is implemented in CMS as a `ClassDefPlusMatcher` method.

External matchers

The most distinctive feature of CMS is its capability of combining ontology/database schemata matching systems. Referred to in Figure 3.1 as external matchers, existing

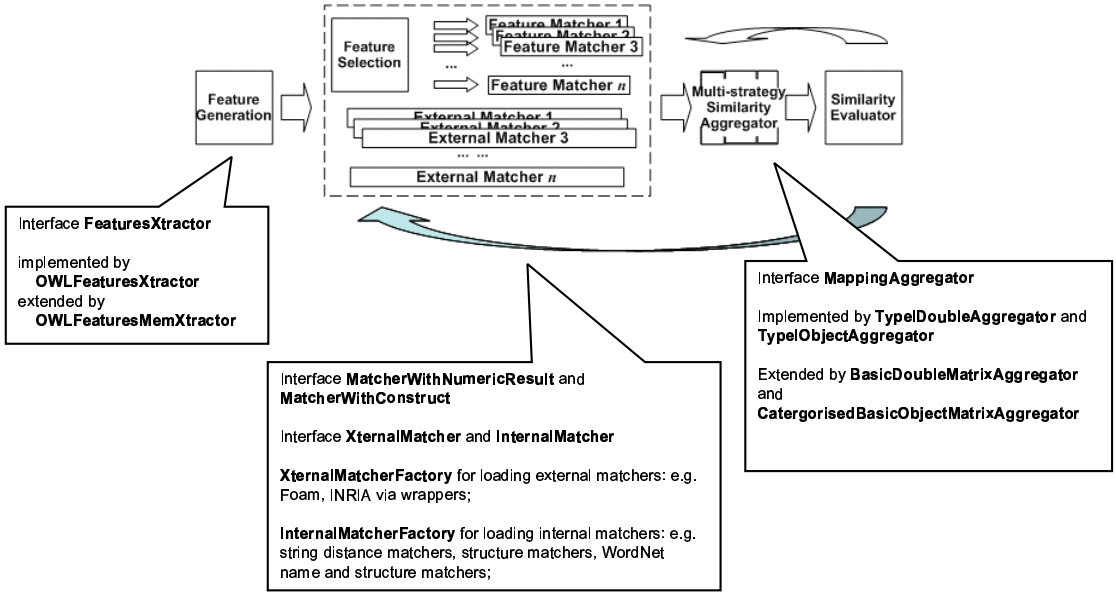
matching systems are wrapped to provide a unique interface with other modules of CMS. In the current implementation, FOAM alignment framework (FOAM hereinafter) and INRIA alignment API (INRIA, hereinafter) are invoked as external sources that matching candidates are drown upon. The reason of using FOAM and INRIA is twofold: 1) both of the systems are programmed in Java making the integration with CMS straightforward; 2) as illustrated in Figure 3.3, although based on similar algorithms, FOAM and INRIA produce results that are disparate enough to make aggregation meaningful. The integration of other ontology/database schemata matching systems is forthcoming.

Method	MIT Bibtex	UMBC Bibtex	confidence
Ont Aligner	hasEditor	editor	0.66
FOAM API	hasEditor	address	0.1396
INRIA API	hasEditor	editor	0.68
Prompt	hasEditor	editor	Map, Directly-changed
Jaro	hasEditor	editor	0.88
MongeElkan	hasEditor	editor	1.0
Ont Aligner	hasEdition	description	0.54
FOAM API	hasLanguage	description	0.1395
INRIA API	hasCrossref	description	0.2727
Prompt	hasEdition	N/A	No, Delete
Jaro	hasEdition	edition	0.90
MongeElkan	hasEdition	edition	1.0
Ont Aligner	howPublished	publishedOn	0.58
FOAM API	hasPublisher	publishedOn	0.287
INRIA API	howPublished	publishedOn	0.7826
Prompt	howPublished	publishedOn	Map, Directly-changed
Jaro	howPublished	publishedOn	0.7449
MongeElkan	howPublished	publisher	0.8888

Figure 3.3: Variant results of different mapping systems.

3.2.4 CMS architectural diagram

In figure 3.4 we depict the association of CMS main Java packages with the abstract modular architecture diagram we described in the previous chapter. We describe in full detail CMS's mechanism and internal Java structure in the accompanied manual document.



uk.ac.soton.ecs.crosi package contains: aggregator, Exception, feature, matcher, OntMatcher, util.

Figure 3.4: CMS Java packages and their association to the modular architecture.

Chapter 4

Evaluation

Evaluating software systems is a subtle task involving many unpredicted factors, ranging from technical glitches (often removed with software testing [49])to external factors related with the socio-organisational aspects of the environment that the software is meant to operate. When it comes to evaluate ontology mapping technology, these issues are, arguably, exaggerated by the intricate nature of ontologies. In addition, we aim to see ontology mapping operational and evaluated in a distributed environment like the World Wide Web, and to its extension the Semantic Web. Having in mind these issues, to evaluate the results of the CROSI project, mainly the CMS mapping system, we opted for publicly accessible, consensual test-beds, which are community-driven.

One such approach is to work on scenarios, that interested practitioners of the community improvise, that require ontology mapping. The most visible ones, are the EON and I3CON initiatives.

4.1 Mapping scenarios and test-beds

The EON alignment contest, re-structured and renamed to OAEI (Ontology Alignment Evaluation Initiative) [1] in recent years, is an initiative partly funded by the EU Networks of Excellence, OntoWeb[2] in the past, and nowadays from KnowledgeWeb[3]. The domains for the test beds vary across the years. In early years, a bibliographic references ontology was deemed enough to test the fragility of ontology mapping systems. However, criticism and uniform results from many ontology mapping system exposed the insufficiency of the domain as a test-bed for evaluating the semantic capabilities of ontology mapping systems. This led to an increase in quality of test-beds as well as in their number. So, in this year's alignment contest, participants were facing with mapping ontologies in three different scenarios: (a) the standardised bibliographic references ontologies, which is also used as a benchmark, (b) a set of Web-extracted ontologies representing fragments of the directories of popular search engines and catalogues, like yahoo, google, and looksmart, and (c) an alignment scenario in the field of life sciences, using two of the world's most complex and large ontologies: the Foundational Model of Anatomy (FMA) and openGALEN.

Bibliographic references: the domain of the first test-bed is bibliographic references. It is based on a subjective view of what must be in a bibliographic ontology. There can be many different classifications of publications (based on area, quality, etc.). The OAEI

[1] http://oaei.inrialpes.fr/
[2] http://www.ontoweb.org/
[3] http://knowledgeweb.semanticweb.org/index.html

organisers, chose one that is common among scholars based on means of publications, and it is reminiscent of BibTeX.

This reference ontology contains 33 named classes, 24 object properties, 40 data properties, 56 named individuals and 20 anonymous individuals. The complete ontology is designed as test # 101[4]. The Semantic Web flavour of this reference ontology is demonstrated by using other external resources for expressing non bibliographic information. For instance, it takes advantage of FOAF ontology[5]) and iCalendar[6] for expressing *People*, *Organisation* and *Event* concepts. The kind of proposed alignments is limited, the OAEI organisers admit: "they only match named classes and properties, and they mostly use the "=" relation with confidence value of 1.". The ontologies are described in OWL-DL and serialized in the RDF/XML format. A number of tests were systematically generated to start from the reference ontology and discard a number of information in order to evaluate how the algorithms behave when this information is lacking.

A number expected alignments in the prescribed alignment format was given to the contest participants. Each ontology was to be aligned with the reference ontology, that of test #101. The participants were also instructed that the only interesting alignments were those involving classes and properties of the given ontologies. So the alignments should not align individuals, nor entities from the external ontologies.

Web directories: the focus of this alignment task was to evaluate performance of existing alignment tools in a real world taxonomy integration scenario. The aim was to show whether ontology alignment tools can effectively be applied to integration of shallow ontologies. The evaluation dataset was extracted from Google, Yahoo and Looksmart Web directories. There are a number of characteristics which make the dataset specific: (i) more than 2000 matching tasks, where each one of them is composed from the paths to root of the nodes in the Web directories; (ii) expert mappings for all the matching tasks; (iii) simple relationships as Web directories contain only one type of relationship which is viewed as classification relation; (iv) vague terminology and modeling principles, the matching tasks incorporate the typical real world facts such as terminological errors.

The matching tasks were represented by pairs of OWL ontologies, where the classification relation was modeled by means of the OWL subClassOf construct. The matching tasks were numbered from 1 to 2265. The task was to find an alignment between classes in the ontologies. In addition, it was allowed to use background knowledge, that has not specifically been created for the alignment tasks (ie no hand-made mappings between parts of the ontologies). Admissible background knowledge were oracles such as WordNet, Cyc, UMLS, etc.

Life sciences: the focus of that task was to confront existing alignment technology with real world ontologies. The task was placed in the medical domain as this was the domain where we find large, carefully designed ontologies. The specific characteristics of the ontologies are: (a) very large models, some of the OWL models were in excess of 50 megabytes in size, (b) extensive class hierarchies as thousands of classes are organised according to different views of the domain, (c) complex relationships, classes are connected by a number of different relations, (d) stable terminology, (e) clear modelling principles that are well defined and documented.

The ontologies to be aligned were different representations of human anatomy developed independently by teams of medical experts. Both ontologies were available in OWL format and mostly contain classes and relations between them. The use of axioms was

[4]The full ontology set is accessible from: http://oaei.inrialpes.fr/2005/benchmarks/
[5]http://xmlns.com/foaf/0.1/
[6]http://www.w3.org/2002/12/cal/

limited.

The Foundational Model of Anatomy (FMA) is a medical ontology developed by the University of Washington. The OAEI organisers extracted an OWL version of the ontology from a Protege database. The model contains the following information: (i) class hierarchy, (ii) relations between classes, (iii) free text documentation and definitions, (iv) synonyms and names in different languages.

On the other hand, the other ontology to align onto, is the OpenGALEN Anatomy Model. It was developed in the OpenGalen project by the University of Manchester. The OAEI organisers created an OWL version of the ontology using the export functionality of Protege. The model contains the following information: (1) concept hierarchy, (2) relations between concepts.

The task was to find alignment between classes in the two ontologies. In order to find the alignment any information in the two models could be used. In addition, it was allowed to use background knowledge, that has not specifically been created for the alignment tasks (ie no hand-made mappings between parts of the ontologies). Admissible background knowledge are other medical terminologies such as UMLS as well as medical dictionaries and document sets.

4.2 CMS evaluation

CMS was designed and developed in the second half of the project. As such, time was limited for a thorough and long term evaluation of the system. On the other hand, CMS's characteristics and its architectural design with minimal commitments to external matchers used and a simple aggregation strategy, enabled us to do a strict evaluation of the system. We were comfortable that the system could be deployed and evaluated in a real-world scenario instead of testing it with designated, in-house, test cases. The time for conducting the real-world evaluation was perfect for us as the annual Ontology Alignment contest (OAEI 2005) had just announced it's alignment test cases, we briefly described in the previous section. We, therefore, decided that we could participate in the contest, align the published alignment test cases, and at the same time evaluate CMS. More on this year's contest results can be found in [2].

4.2.1 CMS in the OAEI 2005 contest

Applying CMS: As expected, different combinations of CMS plug-in matchers perform significantly different due to the nature of benchmark test cases. Table 2 lists the choice of matchers with regard to each test cases while table 3 shows performance values of different matchers[7] with regard to alignment of an ontology of bibliographic references (#303 in contest case 1) , in terms of precision and recall.

In figure 4.1 we include the precision and recall metric for all the alignments in the benchmark case of OAEI's contest. CMS achieved an overall precision rate of 96% and 44% rate for recall for the alignment of bibliographic ontologies. The full analysis of results and details of the settings used are described in [24].

4.2.2 Adaptations made for the contest

We didn't do any major adaptations to CMS in order to align the OAEI contest ontologies. We only did minor, routine programmatic adjustments, as for example running the CMS

[7]Results are obtained with equal weights for matchers.

CMS Matchers	Test Case #
A	103, 201, 210,
A, B	205, 206, 207, 209, 301, 303
A, C, D	225, 228, 233, 236, 239-241, 246, 247, 248-266, 302
A, B, C, D	104, 203, 204, 208, 221, 222, 223, 224, 230, 231, 232, 237, 238, 304

A–Class Definition,
B–Canonical Name,
C–WordNet Hierarchy Distance,
D–Class Hierarchy Distance

Table 4.1: CMS matchers combinations.

CMS Matchers for #303	Precision	Recall
Class Definition (A)	0.6923	0.4736
Canonical Name (B)	0.3243	0.6315
WordNet (WN) synonym	0.06	0.7894
WN Hierarchy Dis (C)	0.24	0.3157
Class Hierarchy Dis (D)	1.0	0.5263
WN synonym + hypernym (E)	0.04	0.8421
A + B	1.0	0.4736
A + E	0.9	0.4736
A + B + E	1.0	0.4736
A + B + D	1.0	0.3684
B + C + D	0.8	0.4210
B + C + D	0.8	0.4210

Table 4.2: CMS performance metrics of different matchers for test case #303.

#	Name	Prec.	Rec.	Time (s)
101	Reference alignment	N/A	N/A	N/A
102	Irrelevant ontology	N/A	N/A	108
103	Language generalization	1.0	0.788	88
104	Language restriction	1.0	0.788	159
201	No names	1.0	0.189	70
202	No names, no comments	N/A	N/A	
203	No comments	1.0	0.697	147
204	Naming conventions	1.0	0.605	153
205	Synonyms	1.0	0.230	85
206	Translation	1.0	0.255	82
207		1.0	0.264	88
208		1.0	0.473	149
209		1.0	0.103	84
210		0.818	0.246	74
221	No specialisation	1.0	0.788	129
222	Flatenned hierarchy	1.0	0.724	169
223	Expanded hierarchy	0.962	0.758	316
224	No instance	1.0	0.788	151
225	No restrictions	0.788	0.788	85
228	No properties	0.788	0.788	76
230	Flattened classes	1.0	0.760	161
231	Expanded classes	1.0	0.788	145
232		1.0	0.788	118
233		0.838	0.788	70
236		0.788	0.788	77
237		1.0	0.724	156
238		0.961	0.757	315
239		0.766	0.793	220
240		0.757	0.757	221
241		0.838	0.788	70
246		0.766	0.793	70
247		0.757	0.757	221
301	Real: BibTeX/MIT	1.0	0.363	30
302	Real: BibTeX/UMBC	1.0	0.348	31
303	Real: Karlsruhe	1.0	0.474	328
304	Real: INRIA	0.85	0.566	131

Figure 4.1: Precision and recall of OAEI test cases

system from the command line prompt in a batch mode to parse and align the hundreds of ontologies in the Web directories case or include specific Java heap size adjustment flags in order to run the system over the vast FMA ontology. Other than that, the system ran as normal.

4.2.3 Results

CMS benefits from the plug and play of modular matchers. In this contest, four different matchers were used, namely `ClassDef` for examining the domain and range of properties associated with classes, `CanoName` for accumulating similarities among class hierarchies, `WNDisSim` for computing the distance between two class names based on WordNet structures and `HierarchyDisSim` for distributing similarity among class hierarchies. The four major matchers were invoked both in parallel and sequentially. When invoked in parallel their results were then aggregated as weight average. On the other hand, when invoked in sequence, `CanoName` and `WNDisSim` give a list of corresponding classes whose similarities were then refined by `ClassDef` and `HierarchyDisSim`. CMS ran each test case with different configurations (combination and sequencing) of the aforementioned four mapping modules and precision and recall values were calculated for each run. In this report, we include the the configurations with the highest precision and recall values.

Case 1: benchmark/BibTex ontologies

For all the ontologies in this case we used a threshold of 0.8.
ontology 202: CMS fails to produce any mapping candidates with high similarity score in test case 202 due to the naming convention. We consider class names as the foundation on which other techniques can be applied (although not the sole and dominant clue for finding mapping candidates). Similarly, cases 248 to 266 also fall into this category: no candidates with high similarity value were found.
ontology 205: CMS does not achieve a high recall rate for benchmark test case 205 due to the restriction of WordNet. In case 205, class names are replaced by randomly selected synonyms. CMS relies heavily on external resources, e.g. WordNet, to provide lexical alternatives for class and property names and thus fails to respond well for synonyms that are not recognised by WordNet. A customised corpus might alleviate the problem and improve the performance with significant efforts and domain expertise.
ontology 301: In test case 301, smaller similarity scores were assigned to mapping candidates. This is due to the fact that although classes have similar names, they are defined with different properties which have different names, domains and/or ranges. It is our contention that for classes restricted with different properties, they should either not be considered as equivalent classes or their similarity value should be reduced to reflect such difference.

Case 2: Web directories ontologies

We do not have any specific comments for Case 2. All 2265 were parsed successfully by CMS and fetched for alignment. However, 29 ontologies did not produced any alignment results due to circular definitions in the original `source.owl` and `target.owl` files. So, a total of 2236 pairs of `source.owl`/`target.owl` were aligned. The system parsed them from the command line in a batch mode, and the results produced after 2 hours and 53 minutes. Each cycle involved reading and parsing the source and target ontologies, find

alignments (if any) and save and write the results in the common alignment format in a file. This was repeated 2265 times.

Case 3: Medical ontologies

This case was the most interesting. The sheer size of the input ontologies (especially that of FMA), the modelling style of OWL, the conventions used, and the complexity of the paradigm made it an interesting adventure from the research point of view. We report in more detail about our experiences in the next section.

4.2.4 General comments

Performance tuning and hardware settings: As we were facing some really large ontologies (i.e., the 72k classes FMA ontology), we had to do certain optimizations to the code and to the computer settings in order to obtain alignment results in acceptable time. We ran the tests on a stand-alone PC running Microsoft Windows XP operating system, service pack II, 2003 version. The PC had 1GB of memory installed (DDR400-SDRAM), an 80GB Serial ATA hard disk, and a Pentium 4, 3.0GHz processor. We used Java VM (version 1.5.0_04) and we had to do certain configurations to adjust the heap size in Java. For example, the standard Java heap size is 64MB. This was not enough though for the Web directory and medical ontologies case. In fact, for the medical ontologies case, the sheer size of the input ontologies (especially that of FMA) forced us to use a 768MB heap size. Settings lower than this threshold caused the system to run out of memory.

Parsing and extracting experiences: FMA owl is a 31MB .owl file comprising of 72545 declarations of owl classes and 100 relations (object and data type properties). These numbers were obtained when using our Jena 2.2 API and probably deviate slightly from other parsers. Parsing and extracting features from the FMA ontology took 9 minutes and 17 seconds with Java Heap Size adjusted to 512MB. However, in order to run the CMS and find alignments with the OpenGALEN we had to use a 768MB heap size setting. While parsing, Jena API was complaining about the syntax idioms used. For example we had a lot of warnings from Jena's RDF syntax handler, or the form "bad URI in qname XXX: no scheme found". We elaborate on the reasons behind this parsing warnings in section 4.2.4.

OpenGALEN.owl is a 4MB .owl file comprising of 24 declarations of owl classes and 30 relations (as previously, object and data type properties, and these numbers were obtained from Jena 2.2 API). Parsing and extracting features from OpenGALEN took just a few seconds. There was no need to adjust the Java heap size.

Comments on the test cases

We do not have any specific comments for test cases on BibTex and Web directories alignments. However, we found interesting the last test case, that of medical ontologies alignment, and we summarize our experiences below.

FMA.owl was a different case altogether. The ontology describes the domain of human anatomy and it aims to provide "a reference ontology in biomedical informatics for correlating different views of anatomy, aligning existing and emerging ontologies in bioinformatics" [44]. However, there are two notable facts regarding the syntactic and modelling idioms of FMA and existing results from previous efforts in trying to align FMA and GALEN. As far as the former is concerned, the OWL version we had to work with was a result of translation from Protege. Previous work has shown that this result is not always

a faithful representation of the original FMA Protege model. For instance, it has been reported that FMA DL constructs are often ill-defined and they lead to inconsistencies when a reasoner parses the ontology [16]. Consistency checking for FMA is an acknowledged problem though, even by its authors: "[...] feedback from these investigators revealed an aggregate of a few hundred errors, many of which related to spelling and only a few to cycles in the class subsumption and partonomy hierarchies." [44].

Leaving aside this fact of life (as it is natural for an ontology of this size and so close to human practice to be inconsistent), we point to a couple of syntactic idioms that we found interesting when parsing the ontology with our Jena-based CMS system. Firstly, the rather unusual use of unique frame IDs for class names (`<owl:Class rdf:ID>` constructs) and the textual description of a class in an `rdfs:label` construct. We also noticed some unusual uses of references to frame IDs. For instance, the declaration of "arterial supply" as an object property: `<owl:ObjectProperty rdf:ID="arterial_supply" rdfs:label="arterial supply">` is used in other parts of the ontology where it refers to a `rdf:resource` which points to a different resource:
`<arterial_supply rdf:resource="#frame_14586"/>`. Tracing that frame ID leads us to a definition of a "Tissue" class, and not the "arterial supply": `<owl:Class rdf:ID="frame_14586" rdfs:label="Tissue">`. The definition of an instance (with frame ID 14586) of an object property ("arterial supply") that is a class ("Tissue") could lead to modelling misunderstandings and confusion (although, syntactically speaking, it is allowed in some versions of OWL).

Going back to our argument for the notable facts, we found that previous efforts for aligning FMA to GALEN reported rather controversial results. For example, in [51], the authors employed two different alignment methods to map FMA to GALEN. Despite of the subtle differences of OpenGALEN with GALEN, the similarity of their work with that of the OAEI contest 3rd case study is high but some of their findings are questionable from the semantics point of view: for example, it was reported that "Pancreas" in FMA matches "Pancreas" in OpenGALEN with 1.0 similarity value which "indicates a perfect match" [51]. When we looked carefully at the definitions of "Pancreas" in both ontologies we saw that "Pancreas" is defined as a class in FMA (`<owl:Class rdf:ID="frame_12280" rdfs:label="Pancreas">`)
whereas in GALEN (OpenGALEN) as an instance of class "Body Cavity Anatomy"

```
<owl:Class rdf:ID="Body_Cavity_Anatomy">
<rdfs:subClassOf
rdf:resource="#OpenGALEN_Anatomy_Metaclass"/>
<Body_Cavity_Anatomy rdf:ID="Pancreas">
```

Even if OWL semantics allow to map an individual to a class (when dealing with OWL Full), such an alignment is misleading especially when we consider the high level of abstraction for the "Pancreas" class in OpenGALEN. It seems that the "lexical phase" parsing used in [51] was the main contributor to this high similarity value when relatively little structure information was taken into account. As a final comment on the case, we also point the reader to observations made by the FMA authors when trying to validate mapping results and differences in terminologies with these two ontologies: "[...]the reasons for the differences have not yet been explored, but at least some of them may be the different contexts of modelling. GALEN represents anatomy in the context of surgical procedures, whereas FMA has a strictly structural orientation." [44].

Chapter 5

Future research directions

In this chapter we identify fruitful research directions (section 5.2), but we also identify ways in which the main deliverable of the project, CMS system, can be extended and what are the lessons learnt for ontology mapping practice, in general. We also put these into perspective and speculate on their impact for future semantic interoperability projects in section 5.3.

5.1 Extending CMS

CMS could be improved in the following aspects: (i) a more sophisticated *aggregation mechanism*; (ii) a unified *alignment representation formalism*; and (iii) parameterised *algorithms for computing class hierarchy distance.*

First, results from multi-matchers are aggregated as weighted average with arbitrary weights to start with. Thus far, the weights are fine-tuned manually relying on the knowledge of the domain of discourse and the underlying algorithms of CMS. A more sophisticated approach would hire machine learning techniques to work out the most appropriate weights with regard to different matchers aiming to solve different sort of mappings. Furthermore, results from different matchers can be sorted locally first which could make accumulating results from different matchers to be reduced to ranking aggregation [**?**].

Secondly, the heterogeneous nature of different matchers – some external matchers produce pairwise equivalence with numeric values stating the similarity score while others output high level relationships, e.g. *same entity as*, *more specific than*, *more general than* and *disjoint with* expressed in high level languages such as OWL and RDF – suggests that output from different matchers has to be lifted to the same syntactical and semantic level. A unified representation formalism equipped with both numeric and abstract expressivity can facilitate the aggregation of heterogeneous matchers.

Thirdly, CMS takes into account the exact position of classes in the class hierarchy. We would like to develop algorithms that penalise mapping candidates that are found to be quite apart from each other, and then propagate their similarity values upwards and downwards in the hierarchy to their descendants and/or ancestors. There could also be pre-defined parameters that as we go up or down the hierarchy we change the similarity values of their descendants and/or ancestors accordingly. We expect that this could reduce the number of false positive results.

5.2 Guidelines for future research

The issues we highlight in this section are not restricted to specifically ontology mapping but address a wider range of issues with regard to semantic integration. All of them though, are glued together with a vision of using ontologies as the main target of integration systems. Although we believe that the technology we developed in this project is not ontology-specific and there are no known obstacles for applying it to DB schemata, we focus on ontologies as these have prevailed as the mainstream means of codifying semantics on the Semantic Web.

- **The emergence of *de-facto* multi-ontology systems:** one of the trends we experience is that we often have to underpin a system's functionality with more than one ontology. The advent of the Semantic Web made that easier to implement as more ontologies are available and accessible online than ever before. The arguments for and against using multiple ontologies are difficult to quantify as it depends on the quality and usage of the ontology in the system. For example, the use of a multiple ontologies structure in the award winning Computer Science AKTive Space application [45] made a difference when dealing with large, heterogeneous data sets extracted from a variety of online resources. These were only made possible to integrate by integrating multiple ontologies describing their semantics. That dictates that we should have appropriate ontology integration and mapping technology available in order to compute alignments between heterogeneous ontologies seamlessly on the Semantic Web.

- **Semantic Web related issues:** the advent and increasing popularity of the Semantic Web poses new challenges for ontology mapping practice. If we accept as a common practice that most ontologies that need to be aligned will be sourced off the Semantic Web, then the following issues need to be accounted for: (a) authority and version control; (b) trust and provenance; (c) inconsistency and incompleteness. All these issues are generic enough and affect the Semantic Web as a whole, hence, ontology mapping systems that operate in this environment should have the appropriate mechanisms to cope with these issues.

- **Semantically rich alignment systems:** a re-occurring theme from the past found new ground in the Semantic Web realm. Semantics aim at revealing and using the meaning of concept across applications to achieve semantically interoperable systems. But, the bulk of the work done in interoperability, in general, uses syntax only. The crux of the problem is that semantics are often not explicitly stated in artefacts but rather tacitly exist in a designers mind. Although ontologies carry the promise that this should not be the case, we still witness alignment systems and algorithms that are heavily inclined to use syntactical features of the targeted ontologies. We need to move more into semantically-rich alignment systems that will exploit, to the maximum degree possible, the presence (if any) of semantically rich constructs found in ontologies.

- **Community-driven systems:** This unique characteristic and a increasing trend on the Web and the Semantic Web, is the engagement of communities in various tasks traditionally associated with engineers. Ontology mapping is no exception (see, for example the early work proposed in [52]) and we believe that this direction could be proved fruitful in the near future. However, the actual role that communities could play in ontology mapping and at which stage they should be involved is not yet clear.

Early efforts showed that communities could validate and verify the correctness of mapping results but more principled work needs to be done to assess the practicality of this direction.

- **Ontology mapping: not an ad-hoc activity:** lastly, but not least, we see that the majority of ontology mapping systems are seen and applied as an ad-hoc component. That is, when ontologies are designed and developed, no action is taken to enrich the ontology with designated ontology mapping constructs (like, for example, the OWL sameAs constructs) or no attention is taken on the potential alignment needs that could arise in the future. Therefore, most publishable ontologies on the Semantic Web are operating as stand-alone artefacts to which is difficult to align other ontologies. We believe that ontology designers should include ontology mapping constructs and guidance as to the integration and interoperability needs they envisage for their ontologies. Then, ontology mapping practitioners could fine-tune their algorithms in order to perform more accurate, efficient and effective ontology alignment.

5.3 On the Future of Semantic Interoperability

Lastly, we speculate on general directions for the whole of interoperability as put forward by major international bodies and industrially driven projects.

The European Commission has put forward an initiative, i2010[1] to boost the digital economy and improve Europe's competitiveness in this vital sector.

According to David White, Director, European Commission, Enterprize and Industry Directorate General:

> The realisation of i2010 goals will very much depend on platforms, services and applications being able to talk to one another and to build an economic activity on the information retrieved. This is what we understand as interoperability. It is complex, not limited to the infrastructure level but encompasses semantic interoperability, organisational interoperability, and even regulatory interoperability.

One could say, that CROSI examined the role of ontology mapping technology to address one of the issues highlighted in the aforementioned quote: semantic interoperability. However, as we saw in chapter 2 ontology mapping is only one of the many technologies that practitioners use to tackle the problem of semantic heterogeneity. A holistic semantic interoperability approach, should ultimately take into consideration all these technologies and provide the best possible links with legacy technology in order to maximize productivity and speed-up utility and adoption by its users. We explored what we see as a first step towards this high level goal: the provision of a modular architecture that different technologies can be plugged in to provide additional functionality.

This is also in line with the initial findings of two major European RTD (Research and Technological Development) projects in this area with heavy industrial participation. For example, in the EU funded Network of Excellence INTEROP (Interoperability Research for Networked Enterprises Applications and Software) [2], issues regarding the use of formal approaches to interoperability are raised ([8], p.167). Furthermore, the EU ATHENA (Advanced Technologies for interoperability of Heterogeneous Enterprise Networks and

[1] http://europa.eu.int/i2010

[2] http://www.interop-noe.org/

their Applications) Integrated Project report DA1.2.1 ([15], p.18-19), exposes deficiencies of OWL formulations of knowledge.

Finally, the call for organisational and even regulatory interoperability that White has put forth, is still in its infancy. Despite a relatively large body of knowledge regarding applications of this technology to organisational knowledge (see, for example, the collection on applications of AI and Semantic Web technology to Organisational Memories - [9]) we still have little experiences and even fewer success stories of how to bring all those three – semantic, organisational, and regulatory – under the same roof.

Bibliography

[1] H. Alani, S. Dasmahapatra, N. Gibbins, H. Glasser, S. Harris, Y. Kalfoglou, K. O'Hara, and N. Shadbolt. Managing reference: Ensuring referential integrity of ontologies for the semantic web. In *Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW'02), Siguenza, Spain*, pages 317–334, Oct. 2002.

[2] B. Ashpole, M. Ehrig, J. Euzenat, and H. Stuckenschmidt, editors. *Integrating Ontologies'05*, volume 156 of *WS*. CEUR, Oct 2005.

[3] J. Barwise and J. Seligman. *Information Flow: the Logic of distributed systems*. Cambridge Tracts in Theoretical Computer Science 44. Cambridge University Press, 1997. ISBN: 0-521-58386-1.

[4] M. Benerecetti and S. Bouquet, P.and Zanobini. Soundness of semantic methods for schema matching. In *Proceedings of the ISWC'04 Meaning, Coordination and Negotiation (MCN'04) Workshop, Hiroshima, Japan*, pages 13–24, Nov. 2004.

[5] S. Bergamaschi, S. Castano, and M. Vincini. Semantic integration of semistructured and structured data sources. *SIGMOD Record*, 28(1):54–59, 1999.

[6] P. Borst, H. Akkermans, and J. Top. Engineering Ontologies. In *Proceedings of the 10th Knowledge Acquisition for Knowledge Based Systems Workshop,Banff, Canada*, 1996.

[7] P. Bouquet, B. Magnini, L. Scrafini, and S. Zanobini. A sat-based algorithm for context matching. In *Proceedings of the 4th International and Interdisciplinary Conference on Modeling and Using Context (Context03)*, 2003.

[8] CRF. Enterprise modelling in the context of collaborative enterprises. IP Deliverable D.A1.2, ATHENA EU IP - FP6 507849, mar 2005.

[9] R. Dieng-Kuntz and N. Matta, editors. *Knowledge Management and Organizational Memories*. Kluwer Academic Publishers, 2002.

[10] A. Doan and A. Halevy. Semantic integration research in the database community: A bried survey. *AI Magazine*, 26(1):83–94, 2005.

[11] 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.

[12] M. Ehrig and S. Staab. Qom - quick ontology mapping. In *Proceedings of the 3rd International Semantic Web Confernece (ISWC'04), LNCS 3298, Hiroshima, Japan*, pages 683–697, Nov. 2004.

[13] 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.

[14] B. Ganter and R. Wille. *Formal Concept Analysis: mathematical foundations*. Springer, 1999. ISBN: 3-540-62771-5.

[15] A. Garcia-Diez. State of the art in enterprise modelling techniques and technologies to support enterprise interoperability. IP Deliverable D.A1.1.1, ATHENA EU IP - FP6 507849, jul 2004.

[16] C. Golbreich, S. Zhang, and O. Bodenreider. Migrating the fma from protege to owl. Technical report, jul 2005. In notes of the 8th International Protege Conference, Madrid, Spain.

[17] T. Gruber. Towards principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies*, 43:907–928, 1995.

[18] F. Guinchiglia, P. Shvaiko, and M. Yatskevich. S-match: an algorithm and an implementation of semantic mathing. In *Proceedings of 1st European Semantic Web Symposium (ESWS'04), Crete, Greece*, pages 61–75, May 2004.

[19] D. H-H and E. Rahm. COMA: a system for flexible combination of schema matching approaches. In *Proceedings of the 28th International Conference on Very Large Databases (VLDB'02), Hong Kong, China*, aug 2002.

[20] B. He and K. Chang. Statistical schema matching across web query interfaces. In *SIGMOD Conf.*, pages 217–228, 2003.

[21] I. Horrocks, P. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. *SWRL: A Semantic Web Rule Language Combining OWL and RuleML*. W3C, May 2004.

[22] W. Hu, N. Jian, Y. Qu, and Y. Wang. Gmo: a graph matching for ontologies. In *in Proceedings of the K-Cap'05 Workshop on Integrating Ontologies, Banff, Canada*, pages 41–48, oct 2005.

[23] Y. Kalfoglou, H. Alani, M. Schorlemmer, and C. Walton. On the emergent semantic web and overlooked issues. In *Proceedings of the 3rd International Semantic Web Confernece (ISWC'04), LNCS 3298, Hiroshima, Japan*, pages 576–591, Nov. 2004.

[24] Y. Kalfoglou and B. Hu. Cms: Crosi mapping system - results of the 2005 ontology alignment contest. In *Proceedings of the K-Cap'05 Integrating Ontologies workshop, Banff, Canada*, pages 77–84, Oct. 2005.

[25] Y. Kalfoglou, B. Hu, and D. Reynolds. On interoperability of ontologies for web-based educational systems. In *Proceedings of the 14th International World Wide Web Conference (WWW 2005) workshop on Interoperability of Web-based Educational Systems (WF05), Chiba, Japan*, May 2005.

[26] Y. Kalfoglou, B. Hu, D. Reynolds, and N. Shadbolt. Semantic integration technologies. 6th month deliverable, University of Southampton and HP Labs, April 2005.

[27] Y. Kalfoglou and M. Schorlemmer. If-map: an ontology mapping method based on information flow theory. *Journal on Data Semantics*, 1:98–127, Oct. 2003. LNCS2800, Springer, ISBN: 3-540-20407-5.

[28] Y. Kalfoglou and M. Schorlemmer. Ontology mapping: the state of the art. *The Knowledge Engineering Review*, 18(1):1–31, 2003.

[29] Y. Kalfoglou, M. Schorlemmer, M. Uschold, A. Sheth, and S. Staab. Semantic interoperability and integration. Seminar 04391 - executive summary, Schloss Dagstuhl - International Conference and Research Centre, Sept. 2004.

[30] P. Lamrix and H. Tan. A framework for aligning ontologies. In *Proceedings of 3rd Workshop and Principles and Practice of Semantic Web Reasoning (PPSWR), Dagstuhl, Germany*, volume 3703 of *LNCS*, pages 17–31, sept 2005.

[31] A. Levy. The information manifold approach to data integration. *IEEE Intelligent Systems*, 13:12–16, 1998.

[32] J. Madhavan, P. Bernstein, C. Kuang, A. Halevy, and P. Shenoy. Corpus-based schema matching. In *Proceedings of the IJCAI'03 Workshop on Information Integration on the Web (IIWeb-03), Acapulco, Mexico*, Aug. 2003.

[33] J. Madhavan, P. Bernstein, and E. Rahm. Generic schema matching with cupid. In *Proceedings of the 27th International Conference on Very Large Databases (VLDB'01), Roma, Italy*, Sept. 2001.

[34] H. Meisel and E. Compatangelo. Conceptool: Intelligent support to knowledge management. In *Proceedings of the 9th Workshop on Automated Reasoning (ARW'02), Imperial College, London, England*, Apr. 2002.

[35] S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity flooding: A versatile graph matching algorithm and its application to schema matching. In *Porceedings of the 18th International Conference on Data Engineering (ICDE)*, pages 117–128, 2002.

[36] T. Milo and S. Zohar. Using schema matching to simplify heterogeneous data translation. In *Proceedings of the 24rd International Conference on Very Large Data Bases (VLDB'98), New York, NY, USA, USA*, pages 122–133, Aug. 1998.

[37] P. Mitra, N. Noy, and A. Jaiswal. Omen: A probabilistic ontology-mapping tool. In *Proceedings of the ISWC'04 Workshop on Meaning Coordination and Negotiation (MCN'04), Hiroshima, Japan*, Nov. 2004.

[38] 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.

[39] 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*, pages 744–751, Aug. 2002.

[40] N. Noy. Semantic integration: A survey of ontology-based approaches. *SIGMOD Record*, 33(4):65–70, 2004.

[41] N. Noy and M. Klein. Ontology evolution: Not the same as schema evolution. *Knowledge and Information Systems*, 6(4):428–440, 2004.

[42] L. Palopoli, G. Terracina, and D. Ursino. Dike: a system supporting the semiautomatic construction of cooperative information systems from heterogeneous databases. *Softw. Pract. Exper.*, 33(9):847–884, 2003.

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

[44] C. Rosse and J. Mejino. A reference ontology for bioinformatics: The foundational model of anatomy. *Journal of Biomedical Informatics*, 36:478–500, 2003.

[45] N. Shadbolt, N. Gibbins, H. Glaser, S. Harris, and M. Schraefel. Cs aktive space or how we learned to stop worrying and love the semantic web. *IEEE Intelligent Systems*, 19(3):41–47, May 2004.

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

[47] P. Shvaiko. A classification of schema-based matching approaches. In *Proceedings of ISWC'04 workshop on Meaning, Negotiation and Coordination (MCN'04), Hiroshima, Japan*, Nov. 2004.

[48] 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.

[49] J. Voas, G. McGraw, L. Kassab, and L. Voas. A "Crystal Ball" for Software Liability. *IEEE Computer*, 30(6):29–35, June 1997.

[50] J. Wang, K. Zhang, K. Jeong, and D. Shasha. A system for approximate tree matching. *IEEE Transactions on Knowledge and Data Engineering*, 6(4):559–571, 1994.

[51] S. Zhang, P. Mork, and O. Bodenreider. Lessons learned from aligning two representations of anatomy. In *in Proceedings of the KR 2004 Workshop on Formal Biomedical Knowledge Representation, Whistler, BC, Canada*, pages 102–108, 2004.

[52] A. Zhdanova. Towards community-driven ontology matching. In *Proceedings of the 3rd International Conference on Knowledge Capture (K-CAP'05), Banff, Canada*, pages 221–222, oct 2005.