Reconciling Implicit and Evolving Ontologies for Semantic Interoperability

Kendall Lister Maia Hristozova Leon Sterling

Intelligent Agent Lab
Department of Computer Science and Software Engineering
The University of Melbourne, Victoria, 3010, Australia
{krl,majah,leon}@cs.mu.oz.au

Abstract
The purpose of this paper is to discuss the current partial solutions to the goal of semantic interoperability on the web. It is obvious that a general solution to the problem on a global level has not yet been achieved. We critically discuss the existing approaches, including technologies such as RDF, SHOE, PROMPT, Chimaera and others, trying to identify the most effective elements of each and noting that since they are mainly closed solutions their ability to succeed on a global web scale is limited. We then review and contrast our own research experiments AReXS and CASA and suggest that as yet unaddressed issues should be considered, such as reconciling implicit ontologies and evolving ontologies. While these ideas may not directly solve the problems of semantic interoperability, exploring them will contribute significantly to the effort.

Keywords
Ontology translation/mapping, ontology maintenance/evolution, classification/comparison of approaches, practical experience, standardisation issues

1. Introduction
The much talked about goal of building a new Internet that is comprehensible to machines as well as humans is generally considered to involve enhancing content and information sources with semantic markings and explicit ontologies. A number of approaches to this goal have been proposed, and these generally involve a new representation for semantically enriched data. Something that seems to be often overlooked, however, is that a single solution is unlikely to be usefully applicable to the entire World Wide Web. It is obvious that business needs are generally quite different to the needs of individuals, and that even within the business community different areas will require solutions of varying sophistication, accuracy and scale.

The widespread success of the World Wide Web and its underlying technologies, HTML and HTTP, has been due in no small part to their simplicity and ease of adoption. By providing a simple architecture that anyone could learn and use with minimal overhead, content flourished on the web. Other information technologies that arguably provided more effective methods for locating and retrieving data simply failed to take off in the same exponential way that the web did. Where the web infrastructure itself doesn't even contain the most rudimentary searching and resource location features, Gopher, WAIS and a large number of proprietary on-line databases that predated the World Wide Web all provided automated indexing, searching, hypertextuality and other information management capabilities. But despite their apparent advantages, all of these technologies were overtaken by the web. In fact, in many cases proprietary databases and indexes have had their interfaces replaced with web-based solutions, to the point that the actual technology is largely hidden. It is more than a coincidence that where the World Wide Web succeeded and grew to become a de facto standard, the more complex alternatives faltered and missed out on popular adoption.

Similarly, we consider that the next generation of semantically-capable global information infrastructure will necessarily be relatively simple in order to achieve the same scale of acceptance. That is not to say that sophisticated technologies have no place – on the contrary, they will be vital for the areas of industry that require them, and their advances will no doubt drive other research efforts even further. Also, the intelligent agents that roam this infrastructure will also be very sophisticated. However, there remains a significant place for simple, flexible and adaptive technologies that do not demand strict adherence to formal standards and protocols and the development and publishing costs that follow. By leaving the majority of the intelligence for semantic comprehension in the interpreting applications rather than the medium itself, we will develop technologies that can operate in any information environment, not just those that are sophisticated and semantically enhanced. There is no suggestion that semantically rich environments are not useful and desirable – but that it is not practical to expect the entirety, or even the majority, of the information landscape of the future to be uniformly structured, as current research seems to imagine.

2. Current projects toward a semantic web
Discussions of the problems of semantic operability on the web have a tendency to become discussions of the problem of managing and integrating ontologies. The reasons for this are not obscure: ontologies are widely regarded as a critical element of the next generation of data integration solutions, and the World Wide Web is a heterogeneous environment in which foreign data (and therefore ontologies) are regularly juxtaposed. What is less clear is how such data can be combined. A number of new technologies have been proposed that extend or replace existing web technologies; prominent among these are RDF, SHOE, PROMPT and Chimaera. However, these tools and techniques either require adoption of a specific standard for ontology representation (RDF [3], SHOE [4]) or are only semi-automated reconciliation solutions (PROMPT [10], Chimaera [9]). In fact, even the ontology representation standards such as RDF and

SHOE appear to require manual construction of the new intersection ontologies that provide mappings between different ontologies.

Each of these tools and technologies has been well described by their authors and developers, and the purpose of this paper is not to repeat the existing descriptions. Each development attempts to overcome the heterogeneity of the web, but they all suffer common problems. For example, although Chimaera successfully addresses the issue of managing different ontologies by reconciling them, it is still a tool that requires manual manipulation and human decision-making. Similarly, PROMPT also is only semi-automatic.

RDF provides a transportable way of expressing information, but the schema itself is separated from the information. Thus maintenance is difficult, as if the data changes the schema must be changed as well to maintain consistency. This problem will be faced by any implementation that uses an explicit representation of ontology. RDF appears to provide a useful medium for expressing data and meta-data, and further layers such as DAML+OIL increase the opportunities for reasoning about the data contained in a document based on its accompanying meta-data. But each ontology must still be constructed either in isolation from other ontologies (in which case reconciling information sources faces all the problems discussed in the introduction to this paper), or the ontology developer must find and look inside external ontologies to choose which to extend. Since this implies that the other ontologies are well-formed and available, there is hardly a need for reconciliation – this paper is concerned with situations in which two (or more) existing information sources are to be brought together. It is not clear that RDF or DAML+OIL can assist with this task.

The biggest disadvantage of SHOE is lack of central management. Because of its flexibility, many different users can create new or extend existing ontologies to annotate the data, which makes it difficult for agents and humans to query this data. As mentioned regarding RDF and DAML+OIL, agents must be aware of the many existing ontologies, and SHOE still does not solve the problem of automatically reconciling diversity. Thus results from a query may remain incomplete or mismatched because the agent is not able to find all the relevant data. Further, SHOE does not yet support a wide range of ontology formats.

3. Reconciling implicit ontologies

Most data on the Internet today appears without any explicit ontology. To integrate data that has no accompanying explicit ontological representation requires either that formal ontologies be constructed for each data source, manually or automatically, or that the conceptual and semantic correspondences between elements in the data be recognised or deduced directly, without resorting to an explicit representation of the ontology. The former process at first seems to be the more reasonable, as it mirrors the intuitive process a person would be likely to define if asked to plan the task. However, we suggest that the latter process is in fact closer to the actual approach a person would take when given two data sources and ask to reconcile them. Furthermore, the first method introduces several of the most troublesome ontology management issues, namely constructing accurate and usable ontologies, choosing a representation, and then aligning different ontologies. If the two ontologies are developed together, some of the difficulties of the development can be stepped over as the engineer juggles and reconciles concepts and relationships as they go, but such a synergy certainly cannot scale far beyond two data sources at a time – in reality it is often desirable to compare and contrast data from multiple sources, such as a variety of on-line book stores. Also, one objective of our research into ontological reconciliation is to automate the process as much as possible so that any solutions are eventually globally deployable.

Another important benefit of an automated, lightweight approach to ontological reconciliation is that it makes whatever technology is deployed very adaptable to changes in the data environment. If an intelligent agent is tasked with retrieving prices of books from three major on-line book stores, traditional ontology development and management approaches require that an engineer assess the data sources and construct mapping ontologies between them. If the companies publishing their stock data does not supply a well-formatted ontology along with the shopping data, the engineer also has to construct three individual ontologies before any mapping can even be considered. If a fourth source of on-line books becomes available, the engineer is required again to construct either another mapping ontology to align the new source and the existing mapping ontologies, or the process must begin again from scratch. Of course, if the intelligent agent has its own ontology for the domain of books sales (which is likely, if it has been designed to search and report on data of this type), it is only necessary for the engineer to construct maps between this ontology and each data source. But each time one of the companies changes their data representation the engineer is again required to manually intervene, unless the company provides sufficient hooks in their ontology for backward compatibility. In the low margin world of on-line commerce, this is hardly likely to be considered a cost-effective effort even though technologies such as SHOE deliberately support this [4]. An automated solution is obviously preferable to one that requires human supervision, and we suggest that in most end-user applications, the required accuracy is generally not high enough to demand heavyweight tools and processes. Additionally, a well-designed user interface could allow the user to touch up the results of the automatic reconciliation on-the-fly, thus harnessing the intelligence of the user for effectively no cost.

4. Practical reconciliation

The remainder of this paper discusses two recent projects that have produced promising results for alternative approaches to ontological reconciliation. Much of this analysis was originally published in [7] but has been updated here to illustrate the practical possibilities for implementing the ideas presented in the first part of this paper. The projects described here are a progression from earlier related work – [13, 14] give some background details.

4.1 AReXS

AReXS (Automatic Reconciliation of XML Structures) is an application that reconciles heterogenous data sources presented in XML documents. It aligns data sources according to their implicit ontological structure. It is able to reconcile differences of expression and representation across XML documents from heterogeneous sources without any predefined knowledge or human intervention [5]. It achieves this by identifying XML elements whose meanings are similar enough to be considered equivalent. AReXS requires no knowledge or experience of the

domain in which it works, and indeed is completely domain independent. It uses Example-Based Frame Matching (EBFM) [6] and is able to achieve very high recall with modest precision on real world data collected from commercial web sites. By requiring no domain knowledge, AReXS is suitable for application to any field; its success relies on its ability to identify and resolve the differences in representation that result from sourcing data from a multi-cultural environment.

AReXS does not fit into the traditional mould of an ontology management tool, as it does not use any type of formal ontology representation. Instead, it more closely mimics the intuitive process that a person is likely to follow when tasked with aligning multiple structured data sources. In formal terms, AReXS attempts to automatically resolve the problems of synonymy and polysemy that are significant hurdles to semantic interoperability [3,7].

For example, a pair of XML documents from different sources, both describing services offered by universities, might contain elements named `SUBJECT` and `UNIT` respectively. If the two elements happen to both signify self-contained units of course work, an agent with no prior domain experience or knowledge will have little hope of realising this. AReXS resolves this discontinuity by considering the values of instances of the elements as well as the element names, deriving confidence in a match from similarities in either comparison. If one document contains the statement `<SUBJECT>Introductory Programming</SUBJECT>` and another contains a similar statement `<UNIT>Introduction to Programming </UNIT>`, AReXS is able to consider the possibility that the two elements `SUBJECT` and `UNIT` are in this context signifying the same concept. If further correspondences could be found between other instances of these same elements, the confidence of a conceptual match would increase.

AReXS works by analysing two XML structures and identifying matching elements, generating a map of equivalence between concepts represented in the two documents. It is important to note that no formal representation of such concepts is attempted – rather, it is assumed that elements represent concepts, and that the equivalence of two elements can be deduced based on similarities between instances of both elements. As a metaphor, AReXS works in much the same way that two people who do not share a common language might teach each other by pointing at objects and saying the names that each person's language gives to that object.

Identification of conceptual equivalence is based on a consideration of lexicographical similarity between both the names and the contents of XML tags in each document. Matches are then assessed to deduce structural similarities between documents from different sources. By repeating this search for semiotic correspondence across other pairs of elements generated from the contents of the XML documents under consideration, AReXS is able to build a local context for data and then use this context to reconcile the ontological differences between XML documents.

To establish the extent of the context shared by pairs of documents, the AReXS engine uses the Character-Based Best Match algorithm [11] to evaluate textual similarity between the names and contents of elements. Such a string based comparison works well to filter out simple manifestations of local cultures; for example, one university web site may choose to include the identification number of a subject in the name of the subject while another may not, opting instead to have a second element containing a numeric identification code for each unit. While AReXS will not be able to realise that the number in the name of a subject from one university corresponds to the numeric unit code from another, it will generally conclude from the similarity of the names that *units* and *subjects* are conceptually compatible in this context.

Applying a textual similarity analysis on real data is likely to generate a large number of candidate concepts that may or may not contribute to the local context of the data. AReXS increases its confidence in a candidate for equivalence depending on the uniqueness of the matches between element pairs. The uniqueness function described by [6] is used to establish the likelihood of a textual match between elements actually revealing a shared, unique concept, based on the principle that the more common a concept is across significantly different elements, the less rich the concept is and thus the less there is to be gained from considering it as part of the data context.

The results of tests based on sample real world data from web sites including amazon.com, angusandrobertson.com.au, barnesandnoble.com and borders.com show that AReXS is capable of accurately identifying conceptually equivalent elements based on both the element names and sample instances of the elements. These web sites were chosen as useful examples for two reasons. Firstly, they are live, international representatives of the types of data source with which people desire to interact (and in fact already do interact) on a regular but casual basis, and secondly they provide data that by its nature is open to subjective decisions during the process of choosing a logical representation. The casual nature of the interaction that people generally have with sites such as these is important, as discussed earlier in this paper. Since AReXS has been tailored to process XML formatted input, the raw data from the sampled web sites was encoded into XML by hand. Although much care was taken not to add any information or structure to the data that might spoil the experiments, extracting relatively free data from web sites and converting it to a structured form is not the focus of the AReXS prototype – please refer to the next section on CASA to see how this extraction can be automated.

The AReXS algorithms allow identification of concept matches regardless of the ordering of concepts or elements, and its consideration of both names and values of elements allows it to identify equivalences even if one of the name or the value is absent (for example, `<>Stephen Hawking</>` ⇔ `<AUTHOR>Stephen Hawking</AUTHOR>`); in other words, AReXS is tolerant of inconsistent data. An element name might well be missing if the XML data has been automatically generated from another source, as happened during the construction of the test input for these experiments – it was not possible to identify from the sample web sites exactly what the intended name of a particular element was, so rather than make one up, the element tag name was left blank. Admittedly, this left the XML documents malformed, but substituting dummy element names would solve this and would actually allow AReXS to cope with more than one missing element name in each data source. The AReXS engine has also demonstrated partial success in identifying many-to-one conceptual equivalences, which can occur in situations like that described earlier in which multiple concepts are represented by

multiple elements in one data source but only one element in the other data source.

Although AReXS only supports reconciling pairs of data sources, the EBFM algorithm on which it is based provides for comparison of multiple sources and so extending AReXS to support this feature is feasible. While AReXS is partially able to recognise many-to-one equivalences, it will require further work to actually capitalise on this recognition. Finally, the principles implemented in AReXS could quite readily be adapted to allow the extension of data structures based on identification of concept matches within element names or values. Drawing on the example described earlier of university service descriptions, if one institution chose to present teaching units with an element of the form `<UNIT>Machine Vision (Semester 1)</UNIT>` and a second institution opts for two elements `<SUBJECT>Machine Vision</SUBJECT>` and `<SEMESTER>1</SEMESTER>`, it is possible to see that a software agent could use analysis techniques similar to those implemented in AReXS to realise that both elements from the second source are encoded within a single element of the first source.

4.2 CASA

Classified Advertisement Search Agent (CASA) is an information agent that searches on-line advertisements to assist users in finding a range of information including rental properties and used cars [2]. It was built as a prototype to evaluate the principle of increasing the effectiveness and flexibility of information agents while reducing their development cost by separating their knowledge from their architecture, and discriminating between different classes of knowledge in order to maximise the reusability of constructed knowledge bases. CASA is able to learn how to interpret new HTML documents, by recognising and understanding both the content of the documents and their structure. It also represents a framework for building knowledge-based information agents that are able to assimilate new knowledge easily, without requiring re-implementation or redundant development of the core agent infrastructure.

CASA classifies knowledge into three categories: general knowledge, domain-specific knowledge and site or source specific knowledge. Each category is independent from the others, and multiple instances of each category can exist. This segregation of knowledge by practical use is markedly different to the usual approach of capturing and representing all high-level knowledge in a formal hierarchy or graph and ignoring low-level knowledge. It provide more seamless integration of different types of knowledge, as well as discriminating between knowledge that is likely to be common across heterogeneous data sources and knowledge that is likely to change. This increases the effectiveness of an information agent equipped with such knowledge, as it does not approach a new data source empty handed, but armed with the ability to make assumptions and deduce correspondences between the new data and sources with which it is familiar.

General knowledge gives a software agent enough information to understand and operate in its environment. General knowledge is knowledge that is true for all information sources, and is independent of specific domains and sites. The set of general knowledge developed for CASA describes on-line web documents, and includes knowledge of the components that make up an HTML document such as what are tables, paragraphs and lines, as well as knowledge of what a web page is and how one can be accessed.

Domain-specific knowledge provides an information agent with a basic understanding of the area in which is required to work. This knowledge is true for a particular field and is independent of site or source specifics. For the case of university services, domain knowledge would generally include the concepts of students, lectures, theatres, semesters, professors and subjects, as well as ontological relationships such as the idea that students take classes, classes cover particular topics and occur at certain times during the week at certain locations, and that particular subjects make up a course. Because domain knowledge is independent of site-specific knowledge, it can be re-used across numerous sites and should remain useful into the future.

Site-specific knowledge is true for a particular information source only. Site knowledge is specific and unique, but necessary for negotiating the contents of a particular information source; it provides a means of understanding the basic data that comprise an information source, for a particular representation. Continuing the university web site example, site-specific knowledge might encode the particular pattern or format in which a certain institution presents a description of a unit of teaching, or of a degree, including information such as table structures, knowledge unit sequences and marker text that locates certain classes of information.

The three categories of knowledge that CASA manages provide different levels of operational assistance for the information agent. General knowledge enables an agent to act and interact in a particular environment, providing the basis for navigation and perception and giving the agent a means by which to internalise its input. Site-specific knowledge permits an agent to assimilate and process information from a particular source, which is a necessary ability if the agent is to perform useful tasks. Domain-specific knowledge sits between general and site-specific knowledge, giving a conceptual framework through which an agent can reconcile information from different sources. Domain-specific knowledge can also assist an agent to negotiate unfamiliar information sources for which it has no site-specific knowledge. Domain knowledge can be used in conjunction with general knowledge to analyse a site's conventions and representations and to attempt to synthesise the site knowledge necessary to utilise the new information source. Because domain knowledge is not tied to a particular representation, it can be adapted and applied to a variety of different sites or data sources, significantly reducing development time for information agents.

A significant benefit of classifying knowledge into categories is that knowledge can be more readily reused and incorporated into other agents. Compartmentalising knowledge also allows agents to teach each other about new information sources or even new knowledge domains. Domain knowledge is reusable by design, and general knowledge is similarly useful. Given the modular approach to information agent construction presented in CASA, once an agent has been taught about a certain domain of knowledge, that knowledge can be applied to a variety of environments just as easily as it can a variety of sites. By plugging in a different general knowledge base, a web-based information agent could easily become an SQL- or XML-based information agent, with the cost of redevelopment greatly reduced by the re-applicability of the domain knowledge base. It also seems quite

feasible for an information agent to be armed with a variety of general knowledge bases permitting it to work in multiple environments as appropriate, or even at the same time, utilising its knowledge as applicable both to process recognised information and to interpret and negotiate unfamiliar conceptual representations.

5. Conclusion

If one of the technologies described in this paper emerged as a unilateral favourite for knowledge representation and data integration, the Internet would quickly cease to be that unstructured wilderness that so many paper introductions claim it to be. Unfortunately, it seems unlikely that any single proposed solution will be widely accepted in the near future. Even if such an event occurred, it is doubtful that many smaller commercial and individual publishers of information would be willing to devote the time and effort required to comply with a standard that requires ontology development. If ontologies developed by leading academics require significant effort to be combined, aligned or otherwise reconciled, as they currently do even with the aid of computerized ontology management tools, how much more the millions of ontologies that would be thrown together by people who just want to get their in-formation on to the web? Ontology engineering is a complicated activity that, while it is clearly important and will definitely play a major role in some areas of information integration, seems likely to always bring overheads that make it unattractive to many publishers, particularly those who move in the global, heterogeneous public space of the Internet. We are proposing and developing technologies and methodologies that cope with heterogeneity and change in information sources by performing implicit ontological reconciliation.

However, it would be a shame to ignore the much work being done to provide information sources with meta-data in the form of explicit ontologies and schema. One important direction for the future development of the tools and algorithms described in this paper is the combination of systems that consider implicit ontologies with ones that understand explicit meta-data. It is imagined that, for example, the uninformed reconciliation done by AReXS could be significantly enhanced by using the knowledge contained in any supplied ontology, schema or DTD as an advantageous starting point for the reconciliation effort.

6. Acknowledgements

This paper was supported by a University of Melbourne Research Development Grant. Thanks to the members of the Intelligent Agent Laboratory at the University of Melbourne for many useful discussions.

7. References

[1] Decker S., Erdman M., Fensel D., Studer R. Ontobroker: Ontology based Access to Distributed and Semi-Structured Information. Kluwer Academic Publishers, Netherlands 1998.

[2] Gao, X., and Sterling, L. Classified Advertisement Search Agent (CASA): A Knowledge-Based Information Agent for Searching Semi-Structured Text. Department of Computer Science and Software Engineering, The University of Melbourne, Technical Report 98/1, 1998.

[3] Hou, D. Automatic Reconciliation of XML Structures. Honours thesis, Department of Computer Science and Software Engineering, The University of Melbourne, 2001.

[4] Ikeda, Y., Itoh, F., and Ueda, T. Example-based frame mapping for heterogeneous information agents. In Proceedings of the International Conference on Multi-Agent Systems, IEEE Press, 1998.

[5] Lister, K., Sterling, S. Agents in a Multi-Cultural World: Towards Ontological Reconciliation. In Proceedings of the 14th Australian Joint Conference on Artificial Intelligence, 2001.

[6] Heflin, J., Hendler, J. Semantic Interoperability on the Web. In Proceedings of Extreme Markup Languages 2000, Graphic Communications Association, Alexandria, VA, 2000.

[7] Heflin, J., Hendler, J. Dynamic Ontologies on the Web. In Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000), AAAI/MIT Press, Menlo Park, CA, 2000.

[8] Klein, M. Supporting evolving ontologies on the Internet. In Proceedings of the EDBT 2002 PhD Workshop, Prague, Czech Republic, March, 2002.

[9] McGuiness, D., Fikes, R., Rice, J., Wilder, S. An environment for merging and testing large ontologies. Principles of Knowledge Representation and Reasoning. In Proceedings of the Seventh Inter-national Conference, San Francisco, USA, Morgan Kaufmann, 2000.

[10] Noy, F., Musen, N. An Algorithm for Merging and Aligning Ontologies: Automation and Tool Support. In Proceedings of the Workshop on Ontology Management at the Sixteenth National Conference on Artificial Intelligence, Orlando, USA, AAAI Press, 1999.

[11] Sato, S. CTM: An example-based translation aid system. In Proceedings of the Fifteenth International Conference on Computational Linguistics, 1992.

[12] Steels L. Self-Organising Vocabularies, C. Langton and T. Shimohara, (eds), Artificial Life V, Nara, Japan, 1996.

[13] Sterling, L. A Knowledge-Biased Approach to Information Agents. In Proceedings of the International Workshop on Information Integration and Web-based Applications and Services (IIWAS'99), Yogyakarta, Indonesia, 1999.

[14] Sterling, L. On Finding Needles in WWW Haystacks, Advanced Topics in AI. In Proceedings of the 10th Australian Joint Conference on Artificial Intelligence, Abdul Sattar (ed.), Springer-Verlag LNAI, Vol 1342, 1997.