Lister, K. & Sterling, L. (2001). Agents in a multi-cultural world: towards ontological reconciliation.

Originally published in M. Brooks, D. Corbett & M. Stumptner (eds.). *Proceedings of the 14th Australian Joint Conference on Artificial Intelligence (AI 2001), Adelaide, South Australia, Australia, 10-14 December 2001.*
Lecture notes in computer science: advances in artificial intelligence (Vol. 2256, pp. 321-332). Berlin: Springer-Verlag.

Available from: http://dx.doi.org/10.1007/3-540-45656-2_28

Agents in a Multi-Cultural World: Towards Ontological Reconciliation

Kendall Lister and Leon Sterling

Department of Computer Science and Software Engineering,
The University of Melbourne, Victoria, 3010, Australia,
krl,leon@cs.mu.oz.au,
Intelligent Agent Laboratory: http://www.cs.mu.oz.au/agentlab

Abstract. In a world in which communication technology is rapidly eroding traditional boundaries, the cultural individualities of organisations become increasingly relevant as the demand for integration and interoperation grows. In order to function effectively, agents, whether human or software, must be able to communicate and interact through common understandings and compatible conceptualisations. In a multicultural world, ontological differences are a fundamental obstacle that must be overcome before inter-cultural communication can occur. The purpose of this paper is to discuss the issues faced by agents operating in large-scale multi-cultural environments and to argue for systems that are tolerant of heterogeneity, illustrating the discussion with a running example of researching and comparing university web sites as a realistic scenario representative of many current knowledge management tasks that would benefit from agent assistance. We then discuss the efforts of the Intelligent Agent Laboratory toward designing such tolerant systems, giving a detailed presentation of the results of two implementations and an assessment of their significance for further progress.

"In an ill-structured domain you cannot, by definition, have a pre-compiled schema in your mind for every circumstance and context you may find ... you must be able to flexibly select and arrange knowledge sources to most efficaciously pursue the needs of a given situation."[8]

1 The Reality of Distributed Knowledge Systems

That useful knowledge systems inevitably incorporate vast amounts of information is becoming a generally acknowledged phenomenon. The evolution of the computer as a data processing device, and computer networks as communication media, has provided the technical means to aggregate enormous quantities of information. Similarly acknowledged is that our capacity for accumulation, storage and reproduction of data and information has out-paced our ability to perceive and manipulate knowledge. This is not a new realisation; Vannevar Bush identified just such a glut of knowledge and information over fifty years ago and proposed a technological solution in the form of the memex, an enlarged intimate supplement to memory that anticipated the hypertext systems of today[3]. The need for contextualising data remains thoroughly applicable to the World Wide Web and other large-scale information networks.

By implementing a (pseudo-)global communication infrastructure that provides means for the publication, comparison and aggregation of apparently limitless amounts of data, we have discovered the potential to ask questions as individuals conducting our daily lives that previously would have been dismissed as infeasible for anyone less than a dedicated organisation. For example, with the entry cost of publishing a web site effectively negligible, the university that does not do so is the exception rather than the rule. This means that dozens, if not hundreds, of descriptions of courses, programs and facilities are available for us to peruse. As we learn this, we immediately see a possibility for comparison, and want to ask reasonable and seemingly simple questions such as "Which faculties offer courses in applied machine vision?" or "Which campuses provide accommodation facilities for post-graduate students?".

To answer questions like these, we could fairly easily compile a list of university web sites; the list might even be complete. We could then visit each site in turn, browsing or searching and recording what information we think will answer our question. Finally, we could compare the results of our research from each site to formulate an answer. Many people perform this very task every day. The question that interests this paper is why our computers can't do this for us yet, and how we can approach the issue of enabling them to do so. The example of university service descriptions is an appropriate one for the purposes of this paper, as the issues described can be readily seen to be present in the real world. Additionally, universities as institutions tend naturally to develop and often then actively promote their individuality; this local culture flavours their presentation of information that must then be reconciled with information from other institutions that apply their own cultural characteristics to their publications. If we are to manage knowledge from a variety of sources effectively, we will need the assistance of software that is culturally aware and is capable of negotiating the conflicts that arise when such heterogeneous knowledge is juxtaposed.

2 How Organisational Culture Affects Communication

The reality of distributed information systems is an environment in which knowledge from large numbers of heterogeneous sources must be integrated in such a way that we can efficiently reconcile any differences in representation and context in order to incorporate foreign knowledge into our own world-view. To be able to work with knowledge from incongruous sources is becoming increasingly necessary[15] as the focus of information processing moves beyond intra-organisational interaction and begins to transgress borders, whether departmental, corporate, academic or ethnic. As they grow, organisations develop unique cultures, whether they are companies, universities, industries or nations. Organisational culture is considered to be both constructive and inhibitive as far as the day to day operations of the organisation are concerned, but in the context of knowledge management it creates significant barriers to inter-organisational communications and transactions.

Organisational cultures arise as individual organisations develop mechanisms, procedures and representations for dealing with the issues that they face. Inevitably, because these cultures are generally developed in isolation, each organisation will invariably arrive at different solutions to what are often very similar problems. In order to stream-line organisational activities and focus group efforts on a common goal, it is necessary for individuals to internalise their own personal intuited approach to a situation in lieu of an agreed common understanding shared by the other members of the group. We do this naturally when we work together on a problem; some are more able than others, and we recognise teamwork and the ability to understand another's point of view as desirable qualities. Such qualities are also becoming desirable in software as agents play an increasing role in our communication and collaboration.

However, the advantages of requiring every member of an organisation to follow a centralised doctrine are accompanied by a downside. The cost of standardisation is inflexibility and a reduced ability to adapt and cope with a wide variety of situations in an efficient manner, which is precisely the working environment in which increasingly we find ourselves. This problem is exacerbated when organisations attempt to interact with external groups. They inevitably find that even when they think that they are working in similar domains and facing common problems, they are unable to communicate effectively. This is due in a large part to the cultural differences between organisations, differences that arise as each organisation attempts to codify its approach to various situations. The streamlining that appeared necessary for efficient operation within each organisation now stands as a barrier to interoperation and sharing of resources. Exactly such an incongruity also manifests in the information and knowledge generated by such organisation, and the problems faced by software agents negotiating such data are analogous. It is our hope that the techniques people use to overcome such problems can be beneficially adapted to software agents.

When we suppress our own intuitive understanding of a situation and attempt to adopt a standardised, agreed upon approach, we increase our ability to interact with others who have similarly adapted their individual understanding to that of the group or community. But we also lose something in the process: context and generality. An efficient understanding of a situation is like a model, in that the more closely it describes a particular situation, the less effectively it describes a general class of situations. Additionally, as we move from a general conceptualisation of a situation rich with semantic flexibility to a specific understanding, we tend to eschew context. We do this because the very generality that gives us the ability to deal with many varied and new situations is a barrier to communication; at the same time that ambiguity allows adaptation, it prohibits individuals from establishing the certainty of agreement that is necessary for confidence that each understands the other.

However, as organisations discover, standardisation of practices and understandings does not create a panacea for the difficulties of communication and collaboration. On a small scale, adoption of standardised approaches helps individuals to cooperate and achieve goals too large for a single person. On a larger scale, the effort required to establish and prescribe global standards and common approaches grows rapidly beyond feasibility as the number of participants and the amount of data being manipulated increases. As our ability to communicate and interact across cultural borders increases, so does our desire to do so. And as we come to terms with the necessities of increased interoperation and develop coping strategies, if our software tools are to scale similarly we must provide them with equivalent reconciliation capabilities.

3 Our Software Colleagues

In many respects, computers are an extreme example of co-workers with poor teamwork and communication skills. When specifying a task for a software application or agent we must specify every step in precise detail, detail that will generally remain constant throughout the life of the software. Whilst humans are able to adjust the level of abstraction at which they conceptualise a particular situation, computers traditionally have the capacity only for comparatively very low levels of abstraction. As machines that follow explicit instructions to the letter, their operation is analogous to the most procedural organisational standards, and unsurprisingly they too have great difficulty adapting to new situations.

Traditional computational paradigms require that computer-mediated representations of information and knowledge be exact and literal; in order for a computer to process information requires simplistic structuring of data and homogeneous representations of concepts. In order to maintain consistency during processing, traditional approaches require that each participant in a system, whether human or software, subscribe to a common understanding of the concepts within the system. In other words, traditional information systems require the adoption of an absolute ontological world-view; deviation from a priori agreed terms and understandings results in a breakdown in communication and loss of consistency through the system.

This ontological homogeneity has worked well for systems with little direct human interaction, when the computers can be left to sort out technical details and humans can work at a level removed from the coal face. In fact, isolating technically detail areas of a system from those areas with which humans interact permits engineering of the technical aspects to create an optimised environment. The World Wide Web is an example of a large-scale system in which the level at which humans interact with the system is greatly separated from the level at which machines interact. We write web pages and read them, navigating along hypertextual paths, while machines manage domain name resolution, protocol selection, transmission of data and rendering of text and images. The gap between the activities of humans and machines is highlighted by the problems that occur when we try to make machines work closer to our level as we attempt to automate various functions that we currently perform manually. The example of this most recognisable to the ordinary web user is searching for information, an obviously difficult problem that has yet to be solved to our satisfaction. But a more far-reaching problem is that of integrating the vast quantities of information available in such a way that we can seamlessly assimilate whatever sources of data are most appropriate to the task at hand, whatever that task may be.

4 Automating Conceptualisation

Automation of data processing is desirable because it frees humans from the morass of detail and permits them to utilise their capacity for abstraction. The ability to manipulate concepts at varying levels of detail and to match the level of detail to the needs of the situation at hand is one of our most effective tools for processing knowledge and communicating. Being able to subsume detail within conceptual units of knowledge allows us to overcome the natural limits of our processing capacity; although there appear to be clear cognitive limits on the number of concepts we can articulate at any given time, we have the critical ability to 'chunk' collections of knowledge into single units[11, 5], effectively providing a a capacity to search through information webs both widely and deeply as necessary. Similarly, when the scope of an information or data problem becomes too great for us to process in a reasonable amount of time, we bring computers to bear on the problem to assist us with storage, recall and simple processing. Automation of data processing provides increased speed and accuracy, and also permits the not insignificant relief of boredom resulting from repetitive tasks.

By handing low-level information processing tasks to machines, humans are freed to consider issues at higher levels of abstraction. If we are to continue to advance the level of assistance that our computers can provide to us as we work, we must elevate our tools to higher levels of abstraction to accommodate the ever-increasing complexity of the situations we face.

As knowledge travels through progressively lower levels of abstraction, its context degrades as generality is replaced by specificity and logical operability. Humans require some specification in order to communicate successfully; the desired degree of consistency of conceptualisations determines the extent of specification that is necessary. Indeed, it is suggested that even consensus between participants is not always necessary for successful collaboration[1,12]. As discussed earlier, one of our greatest strengths as humans is our ability to adapt to new situations and reconcile new ontological concepts with our own history of previous experiences. We also capable of identifying mismatches of understanding in our communications and negotiating shared perspectives as we interact with others[2]. Human natural language is neither precise nor predictable, and this seems to reflect the way that we to understand the world though our internal representations and conceptualisations. When we express ourselves in natural language, we often encounter confusion and difficulty as others attempt to understand us. This requires us to explore alternative expressions, searching for representations that others understand. We do this naturally, and our attention is drawn to the process only when it fails. But we are generally capable of finding enough common ground for communication of knowledge to proceed; we are often even able to convey basic information without a common language, as any tourist who has managed to gain directions to a restaurant or train station with much waving of hands can attest.

Computer mediated communication removes many of the mechanisms that we use to assist our process of reconciling ontological differences during interpersonal communication, and generally leaves us at best with spoken or written language. Anecdotal evidence documents the detrimental affects on effective communication of using a 'low bandwidth' medium such as a telephone or a 'high latency' medium such as the post or e-mail. The effects of limited representation of concepts are only exacerbated when computers are no longer just the communication medium but also themselves participants in the communication and knowledge manipulation, as is increasingly desirable as the amounts of information with which we work are growing beyond our capacity to manage them manually. In order for the processing power of computers to be utilised, knowledge must be reduced to a representation suitable for logical operations. Largely, fitting knowledge to logical representations is a subjective process. Decisions must be made about how to express complex concepts in relatively constrained languages; these decisions are made by people whose choices of representation and expression are influenced by their own cultural background. Consequently, as context is lost problems then arise as other organisations with different cultures, or even just individuals with different conceptualisations, attempt to understand the logical representation and rebuild the original knowledge.

To return to the case of university web sites, it seems reasonable to assume that all universities partake in the teaching of students and in research. Most universities offer undergraduate degrees in the areas of engineering, arts, science and commerce. But when it comes to describing their activities, where one university may use the word *course* to refer to a particular degree program, another will use *course* to mean an individual subject within a degree; a third institution may use *course* to describe a particular stream or program within a degree. Some institutions will say *unit* where others say *subject* and others say *class*. Simply due to their own individual organisational cultures, different institutions use different vocabularies to describe their activities. The researcher wishing to compare the services provided by different universities will generally quickly identify the differences and through an understanding of the knowledge domain concerning university activities and services will be able to translate between terms, usually assimilating them into the researcher's own personal ontological understanding, which itself will be shaped by their personal experiences (if they are from a university that uses *course* to mean a unit of teaching and *program* to describe an undergraduate degree, they will probably translate the descriptions from other institutions into this ontology - if they are not from a particular university, they will probably draw on whatever experience they have of academic institutions, and if they have none, they may build their own ontology from the collection of university representations).

To create software agents that can handle this level of ontological complexity would seem to be very difficult. Why then is it preferable to simply agreeing upon a global ontology to which all agents subscribe, a centralised language of understanding and representation, or even a global directory of multiple re-usable ontologies from which agents select as necessary? Ontology creation itself is very difficult. It requires the ability to define many concepts precisely and consistently. It requires the ability to predict appropriate assumptions and generalisations that will be acceptable to most, if not all, people. It also requires universal access and distribution infrastructure, and a well-established and accepted knowledge representation format. It requires some way to address the desire for agents and humans to interact at variable levels of abstraction as particular situations demand. It requires constant maintenance to ensure freshness and currency, yet also must provide backward compatibility for old agents. It requires that agent developers familiarise themselves with the prescribed knowledge representation formats, ontologies and protocols and adapt their own development efforts to suit them. These issues make a global ontology infrastructure unsuitable as the sole approach, and it is our belief that effort spent adding tolerance of heterogeneity to systems will provide greater benefit as we begin to introduce agents to our multi-cultural world.

In addition to the practical benefits, one of our strongest desires for tolerance of heterogeneity for software systems is rooted unashamedly in idealism: humans manage to resolve ontological differences successfully, in real time and 'on the fly'. This ability gives us much flexibility and adaptability and allows us to specialise and optimise where possible and yet generalise and compromise when necessary. Therefore, it seems both feasible and desirable to have as a goal a similar capability for software agents.

If we are to make effective use of multi-cultural data from heterogeneous sources, we need ways and means to reconcile the differences in representation. If we are to work efficiently to solve large information problems, we need the assistance of automated mechanisms. To achieve both, we need systems that are tolerant of heterogeneity.

Reconciling ontological differences requires understanding the difference between concepts and their representations; in semiotic terms, appreciating the difference between the signifier and the signified. Reconciling ontological differences means reading multiple texts that represent identical, similar or related concepts and being able to work with them at the concept level rather than at the level of representation.

For an XML documents or databases, it might be as simple as realising that two fields in different data sources actually contain the same class of data. On the other hand, it might be as complex as deciding that articles from an economics magazine and an automotive magazine are discussing different topics even though they both have 'Ford' and 'analysis' in their titles, something that current search technologies would be unlikely to realise.

As the number of data sources available to us and our ability to access them on demand and in real time is increasing, the overhead of pre-constructing a complete ontology for a given interaction becomes less and less viable. Large scale interconnectedness and increased frequency of data transactions across organisational and cultural borders leads to a reduction in the useful life of any context constructed for a particular transaction. Just as we are able to establish contexts and construct suitable local ontologies as needed for particular interactions, if we want to be able to include software agents in our higher level communication and knowledge management, they will need to be capable of similar conceptualisation.

5 Results and Thoughts from the Intelligent Agent Laboratory

The Intelligent Agent Laboratory at the University of Melbourne has been working for a number of years on knowledge representation and manipulation for information agents[13, 14]. When considering how best to structure knowledge for information agents, two questions arise: what types of knowledge should be pre-defined and what should be left to be learned dynamically? The work of the Intelligent Agent Laboratory addresses these questions in both theory and practice; the remainder of this paper describes two recent projects.

5.1 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. 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[4]. 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. In a manner that draws on similar principles to object-oriented analysis and design methodologies and component-based development models, an agent shell developed from CASA[9] allows simple construction of agents that are able to quickly incorporate new knowledge bases, both learnt by the agent itself and incorporated from external sources.

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. 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.

5.2 AReXS

Automatic Reconciliation of XML Structures (AReXS) is a software engine that attempts to reconcile differences between XML structures that encode equivalent concepts. It is able to identify differences of expression and representation across XML documents from heterogeneous sources without any predefined knowledge or human intervention[6]. It 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)[7] 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. For example, a pair of XML documents from different sources, both describing services offered by universities, might contain attributes named `SUBJECT` and `UNIT` respectively. If the two attributes 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 attributes as well as the attribute names, deriving confidence in a match from similarities in either comparison. If one document contained the statement `<SUBJECT>Introductory Programming</SUBJECT>` and another contained a similar statement `<UNIT>Introduction to Programming </UNIT>`, AReXS is able to consider the possibility that the two attributes `SUBJECT` and `UNIT` are in this context signifying the same concept. If further correspondences could be found between other instances of these same attributes, the confidence of a conceptual match would increase.

AReXS works by analysing two XML structures and identifying matching attributes, generating a map of equivalence between concepts represented in the two documents. Identification of conceptual equivalence is based on a consideration of lexicographical similarity between both the names and the values of attribute 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 attributes 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[10] to evaluate textual similarity between the names and values of attributes. 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 attribute 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 attribute pairs. The uniqueness function described by[7] is used to establish the likelihood of a textual match between attributes actually revealing a shared, unique concept, based on the principle that the more common a concept is across significantly different attributes, 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, barnesandnoble.com, angusandrobertson.com.au and borders.com show that AReXS is capable of accurately identifying conceptually equivalent attributes based on both the attribute names and sample instances of the attributes. 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.

From a data processing point of view, retail book descriptions provide a good test bed for algorithms that could later be applied to inventory databases, task lists, measurement recordings, product or service advertisements, et cetera; one common characteristic of all these types of data is that although the actual objects or concepts being represented are generally consistently understood, people generally seem to vary greatly in the choices they make when constructing logical representations for them.

The AReXS algorithms allow identification of concept matches regardless of the ordering of concepts or attributes, and its consideration of both names and values of attributes allows it to identify equivalences even if one of the name or the value is absent; in other words, AReXS is tolerant of inconsistent data. 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 attributes in one data source but only one attribute in the other data source.

6 Further Thoughts

AReXS is in reality only a prototype that serves as a demonstration of the potential for automated reconciliation of the ontological differences that manifest in data sources from a culturally heterogeneous environment. Because the effectiveness of the concept matching algorithm is improved by examining more instances of the data, and each data attribute must be examined to increase the confidence of the conceptual matches, AReXS currently suffers from poor scalability as the complexity of data objects increases. The CBBM algorithm used for comparing attribute names and values is heavily biased toward text strings and struggles with variations of numerical data. Due to the modular design of AReXS, this component of the engine could be significantly improved with a combination of simple heuristics, alternative matching algorithms and possibly even the capacity to pre-populate the data context with concepts previously observed or learned. AReXS currently can only work with flat or unnested XML structures, although it is quite reasonable to imagine extending the principles it demonstrates to more complex data structures, or even incorporating the AReXS concept matching engine as a component in a more sophisticated data analysis system. Although AReXS only supports reconciling pairs of data sources, the EBFM algorithm on which it is based does allow for comparison of multiple sources and so extending AReXS to support this feature would appear to be feasible. While AReXS is partially able to recognise many-to-one equivalences, it would require further work to actually capitalise on this recognition. Finally, it seems reasonable to imagine that the principles implemented in AReXS could quite readily be adapted to allow the extension of data structures based on identification of concept matches within attribute names or values. Drawing on the example described earlier of university service descriptions, if one institution chose to present teaching units with an attribute of the form `<UNIT>Machine Vision (Semester 1)</UNIT>` and a second institution opts for two attributes `<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 attributes from the second source are encoded within a single attribute of the first source.

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.

References

1. Beck, Eevi E.: Changing documents/documenting changes: using computers for collaborative writing over distance, The Cultures of Computing (1995)
2. Boisot, Max H.: Information space: A Framework for Learning in Organizations, Institutions and Culture (1995)
3. Bush, Vannevar: As We May Think, The Atlantic Monthly, Vol. 176 No. 1 (1945)
4. Gao, X., and Sterling, L.: Classified Advertisement Search Agent (CASA): A Knowledge-Based Information Agent for Searching Semi-Structured Text (1998)

5. Hofstadter, Douglas R.: Godel, Escher, Bach: an eternal golden braid (1979)
6. Hou, Dominic: Automatic Reconciliation of XML Structures (2001)
7. Ikeda, Yuji, Itoh, Fumiyaki, and Ueda, Takaya: Example-based frame mapping for heterogeneous information agents, IEEE, October (1998)
8. Jones, Robert Alun, and Spiro, Rand J.: Contextualization, cognitive flexibility, and hypertext: The convergence of interpretive theory, cognitive psychology, and advanced information technologies, The Cultures of Computing (1995)
9. Loke, Seng Wai, Sterling, Leon, Sonenberg, Liz, and Kim, Hoon: ARIS: a shell for information agents that exploits web structure, Proceedings of the Third International Conference on Practical Applications of Intelligent Agents and Agent Methodology (1998)
10. Sato, S.: CTM: An example-based translation aid system, Proceedings of the Fifteenth International Conference on Computational Linguistics (1992)
11. Simon, Herbert A.: The Sciences of the Artificial (1996)
12. Star, Susan Leigh: Cooperation without consensus in scientific problem solving: Dynamics of closure in open systems, CSCW: Cooperation or Conflict? (1993)
13. Sterling, Leon: On Finding Needles in WWW Haystacks, Proceedings of the 10th Australian Joint Conference on Artificial Intelligence (1997)
14. Sterling, Leon: A Knowledge-Biased Approach to Information Agents (1999)
15. Weinstein, Peter C., and Birmingham, William P.: Agent Communication with Differentiated Ontologies (1998)