Providing Information for Mixed Initiative Interaction via Interoperable User Modelling *

David Brokenshire and **Jurika Shakya** and **Vive Kumar**
Simon Fraser University,
13450 102 Ave.
Surrey BC, Canada V3T 5X3
604-268-7510
brokenshire@sfu.ca

Abstract

Mixed initiative systems rely on knowledge both to know when to take the initiative and to be helpful when they have the initiative. Knowledge is usefully applied data and there is a huge amount of data available in computing systems on almost any topic. However, data is not always able to be usefully applied to a particular situation. This can occur because data isn't relevant, access to it is restricted in some way, it is not machine readable, or we simply cannot find it. We have created an architecture for making more data and knowledge available by connecting disparate user models and sources of data. We provide a comparison of this architecture to other approaches and discuss their relative strengths and weaknesses. We then describe how our approach can be used to recognize opportunities for initiating mixed initiative interactions.

Introduction

Mixed initiative systems rely on knowledge both to know when to take the initiative and to be helpful when they have the initiative. That knowledge comes in many forms and covers many areas. Some of the forms include user models, domain knowledge encoded in program logic, databases, or ontologies. Knowledge is usefully applied data and there is a huge amount of data available in computing systems on almost any topic. However, data is not always able to be usefully applied to a particular situation. There are several reasons why data may not be useful including:

1. The data is not relevant
2. The data is relevant but cannot be found or accessed because of
   (a) Access restrictions (privacy, ownership, etc.)
   (b) Unusable format
      i. Not machine readable (e.g. standard webpages, offline resources)
      ii. Different meta-data or application specific format (database, different ontology, etc.)
   (c) Usable accessible data which cannot be found because of failures in searching for or identifying that information

Of the many ways of gathering data we have chosen to design an architecture for connecting sources of data and user models together. We argue that this is particularly relevant for attempting to understand the user in order to help choose when to take the initiative. After describing the architecture in section we emplify its use in section .

Motivation for Connecting User Models

In research, education, and commercial systems user modelling has helped create adaptive programs, web sites, and educational experiences (Fink & Kobsa 2000). Further, user modelling allows educational systems to improve student achievement (Brusilovsky 1998). Particularly, Adaptive Hypermedia enabled with modelling the users allows web sites such as Amazon.com make personalized recommendations to users (Brusilovsky 2001). Nevertheless we rarely see the effects of modelling in our daily lives or educational systems. In this paper, we describe how user modelling enhances the ability to recognize opportunities for effective mixed initiative interactions.

Getting information about a user into a model is essential to user modelling but gathering information is slow and expensive. Software using user models collect information through active means such as questioning the user and passive means such as recording what resources are accessed and what requests the user makes. In education a user has motivation to provide information because they are compelled. However on the Internet users are discouraged from providing information by fear of privacy abuses. Even in settings where they have a reason users are notorious for not wanting to enter extra information or entering inaccurate information.

Once information is collected it is often not used effectively. The majority of systems in education are limited to specific institutions or parts thereof. According to (Kobsa 2001) only one shell system was ever used outside its home institution. Furthermore, research projects which collect valuable data often disappear, preventing their ongoing use.

The difficulty in doing useful modelling with little information is referred to as the cold-start problem. Interoperability of user models allowing for data sharing offers a potential

*This work has been conducted with support from a grant funded by the Natural Science and Engineering Research Council of Canada (NSERC) for the Learning Object Repositories Network (LORNET), an NSERC Discovery Grant, and SSHRC INE (LearningKit) grant.

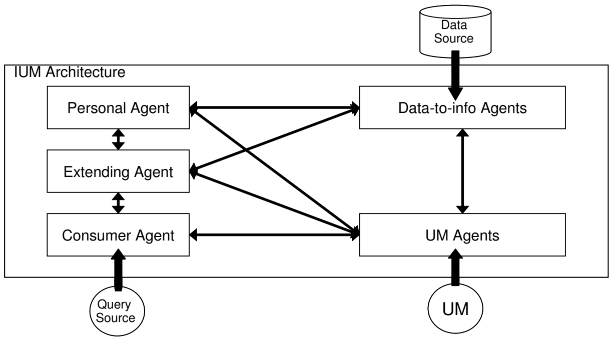

Figure 1: IUM Architecture

solution by allowing models to access previously collected information about a user from other systems instead of collecting it again.

Interoperability can make data from legacy projects accessible and contemporary systems can share data gathered about users. Connecting many systems also allows depth of information collection not possible by a single system. This approach works well with traditional user modelling, where a large amount of data is stored in a single model, and the active modelling paradigm described by McCalla et al. in (Gord McCalla & Bull 2000), where just-in-time models are constituted for a particular purpose. Collecting information from disparate sources also raises issues of model consistency with potentially conflicting information coming from different sources. This is an ongoing issue user models must accommodate though it can certainly be dealt with, as discussed in the aforementioned work (Gord McCalla & Bull 2000).

In section we will describe the underlying architecture of our approach in detail. In section we will describe the communication structure and ontologies of the architecture. In section we will review similar approaches and motivating trends. We then discuss the prototype in section . Finally we discuss our conclusions and future work in section .

Architecture

The current paradigm of user modelling is mostly client/server modelling applications. More recently there has been interest in light-weight user models which follow a user (Kobsa 2001) and in multi-agent user models (Vassileva, McCalla, & Greer 2003).

Our system furthers this trend by dividing the different components needed for communication between user models into several agents. As shown in Figure 1 our system is comprised of six main components, five of which are agent based. The components are data sources, data to information agents (DIAs), user model agents (UMAs), consumer agents, extender agents (EAs), and personal agents (PAs). Designed on top of the FIPA agent specifications these agents provide all the services necessary for use of and communication between user models.

In our system we intentionally do not propose a specific user model technique or specific domain ontologies for user modelling. These tasks are left up to researchers and implementers so they may make the choices which best suit their needs, while still allowing interoperability.

Data Sources

Data sources technically exists outside of our architecture. They can be any source of data or information useful to a user model such as intelligent tutoring systems, student information systems, databases, or web site visitor data. The data source only needs a programmatic method of extracting the information to be retrieved from sources and shared models. There is no explicit need for the data sources to be aware of their participation in this architecture; these sources are decoupled from the rest of the architecture by data to information agents described in section .

Data to Information Agents

Data to Information Agents (DIAs) make use of a modified adaptor pattern (Erich Gamma 1995) to connect data sources to user models. The DIAs use means exposed by the data sources to retrieve data such as database queries or API calls. They do any necessary processing and conversion to supply the information in a form understandable by some user model. This involves converting the results from the native form into an ontology representation. For example a relational database schema can be mapped to an ontology representation by an expert using a tool such as Protege (Protege-Project 2004). User model agents which understand the ontology or an available mapping can then access the information if they have permission.

The adaptor pattern is well known but it is typically a small part of an existing application and not available to other applications. In the eduSource approach (Hatala *et al.* 2004) (eduSource 2003) it is separate from the application but it is still tied to the particular system and is a heavyweight component. By following an autonomous agent approach with the DIAs we allow the adaptors to be written once, and used transparently by many. This The communication mechanisms which enable this are discussed in section . The DIAs can be given the ability to update their known ontologies and mappings and choose the most appropriate for a given situation. Future work will focus on allowing DIAs to automatically adapt to changes in the data model of their associated source, thus reducing the need for human intervention in the system.

User Model Agents

The User Model Agent (UMA) connects the user model to the architecture. The user model agent makes decisions about how to act and whom to interact with. The UMA is again typically an intelligent adaptor to a UM. By separating the UMA from the actual model and making it responsible for communication the architecture permits the use of whatever representation and inference techniques preferred by the developer of the user model. For example a Bayesian user model could connect as easily as a rule based model. The communication mechanisms (described in section) decouple the user model from all of the other components of the architecture.

Consumer Agents

Consumers are software which uses the information in the models. Not all systems will require a consumer agent to access data, as they may be connected to the user model outside of the architecture. If a system just wants to query existing user models which have UMAs it can do so via a consumer agent.

If a new UM application being developed is aware of this architecture it can be developed to use an agent interface. Alternatively, an agent can be created which will interact with this architecture and then provide a legacy system with information using whatever means the legacy system has available. Furthermore, a user model may query other user models by acting as a consumer of different user model agents.

Extender Agents

Extenders are the means for helping agents deal with change. This is particularly necessary with the lack of formal standards in a system that will exist for years. The platform a system runs on may change, as may communication protocols, technical standards, etc. Additionally, the entities a system communicates with may change in any of these ways, forcing the system to adapt or become obsolete. While we do not attempt to automatically surmount all of these difficulties, we do attempt to mitigate their effects.

Presently the extender is responsible for making other agents aware of communication resources including translations between and extensions of ontologies. This functionality is similar to that described in the experimental FIPA Ontology Service Specification (FIPA 2001). We do not prescribe how the ontology mappings should be created. In principle any of the techniques described in (Kalfoglou & Schorlemmer 2003) could be used, or any other technique developed. This is an important to the continuing use of the system as ontology mapping and translation is an open area of research with the methods likely to change and improve in the future.

Communication

FIPA/JADE Platform

The underlying communication system of our architecture is the JADE platform which implements the FIPA protocols and Agent Communication Language (ACL). A large amount of information on the standards is available at (FIPA 2004) and on JADE at (TILabs 2004). The JADE platform provides classes to help construct agents and a thorough platform and protocol implementation used in managing agents and sending messages.

IUM/Control Level Communication

We define interactions among the agent types using the FIPA interaction protocols and semantics. In the case of agent naming and privacy requests we have developed our own ontologies which can be extended and adapted by the community. Figure 2 shows the services offered by and used by the agents. Part A of the table depicts the communication services offered by the different agents. Part B depicts which

Services	PA	UMA	DIA	CA	EA
Subscribe	*No*	*Yes*	*Yes*	*No*	*Yes*
Request	*Yes*	*No*	*Yes*	*No*	*Yes*
Query	*No*	*Yes*	*No*	*No*	*No*

Requests	PA	UMA	DIA	CA	EA
Subscription	*EA*	*UMA, DIA, EA*	*EA*	*EA, UMA*	*EA*
Request	*EA*	*PA, DIA, EA*	*EA, PA*	*EA*	*EA*
Query	*n/a*	*UMA*	*n/a*	*UMA*	*n/a*

Figure 2: Agent Interactions

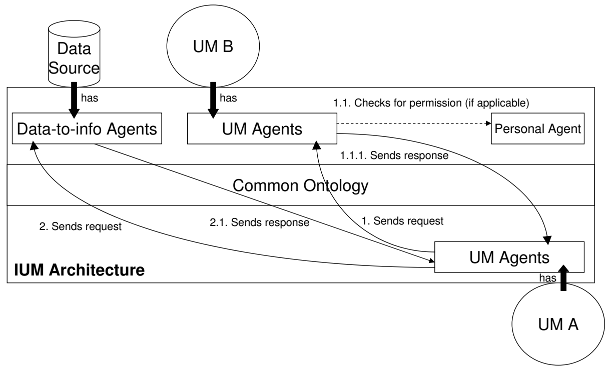

Figure 3: Communication Example

agents make use of the different services offered. Some communication also occurs outside the architecture to external data sources, user models, and querying sources. Figure 3 shows a typical interaction among agents in the system.

Domain Level And Ontologies

The domain level incorporates concepts represented by a model in a particular domain. There is no single user modelling ontology that can cover all possible circumstances and do not seek to define one. Instead projects define their own ontologies and share, extend, and translate them.

The semantic web is making ontologies mainstream and leading to tools which make their creation and use far easier. They allow us to formally represent information thus improving interoperability. However, ontologies must still be created by people, be specific enough to be useful, and be general enough to be reusable.

There may also be interpretation issues. Finally concepts in a domain shift over time and the ontologies must adapt. When attempting to communicate different agents must have a common ontology they can use, or they must be able to access a translation between ontologies they understand.

It is worth noting that interoperabilty between models at the level of common ontologies in no way guarantees that information will be correct, useful, secure, ethically obtained, or even legal. One specific problem with interoperability that may mitigate agaist the usefulness of large amounts of data from multiple sources is consistency. There are no guarantees that the data won't directly contradict. These very important issues are not considered herein and there is a substantial body of work covering all these topics.

Comparison

General User Modelling Systems (GUMS) proposed by Finnin (Kass & Finin 1988) offer a variety of user modelling services which can be used by programs. However, they do not address communication and sharing between various projects using the system differently, or between projects using different GUMS. As mentioned earlier they have seen limited use (Kobsa 2001). Interoperable User Modelling (IUM) offers a simple communication framework based on existing FIPA standards without constraining the domain representation or user model implementation allowing simpler cross-institution connection.

Standards such as PAPI Learner (IEEE-LTSC 2002), IMS LIPS (IMS 2001), and the LOM Metadata standard (IEEE-LTSC 2004) offer broad approaches to representing student and portfolio information. These standards can be used to share information between user models. General standards still need extensions to serve particular needs, but we lack an easy means of sharing those extensions outside of the standards process, which is typically very slow. Standards which can be represented as ontologies can be easily incorporated into our architecture. For example initial work exists on creating ontologies from PAPI and LIPS (Dolog & Nejdl 2003). Unfortunately, the generality of standards leads to unrealistic complexity for many projects to manage. Despite their complexity these standards do not cover domain information, though it can be included via what LIPS calls external formats, which can be ontologies. We expect that with use common ontologies will be created in different domains by communities of practice, in effect creating ad-hoc ontology-based standards. The need for such ad-hoc standards can be seen by the duration of the standardization process for PAPI and LIPS, which is now past its fifth year.

The two primary choices in technology for IUM were web-services and agents. Agents have several conceptual benefits over web-services for this task. Autonomy of agents aligns with a system which must last as a whole but has transient parts. The agents must also be able to adapt independently to their situation, using the appropriate ontology and representation languages, and attempting to adapt to different and changing ontologies with minimal human intervention. The division of labour among agents is cleaner for our system than a web-service based architecture. Agents also offered roughly equivalent ability to be run on a variety of platforms with different processing power. Issues against agents in a practical system include a lack of experienced developers, security and scalability.

More recently the MUMS (C. Brooks & McCalla 2004) project has taken an approach with similar goals to IUM but using the web services approach. Brooks' system requires users to model statements in RDF and uses opinions as the primitive unit for sharing modelling information. In contrast, the IUM takes a less specified approach by only requiring that the ontology be transmissible within a FIPA message structure. This allows the use of OWL, DAML+OIL, RDF, or virtually any other format. Thanks to the open design of IUM it can connect to MUMS as easily as to any other system.

Prototype and Example

The IUM system is now implemented for use in our lab and university. Students in the School of Interactive Arts and Technology use several different systems in their education. To obtain a more complete model of the student we must access information from all of the systems. Data to Information agents have been implemented for the SPARC ePortfolio system (Brokenshire D. 2004), the WebX conferencing system, the internal course management system (CMS), and the gStudy educational technology system. Each of these systems offers different means of accessing information. The SPARC system uses relational database access, WebX and the CMS offer APIs, and gStudy has an combination of an XML based data model and a relational database.

To connect to these systems we first developed an ontology for the project domain. Then we created Java classes to represent the ontology using the Protege ontology editor (Protege-Project 2004) with the JADE Bean Generator plug-in. Finally we extended the DIA base class for each data source. We then used the basic UMA and PA to send requests to the system and simulated load in a series of experiments to analyze the computational and communication efficiency of the system. A user model is under development to use information gathered to support students learning.

Example use: MI-Edna

MI-Edna is a mixed initiative system for helping students improve their self regulated learning incorporated into the gStudy self regulated learning system (Winne *et al.*). MI-Edna attempts to recognize students' learning strategies by analyzing their interactions with gStudy, it then proposes possible improvements and new strategies to the student at identified opportune moments, or when they request assistance.

MI-Edna currently gets all of its data from activity logs generated in real time by gStudy. This data is very detailed down to the level of mouse movements, and also includes semantic data on the topic the student is studying. However, gStudy has limited data available about students learning preferences and their previous work and knowledge outside the system. This data would be useful both in deducing students' strategies and in decided when and how to offer help.

The students using gStudy and MI-Edna also use other computing systems in their education, including grade reporting software, online course management systems, and the SPARC ePortfolio systems (Bogyo, Lougheed, & Kirton 2004). All of these systems could act as sources of additional information, but they do not conform to the gStudy data format or the MI-Edna student profile ontology. As well as collecting information on students interactions the SPARC ePortfolio system has its own ontology describing students, their interactions, and their portfolios (Brokenshire D. 2004).

The SPARC ePortfolio system can be connected to the MI-Edna system via its user model, provided an ontology mapping is completed between the SPARC and the MI-Edna ontologies. The other systems can all be used as data sources by constructing data to information agents which parse the raw data and put it into either the SPARC ontology or the MI-Edna ontology.

The MI-Edna project ontology incorporates information on the domain of reading in gStudy. It also captures vari-

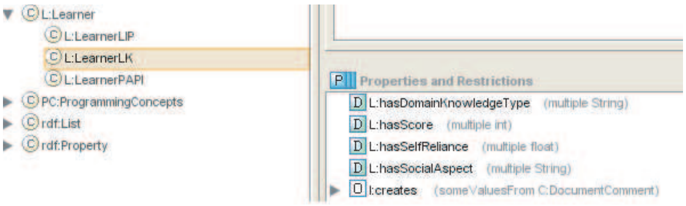

Figure 4: MI-Edna Learner Ontology

ables, tactics, and strategies observed in learner interactions within gStudy that correspond with specific models of self-regulated learning. Presently, MI-Edna encodes partial representations of Zimmerman's (Zimmerman 2000) 3-phase SRL model as well as Winne and Hadwin's (Winne & Hadwin 1998) 4-phase SRL model. 4. Since the goal of the MI-Edna system is to be able to recognize and propose opportunities for mixed initiative interactions based on a wholistic understanding of the learner's self-regulatory capabilities, it is important to unify the two different SRL observations based on the two different SRL models.

The interoperable user modeling architecture can be used to accomplish this goal by performing a semantic mapping between the portions of the two ontologies that represent the variables, phases, states, tactics, and strategies corresponding to the two different SRL models. Based on the mapping, the software agents can automatically populate the learner interactions into ontologies corresponding to the two SRL models.

Conclusion and Future Work

We have discussed herein the use of interoperable user modelling approach to help solve the cold start problem and provide richer modelling. This architecture is a first step towards realizing this goal.

In most cases, the level of effort and expertise needed for ontology creation, maintenance and mapping is prohibitive. To address this problem we propose the means of semantic mapping and sharing of data across ontologies.

We are actively working on testing our ability to recognize opportunities for mixed initiative interaction by experimenting with the MI-Edna system and the SPARC ePortfolio system.

In our upcoming work we will integrate a variety of educational and administrative software using this technology to gather data for richer user models and investigate possible benefits. People rarely put information in ontologies or metadata. Duval recently stated that "Web forms must die" and we must use smarter methods of gathering the information we need (Duval 2004). ePortfolios provide a source of structured data the users have an incentive to enter. We are developing a principled means of extracting ePortfolio information to inform user models. SPARC is currently testing with the TechOne program, the Coop program, and several Secondary school districts in the province of British Columbia providing approximately 4000 users. The Province of BC requires all secondary student to graduate with a portfolio, giving us a potential audience of 150,000 users in the next three years. gStudy is in use by several hundred high school and undergraduate students, and the WebX conferencing system and the course management system are used by approximately 1000 students in the TechOne program and the School for Interactive Arts and Technology.

References

Bogyo, B.; Lougheed, P.; and Kirton, T. 2004. *ePortfolios website at: http://eportfolio.research.iat.sfu.ca/*.

Brokenshire D., Bogyo B. & Kumar, V. 2004. Towards an upper-level ontology for information exchange in eportfolios. In *Proceedings of Europortfolio 2004*.

Brusilovsky, P. 1998. In Peter Brusilovsky, A. K., and Vassileva, J., eds., *Adaptive Hypertext and Hypermedia*.

Brusilovsky, P. 2001. Adaptive hypermedia. *User Modeling and User-Adapted Interaction* 11:87–110.

C. Brooks, M. Winter, J. G., and McCalla, G. 2004. The massive user modelling system. In *7th International Conference on Intelligent Tutoring Systems (ITS2004)*.

Dolog, P., and Nejdl, W. 2003. Challenges and benefits of the semantic web for user modelling. In *Proc. of AH2003 workshop at 12th World Wide Web Conference, Budapest, Hungary, May 2003*.

Duval, E. 2004. Learning objects on the semantic web - keynote speech. In *Proceedings of I2LOR 2004*.

eduSource. 2003. *eduSource Canada project. Available at: http://www.edusource.ca/*.

Erich Gamma, Ralph Johnson, J. V. . R. H. 1995. *Design patterns : elements of reusable object-oriented software*. Addison-Wesley.

Fink, J., and Kobsa, A. 2000. A review and analysis of commercial user modeling servers for personalization on the world wide web. *User Modeling and User-Adapted Interaction* (10):209–249.

FIPA. 2001. *FIPA Ontology Service Specification*.

FIPA. 2004. *Foundation for Intelligent Physical Agents. http://www.fipa.org*.

Gord McCalla, Julita Vassileva, J. G., and Bull, S. 2000. Active learning modelling. In G. Gauthier, C. F. . K. V., ed., *Intelligent Tutoring Systems*, 53–62. Springer-Verlag, Berlin Heidelberg.

Hatala, M.; Richards, G.; Eap, T.; and Willms, J. 2004. The interoperability of learning object repositories and services: Standards, implementations and lessons learned. In *Proceedings of WWW2004*, Proc. of WWW2004. ACM.

IEEE-LTSC. 2002. *Personal and Private Information Learner - Draft 8 Specification. Available at: http://edutool.com/papi*.

IEEE-LTSC. 2004. *IEEE LOM Final Draft Standard. Available at: http://ltsc.ieee.org/wg12/index.html. Accessed on March 20th, 2004.*

IMS. 2001. *IMS Learning Information Package Specification V1.0 Available at: http://www.imsglobal.org/profiles/index.cfm*.

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

Kass, R., and Finin, T. 1988. A general user modelling facility. In *Proc. of CHI'88*. ACM.

Kobsa, A. 2001. Generic user modeling systems. *User Modeling and User-Adapted Interaction* (11):49–63.

Protege-Project. 2004. *Protege Ontology Editor and Knowledge Acquisition System website at: http://protege.stanford.edu/*.

TILabs. 2004. Jave agent development framework (jade) available at: http://jade.cselt.it.

Vassileva, J.; McCalla, G.; and Greer, J. 2003. Multi-agent multi-user modeling in i-help. *User Modeling and User-Adapted Interaction* (13):179–210.

Winne, P. H., and Hadwin, A. F. 1998. Studying as self-regulated learning. *Metacognition in educational theory and practice* 277–304.

Winne, P.; Nesbit, J.; Kumar, V.; and Hadwin, A. Supporting self-regulated learning with gstudy softwrae: The learning kit project. *International Journal of Technology, Instruction, Cognition and Learning*. Accepted for publication.

Zimmerman, B. 2000. Self-regulatory cycles of learning. *Conceptions of self-directed learning, theoretical and conceptual considerations* 221–234.