

General Architecture Supporting Component-based EIS Interoperability

Darina Dicheva¹ and Lora Aroyo²

¹ Winston-Salem State University, Computer Science Department,
Martin Luther King Jr. Drive, Winston Salem, NC 27110, USA
dichevad@wssu.edu

² Eindhoven University of Technology,
Department of Mathematics and Computer Science,
P.O. Box 513, 5600 MB Eindhoven, The Netherlands
l.m.aroyo@tue.nl

Abstract. Web-based Educational Information Systems (EIS) aim at providing the learner with immediate, on-line access to a broad range of structured information in order to support more efficient educational task performance. Currently, more and more efforts are concentrating on bringing all those systems to work together in order to provide better support within the context of web-based education. Our goal is to approach the problem from a rather practical and somewhat minimalist perspective: to better utilize resources and components of already existing EIS we show how through communication protocols, we can realize a general modular architecture comprising components that can be shared and interchanged.

1 Introduction

There is an increasing interest to online learning support systems that are aimed at providing resources and functional components for various educational goals and tasks. Such systems often need to interoperate, collaborate and exchange content or re-use functionality, in order to support a richer set of educational functions and increase their effectiveness.

Adaptive Web-based Educational systems (AWBES) as introduced by Brusilovsky [1] use techniques from Intelligent Tutoring Systems (ITS) and Adaptive Hypermedia (AH) to support courseware authors in combining different systems with different purposes to provide for different aspects of the instructional process and achieve an integrated solution. Another way of achieving such integration is to build the separate systems using component-based architectures, where all the functionality is modularized and well encapsulated.

Within the class of web-based educational systems, a major role in various instructional contexts play the Educational Information Systems (EIS) that are aimed at providing intelligent, task-centered information support for solving problems and performing learning tasks. Consequently, considerable effort is currently focused on defining frameworks and architectures to tackle issues of information support from multiple perspectives. On the one hand, we do have the example of monolith Learning Management Systems (LMS), such as Blackboard and WebCT, which on more or less superficial level cover various teaching, learning, and administrative activities and as a result provide web-enhanced courses. On the other hand, we see multiple examples of specialized and effective educational systems and content providers, which support only one task/function within the entire educational process. Representatives of such systems are adaptive textbooks constructed with AHA! [2], InterBook [3] and NetCoach [4], or adaptive courses within ELM-ART [5], PAT Online [6] and AIMS [7]. There are also more global but still highly specialized efforts, such as ARIADNE and EdNa courseware-reusability frameworks that provide repositories of re-usable educational objects.

Brusilovsky [1] claims that all those systems need to be integrated and that the “university has a clear need in a single integrated system that can support all critical functions in one package”. While we

basically agree, we believe that it is not feasible to expect reaching the one-integrated-system goal in a near future and our claim here is that instead of a complete integrated system, what is currently needed is a good standardized way to allow different specialized systems *to talk to each other*. Such an approach from one side proposes a solution for his concern that “modern AWBES are designed to be used as a whole, not component by component” and from the other fits nicely to his advocacy of “distributed component-based architectures for building adaptive systems”.

In the current efforts targeting integration of various educational systems and content providers, Devezic [8] proposes educational servers (INES), which are based on using standards, ontologies, and pedagogical agents to support interaction between clients (authors and students) and servers (hosting educational content and services). He claims that the interaction in the future educational systems will be between learners and services through educational service directories. We argue that WBES interactions in the near to medium future will be between educational systems’ components - before achieving the large scale service-based integration, we need to explore and exploit possible communication between components.

Another service-oriented perspective on the integration is given by the Elena project [9], which defines a smart learning space of educational service providers based on the Edutella [10] peer-to-peer framework for interoperability and resource exchange between heterogeneous educational applications and different types of learning resource repositories. In the same context, we also see specific efforts trying to fill the gap between adaptive educational systems and dynamic learning repository networks, by proposing service-based architectures for personalized e-learning. An example is the Personal Learning Assistant [11], which uses Semantic Web technologies for realizing personalized learning support in distributed learning environments.

In this paper we try to approach the integration problem of the systems from a rather practical perspective and propose a general framework for supporting communication between ontology-based EIS aimed at utilizing systems’ resources and components. It employs some features of web services and agent-based frameworks, but is intended to be much simpler.

The paper is organized as follows. After identifying the common characteristics of concept-based EIS (Section 2), we outline the current needs and requirements for them to interact and share knowledge and resources (Section 3). In Section 4 we depict a general service-oriented framework to support the interoperability of various concept-based EIS. We conclude with a short discussion.

2 Ontology-driven Educational IS

The main goal of web-based EIS is to provide the learner, on the one hand, with immediate, on-line access to a broad range of structured information and on the other, with domain-related help in the context of her work, thus supporting more efficient task performance. There are a number of concept-based EIS already developed [3,4,5,7,11,12], which typically include:

- Concept-based (ontology-driven) subject domain
- Repository of learning resources (digital library)
- Course (learning task) presentation
- Adaptation & personalization

The fundamental feature of such systems is the subject *domain conceptualization*. It supports not only efficient implementation of required functionality but also standardization: the concept structure can be built to represent a domain ontology providing an agreed vocabulary for domain knowledge representation. Thus the ontology specifies the concepts to be included and how they are interrelated.

The repository contains learning resources (objects) related to the subject domain concepts. We can think of the resources as being *attached* to the domain concepts they describe, clarify, or use. One of the most prominent themes in ontology research is the construction of reusable components. If the attached objects have also a standards-based representation as opposed to a proprietary representation, this will insure that the application’s content is reusable, interchangeable, and interoperable.

Course/learning tasks are typically described in terms of subject domain concepts and some *instructional* relationships (such as ‘prerequisite’, ‘uses’, etc) between the involved concepts. The domain

concepts are also used as a basis for implementing systems' adaptive behavior. The latter involves constructing learner models in terms of domain concepts, performed tasks, and user characteristics.

An EIS user is typically involved in exploring the subject domain ontology and searching the repository for information related to a specific task. Good examples of such systems are AIMS [7] and TM4L (Topic Maps for Learning) [12], which we use as a basis of our discussion. AIMS and TM4L both focus on providing contextual support that enables learners to identify information necessary for performing a specific task (e.g. course assignment). They can be used standalone (for example, as an extension to a traditional or on-line distance course) or integrated in a larger electronic learning environment that allows the users to perform open learning tasks in a specific subject domain. Since both focus on efficient information provision and support for task-oriented problem solving, these systems are quite similar but they can be also seen as complementary in the way they support learning tasks. While AIMS includes course representation and sequencing, TM4L is a kind of digital library, which does not include direct course representation.

3 Need for EIS Communication

Integration and interoperability are very important for EIS systems. If interoperable, two systems can benefit of additional functionality supplied by the other, and especially of sharing resources and common components, e.g. user models. In our example of AIMS and TM4L, TM4L can use AIMS course sequencing model, graphical viewer, and resource metadata, while in turn AIMS can use TM4L external and internal resources, domain and resources merging capability, text and external search.

We start our discussion on EIS system communication with presenting two use case scenarios illustrating the communication between the two considered systems.

Scenario I: A learner uses AIMS as a support tool in her coursework. When trying to solve a specific task, however, she is not satisfied by the informational support provided by AIMS since it is not enough for her to achieve her learning goal. She announces this and AIMS seeks external help (more relevant information). It sends a request to TM4L for more learning resources on the topic at hand. TM4L responds to AIMS providing information that it has. AIMS feeds this information back to the learner. Fig. 1 illustrates this scenario by showing the internal organization of the two systems and the interface which combines the task-based search from AIMS with the resources from TM4L.

As we mentioned already, ontology-driven EIS have concept-based representation of the specific subject domain and learning resources *indexed* by domain concepts. Consequently, such a system would *understand* requests for information expressed in terms of domain concepts and relationships between them. Concerning a specific concept, possible requests of AIMS to TM4L would be:

- Give direct parents (children) of concept X (i.e. the concepts in a direct 'is-a' relation with it).
- Give all parents (children) of X (i.e. all concepts on the same path as X).
- Give all relations in which concept X plays a role (i.e. concept X is involved).
- Give all concepts related (in a relation of any kind) to a given concept.
- Give all available resources connected with this concept.
- Give learning resources of a kind (e.g. using the LOM standard) connected to concept X.
- Give all available information related to concept X (everything listed above).

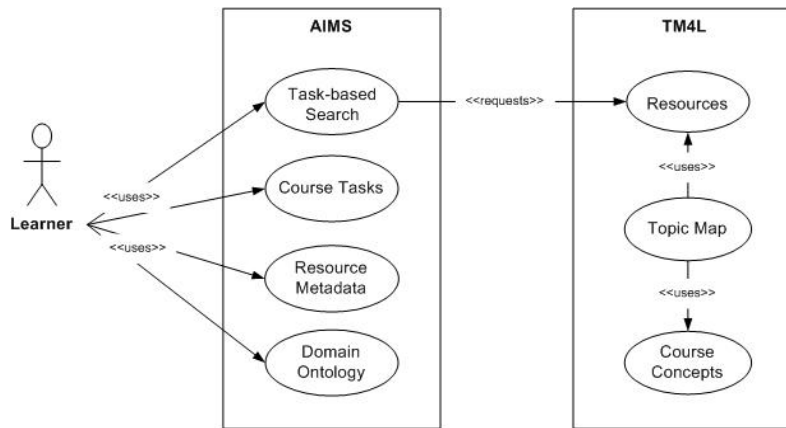


Fig. 1. AIMS – TM4L communication.

Scenario II: An author uses AIMS to construct a course. When preparing the course, she would like in addition to the learning resources she defines to reuse some other resources relevant to the course topics. She announces this to AIMS. It sends a request for information on specific topic(s) to TM4L. TM4L responds to AIMS providing information that it has. AIMS feeds this information back to the author to be approved for storing in the AIMS DM or disregarded.

Possible *requests* in this scenario include:

- Give all available resources connected with a concept X.
- Give learning resources of a kind (e.g. using the LOM standard) connected to concept X.
- Give all available information related to concept X.
- Import the entire domain ontology.

The communication between the two systems can be realized either at a system level or at a component level (in the second case between components with either identical or different functionality, e.g. domain models, user models, etc). The latter imposes additional requirements to the architecture of the involved systems – they should have a clear component-based architecture.

The main research questions related to implementing communication between the systems include:

1. *Level of granularity of information exchange:* what should be the information contained in one communication transaction? For example, in the second scenario, should the author be allowed to ask about importing the entire TM4L domain model in one transaction?
2. *Request semantics:* What kinds of questions the requesting system should be able to ask?
3. *Request syntax:* In what a form the questions should be expressed?
4. *Domain or user model awareness:* Should the requesting system send any indication about what it “knows” or its user already “knows”, so that the responding system doesn’t send information already known? If so, what kind of information and in what a form?

We attempt to answer these questions at two levels – a general one and a minimalist one - providing guidelines to the design of two corresponding frameworks for ontology-based EIS communication support. While the general one provides a powerful service-oriented framework to support efficient communication between component-based EIS, the minimalist one is intended to provide an efficient *current* solution for supporting shareability and exchangeability of systems resources. We started with defining the general framework with the intention to further constrain it to the desired minimalist one.

4 General Architecture for Component-based EIS Interoperability

If we consider the problem of efficient construction of EIS systems that complement (serve as advisors to) each other by sharing resources and components, the obvious answer is modularized building of such systems. This implies a component-based architecture that allows sharing knowledge (e.g. domain ontologies, learning resources, course models, and user models) and components (e.g. user modeling,

course sequencing, ontology visualization, keyword search). The systems will typically have different domain (and other) models and the interoperability between them should include support for:

- translating between systems' domain representations
- sharing learning sequencing components (educational context, results, etc.)
- sharing learning resources
- sharing user models.

4.1. The Architecture

In order to define such a general architecture we address the main research questions formulated in Section 3. With regard to the grain size of exchanged information (Question 1), we consider two issues as important for the communication quality and efficiency between the systems:

- Conciseness of the request and the reply.
- Completeness of the request and reply.

These factors suggest that a system request has to ask for precise information related to a particular user query (topic) and the reply is to be as specific and detailed as possible. This prompts us to propose that a finer level of granularity of information exchange is more appropriate. In some cases, however, as in Scenario II, we might require that the framework supports importing or merging of the complete (domain) models (e.g. TM4L DM merged with AIMS DM). We propose that this is realized through using appropriate communication support services and not through the *ordinary* request/response type of communication between the systems.

Concerning Question 2 from Section 3, in order to “understand” each other both systems must “know” the basic terms for structuring and using ontology-based learning resource repositories, such as ‘concept’, ‘relation/association’, ‘role in a relation’, ‘resource’, etc., which make the common ground for the semantic understanding. In other words, the different systems must know how to *map* their internal knowledge to the basic concepts of this common ground. This lightweight mapping process, as opposed to very expensive reasoning processes, is a key aspect of the proposed approach. To specify precisely the basic terms of this common ground we propose using a *communication ontology*, which we discuss in Section 4.2. It describes what exactly communication input/output can be, i.e. what requests will be allowed and what information will be returned.

Considering Question 3, we propose XML-based protocols, so that any application can “understand” them. In relation to Question 4, we propose that the systems share a common user model, possibly through using a user modeling service. In that case the responding system will not need to ask the requesting system what the user already knows. A common (shared) user model is only feasible since all EIS systems within the framework have concept-based representation of their subject domain.

Thus the proposed general architecture includes (see Figure 2):

- Stand-alone, component-based independent EIS using their private subject domain ontologies.
- Information brokerage bureau.
- Services to support systems communication.
- Information channels.

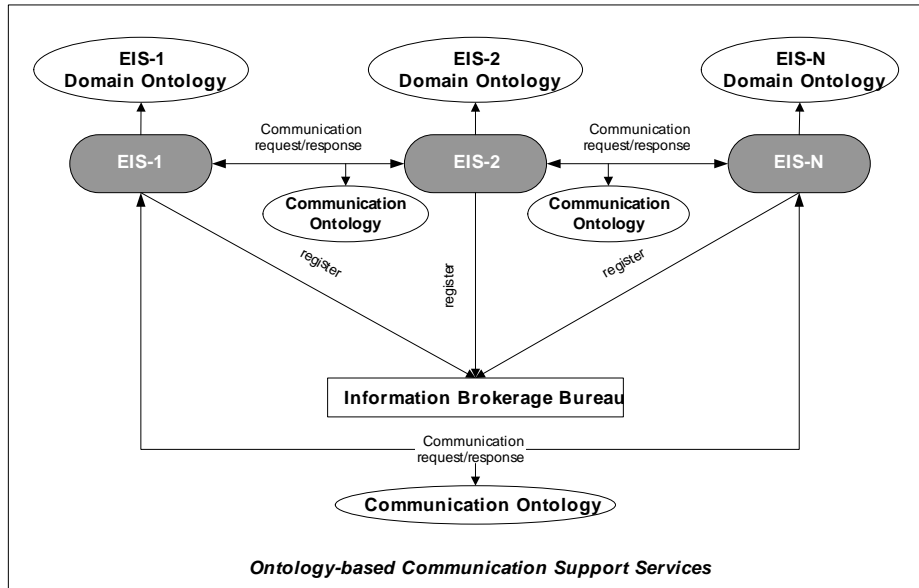


Figure 2. General architecture for component-based EIS interoperability.

The main purpose of the proposed architecture is to support sharing and exchanging information between EIS. This is achieved through communication between the systems (or their components) and services included in the framework to facilitate systems' communication. The services are intended to support different specific aspects of the communication. A significant class of services is ontology-related services since ontology alignment and translation are very important when different applications deal with different ontologies. This is exactly the case here as the different EIS in the framework have different domain ontologies.

A communication is an interaction between two software systems (agents) guided by an interaction protocol. The communication between the systems requires not only standardized transport mechanisms and communication languages, but also common *content languages* (see 4.2) and *semantics*.

Communication between the applications is supported by using *communication ontology* that defines the vocabulary of terms used in the messages at both layers: the message layer and the content layer (see 4.3). To interpret the requests and answers standardized domain ontologies, UM ontologies, as well as upper-level ontologies such as, WordNet, etc. can be used.

The information channels are “bridges” between the systems realizing the actual communication. They support standardized transport mechanisms and a common interaction protocol.

In order to “collaborate” the applications should register within “Information brokerage bureau”. So, when a system needs help, it sends a request to this agency and it distributes the request to the other registered systems. In this regard, the applications can be seen as software agents and their communication can be supported by using, for example, the Knowledge Query and Communication Language (KQML) [13].

4.2. Content Language

The popular languages used to represent the content, embedded in messages in ACL (Agent Communication Languages), such as KIF (Knowledge Interchange Format) [14], SL (Semantic Language proposed by FIPA) [15], and Prolog, are ‘logic’ content languages, which aim at representing knowledge as logic expressions. For our purposes, we consider more attractive ‘information’ content languages, i.e., languages that set rules to describe a particular type of information elements, since we don’t need to represent information as logic expressions. For this reason, we consider XML very appropriate to represent the content embedded in messages in our proposed architecture. The specified XML Schema can then correspond to the ontology of the messages, i.e., content ontology (see 4.3.1).

Since the different EIS (software agents) have to communicate, XML can be embedded within the messages of an ACL, such as KQML, which will be used for agent management.

4.3. Communication Ontology

To define the communication semantics, we propose a communication ontology, which consists of two parts corresponding to the two layers of an interaction, the message layer and the content layer:

- Content ontology - describes the content (knowledge) that can be exchanged by the systems (corresponds to the *content layer*).
- Interaction protocol ontology - specifies interaction communicative act types (corresponds to the *message layer*).

4.3.1 Content Ontology

Content ontology defines the concepts to exchange messages, i.e. gives the meaning of the symbols in the content expression. In our framework, the content ontology consists of two parts: the EIS domain ontologies and a domain-independent ontology describing the concept-based information model of EIS. The latter includes terms such as concept, concept name, relationship type, relationship role, etc.

4.3.2 Interaction Protocol Ontology

Interaction protocol (IP) ontologies describe the input and output data that are processed during protocol's execution together with the actions and the decisions that the agent (application) must perform [16]. A software application that has defined mapping between its internal code and actions and decisions in an ontology would then be able to interpret any IP that is defined with reference to that ontology. The IP ontology defines message types, reasons, and preconditions. While the communication content ontology is generally independent of the framework's functionality, the IP-ontology has to reflect its functionality (e.g. whether it supports agent communication).

Messages represent communicative acts denoting the actions related to communication. In general, communicative acts (performatives) include queries, responses, informational, capability definition, generative, and networking (see KQML [13]).

5 Conclusions

We approach the problems related to systems integration and communication by proposing a service-oriented framework to support efficient communication between component-based EIS. The communication semantics is defined by a communication ontology consisting of content ontology and interaction protocol ontology.

We believe that the proposed framework for supporting communication between applications will eliminate in many cases the need for exporting the entire DM or other application model to another application. Thus, this could be an alternative to interchanging and merging domain models. The advantage of this would be eliminating duplication of stored information, which is unlikely to be often used. In addition, if an application has a specific concept-based application model with no corresponding model in the other system (as is the case with the course model present in AIMS but not in TM4L), import would not work and this would be the only way that the second system (TM4L) could use information from the first one (AIMS). In this way the first system could complement the second one. We believe this would also solve problems with shareability and reusability for already developed applications that don't use standards-based information but rather their own internal representations.

The proposed general architecture can be constrained by considering only two communicating systems that "know" and "trust" each other. We consider this as a common case and will look to find the minimal configuration that will support communication and sharing of knowledge and resources between such systems. We believe that such a *minimalist* architecture will fill the gap between the current technology and realistic situation in the field of web-based educational information systems and the desired future educational semantic web.

Acknowledgements

We would like to thank Geert-Jan Houben for many valuable comments and suggestions on how to improve the paper.

References

1. Brusilovsky, P. (2004). KnowledgeTree: A Distributed Architecture for Adaptive E-Learning. In: Proc. of The Thirteenth International World Wide Web Conference, WWW 2004 (Alternate track papers and posters), New York, NY, ACM Press 104-113.
2. De Bra, P., Aerts, A., Berden, B., de Lange, B., Rousseau, B., Santic, T., Smits, D., & Stash, N. (2003). AHA! The Adaptive Hypermedia Architecture. In L. Carr, L. Hardman (Eds) Proceedings of the fourteenth ACM Conference on Hypertext and Hypermedia (81-85). New York: ACM Press.
3. Brusilovsky, P., Eklund, J., and Schwarz, E. (1998). Web-based education for all: A tool for developing adaptive courseware. Computer Networks and ISDN Systems. In Proceedings of 7th International WWW Conference, 30 (1-7), 291-300.
4. Weber, G., Kuhl, H.-C., and Weibelzahl, S. (2001). Developing adaptive Internet-based courses with the authoring system NetCoach. In Proceedings of the Third Workshop on Adaptive Hypermedia, AH'01.
5. Brusilovsky, P., Schwartz, E., Weber, G. (1996). ELM-ART: An Intelligent Tutoring System on the World Wide Web, Proceedings of the Third International Conference on Intelligent Tutoring Systems (ITS'96), pp. 261-269.
6. Ritter, S. (1997). PAT Online: A Model-Tracing Tutor on the World-Wide Web, Proceedings of the Workshop on Intelligent Educational Systems on the WWW, pp.11-17.
7. Aroyo, L., & Dicheva, D. (2001). AIMS: Learning and Teaching Support for WWW-based Education. International Journal for Continuing Engineering Education and Life-long Learning, 11(1/2), 152-164.
8. Devedzic, V. (2003). Next-generation Web-based Education. *International Journal for Continuing Engineering Education and Life-long Learning*, Vol. 11, No. 1/2, 232-247.
9. Simon, B., Nejd, W., Miklos, Z., Sintek, M., Salvachua, J. (2003). Elena: A Mediation Infrastructure for Educational Services. In Proceedings of 12th International WWW'03 Conference.
10. Nejd, W., Wolf, B., QuLearning, C. (2002). EDUTELLA: A P2P Networking Infrastructure Based on RDF. In Proceedings of 11th International WWW'02 Conference.
11. Dolog, P., Henze, N., Nejd, W., Sintek, M. (2004). Personalization in Distributed eLearning Environments. In Proceedings of 13th International WWW Conference
12. Dicheva D., Dichev C., Sun, Y., & Nao, S. (2004). A Topic Map Authoring Tool for E-Learning, Proceedings of ADMI' 2004, May 20-22, 2004, Orlando, Florida, 62-67.
13. Finin, T., Fritzson, R., McKay, D., McEntire, R. (1994). KQML as an agent communication language. In Proceedings of 3rd International Conference on Information and Knowledge Management, pp. 456-463
14. KIF (1998). Knowledge Interchange Format. Available at: <http://logic.stanford.edu/kif/dpans.html>
15. FIPA-SL (2004). FIPA SL Content Language Specification. Available at: <http://www.fipa.org/specs/fipa00008/SC00008I.html>
16. Cranefield S., Purvis M., Nowostawski M. and Hwang P. Ontologies for Interaction Protocols <http://www.fipa.org/docs/input/f-in-00076/f-in-00076.pdf>