Leaner Model Ontology and leaner Model Agent

Weiqin Chen and Riichiro Mizoguchi
Institute of Scientific and Industrial Research
Osaka University
8-1 Mihogaoka, Ibaraki, Osaka 567-0047, Japan
Email: {wendy, miz}@ei.sanken.osaka-u.ac.jp

Abstract: In this paper we describe the learner model ontology and learner model agent in a multi-agent architecture of learning support systems. The learner model ontology facilitates 1) sharable specification of the functionalities of learner model agent; 2) fluent communication between the learner model agent and other agents. The example of a learner model ontology and learner model agent shows that with the help of ontology, a standard for sharable and reusable agents in learning support systems may be established. The goal of the research presented in this paper is to exemplify the benefit of the learner model ontology through the development of an ontology-based learner model agent and the exchange of information between the learner model agent and other agents.

Biographical notes: Weiqin Chen received her Ph.D. from the Institute of Mathematics, Chinese Academy of Sciences in 1997. Currently she is an associate researcher in the Institute of Scientific and Industrial Research, Osaka University in Japan. She has been working in the domain of intelligence in education since 1995 and a member of the Asia-Pacific Chapter of AACE. Her research interests include educational science and technology, ontology theory and application, and artificial intelligence.
Riichiro Mizoguchi was born in Tokyo, Japan, on October 13, 1948. He received the B.S., M.S., and Ph.D. degrees from Osaka University, Osaka Japan, in 1972, 1974 and 1977, respectively. From 1978 to 1986 he was research associate in the Research Department of Electronics, the Institute of Scientific and Industrial Research, Osaka University. From 1986 to 1989 he was Associate Professor and he is currently Professor there. His research interests include Non-parametric data analyses, Knowledge-based systems, Ontological engineering and Intelligent learning support systems. Dr. Mizoguchi is a member of the Japanese Society for Artificial Intelligence, the Institute of Electronics, Information and Communication Engineers, the Information Processing Society of Japan, the Japanese Society for Information and Systems in Education, Intl. AI in Education (IAIED) Soc., AAAI, IEEE and APC of AACE, currently, President-Elect of IAIED Soc. and APC of AACE. He received honorable mention for the Pattern Recognition Society Award and the Institute of Electronics, Information and Communication Engineers Award, 10th Anniversary Paper Award from Japanese Society for Artificial Intelligence and Best paper Award of ICCE99 in 1985, 1988, 1996 and 1999, respectively.

1 Introduction

The last two decades have seen great progress in computer-based education. From CAI, ICAI and micro-world to ITS, ILE and CSCL, many systems have been built within each

of these paradigms. Moreover, it is believed that computer-based education will be a fact of life in the twenty-first century, and artificial intelligence will play a significant role in this emerging technology [Aiken&Epstein, 2000]. However, building a computer-based education systems still requires a lot of work because it is always built from scratch, and the knowledge and components embedded in those systems are hardly sharable and reusable. From an AI point of view, one of the major cause of the current situation is that most existing computer-based educational systems is lack of an explicit representation of the conceptualisation that each system is based upon [Mizoguchi&Bourdeau, 2000]. The implicit representation not only makes the components in these systems hard to share and reuse, but also makes it difficult for the modules and agents to communicate with each other fluently. To overcome these drawbacks, Mizoguchi and Bourdeau [Mizoguchi&Bourdeau, 2000] proposed the use of ontological engineering in education.

In philosophy, ontology is the study of the kinds of things that exist. In the AI community, the term ontology is used to refer to a set of representation vocabulary, and, more precisely, the conceptualisations that the terms in the vocabulary are intended to capture [Chandrasekaran, et. al, 1999]. In the computer-aided educational field, the vocabularies used in the learner model and teaching activities are organized in an educational ontology [Mizoguchi, et al., 1996]. Educational ontology helps to represent the functionality of components explicitly for reuse and sharing and to facilitate the communication between the components. The learner model ontology described in this paper has the above functions. It facilitates the explicit specification of the functionality of the learner model agent, the reuse and sharing of the learner model agent, and the fluent communication of the learner model agent. The focus of this paper is on the learner model information ontology, which gives external specification of the functionality of the learner model agent, while the internal structure of the learner model is out of the scope of this paper. The goal of the research presented here is to exemplify the benefit of the learner model ontology through the development of an ontology-based learner model agent and the exchange of information between the learner model agent and other agents.

This paper is organized as follows: Following this introduction, section 2 is devoted to the multi-agent architecture of learning support systems, where we describe the architecture of both a stand-alone learning support system and an open one. Once we achieve a better understanding of the multi-agent architecture of learning support systems, questions on the learner model agent and learner model ontology arise in the context of component reuse and sharing. Why do we need learner model ontology? What terms are needed to represent learner model information, specify the functionality of learner model agent and construct the exchanged messages? How can these terms be derived and organized? Some answers to these questions are provided in the sections 3 and 4. In section 5, an example of two systems sharing one learner model agent, based on the learner model ontology, is described along with the message exchanged. After a brief overview of the related work, we conclude by discussing the main issues concerning the design of learner model ontology.

2 Multi-agent architecture

It is obvious that the use of multi-agent architecture, itself, for learning support systems is not new. The concepts of an interface agent, a pedagogical agent [Johnson, 1998][Mengelle, et al., 1998] and a learner modeling agent [Paiva, 1996] have already been studied by many researchers. However, as a starting point of our research, by reviewing the various related works, we re-identified the following list of agents in general learning support (See Figure 1):

- Interface agent
- Learning material agent
- Learning support agent
- Learner modeling agent
- Learner model agent

Figure 1 A multi-agent architecture of learning support systems

(a) A stand-alone learning support system
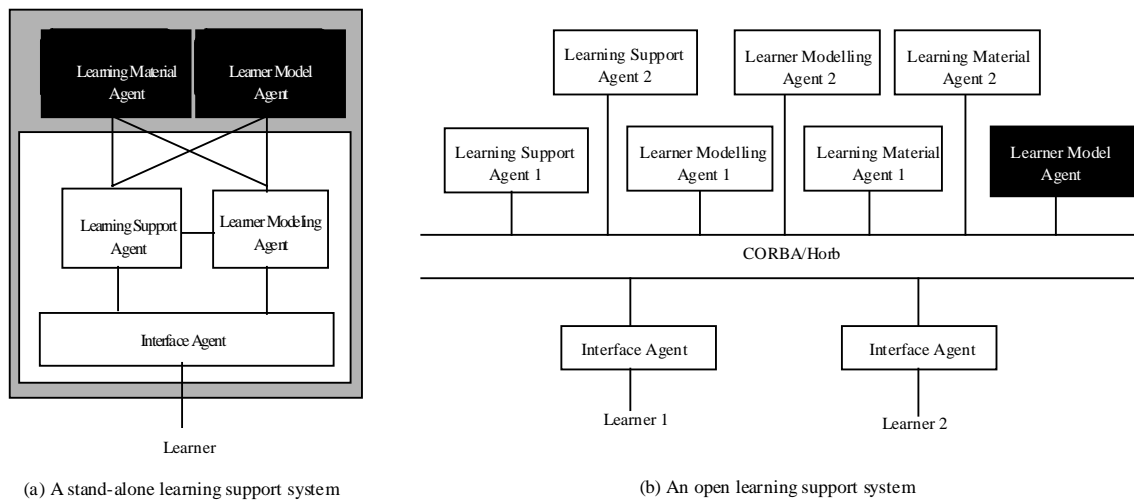
(b) An open learning support system

Figure 1a shows the architecture of a stand-alone multi-agent learner support system, while Figure 1b shows one of an open system. By open, we mean that each agent in the system can be used by other systems as a component, and agents from other systems may also be incorporated into this system as a component. In figure 1, the learning material agent takes charge of the content and organization of domain knowledge. It answers queries from other agents that need domain knowledge to accomplish their functions. For example, the learning support agent takes domain knowledge as its object to plan the instruction. The learning support agent is responsible for planning instructions and creating feedback. It selects appropriate topics to teach, and makes decisions on teaching strategies and learning support actions. The learner-modeling agent is responsible for the construction of a learner model. It derives the learner's cognitive information from the interaction between the learner and the system. When the learner-modeling agent needs extra information about the learner to construct a learner model, it asks the learning support agent to give related problems to the learner, so that the interface agent can transfer the learner's actions/answers to the learner-modeling agent. The interface agent is responsible for the interaction between a learner and the system. It monitors the learner's activities, and passes the messages to other related agents, such as the learner modeling agent and the learning support agent. The learner model agent is responsible for

answering queries from other agents about the information on the learner and the model itself.

Figure 2 Learner model agent
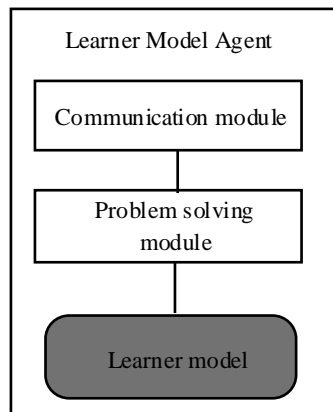
Each agent in Figure 1 consists of several major functional modules. An example learner model agent is shown in Figure 2. The communication module is common to all the agents. It controls the communication with other agents, including determining message performatives, sending out and receiving messages, and interpreting messages performatives.

The problem solving module performs the core functions of the learner model agent. It is responsible for answering other agents' queries. With an inference engine, it can infer answers from the content of the learner model. The main functions of the problem solving module are: 1) analyzing the message content and determining the task; 2) performing the task by reasoning using the learner model content; 3) constructing a message content for answering a query. The learner model contains both facts and abstract information (hypothesis about learner's understanding and mental state). The learner model agent can provide two kinds of information to other agents: 1) Information about the learner, including static information and dynamic information. 2) Information about the model, including the type of the model, the attributes of the model and the model representation.

In the open learning support system shown in figure 1b, one of the main problems is the communication between agents. Concerning this problem, John Self [Self, 1999] stated that:

"The technical questions concern enabling communication between the components, which involves translating between the different representations of the same object in different components. More generally, this leads to research on developing agreed ontologies of educational objects (the subject of AIED 99 workshop 2) to try to minimize, or at least standardize, these differences."

Educational ontology, therefore, plays an essential role in the open learning support systems. In the following sections, we will focus on the learner model ontology and its role in learning support systems.

3 Rationale for having a learner model ontology

The last several years have seen the focus shift in the field of AI from form-oriented research to content-oriented research. Researchers have been excited by the form-oriented research results, such as rule systems, frame languages, neural nets, fuzzy logic, and non-monotonic reasoning. However, they began to realize that however important the form is, it cannot do anything without the content of the field on which it is

to work. Ontology is the quintessence of content-oriented research. There are two kinds of ontologies—task ontology and domain ontology. A task ontology is a theory/system of vocabulary for modeling inherent problem-solving structure for all existing tasks, domain-independently. A domain ontology is a theory/system of vocabulary for specific classes of objects and relations that exist in some domains. In a multi-agent learning support system, the learner model ontology is a kind of task ontology, while the domain ontology is the vocabulary of the subject, such as mathematics, geometry, and chemistry. These two kinds of ontologies are both necessary for communication with the learner model agent. Due to the variety of subjects, in this paper we will focus only on the learner model ontology. Furthermore, since the goal of our research is to share and reuse the learner model agent, the learner model ontology we present here is only the learner model information ontology, which gives external specification of functionality of the learner model agent. The internal structure of the learner model is out of the scope of this paper.

In the multi-agent architecture of learning support systems shown in Figure 1, we separate the learner model agent from the learner-modelling agent. The reason for this is that we believe the learner model agent itself should be an independent entity since the information inside it may also be used by other agents or other systems or even by the learners themselves. In this sense, the learner model agent can be regarded as an agent, just like other agents in learning support systems. In so doing, three benefits are envisioned:
- The information in the learner model agent can be based on the interaction between learners and more than one learning support agent, possibly leading to richer learner model information.
- The information in the leaner model agent can be shared by multiple agents, which avoids the repeated acquisition of the same information.
- The learner model becomes independent of the modelling method, which makes the learner model agent more flexible.

The multi-agent architecture of learning support systems is based upon the assumption that agents share a common ontology. This ontology is used to specify the functionality of agents and support message construction and interpretation, thereby facilitating the reusability and sharability of the agents.

Concerning the learner model agent, the learner model ontology can facilitate:
- The explicit specification of functionality of the learner model agent, enabling other agents to know what they can expect from the learner model agent.
- Fluent communication between other agents and the learner model agent.
- Reusability and sharability of the learner model agent.

In an open learning support system, agents are designed by different programmers or organizations around various ontologies. For fruitful communication with the learner model agent, an explicit learner model ontology is necessary, together with a standard mechanism to access and refer to it.

4   Learner model information ontology

In order to build a learner model ontology, we followed the three steps below:
1. Enumerating the questions that the learner model agent should be able to answer

2. Categorizing the contents of questions

3. Extracting the domain-independent but task-dependent concepts and organize these concepts in an is-a hierarchy

The first question we asked was "what does the learning support agent want to know from the learner model agent?" In all we identified 43 questions that the learner model agent is expected to be able to answer (see Figure 3).

Figure 3 Queries supposed to be answered by learner model agent

Group 1. Queries on static information:
1. What is the name of a learner?
2. What is the ID of a learner
3. Is a learner a male or female?
4. How old is a learner?
5. What is the social status of a learner? (for language learning)
6. What is the education status of a learner? (no degree, bachelor, and master or doctor degree)
7. How motivated is a learner?
8. What experience does a learner have in a certain topic?
9. What's the learning style of a learner? (principle-oriented, example-oriented, general-to-specific or specific-to-general)
10. What kind of media does a learner prefer?
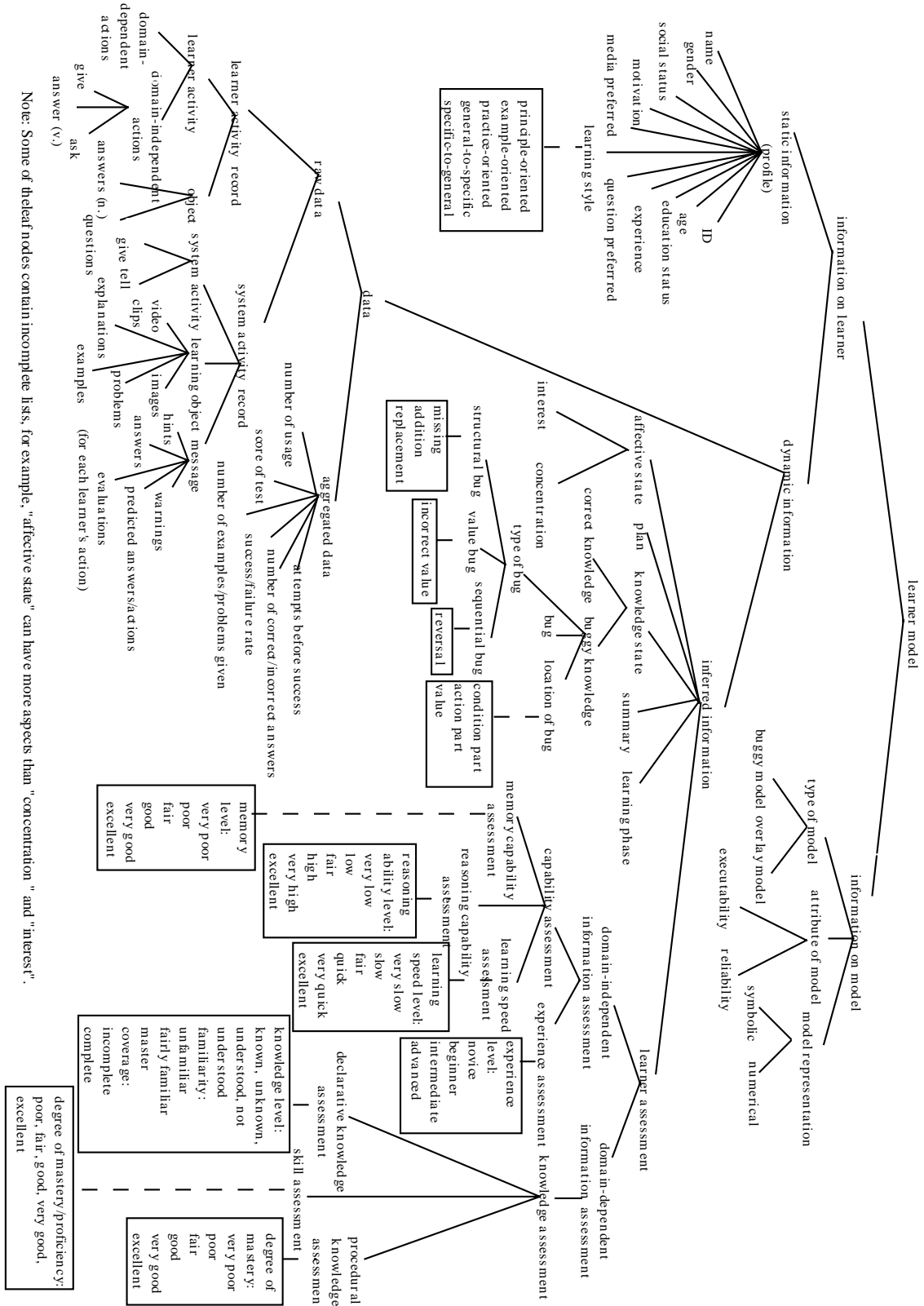11. What type of exercise does a learner prefer?

Group 2. Queries on interaction data:
1. What answer did a learner give to a certain question?
2. What question did a leaner ask?
3. What examples were given to a learner?
4. What problems were given to a learner?
5. What learning objects were given to a learner?
6. What score did a learner get in the test?
7. What is the average score of a learner?
8. How many examples were given to a learner?
9. How many problems were given to a learner?
10. How many times did a learner try before answering a question/solving a problem correctly?
11. What is the rate of success and failure?
12. How much time did a learner spend on a question?
13. What media does a learner use for interaction?
14. How many times did a learner use a specific knowledge unit?

Group 3. Queries on inferred information:
1. What knowledge has a learner mastered? / Has a learner mastered a certain topic?
2. What knowledge has a learner not mastered? / What knowledge has a learner missed?
3. Of what knowledge does a learner have incorrect understanding?
4. At which part of knowledge is a learner weak/strong?
5. What is the cause of a learner's error? / What is the bug?
6. What kind of bug does a learner have?

Figure 4 Learner model ontology

information on learner

learner model

information on model

static information (profile)

name
gender
social status
motivation
media preferred
learning style
question preferred
experience
education status
ID
age

principle-oriented
example-oriented
practice-oriented
general-to-specific
specific-to-general

dynamic information

raw data

data

inferred information

learner activity record

learner activity

domain-dependent actions

domain-independent actions

give
answer (v.)
ask
answers (n.)
questions
explanations
examples

object

system activity

system activity record

learning object message

give tell
video clips
images
hints
problems
answers
warnings
predicted answers/actions
evaluations
(for each learner's action)

number of usage
score of test
number of examples/problems given
number of correct/incorrect answers
attempts before success
success/failure rate

aggregated data

interest

affective state

concentration

plan

type of bug

bug
location of bug

structural bug
value bug
sequential bug

missing
addition
replacement

incorrect value

reversal

condition part
action part
value

correct knowledge
buggy knowledge

knowledge state
summary
learning phase

capability assessment

memory capability assessment

reasoning capability assessment

learning speed assessment

experience assessment

knowledge assessment

memory level:
very poor
poor
fair
good
very good
excellent

reasoning ability level:
very low
low
fair
high
very high
excellent

learning speed level:
very slow
slow
fair
quick
very quick
excellent

experience level:
novice
beginner
intermediate
advanced

domain-independent information assessment

domain-dependent information assessment

declarative knowledge assessment

procedural knowledge assessment

skill assessment

knowledge level:
known, unknown,
understood, not understood

familiarity:
unfamiliar
fairly familiar
master

coverage:
incomplete
complete

degree of mastery/proficiency:
poor, fair, good, very good,
excellent

degree of mastery:
very poor
poor
fair
good
very good
excellent

learner assessment

type of model
attribute of model
model representation

buggy model overlay model

executability reliability

symbolic numerical

Note: Some of the leaf nodes contain incomplete lists, for example, "affective state" can have more aspects than "concentration" and "interest".

7

7. Where is the bug?
8. What is the plan of a learner for problem solving?
9. What answer will a learner give to a certain problem / what will a learner do in his/her next action?
10. Which phase in a learning process is a learner in, initial acquisition, assimilation or mastery (through its use) of long-range process, or problem identification/formulation, hypothesis formation, or verification of short-range process?
11. How interested is a learner in a certain topic?
12. What is the learner's concentration like?
13. How is the overall performance of a learner?
14. How well does a learner master a certain topic?

Group 4. Queries on model information:
1. How confident is the system in evaluating a learner?
2. What type of model does a learner model agent have?
3. Is the learner model executable?
4. How reliable is the learner model?
5. How is the learner model represented? (by symbolic or numerical method)

Figure 4[*] is a detailed version of the concept hierarchy. It can be used as a well-structured, shared vocabulary for specification of the functionality of the learner model agent and construction of the messages exchanged between the learner model agent and other agents. As shown at the top level of Figure 4, information provided by the learner model agent can be in one of the two categories: information on the learner and information on the model. In the questions in Figure 3, the question asking about the reliability of the model (Group 4, #4) would be assigned to the category of information on the model, while question asking the name of the learner (Group 1, #1) belongs to information on the learner.

Information on the learner includes both static information, which is the profile of the learner, and the dynamic information, which refers to the information gained dynamically from the interaction between the learner and the system. In the dynamic information of the learner, there are data, which includes raw data (the learner's and the system's activity record) and aggregated data obtained from the raw data, and inferred information, which refers to the information obtained from the data by reasoning. The reasoning may be done at different levels. For example, from the learner's answer to a question given by the system, the learner's correct or buggy knowledge (bug, bug type and bug location) can be inferred. Furthermore, the ability of the learner (memory, learning speed, and reasoning capability) may be obtained.

With the domain-independent concepts, it is easy to represent a buggy model. Here is a portion of a buggy model represented with the concepts shown in Figure 4:

    Bug: 0-n =0
    Location of bug: action part

---

[*] Note that with the terms in Figure 4, we don't claim that this particular set of terms is the one shared by people. Rather, we show it here to provoke further thoughts and use it as a testbed of further elaboration.

Type of bug: incorrect value

In order to represent an overlay model, which is still popular in intelligent learning support systems, we need to combine the correct knowledge in the information of learner with the knowledge assessment, because the overlay model does not separate the learner model and the assessment of the learner for each learning item.

Obviously, a learner model agent must protect the privacy of learners. It must, therefore, have a facility to decide what kind of questions should be answered and what information can be provided.

5 Example

So far we have re-implemented two learning support systems, WUSOR [Carr&Goldstein, 1977] and IDDEBUGGY [Burton, 1982], which share one learner model agent (Fig. 5). WUSOR is a system coaching a student in the logical and probability skills required playing the computer game WUMPUS. IDDEBUGGY is a system to evaluate a learner's subtraction performance.

Figure 5 An example of two systems sharing one learner model agent.
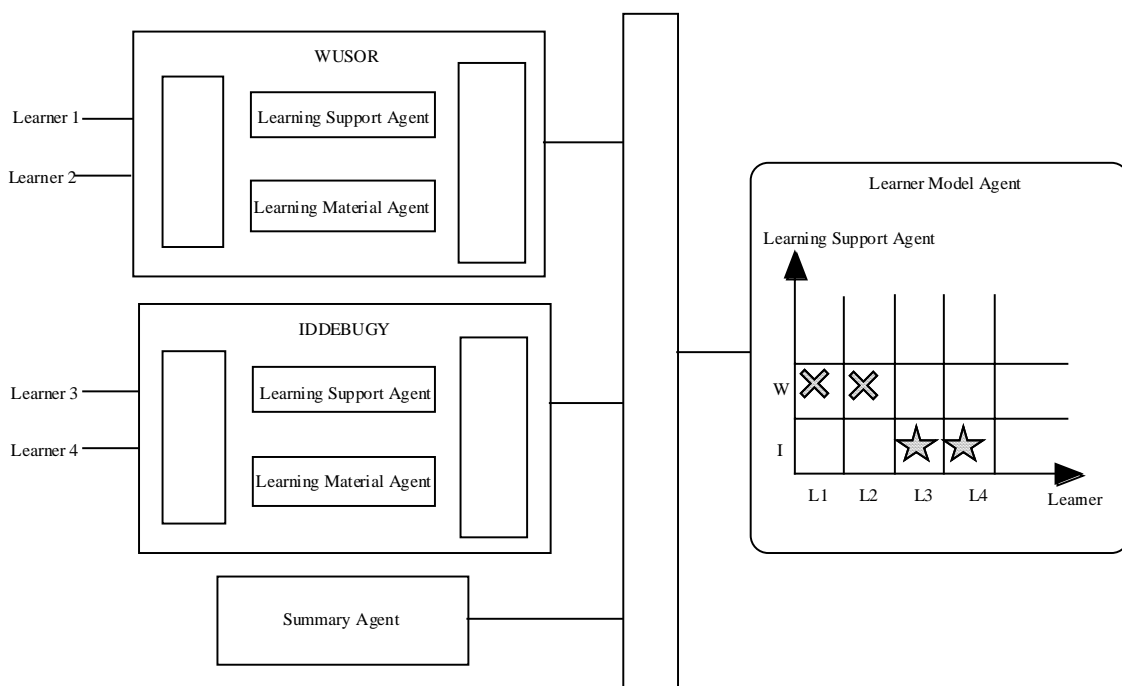
Our purpose is to show that how the learner model ontology supports the communication between the learner model agent and other agents. We choose these two systems as examples because they are famous for two different kinds of student model. WUSOR was the first system to use the overlay model, while IDDEBUGGY was the first system to use the buggy model. We have chosen these in order to cover the two most popular learner models.

As the first step of our implementation, we re-implemented WUSOR, where the communication occurs between agents within one system. Those agents run on different

computers. They exchange messages through a LAN with the communication managed by CORBA/Horb. In order for other agents to share one learner model agent, the leaner model agent itself has a mechanism to manage information on multiple learners and multiple agents. It has a two dimensional structure. The horizontal axis represents learners and the vertical axis represents learning support agents. Each entry represents information from a pair consisting of a specific learner and a specific learning support system. However the information in the entries is not independent. The entries in the same column share the subject-independent information, such as the static information (profile) of the learner.

After the multi-agent WUSOR was implemented, we developed another system, IDEBUGGY. It shares its learner model agent with WUSOR. For example, suppose the learner modeling agent in WUSOR has acquired the profile of a certain learner. When this learner registers the IDDEBUGY, the static information of this learner obtained from WUSOR will be automatically linked to this system by the learner model agent they are sharing. In fact all the subject-independent information on the learner can be reused by multiple systems. For the subject-dependent information on the learner, they can only be reused by systems of the same domain. This reusability helps to avoid the repeated acquisition of the same information.

When a learner registers one of the systems, which shares the learner model agent, the system will send a message to the learner model agent, informing the registration of the learner. The learner model agent will then link all the information related to the specific learner to this system. The linking is transparent to all the systems.

Since WUSOR and IDDEBUGGY adopt different learner models, the information they require from the learner model agent is different. For the overlay model in WUSOR, the information includes raw data, aggregated data, the correct knowledge and declarative knowledge assessment. For the buggy model in IDDEBUGY, the information includes raw data, buggy knowledge and procedural knowledge assessment. Although the two systems have covered more than half of the concepts in Figure 4, some concepts have still not been touched. In order to test the functionality of the learner model agent, we developed one pseudo-agent, which is supposed to require the information not being covered by the two systems. We name this pseudo-agent as summary agent, because it is mainly used to make a summary of the learner's information.

Suppose a learner has registered in WUSOR. The learner modeling agent constructs an overlay model for this learner. During the learning process, the information under the categories of static information, data, correct knowledge and the knowledge assessment are obtained. The learning support agent in WUSOR uses the information to make decision on instructional actions. Later the learner registers in IDDEBUGGY. The learner model agent links the static information of the learner obtained from WUSOR to the IDDEBUGGY. During the learning process within IDDEBUGGY, he is given 10 subtraction problems. From his problem-solving actions, the learner modeling agent in IDDEBUGGY records the activities of the learner and infers his bugs and put the information into the learner model agent. To make a decision on the next instructional

action, the learning support agent in IDDEBUGGY asks the current situations of the learner. The information may under the categories of data (problems shown to the learner) and buggy knowledge (bug). Now it turns to the function of summary agent. The summary agent generates the summary information of the learner according to the information provided by the learner model agent. For this reason, it sends several queries to the learner model agent asking for the information needed to make a summary. The following summary for Learner L in IDDEBUGGY is an example generated by the summary agent:

> Learner L was given 10 problems. He solved problem 1, 3 and 5 correctly, and others incorrectly. From the incorrect answers he provided, it is quite possible that he has bug(s): No.1 (0-n=n) and No. 3 (smaller from larger), and he might have bug(s): No. 2 (adding instead of subtraction). The degree of mastery is poor.

As developers, during the implementation of the two systems, apart from the learner material ontology (domain ontology), we consider only one set of terms for the learner model—learner model ontology. The learning support agents in these two systems commit themselves to the learner model ontology. They construct messages based on the learner model ontology.

Once the learner model ontology has been explicitly incorporated into the communication, it can be used to construct the message content, one of the key points of communication. In our research, we incorporate KQML as the agent communication language. The following example shows the format of a KQML message.

```
(ask
    :sender      agentA
    :receiver    agentB
    :reply-with  messageNo
    :ontology    common_ontology
    :content     "the learning style of learner L1")
```

The message in the example starts with the word ``ask", which is the action (performative) intended for the message. The remainder of the message contains keywords needed for the message interpretation. Keywords used in KQML messages are defined as follows [Finin, et al, 1994]:

sender: agent sending the message.
receiver: agent receiving the message.
in-reply-to: identifier of the message that triggered this message submission.
reply-with: identifier to be used by a message replying to this message.
ontology: identifies the ontology to interpret the information in the content field of this message. In our context, the ontologies are learner model ontology and learning material ontology.
content: context-specific information describing the specifics of this message.

In the following part of this section, we give some examples of communication messages in the multi-agent architecture of learning support systems. With these examples we show how the learner model ontology is used in message content construction.

The communication process starts when the learning support agent sends a message to the learner model agent, asking about the learning style of a learner. In this example, we assume that the type of the learner model is a simple overlay model.

Message1, learning support agent—>learner model agent,
"What is the **learning style** of <u>learner</u> <L1>?"
      (ask   :sender learning-support-agent
                    :receiver learner-model-agent
                    :reply-with m001
                    :ontology learner-model-ontology
                    :content (val (learning-style (Learner learner1))))
Message2, learner model agent—>learning support agent,
"The **learning style** of <u>learner</u> <L1> is **example-oriented**."
      (tell   :sender learner-model-agent
                    :receiver learning-support-agent
                    :in-reply-to m001
                    :ontology learner-model-ontology
                    :content (= (val (learning-style (Learner learner1))) example-oriented))
Note: Since the learner model ontology is not used in Message 3 and 4, the example messages in KQML format are omitted.
Message3, learning support agent—>interface agent
"Show a *multiple-choice question* <M1> of *topic* <T1> to <u>learner</u> <L1>."
Message4, interface agent—>learner modelling agent
"The **answer** of <u>learner</u> <L1> to the *multiple-choice question* <M1> is <A1>."
Message5, learning support agent—>learner model agent
"What is the **mastery degree** of <u>learner</u> <L1> on *topic* <T1>?"
      (ask   :sender learning-support-agent
                    :receiver learner-model-agent
                    :reply-with m002
                    :ontology learner-model-ontology+learning-material-ontology
                    :content (val (mastery-degree (Learner learner1) (topic T1))))
Message6, learner model agent—>learning support agent
"The **mastery degree** of <u>learner</u> <L1> of *topic* <T1> is **low."**
      (tell   :sender learner-model-agent
                    :receiver learning-support-agent
                    :in-reply-to m002
                    :ontology learner-model-ontology+learning-material-ontology
                    :content (= (val (mastery-degree (Learner learner1) (topic T1))) low))
Note that in the following message, we will skip the examples in KQML format.
Message7, learning support agent—>learning material agent,
"What's the *example* of *topic* <T1>?"
Message8, learning material agent—>learning support agent,

"*Topic* <T1> has an *example* <E1>."
Message9, learning support agent—>interface agent
"Show *example* <E1> of *topic* <T1> to <u>learner</u> <L1>."

Note that in the above example messages, the **bold words** such as **learning style**, **example-oriented**, **mastery degree**, **answer**, and **low** are defined in the learner model ontology. Among them, the **example-oriented** and **low** are instances of values. The *italic words* such as *topic*, *multi-choice question* and *example* are defined in the learning material ontology, and <T1>, <M1> and <E1> are instances of them, respectively. The <u>underlined words</u> (<u>learner</u> in the examples) are defined in the ontology of participant of learning support systems. In the messages, not only the concept names (learning style, answer, mastery degree, etc) but also the values of the attributes (example-oriented, low) are included.

6 Related work

Concerning the ontology-aware multi-agent architecture of learning support systems, and reusable learner model, the following related work is worthy of note.

In ITS'96 Workshop on Architecture and Methods for Designing cost-effective and Reusable ITSs, Paiva [Paiva, 1996a] discussed the communication between user/learner modeling agents and application agents. She described the learner modeling tasks, which can be regarded as the functions of learner modeling agent. She raised three main problems with the communication between the user/learner modeling agent and the application agents: the protocol problem, the communicative act problem and the ontology problem. She proposed to address the first two problems with the KQML and KIF languages respectively. She claims that the third problem is more complex because it involves defining conceptualization and language which both the applications and the user/learner modeling system should be able to share. She, therefore, did not consider the third problem in her paper. However, this paper did make an important step toward a communication consensus between user/learner modeling system and application agents.

In CIKM'95 Workshop on Intelligent Information agents, Cheikes [Cheikes, 1995a] presented an agent-based architecture for intelligent tutoring systems--GIA (a Generic Instructional Architecture). He hypothesized that Intelligent Tutoring Systems (ITSs) can be effectively decomposed into collections of independent agents that collaborate and exchange information using an expressive formal language. In GIA, he adopted a federated architecture for agent communication including three types of agents: a facilitator, a kernel agent and an interface agent. The kernel agent, consisting of five independent agents, provides the essential services of intelligent tutoring systems. In order for the agents to collaborate and exchange information, GIA adopts a subset of KQML performatives. With respect to the ontology problem, Cheikes defined four ontologies: the system ontology, the pedagogy ontology, the domain ontology, and the student ontology, which can capture properties of ITSs and domains of instruction and can be reused across many different ITS applications. His work on the agent architecture of ITS and the definitions of the four types of ontology is also valuable, although he did not give details about the conceptualisation.

Ritter and Koedinger [Ritter & Koedinger, 1996] are attempting to build learning environments that incorporate tutoring elements into pre-existing software packages. Their basic idea is similar to ours in the sense that they intend to move toward a set of standards for tutor agents that interact with complex tools. The communication between the tutor agent and tools is domain-independent. They accomplished this domain-independent communication by attaching an accompanying translator, which is domain-dependent, to each of the tools. With the help of the translator, tools in different domains are able to share the same tutor agent. Similar work was done by Peter Brusilovsky [Brusilovsky, 1994], where he presents a student model centred intelligent learning environment. Different components of intelligent learning environment including tutoring, coaching, environment, and manual components use the central student model to adapt their behaviours to the given student. To achieve this flexibility, a projector was built upon the centred student model. Different components use the student information through different projections.

Judy Kay [Kay, 1999] is also working on reusable and scrutable student models based on *um*[Kay, 1995]. She believes that a reusable student model demands core ontologies that are common to all uses of the components in a domain. The core ontologies she mentioned are domain-dependent, and all the concepts in the ontologies belong to subject domains.

The Foundation for Intelligent Physical Agents (FIPA) is working on agent-based application. Similar to work of IEEE Learning Technology Standards Committee (LTSC), the goal of the FIPA research is to achieve internationally agreed specifications that maximize interoperability across agent-based applications, services and equipment. Concerning the user personalization service (UPS) in human-agent interaction [FIPA 98], their aim is similar to ours in the sense that they try to construct a user model based on interaction with multiple agents, and to make user model available to multiple agents. Although they claimed that a user model ontology is necessary for UPSto interpret the user model contents and enable content transfer between user models, there is no explicit representation of what terms are included in user model ontology, apart from the profile of user and the type of model.

BGP-MS [Kobsa&Pohl, 1995] is a user modeling shell system that can assist interactive software systems in adapting to their current users by taking the users' presumed knowledge, beliefs and goals into account. Concerning the communication with applications, BGP-MS defined message labels such as *bgp-ms-tell, bgp-ms-ask, bgp-ms-answer, alert-from-bgp-ms, d-act,* and *interview-response*, which are very similar to KQML performatives. The content of the message is formulated using BGDL (Belief and Goal Description Language) defined by BGP-MS. The communication is carried on via KN-IPCMS, a platform-independent message-oriented communication protocol. Although in BGDL, there are domain-independent terms such as *misconception, inferred-assumption, and answer*, which can be regarded as a subset of learner model ontology, these terms are not well-organized nor explicitly defined. The strength of BGP-MS lies in the general modeling mechanism rather than the explicit representation of the learner model information. Another possible drawback of BGP-MS

might be that it does not allow multiple applications to share the learner model information at the same time.

7 Conclusion

We have described the learner model agent and learner model ontology in multi-agent architecture of learning support systems. The learner model agent is separated from other agents, as an independent agent, for its independence and reusability. We focused on the terms necessary for the external specification of the functionality of the learner model agent instead of its internal structure. In the construction of the learner model ontology, 43 questions, which the learner model agent should be able to answer, were identified. We then categorized the questions and extracted domain-independent and educational-task dependent concepts, and organized them into an ISA hierarchy, which constitute the learner model ontology. These concepts can be used to compose all of the messages necessary to answer the queries in Figure 3. With these concepts defined in learner model ontology, the learner model agent can be reused and shared by other agents, and the communication is made fluent. The role of the learner model ontology in learning support systems was shown with two examples.

We would like to draw several general conclusions from the preceding discussion:

- The learner model ontology helps to specify the functionality of the learner model agent.
- The learner model ontology facilitates the fluent communication between the learner model agent and other agents.

With the example of the learner model ontology, we argue for the continued exploration of the learning support system ontology—explicit conceptualisation in learning support systems. The problems are hard and closely related to the functions of agents in the architecture of learning support systems. The study of the learner model ontology in this paper is still in an early stage. However, it gives us enough confidence and encouragement to continue our research on the construction, implementation and evaluation of learning support system ontologies.

References
Aiken, R. B. & Epstein, R. G. (2000) Ethical guideline for AI in education: starting a conversation. *International Journal of Artificial Intelligence in Education*, vol. 11, to appear.

Brusilovsky, P. (1994) Student model centered architecture for intelligent learning environments. In *Proc. of UM94*, pp. 31-36.

Burton, R. R. (1982) Diagnosing bugs in a simple procedural skill. In D. Sleeman and J. S. Brown (Eds.). *Intelligent Tutoring Systems*. London, UK: Academic Press, pp. 157-184.

Brown, J. S. & Burton, R. R. (1978) Diagnostic models for procedural bugs in basic mathematical skills. *Cognitive Science*, vol. 2, pp. 155-192.

Carr, B. & Goldstein, I. (1977) Overlays: a theory of modelling for computer-aided instruction. *International Journal of Man-Machine Studies,* vol. 5, pp.215-236.

Cheikes, B. (1995a) GIA: An agent-based architecture for intelligent tutoring systems. In *Proc. CIKM'95 Workshop on Intelligent Information Agents*.

Cheikers, B. (1995b) Should ITS designers be looking for a few good agents? In *Proc. AIED'95 Workshop on Authoring Shells for Intelligent Tutoring Systems*.

Chen, W., Hayashi, Y., Jin, L., Ikeda, M., & Mizoguchi, R. (1998) An ontology-based intelligent authoring tool. In *Proc. of ICCE'98.* vol.1, pp.41-49, Beijing, China.

McCalla G., & Greer, J. (1994) Granularity-based reasoning and belief revision in student models. In *Student Modeling: the Key to Individualized Knowledge-Based Instruction*, pp. 39-62, Spring-Verlag.

Finin, T., Fritzson, R., McKay, D., & McEntire, R (1994) KQML as an agent communication language. In *Proc. of the Third International Conference on Information and Knowledge Management (CIKM'94)*, ACM Press.

Genesereth, M., & Fikes, R. (1992) Knowledge interchange format, Version 3.0 *Reference Manual*, Knowledge Systems Laboratory, KSL-92-86.

Genesereth, M., & Ketchpel, S. (1994) Software agents. *Communication of ACM* vol. 37, no. 7, pp. 48-53.

Gruber, T. (1993) A translation approach to portable ontology specifications. *Knowledge Acquisition*, vol. 5, no. 2, pp. 199-220.

Ikeda, M., & Mizoguchi, R. (1994) FITS: A framework for ITS--A computational model of tutoring, *International Journal of Artificial Intelligence in Education*, vol. 5, no. 3, pp. 319-348.

Johnson, L. (1998) Pedagogical agents. In *Proc. of ICCE'98*, vol. 1, pp. 13-22, Beijing, China.

Kay, J. (1995) The um toolkit for cooperative user modelling. *User Modeling and User-Adaptive Interaction*, vol. 4, pp. 149-196.

Kay, J. (1999) Ontologies for reusable and scrutable student model. In Proc. of AIED99 workshop on Ontologies for Intelligent Educational Systems, pp. 72-77.

Kobsa, A. & Pohl, W. (1995) The user modelling shell system BGP-MS. *User Modeling and User-Adaptive Interaction*, vol. 4, pp. 59-106.

Machado, I., Martins, A., & Paiva, A. (1999) One for All and All in One - A learner modelling server in a multi-agent platform. In J. Kay (Eds.) *Proc. of um99*, Springer Wien New York, pp. 211-221.

Mengelle, T., De Lean, C., & Frasson, C. (1998) Teaching and learning with intelligent agents: actors. In Goettl, B. P., Halff, H. M., Redfield, C. L., & Shute, V. J. (Eds.) *Intelligent Tutoring Systems*, pp. 284-293, Lecture Notes in Computer Science 1452, Springer.

Mizoguchi, R., Sinita, K., & Ikeda, M. (1996) Task ontology design for intelligent educational/training systems. In *Proc. ITS'96 Workshop on Architectures and Methods for Designing Cost-Effective and Reusable ITSs*, Montreal.

Mizoguchi, R. (1998) A step towards ontological engineering, http://ei.sanken.osaka-u.ac.jp/english/step-onteng.html.

Mizoguchi, R. & Bourdeau, J. (2000) Using ontological engineering to overcome common AI-ED problems. *International Journal of Artificial Intelligence in Education,* vol. 11, to appear.

Murry, T. (1996) Toward a conceptual vocabulary for intelligent tutoring systems, http://www.cs.umass.edu/~tmurray/papers/conceptual_vocab/conceptual_vocab.html.

Paiva, A. (1996a) Towards a consensus on the communication between user modelling agents and application agents. In Proc. of *UM'96 Workshop on Standardization of User Modeling*.

Paiva, A. (1996b) Communication with learner modelling agents. In *Proc. ITS'96 Workshop on Architectures and Methods for Designing Cost-Effective and Reusable ITSs*, Montreal.

Ritter, S., & Koedinger, K. (1996) An architecture for plug-in tutor agents. *International Journal of Artificial Intelligence in Education*, vol. 7, no. 3/4, pp. 315-347.

Ritter, S. (1997) Communication, cooperation, and competition among multiple tutor agents". In B. du Boulay and R. Mizoguchi (Eds.) *Artificial Intelligence in Education,* IOS Press, pp. 31-38.

Self, J. (1999) Open Sesame?: fifteen variations on the theme of openness in learning environments. *International Journal of Artificial Intelligence in Education,* vol. 10*,* pp. 1020-1029.

FIPA (1998) FIPA 98 Specification, part 8, version 1.0. Human-agent interaction. http://www.fipa.org