Object-Oriented Recommendation (OOR) for Ubiquitous Learning Environments

Moushir M. EL-BISHOUTY[a], Hiroaki OGATA[a] ,
Gerardo AYALA[b] and Yoneo YANO[a]
a) Dept. of Information Science and Intelligent Systems, Tokushima University,
2-1, Minami-Josanjima, Tokushima 770-8506, Japan,
Tel:+81-886-56-7495/FAX:+81-886-23-2761.
b) Universidad de las Américas, Puebla, México.
moushir.elbishouty@gmail.com

Abstract: This paper presents a novel recommendation methodology for learners who are doing tasks in ubiquitous learning environments. The main concept is to deal with a learning task as a set of real objects and to orient the recommendation methodology towards these objects. These objects can be detected using ubiquitous technologies. An intelligent agent can propose diverse recommendations for the learners based on the objects' attributes. These recommendations include assistance from peers, educational materials and learning tasks. In order to do this, the agent constructs and maintains a learner model based on the learner's experiences in using the real objects that conform a task. The proposed ubiquitous learning environment model, for learner modeling and recommendations, is a belief system implemented in DLV under the ASP (Answer Sets Programming) paradigm.

Keywords: Object Oriented Recommendation, Ubiquitous Environments, Intelligent Agent, CSCL, Real Object, Learner Model, DLV, ASP.

Introduction

Educational technology is growing rapidly in order to satisfy the learner' needs. Learning at anytime and anywhere is one of the strongest trends that researchers focus on. Ubiquitous computing is the method of enhancing the computer use by making many computers available throughout the physical environment, but making them effectively invisible to the user. A ubiquitous computing environment enables people to learn at anytime and anywhere.

The most profound technologies are those that disappear. They weave themselves into the fabric of everyday life until they are indistinguishable from it [1]. Ubiquitous computing environments utilize a large number of cooperative small nodes with computing and/or communication capabilities such as handheld terminals, smart mobile phones, sensor network nodes, contact-less smart cards, and RFID (Radio Frequency Identification) etc. [2]. The RFID tag makes it possible to tag almost everything, replace the barcode, help computers to be aware of their surrounding objects and detect the user's context [3]. The RFID system consists of a tag, which is made up of a microchip with an antenna, and an interrogator or reader with an antenna. The reader sends out electromagnetic waves. The tag antenna is tuned to receive these waves. The chip then modulates the waves that the tag sends back to the reader and the reader converts the new waves into digital data [4].

In the *learning by doing* [5] model, teachers identify a specific set of skills to be learned by the learners, and then embed that skills in a learning task, activity, or a goal that the student will find it interesting or motivational. The teachers can evaluate the learner's understanding and skills according to how much the learner accomplishes the learning task and successes to reach to the task goal.

Whenever learners do a task, they use one or more real objects to do it, and to accomplish the task successfully; they should know how to use these objects. Sometimes, a learner could not complete the task because of a lack of knowledge about how to use one or more objects. In that case, he will be in need for getting recommendations in order to find the required knowledge to complete the task successfully. Using the ubiquitous technologies to detect these objects allows the leaner to get recommendations based on his environment without the need of the leaner to define that environment by himself. We have proposed before one of such a Computer Supported Ubiquitous Learning (CSUL) environment, which is called *PERKAM* [6], and it was evaluated in learning *PC assembling* as an example.

In that experiment, a teacher of computer hardware course asked the students to do tasks regarding *PC assembling*. Each student was given many hardware components, and for each component there was RFID tag attached to identify it. The teacher supplied each learner with PDA connected to the wireless LAN and equipped with RF reader. The teacher defined a fixed time to complete the task and to reach the final goal that the computer was working probably. Whenever the student found a difficulty to accomplish the task, he asked the system to recommend a number of related educational materials to refer and peers, whose interests and experiences were matching the information collected from the detected objects, in order to collaborate and complete the task.

1. Object-Oriented Recommendation (OOR)

Object-Oriented Recommendation refers to the construction of personalized recommendations of learning tasks, educational materials and peer assistance for a specific learner that are oriented towards learning how to use the real objects in the domain, rather than performing a very specific task. These objects can be detected using ubiquitous technologies. An intelligent agent can propose recommendations for the learners based on the objects' attributes. This differs from the definition of a task in the PHelpS system [7], where a task is represented as a hierarchical set of steps. Therefore, we would like to define the following terms:

- *Object*: it is the surrounding real object that the learner uses to accomplish a certain task. It could be a basic tool or an instrument. In this last case, a compound object is considered as a single object.
- *Object Attribute*: it defines the object specifications that affect on the way of using this object. In case of the *printer* object for example, its attributes will be *model*, *connection type*, *paper size*, etc.
- *Task*: it is represented as a set of real objects that the learner uses to accomplish a goal.
- *Task goal*: it measures the success of using a set of objects that the task consists of. It depends on the application domain. For example, in case of *PC assembling* application, the task goal is to let the computer works without any problem.
- *Learner model*: it consists of a set of beliefs about the objects that a certain user has used them whether successfully or unsuccessfully.

Utilizing RFID technology, as one of the important ubiquitous technologies, allows the agent to *perceive* the learner's environment and to *act* upon that environment through *effectors*. We believe that in the near future; RFID tags will be attached on almost all surrounding objects. Therefore, we would like to assume the existence of the following facts:

- ✓ RFID tag is attached on any object.
- ✓ RFID tag data is consistent anywhere.

As we defined a task as a set of objects, the similarity between two tasks is based on the similarity between their objects, which can be calculated from their attributes. Therefore, we can make a recommending based on that similarity.

2. Using ASP for Modeling CSUL Environments

Answer Set Programming (ASP) has been recognized as an important contribution in the areas of Logic Programming and Artificial Intelligence [8]. Nowadays, there is a significant amount of research work inspired on this formalism, well known and accepted for non-monotonic reasoning. ASP is more expressive than normal (disjunction free) logic programming and it allows us to deal with the uncertainty. Because beliefs cannot be treated correctly in classical logic programming (PROLOG), ASP is considered as a logical framework for agent modeling. ASP allows us to have a direct and a clear representation of the beliefs of an agent especially in dynamic situations, incomplete information and the uncertainty.

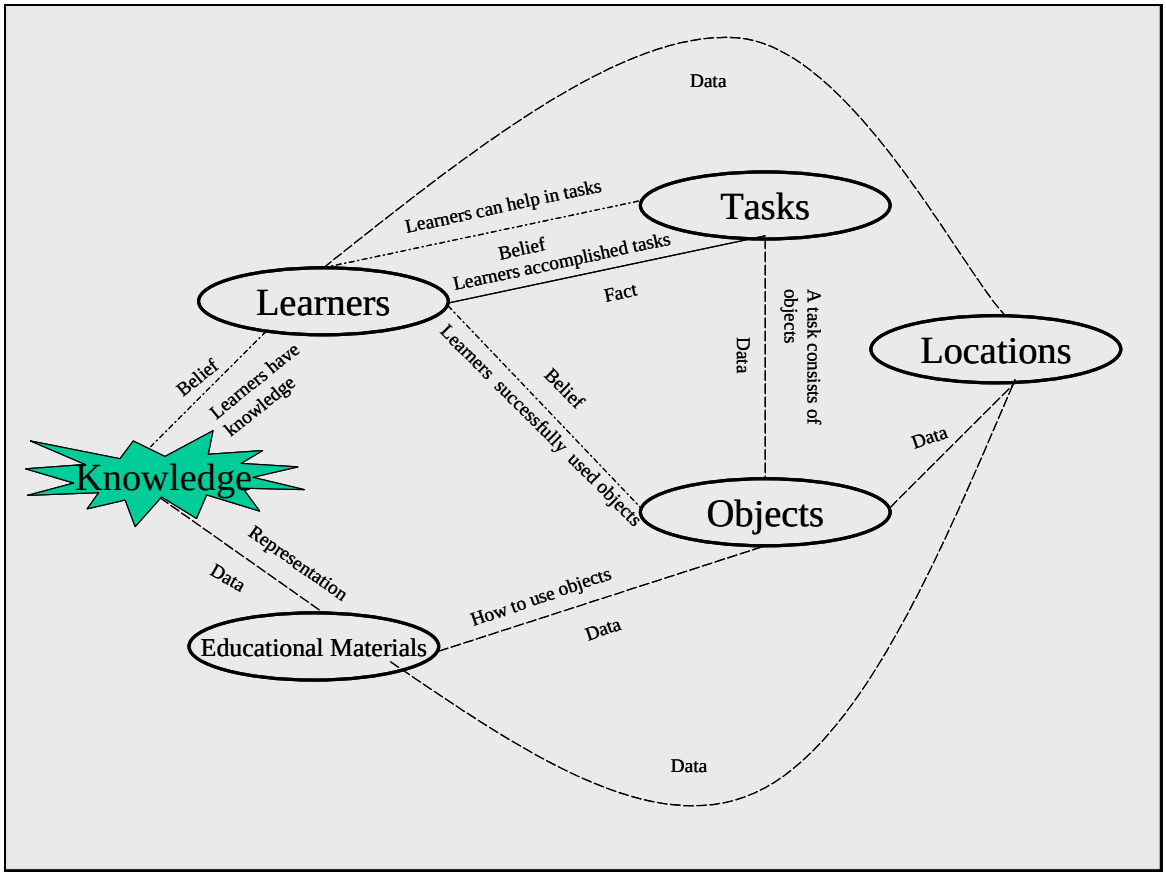

Figure 1. The Object-Oriented Recommendation model

DLV is a system for ASP [9]. The belief system for our agent is implemented in DLV. The result of the computation is a *set of models;* each one is a consistent explanation of the world, as far as the system can derive it. DLV tries to find all the models of the world, which correctly and consistently explain the basic beliefs and belief derivation rules of the program. The model assigns a truth-value to each atom appearing in the program, and it is represented as a set of atoms that are true in a certain model. A model is considered an *epistemic state of the agent*, as a closed theory under the logic defined by a program [10]. Each model is interpreted as implicit (derivable) and explicit beliefs of the agent. Ayala, G. [11] defined the advantages of applying DLV as a logical framework for the agent modeling in Computer Supports Collaborative Learning (CSCL) environments.

3. Object-Oriented Recommendation Model

Analytic models are considered a way to model group activities. According to Hoppe & Ploetzner [12], an analytic model, in the context of collaborative learning environments, is "a formally represented computational artifact that can be used to simulate, reconstruct, or analyze aspects of actions or inter-actions occurring in groups".

We have developed the model in DLV for generating derived beliefs concerning the recommendations that the agent proposes to the learner. The model implies a set of rules for belief derivation, a set of basic beliefs and data corresponding to the objects and the tasks. In addition, the using of *double negation* facilitates the derivation with incomplete information. The agent believes that the learner can successfully use an object if it believes that he has successfully used it before and there is *no* evidence that the learner has failed in using it. Figure 1 illustrated the model, its elements, beliefs, data and the relation between them.

3.1 Objects

The object is identified by the predicate **object**(*objID*). The objects are classified into a number of classes where each object belongs to one class according to the object type. The class of objects is represented by the predicate **objectClass**(*objID,class name*). Each class has its own attributes, **classAttribute**(*class name, attribute description*), and a set of critical attributes is *inherited* from its attributes, **criticalAttribute**(*attribute description*). Based on the hierarchy of the classes of objects, the similarity between *Object* and *AnotherObject* is calculated by the following rules:

similarity(*Object, AnotherObject, allSameValuesOfAttributes*):-
    sameClass(Object, AnotherObject),
    not oneAttributeIsDifferent(*Object, AnotherObject*).

similarity(*Object, AnotherObject, allSameValuesOfCriticalAttributes*):-
    sameClass(Object, AnotherObject),
    oneAttributeIsDifferent(*Object, AnotherObject*),
    not oneCriticalAttributeIsDifferent(*Object, AnotherObject*).

3.2 Tasks

A task consists of a set of objects. The task is identified by the predicate **task**(*taskID, objID*). The task can be either predefined in the system by an expert, or defined by the learner by reading the RFID tags of a set of objects while doing a new task according to the learner interest in using that set of objects. In the last case, we call it a *free task*, which defined by the predicate **freeTask**(*taskID, objID*). In order to let it easy to represent our model, we will consider only the predefined task.

3.3 Educational Materials

The education materials represent the required knowledge to use the objects. The education material is identified by the predicate **educational Material**(*emID,type*). The *type* parameter specifies the educational material form, for example hardcopy, softcopy, website or video. The predicate **howToUse**(*objID,emID*) defines the relation between the object and the education material which contains the required knowledge to use it.

3.4 Learners

The agent identifies a learner by the predicate **learner**(*learnerId*). A set of beliefs about the learner's capabilities of doing a certain task is based on the fact that the learner already successfully accomplished other tasks where these tasks contain the set of objects (or similar objects), which that task consists of. In that case, the agent believes that its learner successfully used that set of objects, and he has the required knowledge to use it that are represented by the following rule:

canUseSuccessfully(*LEARNER, OBJECT*):-
    task(*TASK,OBJECT*),
    accomplished(*LEARNER,TASK*).

On the other hand the agent believes that the learner might not use an object successfully if he did not successfully accomplished a task, which contains this object. In addition, the agent infers that an object may be the problem or may be not the problem in order to accomplish the task.

3.5 Locations

The current physical locations of the objects, the education materials and the learners are defined by the predicate **location**(*id, location*). The information of the current location is received from the RFID data.

4. Object-Oriented Recommendation Methodology

4.1 Peer Assistance Recommendation

The agent recommends assistance from a peer (a free learner who is not engaged in doing any task) for another learner who is doing a task, when the agent believes that the recommended peer has the required knowledge “experience” about how to use the same or similar objects that the learner’s task consists of. We have two categories of the recommended peer: the *expert*, who has successfully used all these objects and we do not have evidence that he has failed in using any of them, and the *familiar*, who has successfully used all of these objects, and at the same time, the agent believes that the peer may unsuccessfully have used one or more of these objects. The DLV rules are:

assistanceRecommendation(*expert, OBJECT, PEER*):-
    canUseSuccessfully(*PEER, OBJECT*),
    not maybeCannotUseSuccessfully(*PEER, OBJECT*),
    free(*PEER*).

assistanceRecommendation(*familiar,OBJECT, PEER*):-
canUseSuccessfully(*PEER, OBJECT*),
maybeCannotUseSuccessfully(*PEER, OBJECT*),
free(*PEER*).

4.2 Task Recommendation

The agent recommends a task for a learner to do if he is not engaged in doing another task. The recommendation of a task mainly based on the current location of the learner and the availability of the objects in that location. We have four categories of the recommended tasks according to the agent beliefs about the learner's knowledge of using the task objects. They are *repair*, *checkIfObjectIsProblem*, c*heckIfObjectIsNotTheProblem* and *tryNewObject.* The first two categories, as examples, are represented in the following rules:

taskRecommendation(*repair,LEARNER,TASK*):-
-**accomplished**(*LEARNER,TASK*),
not notAllObjectsInLearnerLocation(*LEARNER, TASK*),
free(*LEARNER*).

taskRecommendation(*checkIfObjectIsProblem,LEARNER,TASK*):-
maybeTheProblem(*LEARNER, OBJECT*),
task(*TASK, OBJECT*),
not accomplished(*LEARNER,TASK*),
not notAllObjectsInLearnerLocation(*LEARNER, TASK*),
free(*LEARNER*).

The availability of peers for the recommended task is considered while recommending all the previous categories to be aware of the possibility of finding peer assistance for the recommended task, which is represented in the following rule:

taskRecommendationPossibleHelp(*LEARNER,CRITERIA,TASK,OTHERLEARNER*):-
taskRecommendation(*CRITERIA,LEARNER,TASK*),
accomplished*(OTHERLEARNER, TASK*),
learner(*OTHERLEARNER*),
LEARNER <> OTHERLEARNER,
free(*OTHERLEARNER*).

5. Evaluation

Our DLV model is tested and controlled by *NORIKO*, a ***NO****n-monotonic* ***R****easoning software component for* ***I****ntelligent* ***K****nowledge awareness and recommendations* ***O****n the move*, which we have developed in Java. *NORIKO* allows the Java programmer to add and remove beliefs, as well as making queries concerning the beliefs of the model. *NORIKO* calls DLV automatically in order to revise the beliefs every time a new belief is added or removed.

In order to evaluate the consistency of the model and the efficacy of the recommendation methodology, we have performed 10 simulations; each one consisted of 8 iterations. There were 8 predefined tasks, 16 objects and 4 learners. In each iteration, a simulated learner could select and do one of the recommended tasks, propose and perform a free task by selecting a set of objects, or just do nothing. A simulated learner might accomplish or fail to do a task.

In figure 2, x-axis represents the number of iterations (time) and y-axis represents the number of the recommended tasks with diverse criteria. It is clear that the number of the recommended tasks was increasing with respect to the time; it implies that the agent was always able to present a diversity of learning possibilities. On the other hand, in figure 3,

y-axis represents the number of the recommended tasks with possible help. The number of the recommended tasks, where somebody could assist, was increasing with respect to the time; it implies that the agent always proposed tasks where a peer could provide an assistant.

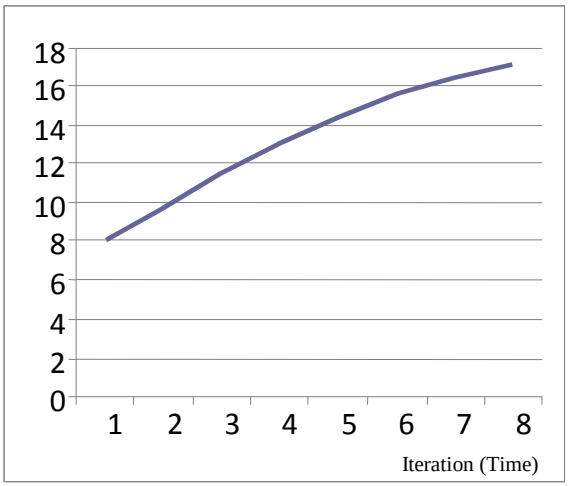

Figure 2. Task recommendations with diverse criteria.

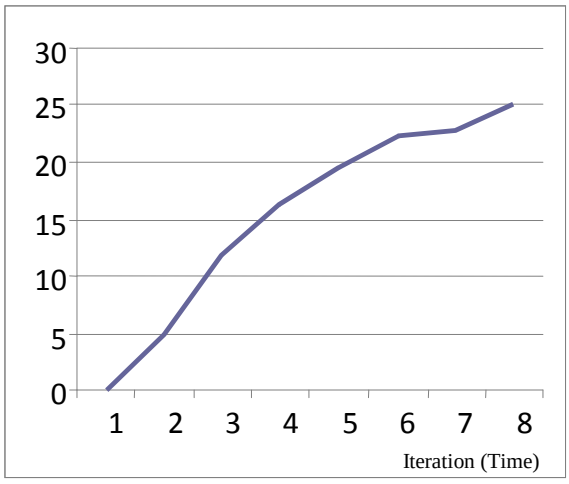

Figure 3. Task recommendations with possible help.

In figure 4, y-axis represents the number of objects that the agents believed that the learners could be recommended as peers in order to assist in using these objects. The increment became more stable when the learners tended to accomplish the same tasks using the same objects.

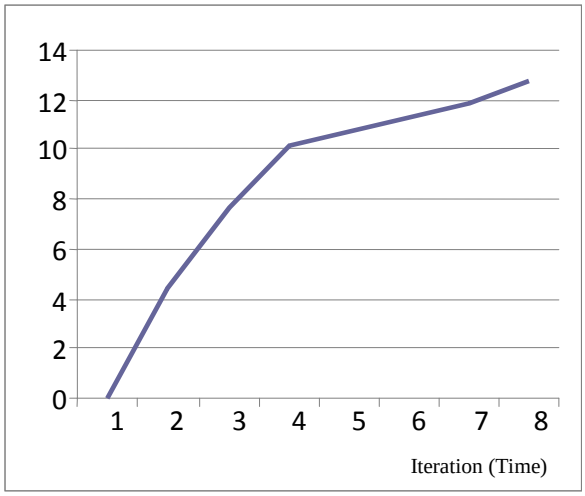

Figure 4. Assistance recommendations

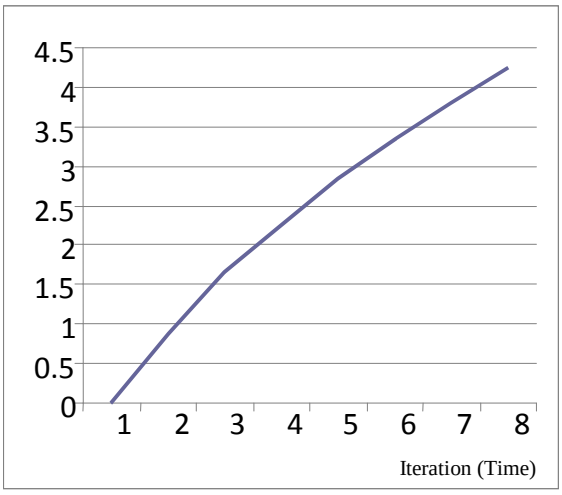

Figure 5. Accomplished tasks

Figure 5 presents the number of the accomplished tasks by the learners. This graph just represents the task accomplishment rate in the simulations, it shows an increasing in the rate of the accomplished tasks by the simulated learners.

6. Conclusions

This paper presents a novel recommendation methodology for learners who are doing tasks in ubiquitous learning environments, which we called *Object-Oriented Recommendation* (OOR). What we mean by the term *Object-Oriented Recommendation* is to deal with a learning task as a set of real objects and to orient the recommendation methodology towards these objects. These objects can be detected using ubiquitous technologies; therefore an agent can propose recommendations for the learners based on the objects' attributes without the need of the leaner to define that environment by himself. The objects are classified into a

number of classes, where each class has its own attributes, and a set of critical attributes is *inherited* from its attributes.

The recommendation of a peer assistant for another learner, who is doing a task, is based on his agent beliefs that the recommended peer has the required knowledge about how to use the same or similar objects that the learner's task consists of. The recommendation of a task mainly based on the current location of the learner and the availability of the objects in that location. We have four categories of the recommended task according to the agent beliefs about the learner's knowledge of using the task objects. They vary from repairing objects to trying new objects. Also, the availability of peer assistance for the recommended task is considered while recommending all the previous categories of tasks. The environment model is implemented in DLV system under ASP paradigm.

We have performed a simulation in order to evaluate the consistency of the model and the efficacy of the recommendation methodology. It indicated an obvious increasing in the recommendations, which implies that the agent always presented a diversity of learning possibilities. On the other hand, it shows an increasing in the rate of the accomplished tasks by the simulated learners.

7. Future Plan of Work

We are planning to include education material recommendations in this model, such as educational videos. On the other hand, we will continue the evaluation stage of this model. The evaluation stage consists of two phases; the first phase is designed to perform several simulations to measure the consistency and the efficacy of the model. The second phase is designed to measure the improving of the learners' skills in using the objects and doing the tasks from one side, and the learners' satisfaction of using the system from the other side.

References

[1] Weiser, M. (1991). The computer for the twenty-first century. Scientific American, September, 94-104.

[2] Sakamura, K. and Koshizuka, N. (2005). Ubiquitous Computing Technologies for Ubiquitous Learning. Proceeding of the International Workshop on Wireless and Mobile Technologies in Education, Japan, 11-18.

[3] Borriello, G. (2005). RFID: Tagging the World, Communications of the ACM, vol.48, No.9, pp.34-37.

[4] Klaus, F. and Rachel, W. (2000). RFID Handbook: Radio-Frequency Identification Fundamentals and Applications, John Wiley & Sons.

[5] Schank, C. (1995). What We Learn When We Learn by Doing. (Technical Report No. 60). Northwestern University, Institute for Learning Sciences.

[6] El-Bishouty, M. M., Ogata, H., & Yano, Y. (2007). PERKAM: Personalized Knowledge Awareness Map for Computer Supported Ubiquitous Learning. Educational Technology & Society, 10 (3), 122-134.

[7] Greer, J., McCalla, G., Collins, J., Kumar, V., Meagher, P., and Vassileva, J. (1998). Supporting Peer Help and Collaboration in Distributed Workplace Environments. International Journal of Artificial Intelligence in Education 9, 159-177.

[8] Baral, C. (2003). Knowledge Representation, Reasoning and Declarative, Problem Solving. Cambridge University Press.

[9] Bihlmeyer, R., Faber,W., Ielpa, G. & Pfeifer, G. (2004). DLV- User Manual. www.dbai.tuwien.ac.at/proj/dlv/man/

[10] Ortiz, M., Ayala, G. & Osorio, M. (2003). Formalizing the Learner Model for CSCL Environments. Proceedings of the Fourth Mexican International Conference on Computer Science, IEEE Computer Society and Mexican Society for Computer Science, 151-158.

[11] Ayala, G. (2005). An Analytic Model based on ASP for Maintaining ZPDs in CSCL Environments**,** Towards Sustainable and Scalable Educational Innovations Informed by the Learning Sciences, Frontiers in Artificial Intelligence and Applications 133 (Chee-Kit Looi, David Jonassen and Mitsuru Ikeda, Eds.) IOS Press, Amsterdam, pp. 3-10.

[12] Hoppe, U. H. & Ploetzner, R. (1998). Can Analytic models Support Learning in Groups. www.collide.info/Lehre/GestaltungInteraktiverLehrLernsysteme/downloads/esf98f.pdf.