Is What You Write What You Get?: An Operational Model of Training Scenario

Yusuke Hayashi[1], Mitsuru Ikeda[1], Kazuhisa Seta[1],
Osamu Kakusho[2], and Riichiro Mizoguchi[1]

[1] The Institute of Scientific and Industrial Research, Osaka University
8-1, Mihogaoka, Ibaraki, Osaka, 5670047, Japan
`{hayashi, ikeda, seta, miz}@ei.sanken.osaka-u.ac.jp`
[2] Faculty of Economics and Information Science, Hyogo University
`kakusho@humans-kc.hyogo-dai.ac.jp`

Abstract. To meet the needs for large-scale, high-quality learning contents, needless to say, we have to sharpen authoring tools. Authoring process can be roughly divided into two phases, a composing phase and a verification phase. A great deal of effort has been made on the support in the former phase. What seems to be lacking, however, is that in the latter. An ontology-aware authoring tool we have been developing has a function called "Conceptual level simulation. " This supports authors in the latter phase by showing the behavior of learning contents not only as a sequence of concrete behavior but also as structured and abstract behavior along the design intention. Ontology lays the foundation for the function by explicating operational and conceptual semantics of a training scenario.

1. Introduction

Contents-oriented research comes to attract considerable attention in information engineering field. The trend has been accelerated with broad diffusion of multimedia and internet technologies. In the research field on educational systems, the transitions from story-board type to knowledge-based type, from individual type to collaborative type and from tutoring type to learning environment type, are symbolic of the trend. Large-scale, high-quality learning contents are becoming one of the critical needs of technetronic society. To meet the needs, needless to say, we have to sharpen our tools to produce high-quality learning contents in a large scale, because the quality of the learning contents depends not only on the author's ability but also on authoring tool's performance. In fact, many researchers address this issue from a variety of viewpoints [6].

Authoring process can be roughly divided into two phases, a composing phase and a verification phase. Existing authoring tools support both phases of authors' work to a some extent. However, compared with the support performance for the former phase, one for the latter does not seem very helpful to the author. A typical support function for the latter is to provide a behavior-level test bed where authors can examine their learning contents step by step along the control structure. In general, of course, it is helpful and absolutely necessary. However, it is not very helpful to

resolve the logical drawback of learning contents. On analogy of programming, it could be described as "semantic-error debugging. " Shapiro explains the hardness of debugging [7]:

A program is a collection of assumptions, which can be arbitrarily complex; its behavior is a consequence of these assumptions; therefore we cannot, in general, anticipate all the possible behaviors of a give program.

This is true in case of authoring of learning contents as well. The key to lightening the hardness of debugging is to shift the load to maintain the design assumptions from authors to authoring tools. To realize this, new functions of an authoring tool to be developed include

− A framework for authors to describe design assumptions including design intention of learning contents.
− A function to show the behavior of learning contents not only as a sequence of concrete behavior but also as structured and abstract behavior along the design intention.

We call the latter as "conceptual-level simulation. "Our idea is that the structured information generated based on author's design intention will lighten the author's major debugging load to compare what he/she thinks (design intention) with what he/she gets (behavior). We use the term "design intention" to prevent confusion of it with more general term "design rationale". Generally, design rationale includes reasons behind design decisions, justification for them, other alternatives considered, the trade-offs evaluated, and the argumentation that led to the decision [3]. Intuitively, design intention is a part of design rationale: limited to reasons behind a design decision and justification for it. Reasons behind concrete learning contents are represented as a hierarchical structure of instructional goals. Justifications for the structure are teaching strategies or pedagogical principles used for hierarchical arrangement of the goals. The benefits of using design intention is that authors can enjoy services to record, maintain, or access their "design intention" behind learning contents and it can thus improve reuse and maintenance of the contents.

An ontology [5] plays an important role to embody the above idea. One of the most important roles of ontology is to lay the theoretical foundation for educational system development process. It maintains continuity from authors conceptual understanding of an educational task including design intention to the computational semantics of educational systems [1]. It provides human friendly vocabulary/concepts for authors to describe the learning contents along with design intention. For the authoring tools, on the other hand, it specifies the operational semantics of the learning contents. This operationality enables the conceptual-level simulation of learning contents. Based on this idea, we have developed an ontology-aware authoring tool SmartTrainer/AT[2, 4].

2. Ontology-Aware Authoring Tool

2.1 Composing a model

Basically, an ontology is a set of definitions of concepts and relationships and a model is a set of instances of them. Roughly speaking, the role of an ontology is to direct the

authors towards the correct model. Our idea is that an ontology-aware authoring tool can help authors to reduce the problems of authoring caused by unintentional error and to improve the quality of the product. Our research on SmartTrainer/AT is an embodiment of this idea. We have developed a training task ontology and incorporated it into SmartTrainer/AT as fundamental knowledge source to yield the intelligent functions to support the authoring process.

The authors' task is to write a "training scenario" for SmartTrainer that is a training system engine we have developed. At the appropriate phase of authoring process, the author is required to clarify his/her own idea from the three fundamental viewpoints listed below.

− What type of learner the scenario of the teaching material is designed for?
− What educational effect the teaching material is supposed to bring about?
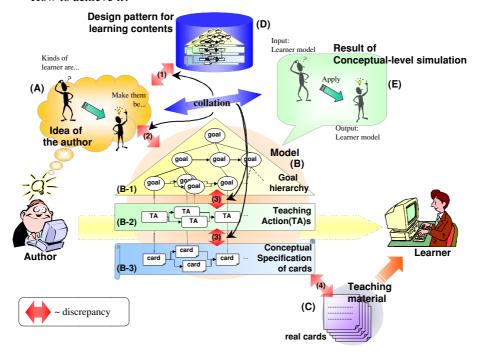− How to achieve it?

Fig. 1. An overview of authoringprocess

Fig.1 shows how the idea (A) is embodied in the teaching material (C). The model (B) can be regarded as a representation of design process. An ontology provides vocabulary and concepts, axioms necessary to describe the model. Firstly, an author describes the idea (A) clearly as a topmost, abstract and instructional goal. Then he/she repeats the expansion of the super-goal into relatively concrete sub-goals until a sequence of the sufficiently concrete goals (B-1) is specified. Secondly, he/she designs a sequence of teaching actions (B-2) which are expected to attain the goals (B-1). Thirdly, he/she embodies the actions (B-2) in a sequence of conceptual specification of cards (B-3).

In (B-1) there are two kinds of instructional goals: a goal for diagnosis[1] (D-goal) and a goal for teaching/learning (T/L-goal). A D-goal is to identify the state of a learner. The structure of the D-goals behind a training scenario implies the classification of the learners assumed by the author. By analyzing the structure, we can know the type of the learner supposed by the author at a certain context of the training scenario. A T/L-goal is to make educational effects on learners. Thus, the author can clarify the last two of the three viewpoints discussed above while describing the model (B-1) using two kinds of goals. The D-goal represents the type of the learner to whom the teaching material is designed for and the T/L-goal represents the educational effect the teaching material is supposed to bring about. At the bottom two levels (B-2) and (B-3), the teaching scenario is characterized from the third viewpoint: "how to achieve it. " The author clarifies how to attain the goal in (B-1) by selecting an appropriate teaching action for the goal (B-2) and specifying the target topics and the level of learner intended for the card (B-3). When the author is very active in referring the ontology, the rationality of the training scenario designed is expected to be quite high. In addition, reusability and sharability of training scenario is also expected to be high, as we have discussed, because an ontology-aware authoring tool stores not only concrete teaching material (C) but also the design intentions and rationale behind it as a model (B) based on the ontology.

2.2 Conceptual-level simulation

As we have seen in the previous section, an ontology-aware authoring tool is expected to be able to bridge the potential conceptual gap between ideas (A), and presentations (C), and suppress the unintended error caused by the gap. Models (B) play the important role as a pivot between ideas (A) and presentation (C): as conceptual representation of ideas or as the design intention behind presentation. Needless to say, the error cannot be suppressed completely and there happen to be discrepancies (1)..(4) during design process as shown in Fig.1. To resolve the - and be close to perfection, it is very important to identify the location of the discrepancies that are not realized by the author during the design process. However, everyone knows the "debugging" is difficult to do and requires huge efforts, because of the conceptual gap between what he/she thinks during composing phase (A) and what he/she observes during verification phase (C). If it is possible for the authoring tool to bridge the gap, the cost of "debugging" can be reduced considerably. As we have discussed, ontology-aware authoring tool knows the model (B) to bridge the gap and can provide the information about the difference between what an author writes and what he/she gets. It can be a good cue to identify the discrepancies. The function of our ontology-aware authoring tool is called conceptual level simulation (E) which shows the behavior of the training scenario. Conceptual-level simulation can demonstrate the behavior of the training scenario from various viewpoints, along the structure of the design model and may expose the three categories of problems caused by discrepancies (1)..(3) to the author's eye. In the case of forth one, it is rather helpless

---

1 In the research area of ITSs, the term "diagnosis" implies the intelligent reasoning process to identify the cause of a wrong answer. However, diagnosis process adopted in SmartTrainer is rather simple. It carries out diagnosis based on simple association patterns of wrong answers with erroneous knowledge.

to resolve the problem caused by discrepancy (4) because conceptual-level model is too abstract to evaluate the quality of real contents. In this section, firstly, we briefly summarize a debugging aid of conventional programming environments. Then, in contrast with it, we will discuss the advanced feature of debug support function of our ontology-aware authoring tool.

2.2.1 Debugger

A debugger of conventional programming environments provides the various functions for authors to enable them to observe the complex behavior of the programs in a systematic manner, such as tracing the process flow, displaying the change of variable, and setting for a break-point of execution. Programmers need to interpret the program behavior, compare it with the design intention, identify and resolve bugs if exist.

2.2.2 Verification support function in an ontology-aware authoring tool

In an ontology-aware authoring tool, the design intention remains in an operational form. This means that tools and authors can interpret the behavior of the product from the common viewpoint and enables the tools to provide useful information for authors to interpret it and identify the problems of design. We call the model with operational form of design intention as a conceptual-level model. "Conceptual-level simulation" is a function that simulates the behavior of the conceptual-level model in various levels of abstraction.

As shown in Fig. 1(E), the conceptual-level simulation shows the behavior of a training scenario as the change of a learner model. We call the learner model which is turned to an input and an output of the conceptual-level simulation as a 'pseudo-learner.' In other words, it is a kind of personification of a stereotyped learner in author's mind while he/she is authoring the training scenario. Of course, it is very different from the real learner because we assume its stable and non-autonomous learning behavior. In addition, the pseudo-learners' understanding does not depend on the quality of concrete contents in teaching material. This means that a pseudo-learner always succeed in learning what a training system teaches as long as the teaching activities are reasonable from educational principle prescribed in training task ontology.

Fig. 2 explains the role of the pseudo-learner in training scenario verifying. It is an ideal situation but almost impossible for authors to be able to examine whether the teaching material has the intended educational effect on all the real learners as shown in Fig. 2 (A). In Fig. 2 (B), by observing the changes taken place in pseudo-learners instead of the real learners, the author can examine whether the conceptual-level model of the teaching material is reasonably designed or not. One might think pseudo-learners supposed in a training scenario could be intractably numerous. It is true if we enumerate them all at once. When verifying, however, the number of the pseudo-learners is not necessarily large, because the author tends to concentrate at a local context of a training scenario. For example, Fig. 3 shows an example of a structure of D-goals, which are represented by a black diamond, and T/L-goals, which are represented by a white rectangle. The example is small because it includes only two steps of a diagnosis. However, training scenarios generally has a complex and large structure of goals. Real learners taking the training would have a long history of
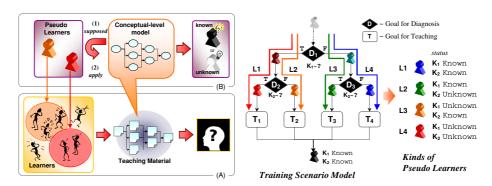
Fig. 2. The role of the pseudo-learner in conceptual level simulation

Fig. 3. Goal structure and kinds of pseudo-learner

learning along the structure. It is intractable to trace all the possible paths in the structure. This is a problem that large software generally has. Common way to solve such a problem is to divide the problem into a set of tractable ones. Well-accepted principle behind this way is called modularity. Our goal hierarchy in a model plays a similar role to modular structure of software. Authors verify the model step by step along the goal hierarchy. Goals that the author concentrates in each step represent the necessary and sufficient local context and include tractable number of the pseudo-learners.

The purpose of the conceptual-level simulation is to show "which part of the training scenario" adds "what type of an educational effect" to "what type of a learner" systematically. The two kinds of instructional goals play the important role to realize the purpose: D-goals and T/L-goals mainly concerns the classification of learners and the educational effect of the teaching activities, respectively. In the following, we will see how the authoring tool interprets the training model and simulates its behavior briefly.

Classification of learners. The purposes of training scenarios are to characterize a learner in terms of understanding status, grade and ability and to teach him/her in a way well adapted to his/her characteristics. D-goals are largely concerned with the former. Fig.3 shows a correspondence of a structure of D-goals to the classifications of learner. Learners are classified into four kinds of pseudo learner and four pseudo learners L1,.., L4 represent the kinds of pseudo learners. For example, L1 are characterized as the pseudo learners who understand both two knowledge units, K1 and K2, based on the two D-goals D1 and D2. The four different T/L-goals, T1,..T4, are set for L1,..,L4 respectively. This enables ontology-aware authoring tools to provide authors with basic information to verify whether the pseudo learner is appropriately characterized by the training scenario.

Effects on learners. Goals for education is to teach knowledge to a learner or to develop his/her skills in a way well adapted to his/her characteristics. Educational effect of the instructional goal and the necessary conditions to achieve the goal are specified as abstract axioms in the training task ontology. After an author specified the goal as a component of the model, the conceptual-level simulator can simulate the behavior of the goal. Intuitively, it adds the educational effects of the goal to pseudo-learner's status if the necessarily condition of the goal is satisfied. In Fig. 3, the four

education goals T1,.., T4 represented by white rectangles are defined. If the training model is well designed, the all the pseudo-learner will understand knowledge units K1 and K2 at the end of the training scenario as shown in Fig. 3. On the other hand, if more than one of pseudo-learners cannot understand both K1 and K2, there might be some problems in the training scenario.

3. An example of the conceptual-level simulation

In this chapter, we will take an example to explain the conceptual-level simulation. SmartTrainer/AT is an ontology-aware authoring tool for a substation operator training system SmartTrainer. A training scenario in SmartTrainer consists of a variety of grain sizes of modules. Typical ones are "backbone stream", which is a sequence of questions along the workflow and a "rib stream", which is a treatment of a learner's erroneous answers to questions in a backbone stream. The goal of SmartTrainer is to help learners to master all the operations in workflow implemented in the backbone stream by giving necessary knowledge in the course of instruction in the rib stream.

A small example of a training scenario is shown in Fig. 4. A question2 (1) of a backbone stream is connected with a ribstream (4) by a treatment (2) based on a diagnosis (3). Diagnosis (3) represents the diagnosis of a learner's incorrect answer of selecting option 1 to the question and ribstream (4) is set up as a treatment for the diagnosis. The purpose of the ribstream is to teach a missing knowledge according to the diagnosis and it is represented by the top goal of a goal hierarchy (4-1) as design intention. An upper goal of the goal hierarchy is expanded into a series of subgoals. In this case, the top goal "Improve Knowledge" is expanded into a series of subgoals "Notice An Error", "Acquire A Correct Knowledge" and "Grasp A Principle" and furthermore the goal "Acquire A Correct Knowledge" is expanded into "Understand A Correct Knowledge" and "Resume The Question". A sequence of teaching actions (4-2) is designed to attain those goals, for example, the teaching action "Teach-Topic" is expected to attain the goal "Understand a correct knowledge". Finally, the sequence of teaching actions embodied in a sequence of card specification (4-3) where the topics to be referred and the level of learner intended are specified.

Let us next show the conceptual level simulation with a pseudo learner who selects an option1 to the question2. Firstly, SmartTrainer/AT specifies the status of the pseudo learner according to a diagnosis and then applies ribstream to it. In this case, it assumes the pseudo learner does not know about the topic "64 Relay" based on a diagnosis (2). When the teaching action "Teach Topic" of a ribstream (4) is applied to the pseudo learner, a status of the pseudo learner is changed by the goal "Understand A Correct Knowledge". The change when "Understand A Correct Knowledge" is realized by "Teach-Topic" is shown in a conceptual level model on the right side of Fig. 4. This model means that the status of the learner is changed from before-status (a), where the pseudo learner does not know about the topic "64 Relay", to after-status (b), where it does.

Let us assume an author wants to examine the case that the pseudo learner is at the novice level. In the following scenario, we also assume that the task ontology prescribes that an average novice learner does not completely master a topic "Relay"

prerequisite to the topic "64 Relay"

When an author does the conceptual level simulation of the pseudo learner's behavior, SmartTrainer/AT shows two problems in the teaching scenario. The author is expected to notice that one is caused by a lack of T/L-goal and the other is by a lack of D-goal. The right of Fig. 4 indicates the former case with a conceptual model. In this case, the pseudo learner (e) does not fill the condition for understanding the topic "64 Relay" (c) because he does not master a knowledge (d) prerequisite to it. Thus, the understanding status of the topic "64 Relay" cannot be changed by the goal. The situation is represented as the status of the pseudo learner (f). In the latter case, the goal "Grasp A Principle" cannot be attained by the pseudo learner because the training task ontology prescribes that the principled knowledge is beyond the limits of novice learner's understandability. A proper way to resolve this problem is to add a new D-goal to classify learners according to the levels of mastery and set the goal "Grasp A Principle" only for advanced learners.

Our conceptual level simulator displays these results in the window as shown in Fig. 5. The simulation monitor window (W1) consists of three panes. The left pane (w1-c) shows a progression of a pseudo learner's learning in the training scenario, which consists of a backbone-stream and rib-streams with a goal structure. A node represents a question, a goal or a teaching activity and a link represents control flow or relation between goals. The top right pane (w1-s) shows the knowledge status of the pseudo learner. The hierarchical classification of pseudo learners is displayed in the bottom right pane (w1-p).

As we have mentioned before, Problems in a training scenario are caused by discrepancies (1)..(4) shown in Fig. 1. The problems are shown to authors as either the unachieved D-goals or T/L-goal. For example, in pane (w1-c), an author is supposed to focus on a problem of training scenario shown as an icon (G1) "Improve
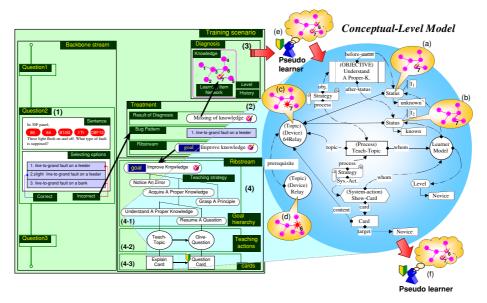

Fig. 4. An example of a training scenario

Knowledge". The icon means that there exist some learners who are not able to achieve the goal specified. At this situation, pseudo-learners have hierarchical structure as shown in w1-c, where a pseudo-leaner A is expanded into three pseudo-learners A-1, A-2, A-3. The learners who are imposed the goal are represented by a pseudo-learner A. The gray icon of pseudo learner A means that the goal could not be attained by some learners represented by it. The learners who cannot attain the goal are represented by pseudo-learner A-2. Pane w1-s displays the knowledge status of pseudo learner A-2 currently selected in w1-c. It shows a knowledge unit K1, which is expected to be taught, is not acquired by the learner after actions are executed under the goal (G1).

The author can reason the cause of the problem in the training scenario by means of getting down into specifics of its behavior along the goal hierarchy from top to bottom in the same way he/she have designed it. The problem can be resolved by adding new goals or correcting the erroneous goals. The author may ask SmartTrainer/AT for a further detailed information about the execution of goal (G1) by double clicking it. Then the goal is expanded into a sequence of subgoals as shown in w2-c. The author can observe the goal closely in W2 and find that the unachievement of (G1) is caused by unachievement of (G2) "Understand A Proper Knowledge" and (G3) "Grasp A Principle". As we have mentioned before, the former problem (G2) is caused by a lack of a T/L-goal a prerequisite topic. The author can reason the cause from the facts that the topic "Relay" has not been taught before and is a prerequisite to the target topic "64 Relay" of the goal (G2). Similarly, the latter problem (G3) is caused by a lack of a D-goal to classify levels of mastery. By
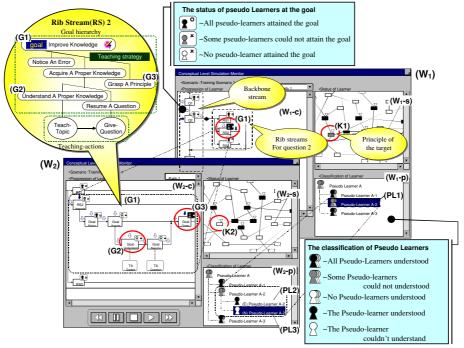

Fig. 5. The interface of the conceptual simulation

observing that a novice pseudo learner (PL3) does not attains the goal but an expert one (PL2) does as shown in w2-c, the author can reason that the problem can be resolved by adding a D-goal to classify the two pseudo-learners. In this manner, SmartTrainer/AT provides useful information for authors to identify and resolve the problems in the model.

4. Conclusion

An ontology-aware authoring tool has functions to effectively reduce the number of the problems which arise during the course of learning contents design. However, there still may remain some problems passed over in the design phase. The conceptual-level simulation helps authors to resolve those problems in verification phase before the training scenario is delivered to the real learner. It simulates the behavior of the conceptual-level model of learning contents in various levels of abstraction and provides good cue to identify the problems.

The conceptual-level simulation of learning contents is enabled by having operational semantics specified by the training task ontology. The most important role of the ontology is to maintain the continuity from authors' conceptual understanding of learning contents to the operational semantics of them. The implication of "ontology-awareness" and "operationality of the learning contents" is deep. To arrange the best collaboration between authors and tools, it is quite important to create an environment for authors to describe the model easily and for tools to operate the model systematically. One of the most important merits introduced by ontological engineering is that an ontology enables human to share the model with computers. We believe that this issue is important in enabling the efficient production of large-scale, high-quality learning contents which will be critical needs of this new millennium.

References

1. Bourdeau, J., and Mizoguchi, R.: Ontological Engineering of Instruction: A Perspective; Proc. of AIED-99, Le Mans, France, pp. 620-622, 1999.
2. Ikeda, M., Hayashi, Y., Jin, L., Chen, W. Bourdeau, J., Seta, K., and Mizoguchi, R.: Ontology More Than a Shared Vocabulary; Proc. of Workshop on "Ontologies for Intelligent Educational Systems", AIED-99, Le Mans, France, pp. 1-10, 1999.
3. Lee, J.: Design Rationale Systems: Understanding the Issues; IEEE Expert, vol. 12, no. 3, May/June, pp. 78-85, 1997.
4. Lai Jin, Weiqin Chen, Yusuke Hayashi, Mitsuru Ikeda, Riichiro Mizoguchi, Yoshiyuki Takaoka, Mamoru Ohta: An Ontology-aware Authoring Tool, ~ Functional structure and guidance generation ~, AIED-99, Le Mans, France, 1999, pp. 85~92.
5. Mizoguchi, R., Sinitsa, K., and Ikeda, M.: Task Ontology Design for Intelligent Educational/Training Systems; Proc. of Workshop on "Architectures and Methods for Designing Cost-Effective and Reusable ITSs", ITS'96, Montreal, pp. 1-21, 1996.
6. Murray, T.: Authoring Intelligent Tutoring Systems: An analysis of the state of the art; International J. of Artificial Intelligence in Education, Vol. 10, pp. 98-129, 1999
7. Shapiro, E.: Algorithmic Program Debugging; The MIT Press, 1982.