Knauf, Rainer; Sakurai, Yoshitaka; Tsuruta, Setsuo

Knowledge engineering technologies for learning processes

Zuerst erschienen in:
Education and technology for a better world : 9th WCCE, IFIP World Conference on Computers in Education, Bento Gonçalves, Brazil, 27 - 31 July, 2009 ; proceedings / [ed.: Ian Setwood ...] - [S.l.] : WCCE, 2009. -
ISBN 978-3-901882-35-7, insges. 10 S.

Knowledge Engineering Technologies for Learning Processes

Rainer Knauf[1], **Yoshitaka Sakurai**[2], **and Setsuo Tsuruta**[2]

[1] Ilmenau University of Technology, Germany, rainer.knauf@tu-ilmenau.de
[2] Tokyo Denki University, Japan, {ysakurai,tsuruta}@sie.dendai.ac.jp

Abstract: Generally, learning systems suffer from a lack of an explicit and adaptable didactic design. Since e-learning systems are digital by their very nature, their introduction rises the issue of modeling the didactic design in a way that implies the chance to apply AI Techniques. A previously introduced modeling approach called storyboarding is setting the stage to apply Knowledge Engineering Technologies to verify and validate the didactics learning processes. Moreover, didactics can be refined according to revealed weaknesses and proven excellence. Successful didactic patterns can be explored by applying Mining techniques to the various ways students went through the storyboard and their associated level of success.

Keywords: Process Modeling, Storyboarding, Learning Processes, Knowledge Engineering, Educational Knowledge Mining

1. Introduction

The design of learning activities in collegiate instruction is a very interdisciplinary process. Besides deep topical knowledge in the subject being taught, an instructor needs didactic skills. In particular, university instruction often suffers from a lack of didactic design. Since universities are also research institutions, their professors are usually hired based on their topical skills. Didactic skills are often underestimated in the recruiting process.

So far, the ad hoc application of didactic skills in teaching situations is not formally modeled for use by less experienced instructors. Moreover, much of such skills are not represented at all, but just "implemented" in the heads of experienced teachers (Chiang, 2006).

To make didactic design explicit, a modeling approach called storyboarding is outlined here. Besides providing didactic support, a (semi-) formal model such as storyboarding is setting the stage to apply Knowledge Engineering Technologies to verify and validate the didactics behind a learning process. The verification may

include both logical consistency issues and formally to check didactic issues.

Moreover, didactics can be refined according to revealed weaknesses and proven excellence. Successful didactic patterns can be explored by applying Mining techniques to the various ways students went through a storyboard and their associated level of success. As a result, future instructors and students may utilize these results by preferring successful ways through a storyboard.

A storyboard provides a road map for a lesson, a course, a subject to teach, or a complete study. According to different learning and teaching preferences, it includes alternative paths and possible detours if certain concepts to be learned need reinforcement. Using modern media technology, a storyboard also plays the role of a server that provides appropriate content material when deemed required.

There are at least three dimensions in which our modeling approach differs from others (1) expressiveness, (2) the degree of being domain based, and (3) IT-based complexity.

The paper is organized as follows. The next section outlines the storyboard concept. It is followed by an overview on Knowledge Engineering Technologies, which have been developed for storyboards. Finally, we summarize the research undertaken so far and outline current work as well as research horizons.

2. Storyboarding

Our storyboard concept is built upon standard concepts which enjoy (1) clarity by providing a high-level modeling approach, (2) simplicity, which enables everybody to become a storyboard author, and (3) visual appearance as graphs.

A storyboard is a nested hierarchy of directed graphs with annotated nodes and annotated edges. Nodes are scenes or episodes. Scenes denote leaves of the nesting hierarchy. Episodes denote a sub-graph. There is exactly one Start- and End- node to each (sub) graph. Edges specify transitions between nodes. They may be single-color or bi-color. Nodes and edges can carry attributes.

A storyboard is the authors' (instructors') design document representing various expectations of the users' (learners') behavior. Storyboards on educational processes can be traversed in different manners according to (1) users' interests, objectives, and desires, (2) didactic preferences, (3) the sequence of nodes (and other storyboards) visited before, i.e. according to the educational history, (4) available resources (like time, money, equipment to present material, and so on) and (5) other application driven circumstances. A storyboard may be seen as a model of an anticipated reception process that is interpreted as follows:

- *Scenes* denote a non-decomposable learning activity that can be implemented in any way; *Episodes* are defined by their sub-graph.
- *Graphs are* interpreted by the paths, on which they can be *traversed*.
- A *Start Node* / *End Node* of a (sub-) graph defines the starting / target point of a legal graph traversing.

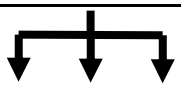
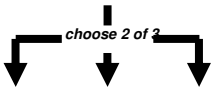
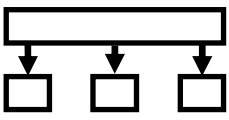
- *Edges* denote transitions between nodes. There are rules to leave a node by an edge: (1) The outgoing edge must have the same color as the incoming edge by which the node was reached. Edge colors express interdependencies between incoming and outgoing edges. (2) Conditions specified as the edge's key attribute have to be met for leaving the node by this edge.
- *Key attributes of nodes* specify application driven information, which is necessary for all nodes of the same type, e.g. actors and locations. *Key attributes of edges* specify conditions, which have to be true for traversing on this edge. *Free attributes* specify whatever the storyboard author wants the user to know: didactic intentions, useful methods, necessary equipment, e.g.

The functionality of nodes and edges specified in Tables 1 and 2.

Table 1. Node Types

Element	Symbol	Behavior on double click	Behavior on hyperlink
Scene		• opening a document • nothing, if just verbally described scene	• opening a document • visiting a website • opening the mail tool
Episode		opening the sub-graph that specifies the episode	• opening a document • visiting a website • opening the mail tool
Start Node		*not meaningful*	*not meaningful*
End Node		jumping to the *Reference Node* that succeeds it's associated *Episode Node* in the related super-graph	
Refe-rence Node		*not meaningful*	

Table 2. Edge Types

Element	Symbol	Interpretation
Simple Edge		defines a unique successor node
Fork		defines several successor nodes, which have to be traversed independently from each other, i.e. in any sequence or parallel
Fork with con-ditions		Defines several successor nodes, which have to be traversed independently from each other, according to the specified condition, e.g. *take n out of m specified paths*
Alter-natives		defines alternative successor nodes, i.e. one of it has to be traversed

3. Knowledge Engineering with Storyboards

3.1 Formal Verification of Storyboards

Our concept of storyboarding is a semi-formal one. The graph hierarchy is completely formal and below the level of scenes is completely informal. Thus, the scenes form the interface between the formal and the informal levels. The formal levels are the key feature to detect logical anomalies.

To ensure consistency and completeness of our storyboards, we developed and implemented several verification procedures:

1. A *Hierarchy Completeness* test focuses questions such as whether every episode has exactly one related graph and vise versa.
2. Also, a *Path Completeness* test the reachability of each node (in particular, of the *End Node*) from the *Start Node* is checked.
3. Furthermore, the *Node Soundness* of outgoing edges, i.e. the completeness and consistence of alternative outgoing edges (with the same beginning color), is checked.
4. Edge colors, which express the *Interdependence of Incoming / Outgoing edges*, are also a subject of formal verification by checking, whether there is a unique (beginning) color of the Start node's outgoing edges and at least one outgoing edge with the same color for each incoming edge's colors.

The above mentioned anomaly tests are implemented for our storyboard development environment (Sauerstein, 2006; Duesel, 2007).

3.2 An Inheritance Concept

Additionally, an inheritance concept within the graph hierarchy was implemented, which distinguishes several inheritance types such as (1) *set union*, (2) *sum*, or (3) *maximum* for inheritance within the graph hierarchy.

(1) In some applications it makes sense to inherit annotations from nodes to their related super-graph as a set of all values that occur in the sub-graphs. For example, material that is used to teach a particular lecture is also material to teach the complete course the lecture is part of.
(2) In other cases it makes sense to inherit the arithmetic sum of a key annotation of all nodes to the related super-graph. For example, this is useful to determine an upper limit for time consumption or for a course fee.
(3) In other cases it makes sense to inherit the maximum value of a key annotation of all odes to the related super-graph. For example, the educational difficulty (basic/easy, medium, advanced, very difficult) of a study needs to be communicated as the maximum value of all mandatory subjects.

Thus, for each key annotation an appropriate inheritance method can be selected

in our Microsoft in our storyboard development environment (Xu, 2006). From a Knowledge Engineering point of view, this is some sort of deductive inference over the knowledge represented as storyboards.

3.3 Towards a Storyboard Development Environment

To a priori ensure soundness, a set of operations were defined, which's exclusive use automatically leads to a "legal storyboard" (Sauerstein, 2006).

These operations are (1) adding paths, (2) adding nodes, (3) turning a scene to an episode, while introducing a related sub-graph, (4) adding a concurrent path, and (5) merging (equivalent) nodes by introducing related bi-colored edges, which make sure that the linkage with the remaining graph isn't changed (see figure 1).

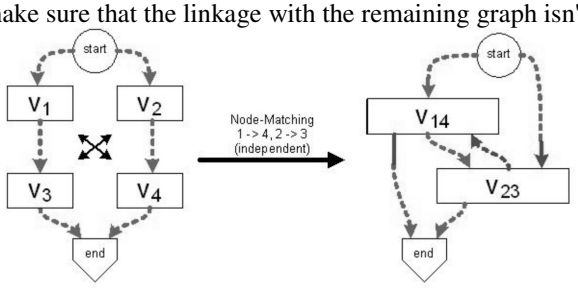


Figure 1. Merging equivalent nodes.

In figure 1, V_1 and V_3 as well as V_2 and V_4 are equivalent. Since different users visit them in different sequences, they are represented as different nodes on the left hand side. By merging the equivalent nodes together, a new color needs to be introduced to express these different sequences.

3.4 Knowledge Mining over Storyboard Paths

A general objective of this storyboard application is to use Knowledge Engineering technologies on the (semi-) formal process models (Knauf, 2008). The particular objective here is inductively "learning" successful storyboard patterns and recommendable paths. This is performed by an analysis of the paths, where former students went through the storyboard and it is based on their success that is associated with these particular paths.

To exemplary show the feasibility and benefit of this approach, a simple prototype was developed to evaluate curricula created or modified by the students in advance of their study (Knauf, 2008). Here, we implemented a concept to estimate success chances of curricula, which are composed by students at a Japanese university in their curriculum planning class in the first semester.

Based on paths of former students and their related learning success, the success chance of intended paths can be estimated as follows (Knauf, 2008).

3.4.1 Construction of a Decision Tree

The construction of a decision tree is based on the paths of former students through the storyboards. Each of those paths can be associated with the degree of success, which has been achieved by the student. In case a set of students went the same path, the degree of success can be estimated by a weighted average degree.

This path begins at the Start Node of the top level storyboard and terminates at its End Node. Each episode on this path is replaced by its sub-graph. This replacement continues throughout the entire hierarchy of nested graphs. Figuratively speaking, the decision tree is constructed on the basis of a "flatten" storyboard, which contains atomic scenes only.

The decision tree is based on the concept of bundling common sequences of the various paths to a node of the tree. Different subsequent following (next) nodes of the paths will result in different sub-trees right below the last node of the common starting sequence. This continues for each lower level sub-tree accordingly.

The final node of the paths are followed by a label-node. Label-nodes contain a list of marks that students received after going through this path. Each mark is along with the number of occurrences (the number of students getting the mark). Since the courses of a semester are usually visited concurrently (which is represented by the fork edges, see Table 2, we consider them as a single node containing a set of courses.

A new path is added to the tree by simultaneously traversing the path's courses sequence and the decision tree down from the root until (1) the path is finished or (2) there is a "next node" in the path that is different from all "next sub-tree roots". In the first case, the related success information for this path is updated accordingly. In the latter case, a new sub-tree is made out of the remaining path and hooked into the tree.

3.4.2 Utilization of a Decision Tree

If a submitted path is completely represented in the decision tree, the success estimation is very easily done through presenting the content of this label. Otherwise, the most similar sub-path in the decision tree will be identified.

In our initial approach, similarity refers to the number of same course sets in sequence, which the path has in common with a path represented in the tree. This similarity measure s is in the range $0 \leq s \leq 1$. In the worst case, there no node in common with any path in the tree ($s = 0$) and in the best case, the submitted path is completely represented in the tree ($s = 1$).

Like in the tree construction procedure, this is performed by simultaneously traversing the path's course sequence and the decision tree down from the root until (1) the path is finished or (2) there is a "next node" in the path that is different from all "next sub-tree roots". In the first case, the related success information of former students is the desired success estimation. In the latter case, a success evaluation is computed by merging the success information of all sub-trees starting from there.

Additionally, we provide a supplement to the submitted path, which is the most

successful rest - path starting at the last node of the tree traversing along with this optimal achievable success.

Also, the user is informed about the degree of similarity of his submitted path and the one found in the decision tree. We call this similarity *significance* and compute it as the number of nodes in sequence that are common in the submitted path and the decision tree, related to the entire length of the path.

Based on this information, the user (student) can make a decision on whether or not holding on to the submitted curriculum or modifying it in accordance with the optimal supplemental path.

3.4.3 An Example

Here, we introduce a small example of a decision tree construction and utilization, which is derived from our application. For better understanding, we

1. refer to the subject compositions as just *episodes*,
2. refer to the atomic storyboard elements, the particular courses, as *scenes*,
3. generalize from concrete episode- and scene names to abstract ones such as *e1, e2, … , s1, s2, …* and
4. use the German numerical students' performance evaluation scale ranging from 1 (very best) to 5 (failed).

Pre-Processing Path Information

First, each given path is decomposed by recursively replacing episodes by their related sub-graph path until the paths consists of scenes only. Concurrent scenes, i.e. subjects that run in parallel, i.e. in the same semester, are united to a scene set and form one element of the student's path. As a result, each path is a linear sequence of elements. Attached to this sequence, there is the associated success label composed of the Grade Point Average (GPA) of the student, who went this path. Figure 2 shows illustrated this procedure.

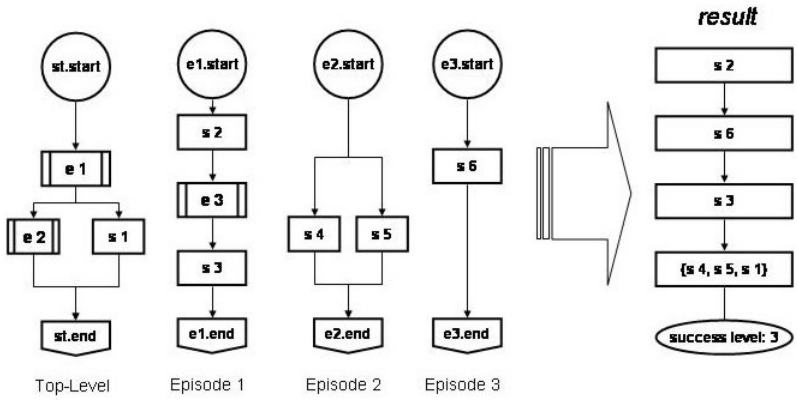


Figure 2. Preprocessing a student's path through a nested storyboard

Composing a Decision Tree of Paths

Next, a decision tree is constructed. Figure 3 shows the result of the decision tree construction in our application. As illustrated in the figure's left hand side, 17 students went through the storyboard on four different paths, namely (1) *[s4, {s6, s7}, s1, s9]*, (2) *[s4, {s6, s7}, s5, s8]*, (3) *[s4, s2, {s3, s1, s5}, s9]*, and (4) *[s4, s2, {s3, s1, s5}, s6]*.

[s4, {s6, s7}, s1, s9]	
1	1 student
2	2 students
3	1 student
4	1 student
5	0 students

[s4, {s6, s7}, s5, s8]	
1	0 students
2	1 student
3	3 students
4	1 student
5	0 students

[s4, s2, {s3, s1, s5}, s9]	
1	3 students
2	2 students
3	0 students
4	0 students
5	0 students

[s4, s2, {s3, s1, s5}, s6]	
1	0 students
2	0 students
3	1 student
4	0 students
5	1 student

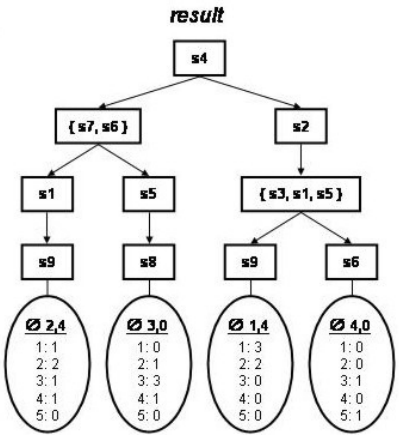


Figure 3: Storyboard paths and a derived decision tree

In the derived decision tree each of these four paths form a path in the tree from the root towards a leaf. Attached to each leaf, there is a label node, which holds the success information of the students, who went this path.

Utilization of the Decision tree

Figure 4 shows the usage of the decision tree for three submitted paths, one which is represented completely in the decision tree and two, which are not represented completely in the decision tree.

The success estimation of the first path is simply performed by providing the related success label of the related path in the tree.

For the second submitted path, there is no identical path in the tree. Here, the estimation procedure looks for a path within the tree, which has the longest starting sequence in common with the submitted path. This is *[s4, {s7, s6}]*. Since this path has only two nodes in common with the submitted one (having four nodes), the significance of the success estimation is calculated by *2/4*. Behind the node *{s7, s6}*, there are two different sub-trees, which led to different success degrees by former students, *[s1, s9]* and *[s5, s8]*. Since the latter is the better one, it is recommended as a rest path to optimize success chances. For the third path, the usage of the decision tree is performed accordingly.

By practicing this way to utilize a decision tree, we realized that we rarely found a path in the tree, which is completely equivalent to a submitted path. This

happened in particular, if the tree contains scene sets with many scenes in parallel. Those sets are mostly never equivalent to similar sets in a submitted path.

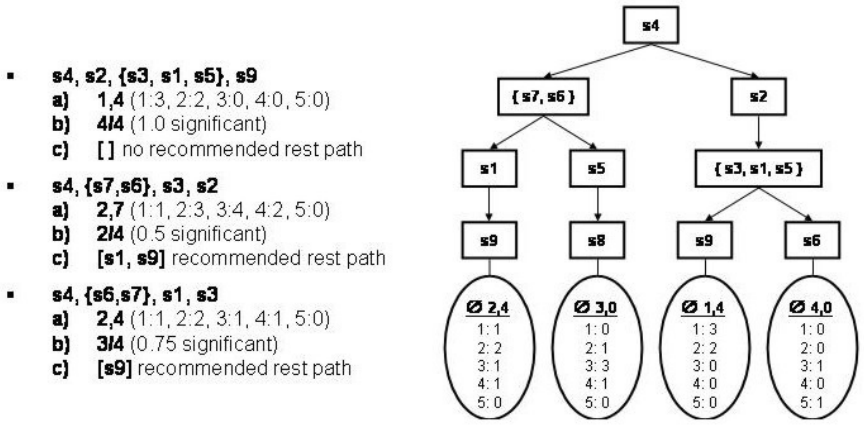


Figure 4 Success estimation (a), its significance (b), and recommended test paths (c)

However, if an element of a node that contains a scene set in the tree is not in the related node of the submitted path, it still could be a subject that the student already passed successfully in a previous semester.

Therefore, the containment in the decision tree was extended with respect to the educational history of a student. A previously taken course may always be considered as an element of a subsequent node:

1. Let $P = [P_1, P_2, \ldots, P_n]$ be a path submitted by a student. Each P_i is a set of subjects planned by the student to be taken in parallel in a particular semester. Semesters with one subject ***s*** only are represented by sets of one element only, i.e. in this case the node is $P_i = \{s\}$.
2. Let $T = [T_1, T_2, \ldots, T_m]$ be a path that is represented in the decision tree.

P and T are *equivalent* ($P \equiv T$), iff

(1) the path have the same number of nodes ($n = m$) and

(2) all subjects ***s*** in a tree node T_i are either in P_i or in another P_j **with** $\boldsymbol{j < i}$**:**

$$\forall s \in T_i : s \in \bigcup_{j=1}^{i} P_j \,.$$

For example,

- a submitted path ***P = [{s1, s2, s3}, {s4, s5}, {s6, s7}]*** and the path within the decision tree ***T = [{s1, s3}, {s4}, {s2, s5, s7}]*** are *equivalent*, because each subject of a node in ***T*** is either in the related node of ***P*** or in a previous one.
- a submitted path ***P = [{s1, s2, s3}, {s4, s5}, {s6}]*** and the path within the decision tree ***T = [{s1, s3}, {s4}, {s2, s5, s7}]*** are *not equivalent*, because the subject ***s7*** in the third node of ***P*** is neither in third node of ***T*** nor in one of the previous nodes, i.e. the second or fist node.

4. Outlook

Storyboards are an approach to make the didactic design of university courses explicit. Since their scenes are not limited to the presentation of electronic material, but may represent *any* learning activity, the application of this concept goes far beyond the IT approaches to support learning so far.

This modeling concept is appropriate to be used by topical experts (university instructors, in our case) without an IT- or software engineering background. Didactical intentions and variants can easily be specified as a nested graph-structure. This formal character allows the application of Knowledge Engineering technologies to didactic knowledge.

Our current work focuses the integration of a cognitive user (student) profile to provide success estimations and refinement suggestions due to a student's individual learning needs, learning desires, preferences and talents.

References

Chiang, F.-K. and Chung, S.: A Case Study on An Experienced Elementary Teacher's Teaching Strategies and Teaching Reflection in Mathematical Classroom. C. Crawford et al. (Eds.): *Proceedings of Society for Information Technology and Teacher Education International Conference 2006*, Chesapeake, VA: AACE, pp. 3697-3703, 2006.

Dohi, S., Nakamura, S., Sakurai, Y., Tsuruta, S., and Knauf, R.: Dynamic Learning Need Reflection System for academic education and its applicability to Intelligent Agents. *Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies (ICALT 2006)*, Kerkrade, the Netherlands, Los Alamitos, CA: IEEE Computer Society, ISBN 0-7695-2632-2, pp. 459-463, 2006.

Duesel, H.: *Konzeption und Realisierung von Methoden der formalen Verifikation von Storyboards.* (A Concept of formal Storyboard verification and its Implementation), Diploma Thesis, Ilmenau University of Technology, Faculty of Economic Sciences, 2007.

Kasperski, S.: *Entwicklung eine unabhängigen Storyboardrepraesentation.* (Developing an Platform Independend Storyboard Representation), Study, Ilmenau University of Technology, Faculty of Computer Science and Automation, 2007.

Knauf, R.; Boeck, R.; Sakurai, Y.; Dohi, S.; Tsuruta, S.: Knowledge Mining for Supporting Learning Processes. *The 2008 IEEE International Conference on Systems, Man, and Cybernetics (SMC 2008)*, Singapore, 2008, IEEE Catalog number CFP08SMC-USB, ISBN 978-1-4244-2384-2, Library of Congress: 2008903109, IEEE, Piscataway, NJ, USA, pp. 2615-2621.

Sauerstein, G.: *Formale Modellierung und deduktive Interpretation von Storyboards. Modellierung eines komplexen Prozessmodells.* (Modeling a Complex Process Model: Formal Modeling and Deductive Interpretation), Study Work, Ilmenau University of Technology, Faculty of Computer Science and Automation, 2006.

Xu, G.: *Ein Vererbungskonzept für Storyboards.* (An Inheritance Concept for Storyboards), Study Work, Ilmenau University of Technology, Faculty of Computer Science and Automation, 2006.