DESIGN AND EVALUATION OF AN ADAPTIVE LEARNING SUPPORT AGENT

Dejan Ljubojevic, John Cook, Tom Boyle
Centre for Excellence in Teaching and Learning in Reusable Learning Objects,
Learning Technology Research Institute,
London Metropolitan University

ABSTRACT

This paper introduces a multi-agent approach to adaptive, problem-based learning support based on context-centric view of tutorial interactions and Elaboration Theory (Reigeluth, 1999). It describes (i) a framework for describing and analysing resource-use inside a (computer based) tutorial, (ii) a novel idea for a computer-based learning activity configuration agent, (iii) an agent that adaptively suggests learning resources and activities to learners at run time, and (iv) an empirical study that tested the second agent.

KEYWORDS

Contextual Metadata, Reusable Learning Objects, Adaptive Learning Support Agent

1. INTRODUCTION

In this paper we present an agent that adaptively suggests Learning Objects and learning activities to learners at run time, and a study that evaluated the agent with learners. Consequently, we see this work as fitting in with the conference tracks of ‘Pedagogical Issues Related with Learning Objects’ and ‘Assessment of exploratory technologies’.

Adaptive learning support is a type of learning guidance typically found in tutorial sessions. The provision of adaptive learning support is widespread in higher education and generally takes the form of tutorial sessions (labs, seminars, etc.). Laurillard (1993, p.148) suggest that: “the ideal teaching system is a one-to-one teacher-student-dialogue”. However, on the popular courses in Higher Education, the student-staff ratio can be very high and in the extreme a single tutor is providing adaptive learning support to up to 50 students. In such extreme cases there is a genuine need to broaden access to this type of learning support, as its availability and quality is a crucial component of education. In order to design computer-based agents that adaptively support a learning session, the following key question must be addressed: How to describe the resource use inside the tutorial so that the description drives the adaptive use of the resource at run time (when learners engage in problem-solving activity)?

Following a brief review of the background to our work, the paper describes (i) a framework for describing and analysing resource use inside a (computer based) tutorial, (ii) a novel idea for a computer-based learning activity configuration agent, (iii) an agent that adaptively suggests learning resources and activities to learners at run time, and (iv) an empirical study that tested the second agent. An objective for the study was to determine how ‘adaptive’ our agent actually is. This was to be determined from an analysis of the successful and unsuccessful interactions between the software agent and learners. By successful interactions we mean where recommended sequences of learning proved instrumental to advancing student’s problem solving activity. Conversely, by unsuccessful interactions we mean the instances of a learner following a suggested learning sequence that did not aid students in their problem-solving context (in such circumstances the analysis tried to establish the root cause).
2. BACKGROUND

Elaboration Theory (Reigeluth, 1999) has been used in the research described in this paper as design approach. Elaboration Theory applies to the design of instruction for the cognitive domain and is an extension of the work of Ausubel (1993) work on advance organizers and Bruner (1960) spiral curriculum.

“It is claimed that the elaboration approach results in the formation of more stable cognitive structures and therefore better retention and transfer, increased learner motivation through the creation of meaningful learning contexts, and the provision of information about the content that allows informed learner control.”

(From http://tip.psychology.org/reigelut.html, accessed 20th May 2005)

Elaboration Theory offers guidance when decomposing the overall problem or task into a set of sub-tasks. One method from the Elaboration Theory, the Simplifying Conditions Method (sequencing strategy), integrates task analysis with design for the purpose of guiding the identification and sequencing of sub-problems that leads a student into solving the central problem in a learning activity. Two processes that make up the Simplifying Conditions Method are: i) Epitomising, the process of identifying the simplest version of the task that is still fairly representative of the whole task, and, ii) Elaborating, the process of identifying progressively more complex versions of the task. The Simplifying Conditions Method approach centres around the questions: “What is the simplest version of the task that an expert has ever performed?” and “What is the next simplest version?”, and so forth (Reigeluth, 1999, p.445).

The whole point of developing an agent that adaptively suggests learning resources and activities to learners at run time is not to replace the human tutor, which in any case is presently impossible, but rather to provide an additional, value-added tool in a situation where there is a high student-staff ratio. The approach described in this paper is presented as an alternative to methods of achieving adaptation in hypermedia-based learning environments that involve learner modelling and as advancement to the notion of ‘activity nodes’ described by Koper et al., (in press). Adaptive Hypermedia Systems seek to “build a model of the goals, preferences and knowledge of the individual user and use this through the interaction for adaptation of the hypertext to the needs of the user” (De Bra, Brusilovsky, et al., 1999). The goal of the work described in this paper, on the other hand, is to orchestrate the available reusable learning resources so as to meet a particular learner’s context and learning needs. Koper, et al. (in press) describes the next step for Learning Design, calling it Learning Networks. Learning Networks provide a track of completed ‘activity nodes’ for a single learner. Each activity node can consist of an actor (e.g. learner) and a Unit of Learning, the latter being a learning facility that is abstractly defined for any set of learners at any given time. The work described in this paper ‘drills’ down into an activity node and presents a proposal for what we believe needs to be provided to adaptively support an independent, self-motivated learner as they negotiate their way from one learning resource to another in a problem-based learning activity context.

The work described in this paper made use of a repository of Learning Objects developed in a previous project (Bradley and Boyle, 2004). The IEEE (2001) provide the following, widely quoted definition – “Learning Objects are defined here as any entity, digital or non-digital, which can be used, re-used or referenced during technology supported learning”. In our work, however, we treat reusable Learning Objects as web-based interactive “chunks” of e-learning designed to explain a stand-alone learning objective (Boyle, 2003). The fact that the Learning Object has been broken down to a low level of ‘granularity’ facilitates its reuse in different learning and teaching situations.

3. PROBLEM SOLVING UNIVERSE FRAMEWORK

The Problem Solving Universe (PSU) framework (Ljubojevic, Cook, et al., 2004) is a context-centric framework for describing resource use inside the tutorial, in our research it is used as means for configuring (i.e. describing) tutorial-like computer based adaptive learning support.

The framework space is multidimensional and on the top level it is divided in two halves along vertical axis, called Quadrants and in our metadata scheme it is denoted Q. The Q variable can have values 1, or 2 (see Figure 1).
The value of $Q=1$ denotes a Conceptual quadrant or space for ‘placing’ the learning resources and associated activities that engage students in conceptualising the particular concept in relative isolation from other concepts. Conversely, the value $Q=2$ denotes Contextual quadrant or space for ‘placing’ the learning resources and associated activities that engage students in conceptualising particular concept in relation to other concepts. Essentially, placing a resource into the Conceptual quadrant implies a standalone treatment at learning run-time, whereas placing a resource into the Contextual quadrant implies that the learning activity at run-time would link the concept under discussion to other concepts in order to ‘contextualise’ the target concept.

To aid our explanation of the framework we will make use of an example configuration used inside the study described in this paper. The configuration is aimed at supporting the 1st year Java Programming course students in solving a problem titled ‘Write Java Applet that Displays Three Rectangles using Repetition Loops’. Before the configuration process starts the problem setting specification document is produced (typically HTML document placed on the web), detailing the size of the rectangles, the coordinates where the rectangles are to be placed, etc. The configuration process starts with decomposing the central task into a set of sub-tasks/topics (Figure 2); in our study we used Simplifying Conditions Method (Elaboration Theory, Reigeluth, 1999). In this example the overall task is decomposed into three whole, progressively more complex subtasks, we call these Topics. In our example, the Topics are: i) Write Java Applet (Topic T=A in Figure 2), ii) Write Java Applet that Displays Graphic Element (Topic T=B in Figure 2), and, iii) Write Java Applet that Displays Multiple Graphics using Repetition Loop (Topic T=C in Figure 2).

Once these initial configurations are completed the resources can be inserted (contextualised) inside the framework space. Depending on the tutor-intended manner of resource use the resource is placed inside appropriate ring (Figure 2), outer, middle, or inner one. These rings are called Orbits; the outer Orbit is denoted X, the middle one Y, and the inner one Z (Figure 3).
The Orbital placement describes the resource-use type that is jointly determined by i) the resource-type and ii) the commentary type associated with that resource placement. The resource-use types are defined as follows:

- (Orbit X) Resource-use of **descriptive type** is made up of (i) a resource whose content (typically text resource) is describing a generic conceptualisation or generic contextualisation, and, (ii) the commentary that highlights the issues and prompts student reflection on the issues as presented in the resource,

- (Orbit Y) Resource-use of **prescriptive/descriptive type** is made up of (i) a resource whose content (typically interactive Flash Learning Object) is detailing steps of a given learning activity (leading to student acquiring conceptualisation or contextualisation knowledge) together with explanations of these steps inside the resource, and, (ii) commentary that sets a micro-goal and prompts student to learn through activity (interaction with the interactive resource),

- (Orbit Z) Resource-use of the **prescriptive type** is made up of (i) a resource whose content (typically the resource containing lines of code) is detailing the steps of the activity (leading to student acquiring conceptualisation or contextualisation knowledge) without explanations of these steps inside the resource itself, and, (ii) a commentary that sets a goal, usually to modify the code in the resource presented and to reflect on the outcome of that activity.

Figure 5 presents a software agent that supports the configuration process described above.

![Figure 5. Configuration Support Agent](image)

The Configuration Support Agent allows tutors to drag-and-drop the resources (from the list on the right-hand side in Figure 5) into the framework space (the left-hand side in Figure 5). Each placement of a resource inside the framework space results in a description of a micro activity, and the software prompts the tutor (i) to name the activity that is being created, and (ii) to provide the comment that will drive the newly created activity.

At the end of each session with the Configuration Support Agent, the learning configuration (as designed by tutor) is captured in the XML file format that contains the data regarding the resource contextualisation decisions (by the tutor) and the commentary associated with these. This data-capturing method is aimed at recording the outcome (learning configuration files). The contextualisation decisions are captured in the form of ‘coordinates’, and at each coordinate there is a URI (Universal Resource Indicator) reference to a learning resource and associated tutor commentary (see Figure 7). Further, these contextualisation decisions are captured in, what we call Contextual Metadata, the format that is interpretable by the Adaptive Learning Support Agent (described below). Example of Contextual Metadata is presented in Figure 6.
Figure 6. Mapping between PSU framework and Contextual Metadata

Figure 7. The ‘resource’ element exploded

Note – The evolution of the framework redefined the meaning of the ‘resource’ tag. The ‘resource’ element, as shown in Figures 6 and 7 is actually the ‘activity’ element. Due to the time limitations the metadata schema has not been modified to reflect this.

4. ADAPTIVE LEARNING SUPPORT AGENT

The Adaptive Learning Support Agent (ALSA) (Ljubojevic, Cook, et al., submitted), shown in Figure 8, uses the output from the first agent (what we call contextual metadata) to personalise learning support for a student who is engaging in a problem-based learning activity. That is to say, the second agent provides adaptive learning support by suggesting appropriate learning resources and activities associated with these resources.

Figure 8. Adaptive Learning Support Agent
The top part in the Figure 8, **part 1** contains the title of the tutorial session and the button for accessing the tutorial task specification. This part 1 of the interface is used to clearly explain the task and to provide the title to the tutorial.

The middle part (Figure 8), **part 2**, is used to engage the student in a form of a dialogue. By placing a question on top of the part 2 (‘What do you need help with?’) the student is expected to, in answering the question, choose one out of three suggested topics. The software dynamically puts these three topic-options together, by processing appropriate elements in the contextual metadata. The number of the topic-options may vary depending on the number of tutor-identified sub-topics during task-decomposition (part of configuration process). One conclusion from the design of this particular dialogue-related interface issue is that the titles of the sub-topics should start with a verb in order to semantically plug-in with the question on the top. So the question is – What do you need help with? and the answer is, for example – I need help with ‘Writing Java Applet’. It is at this point, when the student selects the sub-topic to be helped with, that the initial, and sketchy, diagnosis of the student’s skills, with regards to the tutorial problem/task, is made by the software. In other words, by student choosing the help about ‘Writing Java Applet’ (which is the least complex topic) it can be inferred that the student lacks the knowledge and skill deemed as basic inside this particular session. Consequent to the topic selection when the button ‘Get Help!’ is pressed (bottom of part 2 in Figure 8) a dialogue box ‘Rate your Confidence’ is presented to the student (not shown in Figure 8). The aim of this dialogue component is to establish student’s confidence, as perceived by the student, with the chosen topic. This dialogue prompts the student response with a question: ‘How would you like to be supported with <topic name>?’ and allows two options for student to choose from, ‘Introduce it to me’, and, ‘Suggest activities to practice it’; the options/answers correspond to the type of learning support (Conceptual and Contextual), that is, if the student opts for ‘Introduce…’ the software will start by suggesting the resources that will support the student with conceptualisations of the chosen topic. In other words the dialogue so far is structured in such a way as to:

- Elicit student’s perception of his/her knowledge and skills in relation to the tutorial task, and,
- By means of comparison of the student feedback and the contextual descriptions available in metadata, allocate the coordinates in the PSU space from which the ensuing learning support is to start.

Once the starting point for the learning support is determined (using student feedback and the contextual descriptions in metadata) the lower-most part of the interface, **part 3** (see Figure 8), is used by the software to display the activities (resources and associated comments) that, according to the descriptions in metadata, fit the learning need (identified through structured dialogue). These resources have coordinates that correspond to the learning support starting point as determined by student feedback and tutor’s contextualisation descriptions in metadata. When student selects activity, the resource associated with the activity is displayed in Internet Browser and the activity is driven by the comment associated with that activity. In the study, the actual activity takes place inside the JCreator Integrated Development Environment and when the student performs the activity and returns to the ALSA for further support, the ALSA presents the feedback dialogue. The feedback dialogue is prompted with a question ‘How well did you perform/understand the tutor suggested activity/concept of <name of activity> activity?’ two feedback options are available within this dialogue, ‘Not so well’, and, ‘Very well’. This feedback, together with the activity’s coordinates is recorded by ALSA in XML format inside Learning History Metadata.

Processing the Contextual Metadata descriptions (Figure 6) and the Learning History Metadata records the ALSA can monitor student progress through the framework space and adaptively recommend next set of activities depending on the student’s feedback and the coordinates of the last interaction. The recommendation function is made available with ‘Recommend Next’ button (bottom-right in the Figure 8). The Learning History Metadata can also be saved so that the exercise can be continued at another time from the point where it was left off.
5. THE STUDY, RESULTS AND DISCUSSION

5.1 Overview

The study involved 15 sessions and took place at the North Campus of London Metropolitan University, The Department of Computing, Communications Technology and Mathematics, over 1 week in November 2004. The participants for this study were selected from the first year computing module ‘Introduction to Programming using Java’ at London Metropolitan University. The sessions lasted 45 minutes each, broken down as follows: 5-10 minutes introduction to the software and the study, 25-30 minutes subjects engaging in problem solving activity supported by the software agent, and, 10 minutes debriefing time which included subjects completing a short questionnaire. Each subject was presented with a task (“Produce an Applet that Displays Three Rectangles using Repetition Loop”) and asked to use the ALSA software for any support that they needed. The coding of the solution was performed with JCreator (Java IDE), which was also made available to students. The study set-up was aimed at observing the student use of ALSA.

The first author was present throughout each session and recorded (pen and paper) the incidents relevant to the research. Other means of recording the sessions were: HyperCam software recording the terminal screens and the learner history metadata with the complete record of all the resources viewed by the subjects. The analysis of the data set obtained starts with identifying the critical incidents (something that stands out, e.g. a successful or unsuccessful learning incident), as recorded by the researcher (the paper notes) and the subject’s response data at debriefing part of the session. Any ambiguities that may occur during the analysis of these principal data records were correlated to the HyperCam data and the session metadata. Also, the comparisons of performance between subjects were expected to further illuminate the utility of ALSA in responding to the learning need. Thus a system of triangulation was put in place where by i) data was captured automatically in the Learning History metadata and ii) data was captured in the screen-grab video files and (iii) the pen-and-paper notes of the significant session incidents (recorded by session observer). An example of interpretative-synthesis of the data from several sources is presented in Table 1.

Table 1. Analysing student progression through problem-space

<table>
<thead>
<tr>
<th>Student 3</th>
<th>Student 4</th>
<th>Student 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>IntCtx</td>
<td>Q/T/O</td>
<td>IntCtx</td>
</tr>
<tr>
<td>Initiate</td>
<td>1</td>
<td>1/C/X</td>
</tr>
<tr>
<td>Explore</td>
<td>1.1</td>
<td>1/C/Y</td>
</tr>
<tr>
<td>Explore</td>
<td>1.2</td>
<td>2/C/X</td>
</tr>
<tr>
<td>Resolve</td>
<td>1.3</td>
<td>1/C/Y</td>
</tr>
<tr>
<td>Initiate</td>
<td>2</td>
<td>1/C/Z</td>
</tr>
<tr>
<td>Total Task solved 75%</td>
<td>Total Task solved 100%</td>
<td>Total Task solved 100%</td>
</tr>
</tbody>
</table>

In this table: IntCtx – Interaction Context, Q/T/O – Quadrant/Topic/Orbit – see section 3 for details

5.2 Results

An evaluation of the ALSA with 15 learners found that the student-group enjoyed the experience of agent-aided problem-based learning (93.4%). Furthermore, evidence was obtained through post-session questionnaire that suggests that the second agent design was problematic for more than a third of the student-group (39.9%). Overall, the majority of learners found the software a useful tool for learning (86.6%). Finally, responses to the question ‘Was the exercise difficult?’ brought to the surface the inadequacy of the chosen problem-task in that more than a half of the student-group (53.3%) found the exercise simple.

5.2.1 Key Findings

The study yielded several key findings, these are summarised below.

**Key finding 1 – Interaction-Context Bugs.** The occurrence of Interaction-Context Bug is defined as: an obstacle to exploring/resolving an agreed Interaction-Context that lays outside an agreed Interaction-Context. Two types of Interaction-Context Bugs have been identified:

- The obstacle that lays outside the agreed Interaction-Context but inside the agreed sub-topic, and,
- The obstacle that lies outside the agreed Interaction-Context and outside the agreed sub-topic.
The identification of Interaction-Context Bugs has implications on the design of the student feedback dialogue in that the provisions for capturing this detail would further increase the system’s level of learning support.

Key finding 2 – Sub-topic Objective. The Sub-topic Objective finding stems from the observation of student-disorientation, that is, some students failed to recognise when a particular sub-topic has been sufficiently explored for the purpose of solving the overall task/problem. Provision of the Sub-topic Objective would ensure better learner’s cognitive-grip on the way the support is designed.

Key finding 3 – use of Simplifying Conditions Method should not be made mandatory. The choice of the problem/task for the student group proved to be simple for more than 50% of the student group, this prompted re-evaluation of the mandatory status of the SCM within the framework. The SCM is intended, as its author (Reigeluth) claims, for medium to complex cognitive learning. Enforcing a single decomposition method would greatly limit the utility of the framework; instead the alternative approach is suggested. At the outset of learning configuration activity (tutor activity) the software could prompt the tutor to describe the type of audience and the perceived relationship between the audience and the complexity of the knowledge treated by the tutorial being designed. Following this several decomposition methods could be suggested to the tutor.

6. CONCLUSIONS

The context-centric approach to understanding tutorial interactions and the resources that mediate these interactions, entailed in the PSU framework, helps move away from the disjointed investigation of the relationship between the learning resources (LO) typology and the resource-use typology.

The generally positive results obtained from the evaluation of the ALSA, described above, suggest that the approach taken in our research is generally a productive one. Consequently, we claim that our evolving multi-agent support system has the potential to bridge the gap between Learning Design/Networks (i.e. drilling down into activity nodes) and Learning Objects. Furthermore, we claim that our multi-agent system appears to have the potential of providing a pedagogically effective approach to the management of learning resources inside a learning activity.

REFERENCES