PROJECT-BASED COLLABORATIVE LEARNING WITH ARGUMENTS

Flexible Integration of Digital Resources

Gert Faustmann

Department of Cooperative Studies, Berlin School of Economics and Law, Alt-Friedrichsfelde 60, 10315 Berlin, Germany

Keywords: Project-based learning, Collaborative learning, Cscl, Arguments, Digital resources.

Abstract: Success in project-based learning strongly depends on the experience of the project participants. Experienced learners often develop results that are not understood by less experienced participants. In this situation arguments can show the solution's approach by connecting parts of the given task with individual results. By using digital documents of arbitrary format it is possible to build a flexible net of arguments for the whole project. This paper describes an approach to integrate arguments with project-based learning and shows how to realize the use of digital resources in a web-based learning platform.

1 INTRODUCTION

Project-based learning is a widely accepted approach to teach and train problem solving capabilities. Especially in higher education complex tasks should be a subject in order to simulate and train real world problems. They are open tasks in the way that on the one hand the initial question may change according to a later research process and on the other there is not just a single solution (Jung et al., 2001). This kind of task is particular suitable for universities, because students can be prepared for today's business world's requirements. Additionally the integration of projects in class is motivating for the students and considerably improves their performance (Doppelt, 2003).

But some problems occur in realizing this learning approach: when using project-based scenarios in higher education, varying levels of student knowledge often are evident. An example is design and implementation of software systems in early computer science classes. Students have different prior knowledge according to their interests and education in school. This results in different competencies and volume of output. As a consequence after some time working in a project students with low experience in software development cannot further contribute ideas to the project, because they do not understand most of the parts of the solution already completed.

This paper introduces a concept to compensate different levels of prior knowledge by making project decisions more comprehendable. Less experienced project members will then be able to follow the solution process. This support will be implemented by arguments that combine parts of a solution with the requirements formulated in the project assignment or parts of it. An argument in this case need not be a logical assertion or a rule, but is information that helps to understand the particular result. We implemented the web-based learning environment WeCoLAr (Web-based Collaborative Learning with Arguments) that allows storing arbitrary digital documents as arguments and that gives the opportunity to navigate through a clearly arranged solution structure.

The following chapters at first introduce the constructivistic approach in collaborative online learning and show why and how to incorporate arguments into collaborative learning environments. Then the concept of arguments as supporting knowledge in problem solving is developed. Especially the integration of digital documents of arbitrary format will be shown. The paper concludes by discussing the presented approach and showing possible extensions.

Faustmann G..
PROJECT-BASED COLLABORATIVE LEARNING WITH ARGUMENTS - Flexible Integration of Digital Resources.
DOI: 10.5220/0003483403590364
In *Proceedings of the 3rd International Conference on Computer Supported Education* (CeLS-2011), pages 359-364
ISBN: 978-989-8425-50-8

2 RELATED WORK

While collaborative learning based on constructivistic principles is an accepted concept in learning theory there are different ways how arguments are used to support learning. Some approaches exist how to present these arguments to the students.

2.1 Constructivistic Learning Approach

Today's theory of learning favours a constructivistic approach, in which the student uses his/her experience for problem construction and problem solving. The teacher takes the role of a coach supporting the learners in managing the complex situation (Hmelo-Silver, 2004). In (Gilbert and Driscoll, 2002) constructivistic principles of learning are defined by

1. A group objective,
2. A cooperative group with interactions,
3. Individual decisions and means,
4. The use of integrated tools for saving results and for communication.

These principles are to be realized in tool-based online learning environments facilitating distributed learning. Especially wikis as tools to allow the distributed creation of documents play an important role (Augar et al., 2006). They are used in settings with different learning objectives (e.g. journalism (Ma and Yuen, 2008). Extending wiki tools for learning scenarios is investigated by some authors (e.g. in (Larusson and Alterman, 2009)(Pusey and Meiselwitz, 2009)). Wiki platforms can therefore be judged as an elementary part of online learning plattforms.

2.2 Arguments in Collaborative Learning

While wikis support a distributed collaborative creation of documents they do not offer explicit support in understanding the knowledge building process. There is plenty of work on how to use arguments to support project based learning. (Scheuer et al., 2010) give a very detailed insight to approaches and systems in this area. One common goal is to support the akquisition of argumentation skills in computer supported distributed learning platforms. The research questions in this area are

- the visualization of argument structures,
- the interaction between students via their arguments,
- the connection of arguments with ontologies and,
- the analysis of arguments with feedback to the students.

For the goal of balancing very different prior knowledge of the learners particularly the presentation (and with it the visualization) of the arguments and the knowledge may be of interest.

2.3 Visualizing Arguments

There is concrete research on how to visualize argument structures. Visualization supports the awareness of the other participants' knowledge and leads to a more efficient generation of new knowledge. (Sbarski et al., 2008) design a visualization tool to support a better understanding of argumentation structures. (Suthers et al., 2008) carry out a study investigating the support of knowledge maps in threaded discussions. (González et al., 2007) construct dialectical trees representing arguments automatically extracted from discussions.

An interesting software tool to incorporate and visualize arguments is SenseMaker (Bell and Linn, 2000). Arguments are web resources that are presented with their URLs. The overall structure is a container style where arguments can contain further arguments and so on.

(Land and Greene, 2000) investigate the knowledge building process in project-based settings from the perspective of information resources. They observe a topic drift while learners search for new information. Structuring the information is no subject in their study.

3 LEARNING SUPPORT WITH ARGUMENTS

3.1 Arguments as Explanations

Arguments in the proposed concept are used as explanatory statements for project results. They connect parts of the project assignment with parts of the solution. Beyond this connection arguments have a meaning that gives background knowledge for the way the part of the solution they are pointing to is designed.

Arguments need not be a logical statement giving a proof for the combined result. They support other project participants in understanding the result and its relation to the task.

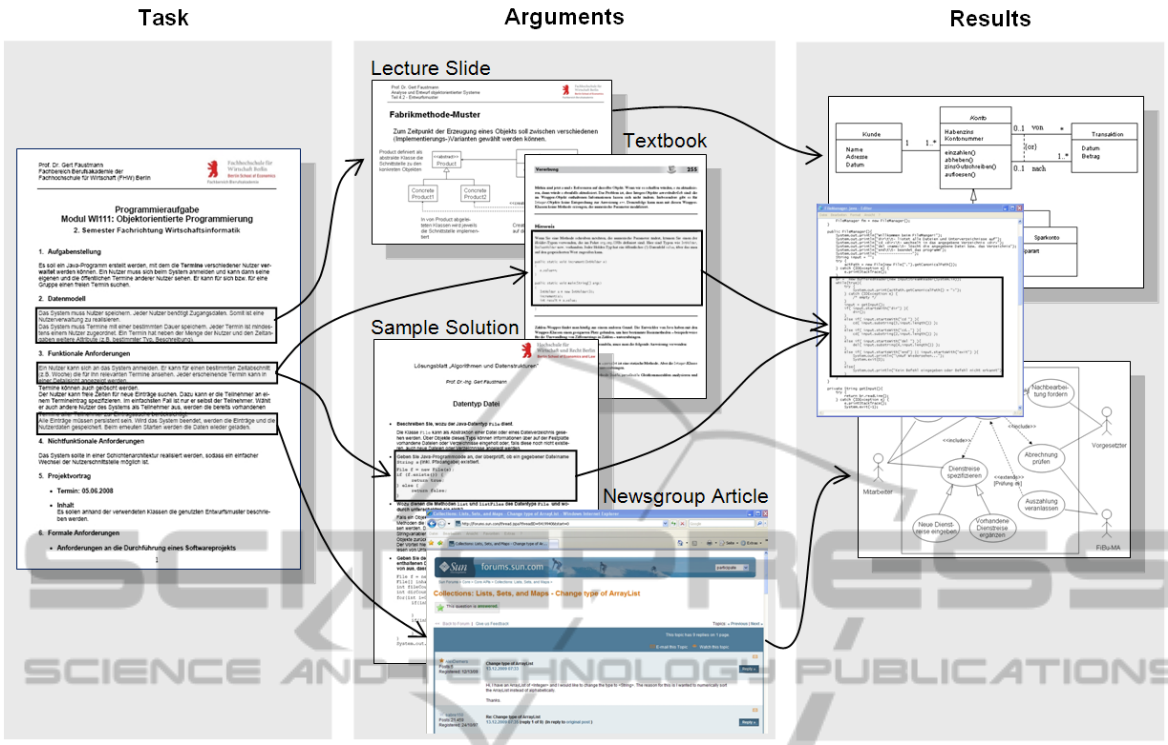

Figure 1: Arguments combining Tasks and Results.

In this way one can think of many types of documents being an argument. Knowledge from textbooks or lecture slides can explain a result. Active knowledge showing a method to solve a problem that is relevant for the given task gives good support for understanding a result. Examples for active knowledge documents are sample solutions and newsgroup articles. Finally, it should be possible to use arguments that are developed by the author of a result herself/himself. This can be an explanation in textual or graphical form.

3.2 Integration of Arguments

At this point we combine documents to show a development from tasks to results. In practice it is difficult to relate information on document level with one another. A task provided in an individual document may have different sub tasks. The same is with a resulting document that may have different solution aspects explained by different argument documents (see Figure 1).

Let us look at the following *example*: The project task is to develop a small software system that provides a command line interface (CLI).

This interface offers commands that allow the management of directories and files (e.g. deleting files and directories, showing the content of a file, making a new directory). Not only the program text is a result of the project but also some graphical design documents visualizing the algorithms used to implement the different commands. Furthermore there is a requirement that the interface should easily be extended by further commands. These assignments are formulated in one document. Resulting documents can now be the whole program code and graphical documents (e.g. in jpg format). Argument documents can be documents about how to design algorithms in a graphical way (e.g. basics of Nassi-Shneiderman-diagrams), parts of textbooks about programming basics and pattern documents showing how to make software programs easily extendable.

It is obvious in this example that linking whole documents that only are partially relevant is not useful. It should be possible to link just a piece of a textbook (e.g. how to realize an iterating program structure) or a sub task of the whole project task (e.g. just the “extension requirement” to link it with a command pattern document).

Figure 2 shows the relation between tasks, arguments, results and documents in the conceptual approach. Tasks, arguments and results are in a

direct relation with arguments combining tasks and their results. They all three are parts of documents

which themselves belong to exactly one document. The approach is to make a difference between a whole document and the parts of this document. Later on, a technical solution will be presented to implement these document parts.

Note that every resulting document must at least have one argument to be related to. This enforces the development of a net of arguments that can explain the whole set of results. But it is also possible that a result has more than one explaining argument. In our example it may be difficult to find the parts of the resulting program text corresponding to documents explaining iteration (for repeated reading of commands) and string analysis (for analysing the user's input). So it should be possible in this example to specify arguments pointing to the reading routine and also pointing to the routine analyzing the string typed in by the user.

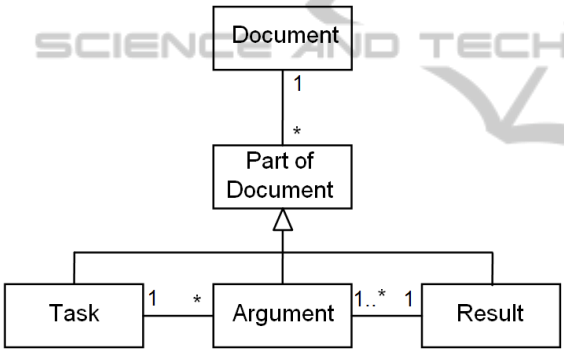

Figure 2: Integration of document artefacts.

3.3 Collaborative Learning Process with Arguments

The provision of arguments in the learning process can be twofolded: On the one hand the author of a new result in the project should present one or more arguments for her/his particular result. On the other the teacher may want to give some hints in the beginning of the project by providing arguments that may lead to results later on. This may also be done by the teacher while the project is already taking place. She/he is then able to lead the work of the project participants in the intended direction.

Taking the example of a software development project the teacher can provide the assignment document to develop the CLI together with documents describing the basic functionality of a command line interface. Moreover she/he can provide lecture slides showing some alternatives for making a software system extendable (e.g. some software patterns like command pattern or decorator pattern). Then the learners work on the project by adding results to the project space. Suppose the project members provide the program text implementing the input and analyzing routines and three commands. They also contribute digrams to show how the command routines are implemented. They must also provide argument documents for each result. An argument document can be a document provided by the teacher or a document that the author of the result loads into the learning platform. While accompanying the project the teacher may give hints how to implement the software system. Suppose she/he looks at the program text so far and is not satisfied with the solution. Then she/he may load a sample implementation of a command pattern into the project space to be used by the participants as an argument for their own implementation of an extendable system.

4 USING DIGITAL RESOURCES

This chapter proposes a concept that allows to implement document parts of arbitrary format. Especially the problem of a simple and ergonomic use of these document parts is discussed.

4.1 Format Heterogeneity

When trying to use portions of documents in a learning environment some problems appear: Many types of documents can be arguments within the suggested system approach. So there also may be very *different document formats* that have to be integrated in the system. The problem of representing *parts of a document* is even worse under the condition of managing multiple formats. Besides it should be possible for every project member to work at the documents and e.g. extend them. This means that every document must be *available in its native format* (e.g. the format of the used text processing system). At least there should be *no redundance* if a document is used in many contexts in a project.

The approach to deal with these requirements is visualized in Figure 3. The system stores not only the original document but also a universal representation of it. This will then allow to use a general method to handle the documents in the user interface (viewing, moving, etc.) and to work with parts of a document.

Each native document within the proposed

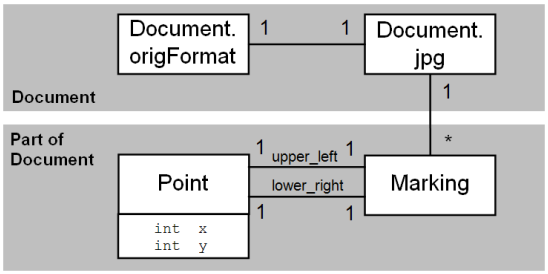

Figure 3: Realizing Document Parts.

system has a representation in a graphical format (here jpg). This is a 1:1-relation meaning that a document presented on the graphical user interface (the jpg-document) can easily be loaded in its native format. The graphical document can then easily be marked to define sub parts of the whole document. One document may have many markings. A marking just denotes a rectangular area in the graphical document by storing two points defining this rectangular area. Considering the argument net in which tasks are combined with results using arguments one can see from this implementation view that the nodes in the net are just markings related to graphical documents. On the other side, comparable to a document pool, there are the original documents and the graphical representations of them.

4.2 Overview and Navigation

While other approaches combine the graphical argumentation structure with text based arguments, arguments in this paper often are based on a graphical format (e.g. a lecture slide with some graphical portion).

As a consequence the technical concept includes the visualization of documents when presenting the whole argument net to the user. This is supported by the previously offered design rationale to present documents by their graphical representations. So the project member is able to see the dependencies between results and sub tasks as well as the contents of the documents respectively the parts of the documents.

The requirement to work with the document results in the approach to be able to have different zooming levels to look at documents, to get a full view of documents in the learning system and finally to save the original document into a participant's local memory. Because a net of arguments can grow rapidly over time it is possible to filter the perspective acording to the structure.

Dimensions in filtering may be the author (e.g. just see one's own results), or results and arguments based on one part of the assignment (e.g. show all about realizing an extendable software system).

4.3 Learning Platform WeCoLAr

We implemented a first prototype in a web-based platform for collaborative learning. The WeCoLAr-system is programmed in PHP and AJAX for building an interactive user interfaces. It is for example possible to freely move documents within the argument net to position new arguments.

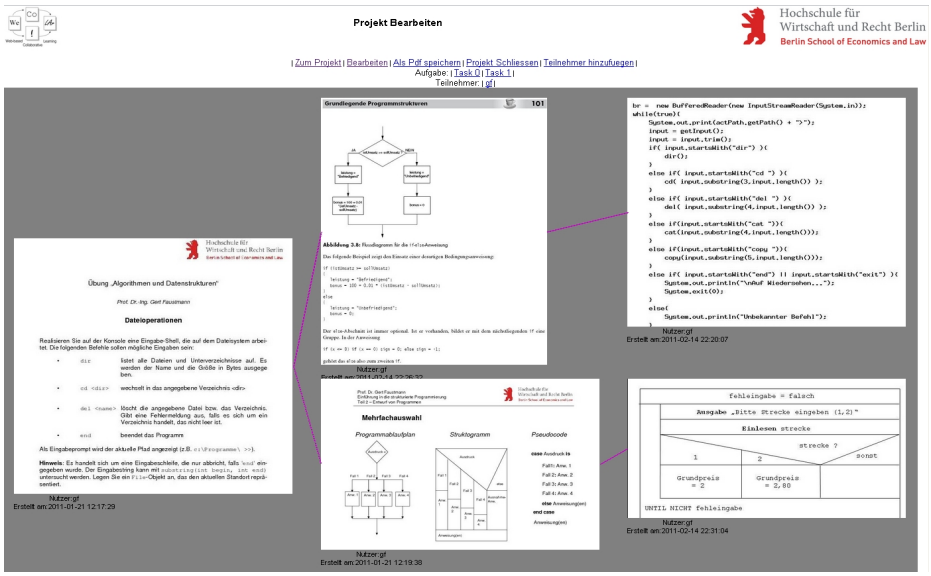

Figure 4: WeCoLAr User Interface.

Figure 4 gives an idea of the user interface of WeCoLAr.

5 SITUATION AND OUTLOOK

The presented approach is based on the constructivistic idea of wikis in collaborative learning environments (Augar et al., 2006). It contrasts the so far use of arguments in assisting discussions (e.g. in (Sbarski et al., 2008)) and supports the understanding of project results of arbitrary type and format. While SenseMaker also uses arguments for collaborative problem solving, it focuses on the logical argumentation process in scientific problems (Bell and Linn, 2000).

WeCoLAr will be tested in real world settings at university level. Beside the topic of programming software systems, we will try to use the system in business courses e.g. macroeconomics. Future conceptual work will incorporate a scripting approach to support the project members but also to enforce everyone's contribution to a project. These scripts will enforce time aspects (e.g. working on assignments one by one), and role assignment (e.g. defining who has to do which part of a project). A further control of the project work may also result in new learning models (e.g. provide a solution and let students find arguments for the solution). Further research is the implementation of conditions for the results of a project. Especially in software engineering many documents are compiled that all depend on one another. A problem for beginners when learning software development is exactly this dependancy between the different documents (e.g. program code and design documents). We will make these conditions also part of the learning process.

ACKNOWLEDGEMENTS

The project WeCoLAr is funded by the European Social Fund (ESF 2009000038) as a sub project within „E(r)lernen: Kompetenzvermittlung zum Einsatz neuer Medien". I like to thank Liangliang Gu for his contribution to the WeCoLAr system.

REFERENCES

Augar, N., Raitman, R., Zhou, W., 2006. Wikis: Collaborative Virtual Learning Environments. In *The International Handbook of Virtual Learning Environments*. pp. 1251-1269.

Bell, P., Linn, M., 2000. Scientific arguments as learning artifacts: Designing for learning from the web with KIE. *International Journal of Science Education*, 22(8), pp.797–817.

Doppelt, Y., 2003. Implementation and assessment of project-based learning in a flexible environment. *International Journal of Technology and Design Education*, 13(3), pp.255–272.

Gilbert, N., Driscoll, M., 2002. Collaborative knowledge building: A case study. *Educational Technology Research and Development*, 50(1), pp.59-79.

González, M. et al., 2007. Modelling Shared Knowledge and Shared Knowledge Awareness in CSCL Scenarios Through Automated Argumentation Systems. In *Groupware: Design, Implementation, and Use*. pp. 207-222.

Hmelo-Silver, C.E., 2004. Problem-based learning: what and how do students learn? *Educational Psychology Review*, 16(3), pp.235–266.

Jung, H., Jun, W., Gruenwald, L., 2001. A design and Implementation of Web-Based Project-Based Learning Support Systems. *The Human Society and the Internet Internet-Related Socio-Economic Issues*, pp.354–367.

Land, S.M., Greene, B.A., 2000. Project-based learning with the world wide web: A qualitative study of resource integration. *Educational Technology Research and Development*, 48(1), pp.45-66.

Larusson, J., Alterman, R., 2009. Wikis to support the "collaborative" part of collaborative learning. *International Journal of Computer-Supported Collaborative Learning*, 4(4), pp.371-402.

Ma, W., Yuen, A., 2008. A Qualitative Analysis on Collaborative Learning Experience of Student Journalists Using Wiki. In *Hybrid Learning and Education*. pp. 103-114.

Pusey, P., Meiselwitz, G., 2009. Heuristics for Implementation of Wiki Technology in Higher Education Learning. In *Online Communities and Social Computing*. pp. 507-514.

Sbarski, P. et al., 2008. Visualizing Argument Structure. In *Advances in Visual Computing*. pp. 129-138.

Scheuer, O. et al., 2010. Computer-supported argumentation: A review of the state of the art. *International Journal of Computer-Supported Collaborative Learning*, 5(1), pp.43-102.

Suthers, D.D. et al., 2008. Beyond threaded discussion: Representational guidance in asynchronous collaborative learning environments. *Computers & Education*, 50(4), pp.1103–1127.