Allocating an Industry Based Project to Student Teams of First and Second Year

Tomas Benz*, Ulrike Jaeger*
*Heilbronn University, Software Engineering, Heilbronn, Germany

Index Terms: collaborative learning, project based learning, different perspectives, wikis.

Abstract

Is it possible to allocate a single industry based project to several student teams in the first two years? Which can be the positive results for the students? How can we implement the approach into the rigid German university system?

These were some of the questions we had to face when we decided to base more classes in our software engineering bachelor course on project based learning. We aimed at an industry based project for our students in first and second year. The project requires skills in different fields, so we combine the power of two different classes and their different perspectives.

The actual project is to develop a project management and accounting system for a software company with a special software development tool and environment. To be able to complete the system development we also needed database functionality. We wanted the business and database parts to motivate each other, the customer company to really use the product.

The business class students are absolute beginners and will model the front end of the application. They use a wiki tool that helps to fill the gap between their limited knowledge and the database implementation supporting the business process. The other class is a second year database class with basic software engineering skills, experience with team work, and sufficient programming skills. They use the project to intensify their new database knowledge.

I. Educational Concept

A. From Subject Centered Teaching to Project Based Learning

Our discipline, software engineering, adds a special flavour to computer science. From the beginning, students learn to work on projects and apply engineering virtues to the software. Working in teams reflects the working life situation we want to prepare them for. Our curriculum reflects this to a certain point:

- In the second year we offered a big project subject as a learning stage for project work in IT.
- Almost all other subjects in our curriculum already used small projects.

This lead to a considerable number of single-subject projects per semester and a heavy work overload for students. Also, the single projects were mostly designed to illustrate their subject only and therefore were artefacts with at most some similarity to real life projects. In consequence, students did not learn to use the subjects' contents in combination. They did not develop strategies to use different subjects for one given complex problem.

So we started to look into teaching methods that emphasize project work on a general basis. We were influenced by the concept of Problem based Learning [1],[2], as implemented in Aalborg, Denmark [3] and the 4C/ID Model as implemented in Groningen, Netherlands [4]. However, the concept has to be adapted to the rigid German University system and its limitations in resources.

In order to prove that project centred teaching works, we started a series of experiments. The goal is to implement industry related projects in the first year, and real industry based projects in the second and third year. We aim at one carefully designed project per semester. That requires many changes in our way of teaching, our time and resource management as well as changes in the curriculum.

[1] E-mail: benz@hs-heilbronn.de

Before we start adapting the curriculum we decided to work with the present one and cooperate between different subjects if a given complex project requires. The current project is industry based and therefore suitable for students in the first year. However, it requires skills we teach in the second year. So we cooperate at the teachers' level and bias our teaching towards this single project instead of two different ones.

Besides the fact that cooperation among (German) university teachers is not very common, and that we are not (yet) used to plan our teaching around a complex application project, this situation is also novel for students. So we learn a lot from the experiment, but we think that students benefit all the more from it.

B. A Complex, Cooperative Project

Kristen Nygaard says, you must start with "sufficiently complex examples" [5] in order to give the students an idea about the real world. We agree. Cooperation between subjects not only relaxes the number of "subject projects", as stated above, but also helps the subjects to mutually motivate each other.

More important, we want the student to get a realistic picture of their profession as soon as possible, so that they experience the subjects as useful. Especially first year students have an unrealistic dream of coding all the time, and coding game software, preferably. Well, some tiny fraction might actually work in that field. The majority however, will develop software solutions in either business applications, or embedded software for automotive systems. Many students suffer through the first and second year, before their minds change during the internship. At that point, at last, they meet with real life in a company.

So the main goal for the actual project was to make beginners as well as second year students aware of their professional field and prepare a mental roadmap towards software engineering.

1) Benefits for First Year Students

In first year, our students struggle with their new situation and the content of our subjects is only part of the problem. Many underestimate the amount of time their learning needs. The failure rate is high and the overall message seems to be: *many of you are not fit to study here*. One of the most important new learning goals is therefore: *you can contribute to the success of a nearly realistic, complex project*.

First year students are used to business processes on the web, mostly as users, buyers at web shops, participants in collaborative play or work. Now students switch sides and model business processes themselves. And furthermore, they switch sides again by being customers to a group of more experienced second year students who will implement the more technical side of the project.

We want the project to succeed; therefore we decided to use an industry-related project rather than a real project with real customers. However, the industry is real: there is a small company which will later use the software. But the students are shielded from that customer by the teacher who plays the role but is not as demanding and hard on them as a real customer would.

2) Benefits for Second Year Students

Second year students perform in programming and other technical subjects quite well. However, they are not used to communicate with customers from an application field; they are not used to model real world situations and transform those into complex and appropriate software solutions. Again, there is no real customer too hard on them, but a group of first year students. They discuss concepts and ideas, and then translate the ideas of the not-yet-profesional beginners into software while using the newly achieved database knowledge.

C. Wiki as a Communication and Collaboration Platform - and More …

Problem or project based learning leads to special skills which Bruns calls "Produsers and Produsage in the Knowledge Economy" [7]. Produsers/Produsage means to educate our students not only to *USERS* of knowledge, not only to teach them the *USAGE* of knowledge and sources but as well to *PROD*uce their own content. So produsage has four fundamental characteristics [7]:

- It is community based; it is based on the collaborative engagement of (ideally large) communities of participants in a shared project.
- Participants occupy fluid roles. Different users or produsers will occupy different roles throughout the life of a project.

[1] E-mail: benz@hs-heilbronn.de

- The artefacts are unfinished. In this dynamic model products are always subject of ongoing work, adjustment, and updating, fixing, expansion and so on.
- The product is common property, although recognition of individual merit of contributors and contributions is a standard feature of produsage environments.

There is a growing need for education to discuss the process and practice of user-led content creation. Learners develop a more informed, self-reflexive, and critical perspective on their own practices as information seekers, users, and providers. A wider range of participants engage successfully in user-led environments [7].

To act as produsers requires special capacities [7]:

- Creative: content creation in a collaborative environment
- Collaborate: collaborative engagement under fluid, heterarchical rather than hierarchical structures
- Critical: critically engaging with content produced by others
- Communicative: the need for an effective and successful communication between participants

We use a wiki as the platform. Starting with the content provided by the teacher and a description of given problems, tasks or projects we follow the steps of project based learning [3] adapted to our possibilities and needs and try to motivate our students to add the content, results of their projects, their discussions within a workgroup as well as students' opinions about the results of other groups [7]. Providing the students with the technical framework of a wiki is only the first step, which makes life easier for the teacher:

- Students publish their own content themselves.
- Results, discussions and meeting protocols are all part of the material in the wiki.
- The process can easily be followed by the teacher

The wiki is a basic process framework for blended learning [9]. It still needs the role of the teacher or tutor as an observer, mediator and reviewer. The "scaffolding" process to slowly withdraw support is implemented in three phases:

- We create initial page templates with simple guidelines on the content. Students are encouraged to fill in their ideas.
- We set the topics for discussion. Students will detect discrepancies between concepts and models and will eventually find a common basis.
- We provide feedback on the work in progress [9].

Students will gain confidence along the process and need the teacher's feedback only in some cases. This requires a clear understanding of the project's learning outcome and its evaluation. Otherwise they are insecure and try to feel their way along possible exam questions or requirements. The learning goals are discussed before the class really starts. As the project proceeds, any changes in the learning goals are clearly communicated. Their evaluation is related to the learning goals and contains oral and written feedback for formative reasons, as well as an exam for grading.

Students in the first year come to university with inhomogeneous programming skills. We therefore provide this course with a specially featured wiki, which allows us to implement prototypes of user interface and workflow of software without any previous knowledge about programming.

II. ORGANIZATION

We do not join the two classes in mixed teams; both classes work separately and concentrate on their part. We organized the schedule of both classes so that the first year student groups develop the database specification. Next, the second year students complete the database part at a certain point.

[1] E-mail: benz@hs-heilbronn.de

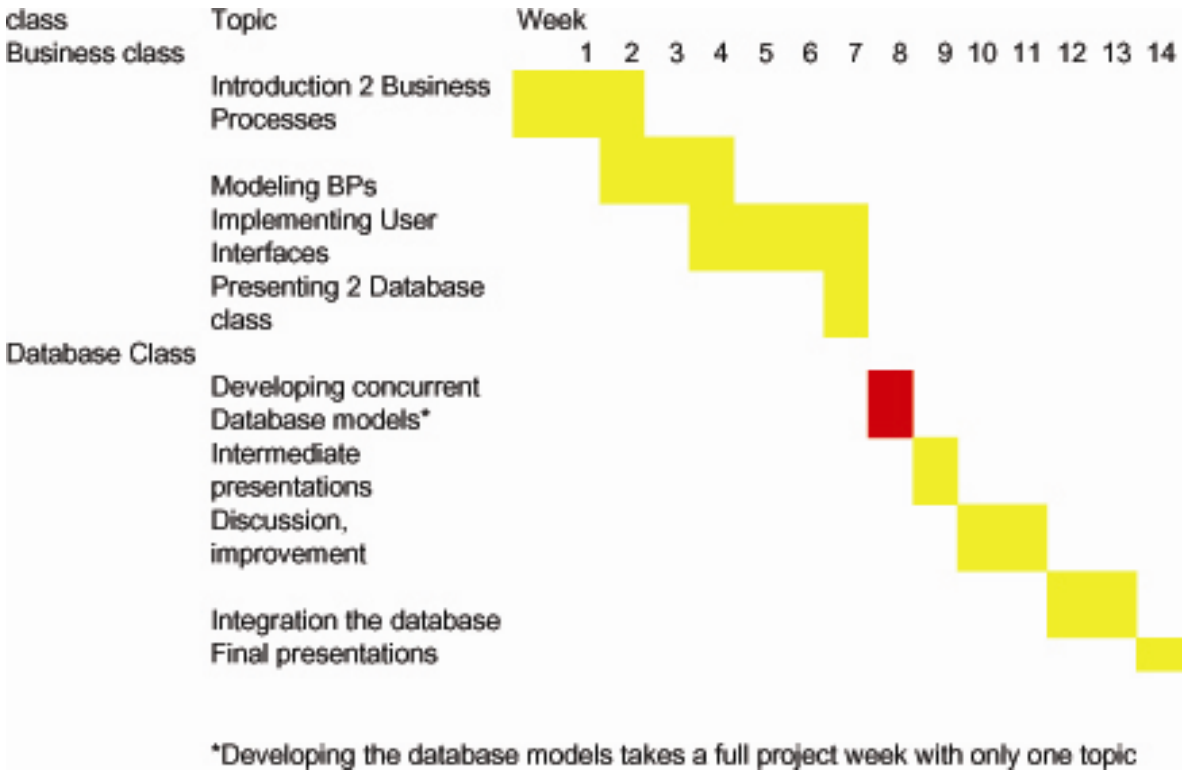

Fig. 1. Schedule of cooperative project

The first year students implement the interface using the company's tools. The classes worked in teams each, the instructors guide the team learning process and provide the necessary information on demand. Task description, team meetings, documentation, discussions and prototyping of the user interface use the wiki environment.

A. Business Class

The business class first identify the methods and models needed for developing a complete chain of user interfaces of a complex software product. For them, the topic, the teamwork and the kind of result is completely new. They complete a partly developed application including the user interface. Meanwhile, the database students get accustomed to the basics of database management systems. The business class evolved along four steps:

- identifying, analysing and modelling business processes
- free trial to implement user interfaces for the basics steps of each process
- identifying the key abstractions of each process and linking the user interfaces to the key abstractions
- crosschecking and refactoring the user interfaces for the final release (entry for the database class)

Step 1: We started with an introduction to the usage of the wikis in general and Wiki Creole [10] as quasi standard for a wiki mark-up language. In the wiki and in teacher presentations we provided the basics of business processes and how to make a simple model of a process in a flowchart. The students had to identify the basic processes in an enterprise and to find out the essential steps of processes like sales, procurement or accounting.

To identify the essential processes for the project, a project management and accounting system for a small software company, the students had to interview the "customer" (the teacher) about the way the company

[1] E-mail: benz@hs-heilbronn.de

manages projects, pays employees and bills its customers. So they found out processes like “offer, order or billing”

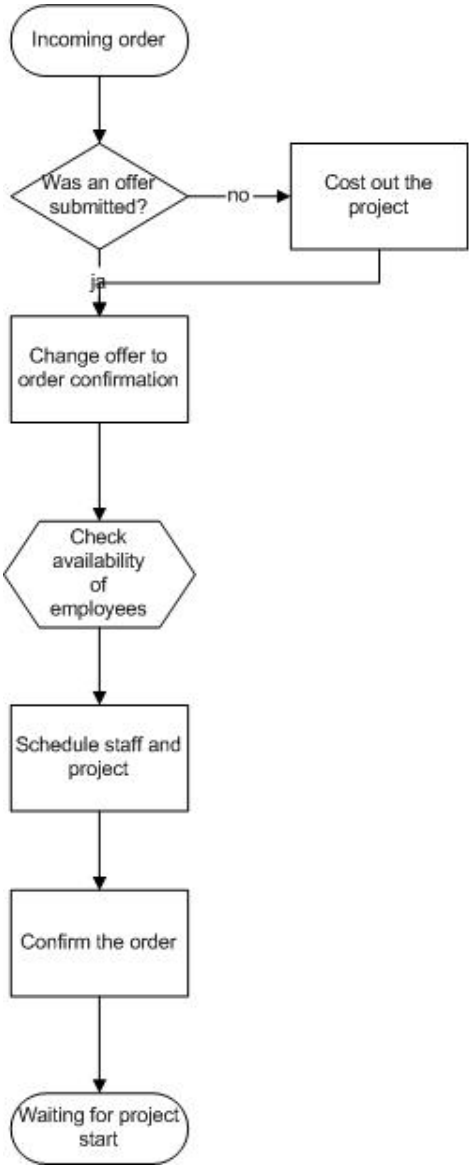

Fig. 2. example for a business process

In **Step 2** we build working groups of four students who have to manage two processes each. We provide the students with some extended markup for standard interaction items in the user interface like buttons, pull down menus, tables etc. as basic patterns. The students have a period of trial and error in building user interfaces and connecting them in a work flow. Every user interface is a single wiki page. Simple rules given by the supervisors help to name the pages properly.

So the architecture of the wiki leads students to conflicts. For example, teams with similar content want to name two wiki pages the same way. Or some teams talk about finding a common solution for the content for such a “common” page. Most teams try to find a workaround in breaking the naming rules and creating their “private” pages. They realize that they have forgotten some important internal processes like customer registration, time recordings of employees or administration of employees and suppliers.

So we force the students to discuss the conflicts or their work around in public. This leads to the next phase of the project.

[1] E-mail: benz@hs-heilbronn.de

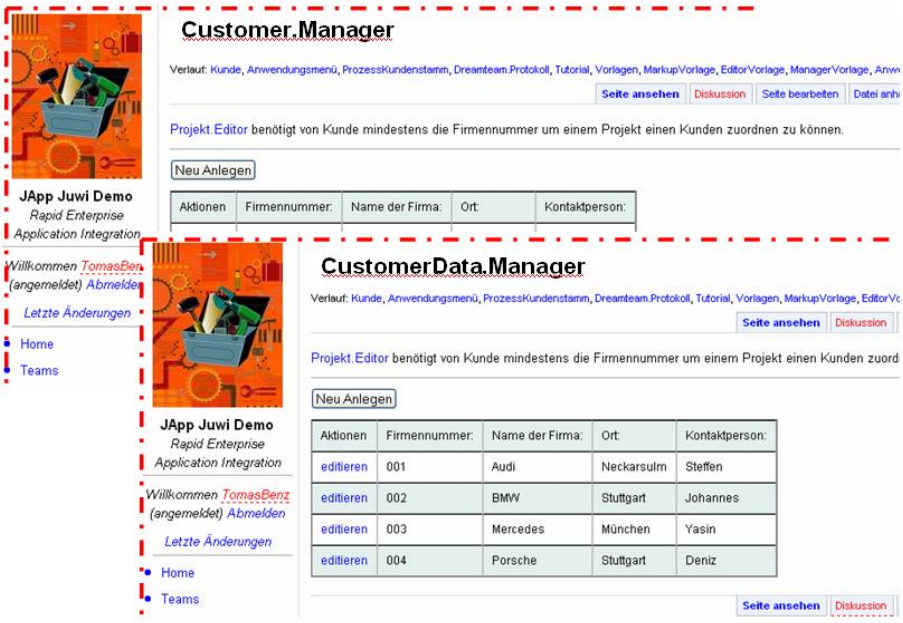

Fig. 3. conflicting wiki pages and their “work around”

Step 3: During the discussion started in step 2 we give the students an idea of some key abstractions for one of the processes as a basic solution pattern to their problem. These abstractions could serve as central objects for different processes where they can attach the user interface objects, following the naming conventions. So without naming it with the technical terms of software engineering the students become aware of the need of basic elements of an object oriented software system and its relationship to real world as an abstract model. In the following period of time the students identifiy the abstract objects step by step, and implement the user interfaces. Each group learns now, that they can easily use objects from other groups and attach their interfaces or integrate their aspects to an existing interface.

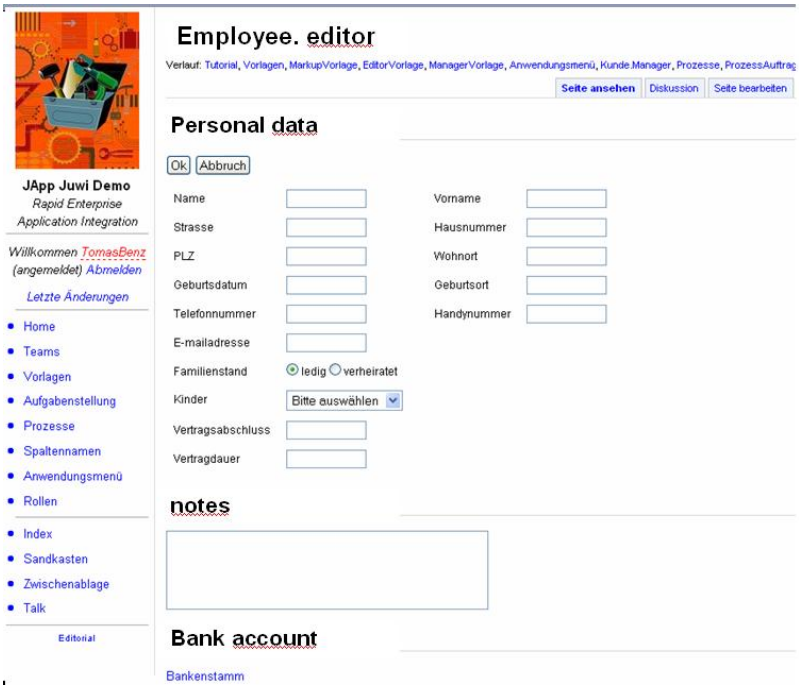

Fig. 4. example of a user interface

[1] E-mail: benz@hs-heilbronn.de

Step 4 starts with the presentation of the different objects to the other groups of the business class and another discussion about the optimal content (data elements) and connection of the user interfaces. The teams revise their key abstractions or add some missing objects and to complete the description of processes, key abstractions and user interfaces. So they provide all input information for the database class to be able to continue the project. The business class then presents the results to the database class. After implementing the database and connecting it to the front end the database teams present the completed solution to the business class again and to the final customers, which are real employees of the software company.

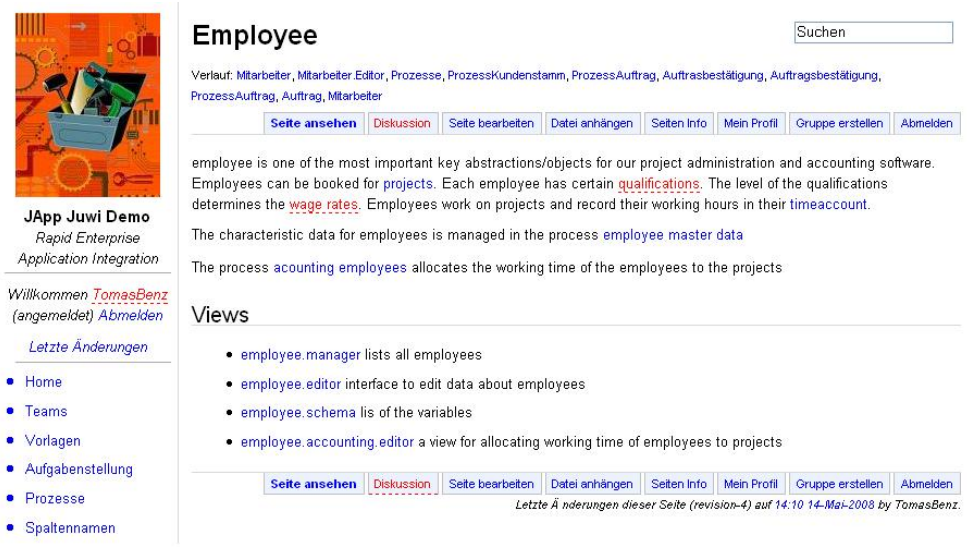

Fig. 5. a key abstraction object and related interfaces

B. Database Class

Until mid-term, this class is taught basic database techniques, especially modelling applications from small examples, and SQL queries, also from a ready-made small example. In mid-term we gather as a group during a complete week of project implementation. The necessary input is given in short lectures as teaching on demand. The different perspectives of the project are represented by a team of different instructors. This follows the *team teaching pattern* of the pedagogical pattern project (http://www.pedagogicalpatterns.org/). It demonstrates teamwork on the teaching level. Furthermore, the different persons, styles and settings help to organize the different kinds of knowledge and reduce the cognitive overload [2].

- Tool usage: one instructor shows the special database adapter to the wiki tool.
- Technique: the database teacher polishes their SQL knowledge.
- Application: the business teacher and the first grade students discuss business implications of the implementation.

Based on their analysis the group develops versions of the database part during this week in competing teams. During the rest of the semester, those solutions are presented, discussed, and merged into better solutions. The best result is integrated into the overall project, which will be used by the customer company.

[1] E-mail: benz@hs-heilbronn.de

III. LESSIONS LEARNED

Our approach is a first step towards "apprenticeship perspective" [6]. Students work on projects and gradually evolve from novice to experienced learner.

A. Observations

The first feedback from students tells us that we could spark the interest in business applications and the understanding that they need to know about the basic terms and their interrelation. In the first approach we could not really provide all the basic knowledge of business administration in the context of the software to be developed. But we initiated the interest to look for some of these informations on their own. On the other hand we achieved more than we intended: a strong curiosity to learn more about the software development process. Interesting aspects were also:

- Starting with a complex project in beginning of the first year gives students a mind-blowing kick. They became aware of complex possible application of our subject contents.
- For about two thirds of our students readily followed our teaching change from push to pull principle. The last third at least did understand what we were talking about. In classical teaching we would have possibly lost them.
- We kept close contact to the teams in order to provide them with just in time information and hints about useful solutions, or, even better, ideas to fire their own knowledge and imagination on the path to a solution of their own.
- The wiki was a powerful basis for sharing ideas and detecting contradictions in concepts and models.

B. Next Steps

We found the project too complex for the available amount of four contact hours per week for the business class. Hence we had to reduce the theoretical background to the necessary minimum for understanding the project. Since this was the very first project or our beginner students, they would have done much better with more input about software project steps and methods (and, they asked for it!).

Next semester we will start a similar project. Instead of integrating two subjects we will have three supporting subjects: business processes, databases, and basics in software enginering. This is a chance to integrate more supporting information as well as giving them more contact hours and time for a project of comparable complexity.

Our hope is that this approach proves so successful for both students and supporting teachers that other teachers will join in the initiative. In the end we might have the one project per semester supported by many subjects. Even if this means to adapt the curriculum, because no teacher wants to be left out, and some subjects are needed at a different moment during the learning process of becoming a real software engineer.

REFERENCES

[1] B. J. Duch, S. E. Groh, D. E. Allen (ed.): *The Power of Problem-Based Learning*. Stylus Publishing, Sterling, Virginia, 2001.

[2] H.S. Barrows, R.M. Tamblyn: *Problem-based Learning: An approach to Medical Education.* Springer Publishing Company, New York, 1980.

[3] A. Kolmos, A., Fink, F.K.& Krogh, L. (eds.): *The Aalborg PBL model. Progress, Diversity and Challenges*. Aalborg University Press, 2004.

[4] J.J.G. van Merriënboer, *Training complex cognitive skills.* A four-component instructional design model for technical training. Educational Technology Publications, Englewood Cliffs, New Jersey, 1997.

[5] K.Nygaard: "A Sufficiently Complex Example", http://heim.ifi.uio.no/kristen/FORSKNINGSDOK_MAPPE/F_COOL1.html, visited 2008-05-06.

[6] D.D. Pratt,. and J.B. Collins: *The Teaching Perspectives Inventory*. In Proceedings of the 41st Adult Education Research Conference, Vancouver, B.C., 2000.

[7] A.Bruns, S.Humphreys: *Building Collaborative Capacities in Learners*. In Proceedings of WikiSym 2007, Montreal, Canada

[8] T.Benz: *Wikis as a platform for problem based learning*, In Proceedings of ICEE 2006, Puerto Rico, 2006.

[9] M.Cubric: *Wiki-based Process Framework for Blended Learning.* In Proceedings of WikiSym 2007, Montreal, Canada, 2007.

[10] C. Sauer, C. Smith, T. Benz: *WikiCreole: A Common Wiki Markup*, In Proceedings of WikiSym 2007, Montreal, Canada, 2007

[1] E-mail: benz@hs-heilbronn.de