Research and Teaching with Remo:

Student research projects and teaching for and by undergraduate students

Dagmar Monett*
Computer Science Division
Department of Cooperative Studies, HWR Berlin
Berlin, Germany
Dagmar.Monett-Diaz@hwr-berlin.de

Mario Sänger
Student trainee at ITSO GmbH and
Computer Science Division, HWR Berlin
Berlin, Germany
Mario@itso-berlin.de

Abstract—**In this paper we give insights about our experiences in both undergraduate research and supervision of Computer Science students. We focus on how undergraduate students support teaching when this is included in their student research projects. Concrete settings for an Artificial Intelligence course give challenging ideas for further action.**

Keywords-undergraduate teaching; student research; sandwich principle; supervision

I. Introduction

Delegating responsibilities to the students and making them a part of the decision-making process seems to be a common practice in both student research and teaching. Even so, combining both of these aspects in a successful way is not as simple as expected when considering research and teaching together.

Berrett comments in [3] that teaching is no longer a distraction from research, since a recent study demonstrates that students involved in both have better improvements in research skills than those conducting only research. Although these findings were analyzed for graduate students teaching in undergraduate courses, we believe the same applies for their undergraduate mates. Certainly, no few skills, attitudes, and methodologies are important for both research and teaching, disrespecting the awarded level.

Would the relationship between research and teaching be strengthened when students incorporate in their research teaching those contents they research in? What happens when subjects for research proposals are discussed with undergraduate students in advance and they include developing computer-based software for improving teaching? How do co-teaching models [5, 9] for instruction can be applied considering student research projects and teaching for and by undergraduate students?

We have answers to these questions regarding our experiences in an Artificial Intelligence (AI) course at the Berlin School of Economics and Law (BSEL).[1]

* Contact author.

[1] The BSEL is a University of Applied Sciences. In German: HWR Berlin.

II. Computer Science at the BSEL

The Computer Science Division from the Department of Cooperative Studies (DCS), former Faculty of Company-Linked Programs, at the BSEL, offers since 1993 a solid education in the field of Computer Science, from software development to systems support [1]. A clear focus is put on applied skills, since students are part of a special cooperative program that combines full-time classroom study with regular practical on-the-job trainings at business enterprises. Upon successful completion of their studies, students graduate with 210 ECTS-credits[2] have access to all the classic professions of a computer scientist, with both deep theoretical business knowledge and company-based practical working experience.[3]

A. Study Research Projects

Undergraduate students from Computer Science at the DCS have to accomplish two study research projects (SRP) as part of their curricula. Such projects allow students to research in a specific topic that can be proposed by educators from the BSEL, or by their direct supervisors at the training companies, or, in the ideal case, by both. With no doubt, the last constellation encourages applied research in a stronger way.

SRP duration takes a maximum of six months in the 4th and other six months in the 5th study semester. Each semester comprises both a 3-month theoretical and a 3-month practical phase. During the theory periods, students profit from an intensive, on demand coaching from their DCS academic supervisors because they are almost always at the university. During the practical phases, the contact is kept mainly through email, since students are at their training companies. There, they profit much more from their company advisors and real-live practice.

Students get 14 ECTS-credits with each SRP. They should learn how to work on a subject-specific or cross-disciplinary task, independently. Although research in a new topic or field is encouraged for the second SRP, an in-depth analysis of a

[2] European Credit Transfer and Accumulation System. In Germany, one credit point is equivalent to 30 hours of study.

[3] See more about the Department of Cooperative Studies at http://www.hwr-berlin.de/en/department-of-cooperative-studies/. The enterprises spectrum comprises more than 650 companies from Berlin and Germany.

complex topic, as natural continuation of the first SRP, is also possible.

Partial results should be presented in the context of a short talk and in a comprehensible, stylistically convincing form. It takes place on the *projects' week*, where a deep discussion is expected and other classmates or interested guests can be part of the audience. At the end of the practical phase, students should submit a written report (like a small thesis of about 4000 words or 25 pages long) for evaluation by a BSEL professor.

For evaluating a SRP, BSEL professors weight the following criteria:

a) Theoretical considerations (8 points): Literature review, decision making, discussion of implementation approaches, and argumentation.

b) Practical realization (32 points): Design, practical application of technical knowledge, selection and use of tools, and implementation.

c) Initiative and commitment (8 points): Original ideas, ability to comment and to criticize, and independence.

d) Working style (8 points): Work organization and style, cooperation within the team and with other persons.

e) Subject-specific content (16 points): Definition of constraints and termini, evaluation and analysis of methods, description of goals, time management in reaching these goals, and discussion of results.

f) Work structure and organization (8 points): Outline, structure, style, and bibliography.

g) Oral presentation (20 points): Selection of materials and information density, use of media, verbal expression, and time management.

Each criterion deals with several aspects but is not limited to them; they serve as an orientation to the BSEL professors. Criteria related to the written report, i.e., from *a* to *f*, represent 80% of the SRP grade. The last criterion completes the final evaluation on 20%.

In the following sections, we give insights from our experiences supervising SRP and how undergraduate students strengthen both research and teaching, based on an Artificial Intelligence course at the DCS.

III. THE AI COURSE

The AI undergraduate course is an optional 7-credit course and is part of the curricula in the 5th study semester. Students learn there the focus, history, termini, and applications in AI. They develop theoretical and practical basics on modeling and representing knowledge, as well as on processing it using both problem solving and learning strategies, from knowledge to agent-based systems as well. This is why the AI course comprises two main areas: autonomous agents and multi-agent systems, and knowledge-based systems. We have located subjects for SRP mainly in the former area since 2008. Topics for SRP in the latter have been centered on the inference in expert systems.

The book from Beierle and Kern-Isberner [2] on knowledge-based systems is followed for introducing the theory and practice on expert systems in our AI course. Special attention is put on the inference mechanisms using forward and backward chaining algorithms. The need to better explain these reasoning algorithms to the students gives us the idea for a SRP's subject: how to visualize their functioning, as well as studying the impact of visualization in the classroom. We even think beyond these limits and ask ourselves, what happens when the student self teaches these contents, too! This way, we define the tasks, goals, and general requirements for a SRP, including the implementation of a software program.

The second version of the resulting software product is Remo (Rule-based expert system modeler), a program for modeling rule-based knowledge and for analyzing reasoning algorithms' functioning. We will describe Remo further in the next section. Its main features comprise the definition of condition-action rules, the graphical representation of rule-based knowledge, the visualization of inference processes, and a step by step execution of both data- and goal-driven algorithms. We used it successfully in the AI course, which we will also explain in the following section.

IV. REMO

Remo was entirely developed by Sänger[4] as part of two student research projects [10, 11]. It was of special importance for the AI course (and as SRP, too) because of the lack of adequate software support for both visual modeling knowledge and for analyzing inference algorithms for rule-based expert systems. To our knowledge, there is no computer-based software tool available for teaching these contents in depth.

The first SRP [10] was concerned with the development of basic functionalities for Remo, like a graphical editor for modeling rule-based knowledge, the implementation of algorithms for backward and forward chaining inferences, and their debug visualization. During this SRP, the student had to run through the four major phases of any software development process: analysis, design, implementation, and testing. The analysis phase was devoted to the collection of requirements for the application. In the design phase, a concept for the implementation of the requirements was conceived, i.e., relevant solution methods were selected and evaluated. A model-driven approach using the Eclipse Modeling Framework (EMF) [7, 12] and the Graphical Modeling Framework (GMF) [6] was used. Afterwards, the concept was implemented and tested. During the implementation phase, there was much attention paid to a modular component structure and a good extensibility of the application. Therefore, each main feature, like the graphical editor or the inference algorithms, was implemented as a separate plug-in [4]. Moreover, the extension point mechanism of the Eclipse platform was used to provide an easy way to extend the application, e.g., for adding new inference algorithms.

[4] Mario Sänger currently attends the 6th and last semester in the Computer Science career at the Computer Science Division, DCS.

In the second SRP [11] additional functionalities were developed. The application was extended with different export features, which allow the user to transform the graphical model of the application into other (textual) representations. Two transformation options were provided: the user can transform the model into simple "IF … THEN …" rules or into Prolog syntax. The exported Prolog model works with various Prolog implementations like SWI-Prolog [13]. It is also possible to validate the created models, e.g., to check whether each element has a unique name and whether it is connected to other elements or not. Resulting error and warning messages are shown at the top left corner of the graphical elements. Furthermore, an event log console was developed; it increases transparency of algorithms functioning to the user. Moreover, the use of Remo in the AI course was also a major activity in the second SRP.

Fig. 1 shows an excerpt of a knowledge base created with the graphical editor of Remo. Ellipses represent facts and rectangles are conclusions or further conditions in rules. Check marks on facts indicate that they are true, i.e., that they are evidential facts used for reasoning. Cross marks specify that a fact is false. If a fact node has neither a tick nor a cross, then the state of the element is unknown, and it will be determined during the inference process according to user information. Rules relate conditions to actions. A conjunction symbol in a rule node means that all connected nodes from the left should be satisfied as preconditions in order for the action part of the rule to be executed. If a node has a disjunction symbol, only one precondition needs to be satisfied for the rule to be executed.

The given excerpt is taken from an exercise which was created by Mario Sänger and that deals with the evaluation of visiting requests in a museum. It shows that, for a valid visiting request (VVR), both a full list of participants (PL) and a guide by a trustworthy group leader (GL) should be satisfied. A trustworthy group leader is either a teacher from a local school (LT) or a person who has already guided a group in the museum (AG).

Fig. 2 shows the debug console of Remo. With its help, the user can better follow the inference algorithms' functioning. This allows a gradual walk through the inference process. The console works together with the graphical editor, i.e., the currently focused node in the debug console is simultaneously highlighted in the editor by filling its background to gray (see also Fig. 3.)

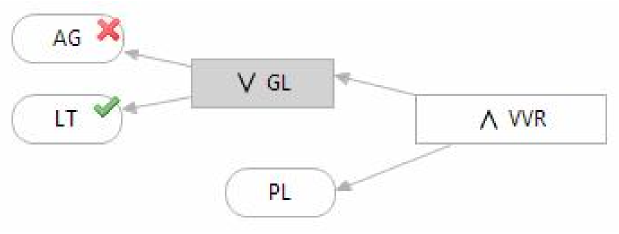

Figure 1. Example of rule-based knowledge in a graphical representation using Remo.

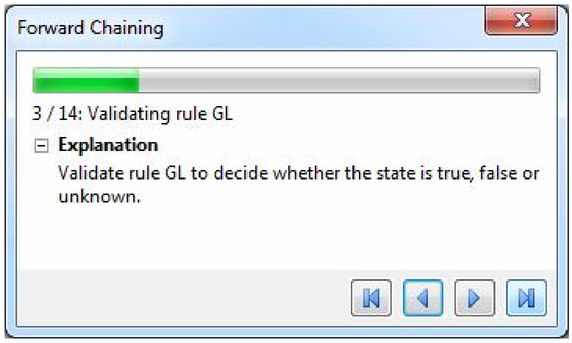

Figure 2. Debug console for the forward chaining inference algorithm.

Fig. 3 shows the complete user interface of Remo when the execution mode is activated. Further views, like a properties view of the selected node or an outline view, are located at the bottom in the user interface.

During both research projects the student gained valuable experiences regarding the software development process and the presentation of project results in suitable form. These experiences are also very useful in a variety of tasks and projects at both the BSEL and the business enterprise students are enrolled at during their dual studies. The following section discusses one of these applications, i.e., the use of Remo in teaching.

A. *Remo in the classroom*

AI contents related to expert systems were prepared and taught in a collaborative way, profiting from the benefits of co-teaching. The supervisor and the undergraduate student divided responsibilities for planning, for teaching, and for moderating learners' work on exercises. A combination of co-teaching approaches like "one teach, one observes" and "team teaching" was followed.

Furthermore, we applied a sandwich principle consequently in our lecture on expert systems, i.e., we combined theory units of about 20 minutes with practical examples and exercises with Remo in order both to motivate and to increase student attention. For this, separate meetings and email communication between the undergraduate student and the supervisor were planned and carried out. Such discussions strengthen the student's preparation on the contents to teach (Remo related, but also about inference in expert systems). They helped in designing course slides, examples, and exercises for the lecture as well, the latter especially conceived to support active learning in the classroom.

The student considered the following schedule when teaching new contents to the other students:

1) Remo I: Introductory part with focus on the SRP subject, on the main features of Remo, and on how to work with it. Duration: 10 min. Media and materials: lecture slides, data projector, Remo.

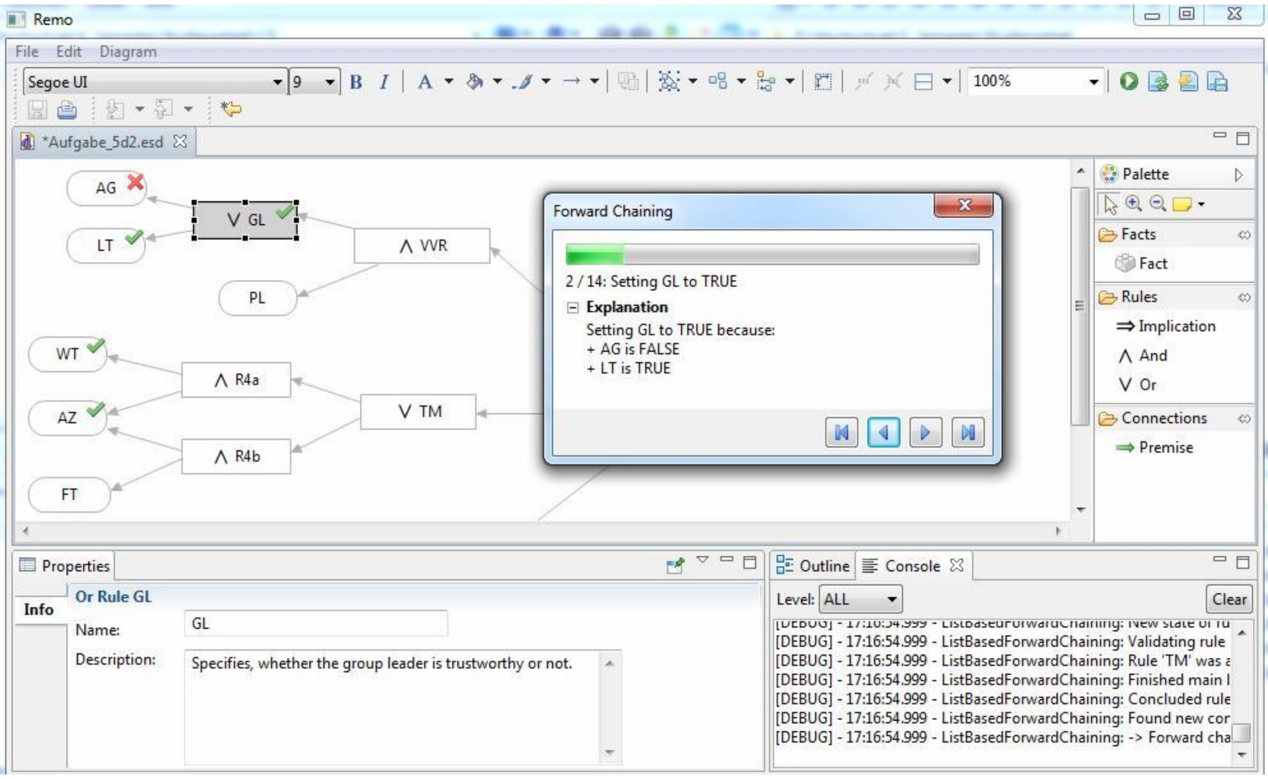

Figure 3. Remo's user interface in execution mode.

2) *Forward chaining:* Theory part with focus on the forward chaining reasoning algorithm and on its functioning. Duration: 10 min. Media and materials: lecture slides, data projector.

3) *Exercise forward chaining:* Practical part with exercise on forward chaining, first manually and then with Remo. Duration: 30 min. (20 min. solving; 10 min. discussing). Presentation of solutions and discussion: in plenum. Media and materials: individual materials, blackboard, Remo, data projector.

4) *Remo II:* Theory part with focus on implementation aspects in Remo and on related technical details. Duration: 5 min. Media and materials: lecture slides, data projector, Remo.

5) *Backward chaining:* Theory part with focus on the backward chaining reasoning algorithm and on its functioning. Duration: 15 min. Media and materials: lecture slides, data projector.

6) *Exercise backward chaining:* Practical part with exercise on backward chaining, first manually and then with Remo. Duration: 25 min. (15 min. solving; 10 min. discussing). Presentation of solutions and discussion: in plenum. Media and materials: individual materials, blackboard, data projector, Remo.

7) *Integrated exercise:* Practical part with complex exercise integrating all contents together, in the form of a mock exam. Duration: 80 min. (30 min. solving; 50 min. discussing). Presentation of solutions and discussion: in plenum. Media and materials: individual materials, blackboard, data projector, Remo.

8) *Questionnaire to evaluate Remo and teaching:* Duration: 10 min. (see next Section for details.)

Schedule points one to six were accomplished in the first day the undergraduate student taught. The last two were accomplished a day after, at the beginning of the next lecture. We prepared also several exercises for individual learning and distributed them in the classroom, together with other teaching materials. All of them were additionally uploaded to the BSEL's E-learning platform, i.e., to the Moodle[5] course site.

Short discussions and deep reflection about the schedule completion and other organizational issues were carried out before, during (in the breaks), and after the lectures. Through them, a dynamic adaption of the already planned course's guide was possible, depending on the current development and on the invested time. Feedback and coaching was also offered to the undergraduate student for improving and refining didactic methods to use and in order to reduce uncertain feelings about own teaching techniques.

Several examples and exercises were carefully selected from [2]. The undergraduate student also had the possibility to design new examples and exercises. This was included in his evaluation, i.e., the ability to create new, adequate content to teach. There also evaluated his teaching methods, his answers to questions, his preparation in the field (AI contents to teach were new to him), as well as additional teaching materials prepared by him.

[5] Modular Object-Oriented Dynamic Learning Environment, a free source course management system.

V. Evaluating both Teaching and Remo

At the end of the AI course part taught by Sänger, we distributed a short survey for evaluating both teaching by students and Remo as well. The questionnaire's primary objective was to get feedback from students mainly about the impact and use of Remo, as well as about using undergraduate students for teaching contents in the AI course.

The survey questions were proposed by the student and they were discussed in advance with the supervisor. Conceiving, applying, and analyzing survey results were also included as part of his evaluation in the AI course.

A. Content

Students were asked to answer up to eleven questions divided in two major areas: teaching and Remo. Table I shows the main contents and the questions' scope, as well as the answer types that were considered.

B. Results

Although the survey was applied to a relatively small group of students (13 attended the AI course), its results were very inspiring to us.

Testing the application with the hope of getting valuable feedback for Remo improvements was, as expected, positively welcomed by the students. Their opinion and suggestions (during the lecture and after processing the survey results) helped us to further complete a to-do list with new features, changes, and necessary modifications to Remo. For example, while visualizing reasoning algorithms with Remo was rated from "good" to "very good" by 9 of 12 students, the usability of the user interface still needs to be improved: only 4 students rated the Remo usage as "good" or "very good"; 7 evaluated it as "satisfactory", instead.

TABLE I. Questionnaire Content and Scopes

Nr.	Main content	Type[a]	Focus on
1	Teaching by undergraduate students.	Rating scale	Teaching
2	Quality of the lecture when students teach.	Rating scale	Teaching
3	Sandwich principle and lecture's structure supporting the learning process.	Rating scale	Teaching
4	Using sandwich principle and lecture's structure in other courses.	Yes/No	Teaching
5	Suggestions for teaching improvement.	Open-ended	Teaching
6	Remo supporting the learning process.	Rating scale	Remo
7	Remo usage.	Rating scale	Remo
8	Suggestions for usage improvement.	Open-ended	Remo
9	Visualizing reasoning algorithms with Remo.	Rating scale	Remo
10	Suggestions for visualization improvement.	Open-ended	Remo
11	Suggestions for Remo improvement.	Open-ended	Remo

a. All ratings based upon a six-point rating scale.

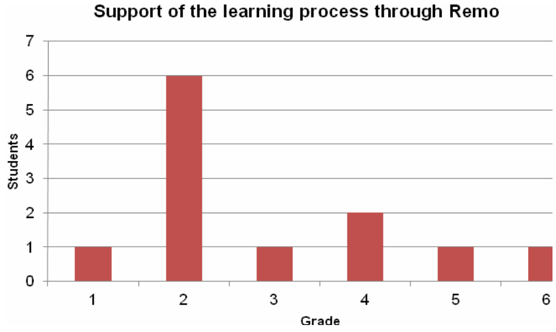

Figure 4. Results to the question "To what extent Remo supported your learning process in the subject field?" from 1 (very much) to 6 (no support).

Suggestions in this concern were introducing shortcuts for the creation of facts and rules and drag-and-drop functionalities, for instance. Nevertheless, more than half of the students (7 of 12) felt that the support of the learning process by the application was "good" to "very good" (see the bar chart in Fig. 4 for detailed results).

Regarding teaching, 11 of 12 students reported that teaching by undergraduate students made the lecture more interesting. That both the sandwich principle and the lecture's structure support the learning process was also in corroboration with the respondents. Furthermore, also 11 of 12 students wish such a course design in other modules of the curricula.

VI. Lessons Learned and Discussion

When students teach students, all they share is a common language that helps them to understand new contents "differently," often even better. On the other hand, explaining others any content, preparing it in advance, and asking questions from others is a very powerful way for individual learning. Furthermore, classmates' questions flow more openly and unbiased because students mostly know each other and there already exists a relationship between them. Thus, active participation and questions in such a relaxed atmosphere is encouraged, as well as friendly discussions that enrich effective understanding, even when a student in front of the class is being evaluated, too. In general, we could perceive a positive group reaction to questions, comments, and exercises. We believe this contributes positively to both education and learning quality.

A bidirectional support is very motivating for all students: SRP students that teach support others when learning how to work with the new software tool; they answer questions in the classroom and beyond the course hours, they broadcast their own positive experiences and enthusiasm proudly, feeling they are like real experts (they are in what they have developed and researched on for months!). But they also receive ideas and suggestions for further work in their SRP.

Interestingly, classmates are part of the software development willingly because they contribute to the testing process as end users while they solve exercises in the new contents. This kind of *beta* testing, i.e., testing in different environments and computer constellations, as well as usability testing concerning the user interface and Remo usage, contribute enormously to software validation. Moreover, bugs or problems were welcomed in order to improve Remo, some of them being rapidly fixed or modified as appropriate.

Students are in general more motivated in such a lecture for another three reasons. First, some of them usually compare the current work to their own in a different SRP and often times with different supervisors. Second, this internal competition stimulates favorably the emergence of new ideas and discussions between students, too. Finally, understanding new contents by visualizing the way some algorithms work contributes to the learning process in the field and is welcomed when students assist.

A well-planned sandwich structure helps supervisors to find a better balance between frontal lecture and students' needs. While a student is teaching, the supervisor could invest more time in observing other students' behaviors, in completing ideas with parallel comments or alternative explanations when needed, in helping to moderate discussions, in annotating successful or critical lecture development for further analysis, in assessing student's achievements and performance, as well as in simply thinking on new challenges that can improve teaching in the future. Exploiting combinations of co-teaching models and approaches is also a key to success for lecturers when working with undergraduate students.

We suggest planning a differentiated evaluation schema in advance. This is of special importance when the SRP student should also attend the course he or she is working in his/her SRP for. In what extend which contributions and related work belong to the SRP or not, as well as what will be part of the course evaluation, should be well-defined and discussed with the student before both the SRP and the course start. Seeking a balance between course work and research as part of a SRP should appropriately be decided upon by the supervisor. We could consider the role as teaching assistants and the active development in the class as part of the course evaluation for those students working in SRP for the AI course. Thus, we did not include questions related to expert systems in their final exam, for instance, because such students are "almost experts" in that area.

Critic discussing and planning together SRP subject, tasks, goals, and applications in the classroom have several benefits for lecturers but also for students doing SRP. Giving the students enough freedom to propose new features, to suggest didactic methods according to their learning types, to decide on what to teach and how, and even to break traditional teaching schemas, enrich their willingness to both teach and research, as well as their skills, talent, and disposition. Depending on their abilities and needs, for example, re-engineering an existing software program in order to include new functionalities or enhancing existing features to be applied in teaching, could be well part of SRP, too. We believe delegating such responsibilities to the students welcomes decision-making and decentralizes the lecturer's tasks and efforts in an innovative way. In addition, such involvement will better prepare undergraduate students for academia.

We suggest seeking topics and subjects for SRP with enough time in advance to a course start. This contributes to better planning of resources and content to be taught. On the other hand, it leaves necessary time for the conception, design, and implementation of a software tool (if this is the case in the SRP). When giving a subject of SRP, we also recommend having prepared materials and literature about the topic, past documents, or developments when needed, as well as useful contact information to give to the students by the first meeting.

We also recommend, if it is a case of further development for a project, concerting meetings with former developers and teaching assistants in order to exchange ideas in new discussions, as source of inspiration for further work. In such meetings, former students could also inform widely about the state of the art of their work. With this respect, we suggest documenting and registering essential data like date, duration, main topics, and to-do list, for instance, for generating interesting statistics about it. This will be a good estimator to better plan both resources and time to dedicate to SRP and to supervision in general. In particular, Monett invested 2011 almost 6 hours per student in the first SRP, in average. In the first SRP subject of this paper, a total of 8 hours and 40 min. were invested in the whole supervision, including discussions, written report revision, and preparation of contents, to name a few.

Combining research and teaching together in student projects gives undergraduate students the opportunity to be coauthors of future research papers and, even better, to be oral presenters at international events. This was already the case after the 2009/10 edition of our AI course. Such an experience as in [8] was the first of its type at the DCS. Furthermore, it received very positive feedback, also from the AQAS e.V., a German agency for the accreditation of study programs. This paper is the second educator-student attempt in this sense, where we also reflect on the aspects that should characterize a successful combination of undergraduate student research, course work, and teaching.

We should mention some initial fear and resistance to accomplish these tasks. In other words, writing "real" papers or parts of them and presenting research results at international conferences are often rejected, because the benefits of scientific contributions for the future professional life are not well known to undergraduate students yet. However, they turn in a positive stimulus when good supervision, constant support, and co-work are offered to the students.

This is why we suggest not including this higher phase as part of an undergraduate course evaluation, but to give an incentive, from the first discussion and assignment of the research topic on, about the possibilities, conditions, and advantages of a scientific publication on the field. To our opinion, even when a solid research is available and mature for publication, approaching deadlines often gives stress to the students, which could negatively influence their scores in other course topics or even in other parallel courses. Thus, sending contributions to conferences should take place after the course finishes, when possible. For a successful project continuation it is very important to maintain the contact with the students in case they are not part of the department staff. Any kind of further collaboration with them would be as positive as effective if communication, interest, and support exist. The topics and content are of course of utter importance as well.

VII. Conclusions

Bringing together student research projects and teaching gives decisive, explicit input to outstanding teaching in Computer Science. We presented, so far, our experiences on these topics at the Computer Science Division, Department of Cooperative Studies, at the BSEL. We focused on a particular software application, Remo, for modeling rule-based knowledge and for reasoning about it, subject of SRP. We successfully used Remo in an AI course for teaching inference in expert systems. It was challenging for us to involve undergraduate students, authors of SRP and partially training at business enterprises, in teaching.

When a SRP is combined with teaching activities, it is especially attractive for the following reasons: (1) it supports the academic staff, not only in research interests, but also in their lectures, (2) it incorporates and enhances students' soft skills like the ability to teach, (3) it prepares undergraduate students for academia and further steps in this field, (4) it stimulates research essentially focused in current applied topics, and (5) it encourages early interest in publishing research results, amongst many other benefits. We could verify all of them in our concrete settings and hope several recommendations for concrete actions are useful to others.

Further work will be devoted to the application of other co-teaching models and approaches as well as to their evaluation in the classroom. For this, and for the further development of Remo, we already have a new student working in a successive SRP who will enhance and complete both Remo and its documentation, already started online at http://code.google.com/p/hwr-remo/. We also plan to extend our experiences to other curricula modules, as positively perceived and signalized by the students in the survey.

Acknowledgment

We thank Alexander Müller and Markus Beckmann. They programmed in 2010 a first software prototype, predecessor of Remo.

References

[1] -, Bachelor of Science in Computer Science. BSEL's flyer (in German), September 2011. Available online at http://www.hwr-berlin.de/fileadmin/downloads_internet/studiengaenge/flyer_fb2/BRO_FB2_Informatik.pdf.

[2] Ch. Beierle and G. Kern-Isberner, Methods of knowledge-based systems: Foundations, algorithms, applications (Methoden wissensbasierter Systeme: Grundlagen, Algorithmen, Anwendungen), 4th ed., Vieweg+Teubner Verlag, 2008.

[3] D. Berrett, "Want to Be a Good Researcher? Try Teaching," The Chronicle of Higher Education, August 18, 2011. Available online at http://chronicle.com/article/Want-to-Be-a-Good-Researcher-/128753/.

[4] A. Bolour, "Notes on the Eclipse Plug-in Architecture," Eclipse corner article, Bolour Computing, July 2003. Available online at http://www.eclipse.org/articles/Article-Plug-in-architecture/plugin_architecture.html.

[5] M. Friend and L. Cook, Interactions: Collaboration Skills for School Professionals, 6th Edition, Pearson, 2010.

[6] A. Hunter, Graphical Modeling Project. The Eclipse Foundation, 2012. Available online at http://www.eclipse.org/modeling/gmp/.

[7] E. Merks, Eclipse Modeling Framework. The Eclipse Foundation, 2012. Available online at http://www.eclipse.org/modeling/emf/.

[8] D. Monett, R. Janisch, and S. Starroske, "NL-Analyzer: Enhancing Simulation Tools to Assist Multiagent Systems' Teaching," in Proceedings of the Workshop Multi-Agent Systems for Education and Interactive Entertainment (MASEIE), 9th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2010, Toronto, Canada, May 10-14, 2010.

[9] O. Raia, "Co-Teaching," Keys to inclusion, vol. 3, issue 2, NJCIE, East Brunswick, NJ, May 2010. Online at http://njcie.net/pdf/resources/co-teaching.pdf.

[10] M. Sänger, "Inference in Expert Systems: Further development of an existing expert system," 1st student research project, Computer Science Division, DCS, BSEL, 2011, unpublished.

[11] M. Sänger, "Inference in Expert Systems: Further development of an existing expert system – II," 2nd student research project, Computer Science Division, DCS, BSEL, 2012, unpublished.

[12] D. Steinberg, F. Budinsky, M. Paternostro, and E. Merks, EMF: Eclipse Modeling Framework, 2nd Edition, Addison-Wesley Professional, 2008.

[13] J. Wielemaker, SWI-Prolog 6.0: Reference Manual, University of Amsterdam, The Netherland, 2012. Available online at http://www.swi-prolog.org/.