Computer Supported Collaborative Learning in Software Engineering

(ETC - Enforcing Team Cooperation)

Mauro Coccoli and Lidia Stanganelli
Dept. of Communication, Computer, and Systems Science
University of Genoa
Genova, Italy
{mauro.coccoli, lidia.stanganelli}@unige.it

Paolo Maresca, IEEE senior Member
Dept. of Computer and Systems Engineering
University of Naples, Federico II
Napoli, Italy
paolo.maresca@unina.it

Abstract— **Collaboration is one of the keywords in education as well as in computer-assisted instruction. In this respect, e-learning platforms provide users with specific tools, enabling them to collaborate and/or to cooperate so to reach common objectives. Collaboration is considered as a teaching strategy but, in many cases such as in the software engineering classes, collaboration has to be a learning outcome itself, since students must acquire a specific ability in team working. Thus a suited working environment is needed, that has to be much more than just a flexible Learning Management System. Consequently, a specific project has been launched within the Italian Eclipse Community in the framework of the Enforcing Team Cooperation (ETC) activity. The aim of the project is that of enforcing and enlarging cooperation activities among a large number of students, all attending software engineering courses at different Universities in Italy. The main idea behind the project is the implementation of a really effective Computer Supported Collaborative Learning (CSCL) paradigm, to be used for higher education on team cooperation, in software engineering classes for the analysis, design, and development of software programs along their lifecycle.**

Keywords-component; Team Cooperation, E-learning, Knowledge Management, Software Engineering, Eclipse-Jazz.

I. INTRODUCTION

A concrete experimental activity, conducted in suited laboratories duly equipped with related facilities, is a core component for the education in engineering and, generally speaking, in all of the scientific disciplines. Experimentation enables students to learn through both simulations and hands-on experiences. This is the basis for practical educational activities as well as for training personnel at their workplace. In this paper, the focus is on software engineering education, where the laboratory equipment is not made of machineries; it is all about computers, networks, and software systems. In particular, a collaborative learning environment will be considered as an ecosystem for the software engineering education. Moreover, within the introduced ecosystem, the term collaboration has multiple meanings. It refers to: *i)* collaboration among students, *ii)* collaboration among Universities, and *iii)* collaboration between industry and University.

Today's software engineering students, which are going to be the tomorrow's professionals, have to acquire and maintain a variety of skills, aiming to become proficient in the design and development of software applications, tools, and services. As an example, let us consider programming languages, software architecture, design patterns, operating systems, compilers, interpreters, graphical user interfaces, data base management systems, rapid application development environments, libraries, drivers, tools for analysis and design, modeling languages, etc.; a sub-set of them, or all, has to be studied so to acquire ability in fully exploiting their potential. Moreover, to merge together some of them, an Integrated Development Environment (IDE) is needed, and the IDE has to be studied too. In order to do that, a continuous update is necessary, to keep on the edge with novel technology, trends, and paradigms, and it may be not enough to be a good software engineer. In fact, such competences have to be exploited in a workgroup and the project is driven by collaboration and interactions. Professional software developers give just small contributions to the construction of huge products, and they become part of a specific workgroup (identified by an assigned task), which has to interoperate with other groups devoted to other connected tasks. Nevertheless, team cooperation is not considered as a core competence in software courses, while developing such skills in students at the University level is strongly needed.

All of these considerations activate new remarks on how interaction between Universities and industries happens, both in the field of education and in the field of the work experience, during the individuals' educational processes. Nowadays, corporate and industries are involved with universities in the educational and training activity by means of the internship. In fact, every student has to spend a short period at a workplace, supervised by an academic tutor. Moreover, the internship is the last part of the education process and it gets hard to exploit the results achieved during that period so that one can compare this model to a "closed innovation network". On the other hand, one can observe what happens in the open-source developers community in which each member gives his/her own contribution on the basis of a cooperative and self-organizing model. The participants to a project do not know each other, but every one knows what the others can do for the

978-1-61284-643-9/11/$26.00 ©2011 IEEE April 4 - 6, 2010, Amman, Jordan
2011 IEEE Global Engineering Education Conference (EDUCON) – "Learning Environments and Ecosystems in Engineering Education"

whole community, in terms of competence, skill, and ability. Moreover, collaboration and cooperation happen on the basis of asynchronous and distance communication. Common interests, regardless of the geographic position of its members, drive the community. The same model can be adopted at the University too, so to improve a winning behavior of knowledge sharing and flow among peers. A "networked University" will act as an "open innovation network", in which ideas are shared among groups of teachers and students of different schools, with common projects and scientific interest. Given this paradigm, a technological support is needed to implement such a collaborative learning model, so to enable the aggregation of small groups in one distributed team.

The remainder of the paper is the following: after this general introduction, the work refers to an experience carried on in Italy; hence a short overview of the actual Italian situation will be presented in Section II. In Section III the Collaborative Learning paradigm is stressed, with specific reference to software engineering. Section IV describes the ETC Project. Results and future development are reported in Section V as well as final remarks and conclusions.

II. University and Industry Collaboration

At present, most traditional models people are using are not any more feasible. This is very clear for economy and it is true for scientific education and technology too. Universities, companies, industries and public administrations are called to a strict collaboration to cope with the evolution of markets and products. To this aim, the cooperation among industry and university to the education of future workers appears a crucial objective. Higher education in Europe is characterized by three-years courses followed by two-years specialization courses (master degree) and, many times, the specialization field may be very narrow, so to give a fast response to the needs of both market and production. Besides, given the flexibility of modern production systems, industries too have the possibility of rapidly changing the way products are made and distributed, so to fully exploit the results of academic research. In this scenario of rapid changes, pushed both by social needs and by technological progress, education should be as flexible as production. In the specific field of software engineering, innovation is based on immaterial things and may be sudden. Let us consider the programming languages as an example. They are continuously evolving, pushed by the trends of developer communities or according to specific needs in specific application fields. The syllabus of a course should be able to react to changes and to adapt its content and structure year after year, making knowledge dynamic and always up-to-date. To this aim, the cooperation among university, industry, and public institutions is mandatory. Moreover, in such a participated model, students and teachers can keep in touch with the real world and the work market. In this respect, students are called to have a "T-shaped" education [1]: a specialty, the leg of the "T" is complementary to a more general knowledge, including interdisciplinary competences. Besides, through collaboration with industries, the academic researchers can improve their results making them market-ready. University and industry, the pure academic research and the real world experiences, influence each other with the result of enhancing the professionals, according to the principle of adaptability to the environment, to the contextual conditions, or the market needs. Based on the theory of evolutionism, this is the winning strategy in the survival of the fittest.

Focusing on the Italian situation, one can observe a fragmented scenario in which there are 94 Universities and a relatively small number of students for each. In most cases, different universities offer similar curricula, especially for the mainstream courses. This results in many classes with few students. Therefore it can be hard to plan students' activities in workgroups and even harder to make different workgroups interact. Moreover, let us consider the fact that groups are made of homogeneous individuals, while the cooperation to reach the common objective may happen among heterogeneous ones. An orchestrator is needed for the activities to be scheduled and the information flow among the different groups with their specific responsibilities in the project (Fig. 1).

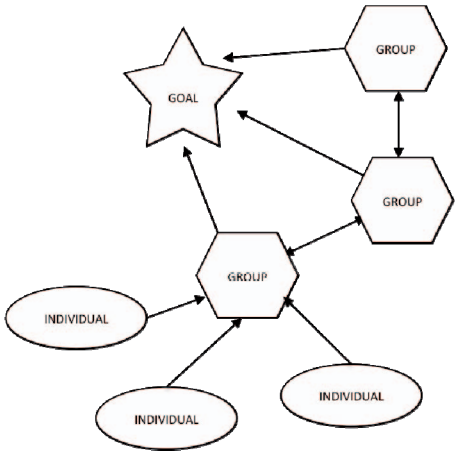

Figure 1: collaboration among individuals and groups, to reach a common objective

In this paper, a specific educational experience is taken into account as a possible model for closer interactions among industry, Universities and academic research. Such an experience was led by the Dept. of Computer, Communication and System Sciences, University of Genoa together with the Dept. of Computer and Systems Engineering, University Federico II Naples, from the University side, and IBM from the company side, thanks to its IBM Academic Initiative. Therefore, the proposed model can be replicated in other situations. The proposed model is called ETC, which stands for Enforcing Team Cooperation and it represents a concrete experience of collaborative work and collaborative learning, based on the joint usage of the open-source Eclipse framework integrated with the Jazz platform.

III. Collaborative Learning

A. Selecting a Collaborative Paradigm: Motivation

Generally speaking, the collaboration among people can be an essential activity for the learning [2]; this is true especially for teachers and students that want to achieve good results in software engineering. During the lessons that include hands-on activity to be carried on in a laboratory, students may

experience some difficulties in applying the critical concepts learned (*to know*) in order to build artifacts (*to be able to do*). More criticisms arise when the job has to be done by a group in which the members of a team have to transfer the knowledge they acquired to other people (*to enable to do*). When dealing with a network of groups (*knowledge network*) in which every team has acquired some specific skills, a dissemination activity has to be done, that is this to transfer such knowledge to others (*to inform outside*). In this scenario, a specific paradigm has to be defined; it must be able to represent these interaction levels in a virtual laboratory and, moreover, it must keep into account the skills requested by industries interested to participate to the cultural and professional growth of students, so to make them suited to the market needs and, more important, ready to work. Many universities should participate to the construction of the above network of knowledge. The same faculties (i.e., engineering) can put together people in large and spread workgroups, and share contents and materials. Interaction among different faculties (i.e., engineering and computer science or education science) may have the effect of bridging the gaps existing between people who have studied the same disciplines in different schools, thus with different aims and points of view. A distributed virtual laboratory will be the core of the activity.

This is the target of the ETC project: the implementation of a virtual laboratory in which the Computer Supported Collaborative Learning (CSCL) model is applied to software engineering and gives support to knowledge engineering. In smarter planet university jam [3] was asked a question: *«What interdisciplinary skills are required for students to compete in an increasingly interconnected, intelligent and instrumented smarter planet?»*. This question raises some useful remarks on the collaborative paradigm that can be chosen to achieve this objective. More and more, students are expected to have a T-shaped competence profile, when approaching their first work experience. The base of the "T" involves a breadth of knowledge in a number of topics (e.g., personal relationships, project management, leadership, etc.) while simultaneously having a depth of knowledge in a specific discipline. In this respect, the team-based project is candidate to be the best-suited interdisciplinary reference models for an effective T-shaped education. This is even more important if one considers that the world is moving toward an economy based on services, where the network of relationships is open and is also the mean knowledge flows through, in order to make innovation. Each person should better collate own technical skills (*to know*) and the relevant practice (*to be able to do*), with creativity and innovation perspective, in an open environment of relations without boundaries due to geographic matters, language, and belonging (e.g., university, companies, public administration, citizens, etc.). The meaning of collaboration is also to merge experiences coming from both university and industries, so to make ideas freely flow, in order to create open innovation networks.

B. Computer Supported Collaborative Learning

The Adoption of a collaborative learning paradigm can be useful in many situations, and it can be realized with or without the use of technology. In a student-centered model, learning is also achieved through the interactions that happen among peers, while cooperating to cope with any given problem. This is considered a winning strategy due the fact that, according to Silverman [4], *«to perform competently in many professions requires one to keep abreast of new knowledge, learn constantly, interact with other specialists, and group problem solve. That is, jobs in the information age require learning to learn, and life-long collaborative learning and problem solving skills»*. Many different pedagogical models exist to make effective learning through cooperation [5]. Such models assign to teachers and students their respective roles in the learning process, as well as they describe how the cooperation among students should happen [6]. A number of these models, based on cooperation, can be exploited both in the "cooperating to compete" paradigm and in the "cooperating to cooperate" paradigm. Moreover, in the case in which the collaboration among the people involved in the learning process is favored by the computers, collaborative learning is called Computer Supported Collaborative Learning. In this respect, specific tools and communications systems exist for the CSCL to be made effective. It appears obvious that this approach has both weak points and strengths [7], mainly connected to the responsibility that, compared to a traditional model, is shifted from the lecturers to the students themselves, which take a more active role in their own learning process. Basic principles of CSCL are fully in concordance with the fact that one's knowledge is created also based on other's experience [8] as well with the concept of distributed constructive learning that explains how knowledge can derive from doing, rather then receiving [9].

Summarizing, CSCL is a powerful didactic strategy for both the learners and the teachers, when communicating and sharing are considered added value and where the social aspects go beyond the learning activity itself. By means of the Information and Communication Technology, communication, collaboration, and thus the learning experience, can be greatly enhanced so to achieve better results in education.

C. CSCL in Software Engineering

Traditional CSCL is based on the use of a set of standard collaboration tools integrated in the learning process or in the Learning Management Systems. Such tools include synchronous and/or asynchronous communication systems, discussion boards as well as forum, blog, wiki, and social networking, resource sharing systems such as teleconference services and shared whiteboards. A slightly different approach is needed when students are supposed to work in a collaborative environment, in which the collaboration is part of the task, not just a part of the learning process. That is the case of software engineering and other disciplines, in which code development is a creative, social, and collaborative process and teams of developers have to work together to design, solve problems, and produce code as well. Team cooperation is at the basis of the development of any software product [10]. Moreover, recent trends are driving software engineering towards the use of non-traditional development methodology (e.g., agile programming, eXtreme programming) [11] where computer-based collaborative tools are needed to support coordination of work as well as communications, both

structured or unstructured, and synchronous or asynchronous. In complex projects, as well as in simpler ones, different persons have to interact in a group (e.g. analysis, design, development, graphics, documentation, user interface, accessibility, interoperability, APIs) and the outcomes of independent groups have to be integrated so that they can work together in just one product. This can be achieved by applying the unified software development process [12] and the Unified Modeling Language (UML) [13] so that different groups, at different levels of detail, can use a common language to give each other instructions as well as to exchange information.

At engineering Faculties, collaboration has to be considered in terms of the construction of collaborative parts in a collaborative environment that can realize the spiral model adopted for software development [14]. Software engineering and the use of the CASE (Computer Aided Software Engineering) model, make available specific tools (e.g., common versioning systems, source control, bug tracking systems,) and specific development environments offering advanced collaboration functionalities. The process is meant to be the collaborative construction of one "big thing" for which objects are developed in a collaborative way.

D. Communication and Collaboration

A good communication is the key-point in collaboration and in cooperation. For software engineers the development platform is expected to be the enabler for good communication to take place at different levels: developers working on the same application; practitioners across the life cycle; business analysts; architects; developers; testers; project managers. Collaboration in a workgroup is made possible by means of technology enabling the people in that workgroup to a seamless work, as if they were co-located, in order to: improve communications; create a common context; share information; notify events.

IV. Enforcing Team Cooperation Project

A. Eclipse with Jazz as a Learning Environment

Eclipse is an open and extensible Integrated Development Environment, as well as an open source community. The aim of Eclipse developers, and hence of the Eclipse-based tools, is giving developers the freedom of choice in a multi-language, multi-platform, multi-vendor environment supported by multiple vendors. The operating system platforms Eclipse have been targeted to include MS-Windows, Linux, Solaris, HP-UX, AIX, and MacOS. In addition, Eclipse provides a unique environment for members of the academic community to build new tools for teaching, research, and further growth of the Eclipse community [15]. Collaboration is an integral part of software development, occurring through tools inside and outside the IDE. The Jazz project, which seeks to integrate collaborative capabilities into the Eclipse IDE, enables small teams of software developers to work together more productively. A very clear image of the Jazz philosophy is that *«Jazz is based on the metaphor of an "open office" approach to application development. (...) Within the office, each developer will have his or her own workstation, while elsewhere in the office will be space for team meetings, shared whiteboards, schedule information, etc. A key aspect of this setting is team awareness. Even while focusing on their personal work, all team members have a sense of what is going on elsewhere in the room, who is talking to whom, what others are working on, etc. Communication among team members is also facilitated in this environment, whether in the form of a question shouted out to the team in general, or in calling a colleague over to consult on an issue at a particular workstation»* [16].

B. The ETC Project

The ETC (Enforcing Team Cooperation) Project started in March 2010. It can be described as a collaborative learning experimentation among different Italian universities. This experimentation has been conducted in collaboration with IBM Rational, the IBM Academic Initiative, the Italian Eclipse Community, and seven Italian Universities (i.e., Genova, Milano Bicocca, Bergamo, Bologna, Napoli Federico II, Salerno, and Bari). The aim of the project is to develop reliable software products, using Rational tools, enabling collaboration among all the participants: students, teachers, researchers, and professionals as well. All the students involved are attending the first level courses for Software Engineering, Database, Computer Science Fundamentals, and Web Design. The aim of the activity is to test their ability of conducting a team-working activity, by using the Rational tools, which enable collaboration among all project participants: students, teachers and professionals. In support to the ETC Project, IBM has provided both the hardware and the software architecture. The hardware architecture is built around a X3650M2 IBM server (Fig. 2), which allows a number of concurrent accesses between 150 and 200. The software architecture (Fig. 3) is based on the Jazz platform allowing collaboration among multiple teams [17]. It is an Eclipse application composed by three tools: *i)* the Rational Team Concert (RTC), *ii)* the Rational Requirement Composer (RRC), and *iii)* the Rational Quality manager (RQM). The three tools assist different teams to co-operate in developing software specification respecting quality constraints. More in details: RTC allows collaboration between teams: it is an integrated development environment for the creation and distribution of resources. RRC enables teams to define requirements and manage the entire lifecycle of software. The stakeholders in requirements process can use textual notations, visual and intuitive in a collaborative online

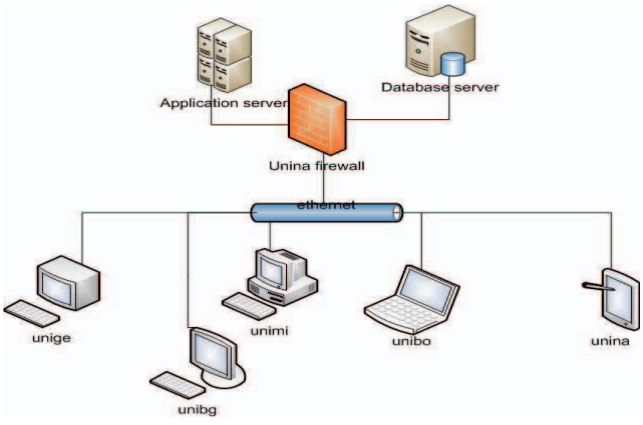

Figure 2: hardware architecture

environment. RQM is a test management environment that improves the efficiency and quality of software and the release phases through test planning, workflow control, tracking and traceability, and reporting metrics. The RTC, RRC, and RQM enable project teams to shorten planning cycles and reducing the work, by aligning the development and test with requirements and objectives. In addition, two other architectures are implemented: the peopleware-toolware and the knowledge-sharing. The first implements the combination of courses and tool sets, also assigning roles to the actors (7 universities, 483 students, 8 champion students – typically PhDs or PhD students, 1 system administrator and 15 professors and/or researchers). The second one includes a team of undergraduate students creating plug-ins targeted to providing documentation and innovative solutions, so to improve the project itself, and the exchange of knowledge among students.

The experimentation in collaborative learning, which is still running, has made possible to implement both *intra*-University and *inter*-University projects. Among these, it is worth noticing the work carried on within the OTRE Project [18], developed thanks to the joint effort of the Universities of Genoa and Naples Federico II. The project has joined "communication science" and "engineering" courses in the experience of developing web sites and services in the collaborative environment provided by the ETC.

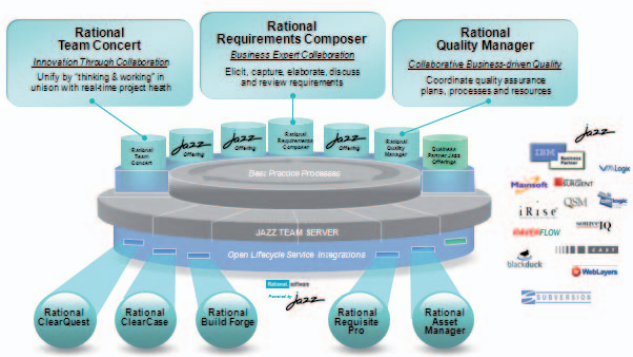

Figure 3: software architecture

C. Tool Installation and Projects Organization

The RTC, RRC, and RQM can be installed and used in a variety of configurations. For example, there is a thin client RTC, which can be used in any web browser. Alternatively, a RTC rich client can be downloaded from the Jazz web site and installed on your computer. A third version of RTC is a plug-in to be installed in an Eclipse platform. RRC is released with both thin and rich client. RQM comes only with the thin client. The tools are platform independent and they are made available for Windows, Linux, and MacOS. The project started with the use of the RTC, to enable the implementation of projects and collaboration. Each project is assigned a name, a team, the beginning and expiration date (timeline) and the process model. In the present situation, each team is composed by 1 professor, 1 champion student, and a group of students (4 to 6). For each team member a process role is assigned: the analyst, the designer, and so on. The process model may be common, openUp or scrum. Each project can be assigned one or more Work Item (WI), each of which will have an owner who is a member of the team. Priorities are assigned to each WI and it may be associated to one or more subscribers. All the information and all the changes relevant any WI are communicated to all the team members associated with the WI itself. Communications among the participants happen inside the RTC. RTC allows automated mailing to the participants of any project, whenever a change occurs, keeping track of all activities. It also has a board for discussion and this is an alternative way to communicate.

Some final remarks are on the obstacles encountered by young students at the first approach with such tools. The Jazz environment is designed for complex projects and a huge didactic support has been necessary, as well as the creation of tutorial and hands-on manuals. To overcome these start-up difficulties, the first project in RTC has been devoted to self-teaching; it was called "Prelude" and it involved all 483 students from all the participating Universities. After this short start-up period, a team of undergraduate students engaged the realization of a plug-in in order to create facilities useful for all the stakeholders. It is a team who has already succeeded in one Eclipse plug-in called "IngSoftwarePlugIn" that collects, ranks and maintains all existing documentation on Rational tools. Besides, an activity has started, with the aim of creating an Academy-oriented light-RTC, slightly simplified and more user-friendly. The objective is to decrease the start-up time and to make smooth the learning curve for new experiments that will take place in the forthcoming years, with new students and, possibly, with an increased number of Universities, that will exploit the past experience.

D. The methodology of the experience

The experiment has been conducted in two phases, the first used to train student to the use of the RTC platform in programming laboratory lesson. In this lab, students have developed code and managed their workspace independently; moreover they have learned concepts such as versioning of their elaborate (e.g., check-out, commit, etc.). Each student submitted 23 homeworks in 4 different iterations, each given every 15 days starting on Dec. 5, 2010 and until Jan. 21, 2011. In the second phase a project was assigned, the students were divided in groups and each group was assigned a portion of the project. Then they all started to define the specifications and to develop cooperatively.

E. Project development

The project is relevant the system software analysis, design, and development of a conference management system. The involved groups are 9, the students are 40, each group consisting of 3-4 students; every student has a specific role in the project (e.g., analyst, designer, developer and so on) indeed we used Open Unified Process as project organization framework. Figure 4 shows all the projects assigned to the students. Moreover, the students were provided with a draft document of specification, in order to better understand the project, to produce more detailed specifications, and to provide innovative ideas on the specifications themselves. Then, a group of students was responsible for the formalization of the detailed specifications, made by using a tool of ETC project

(Rational Requirement Composer and Rational Requisite Pro). Such specifications were discussed and approved in order to address the forthcoming development phase: the software architectural design. The next step (still a work in progress at the time of the writing), is the abstract class architecture development.

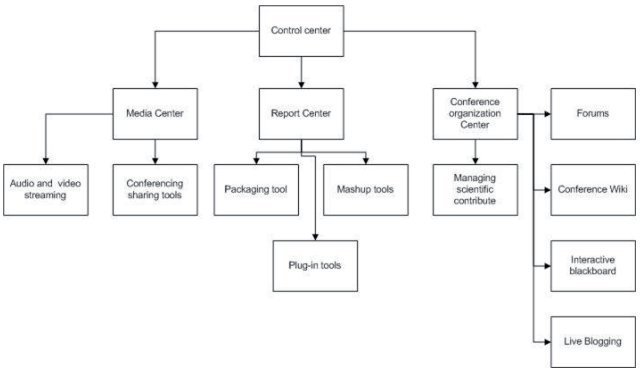

Figure 4: The OTRE software architecture

F. Participant profiles and experimental results

Students are enrolled in the second year of a degree in computer engineering. The final results are still being collected. At present, the only module that has been fully developed is a "single-sign in" propaedeutical to the construction of all the other modules. From a teacher's point of view, the results were above expectations and, more important, the Project has put together Universities located in different places on the basis of a common educational objective. There was the possibility of agreeing common experiments and laboratories for students from different Universities, under the same curriculum. The experimentation carried on during this first year has been interesting for all the stakeholders. Students, teachers, and researchers have had the opportunity to work together, to implement joint projects. About 500 students were trained in the virtual laboratory and nearly a dozen professors from several Universities have collaborated in this activity. In the forthcoming year, the ETC project will be revamped with a new experience involving more Universities and therefore a greater number of people. Next release of the ETC will also exploit new tools such as the RAM (Rational Asset Manager) and the RRP (Rational Requirement Pro).

V. CONCLUSIONS AND FUTURE DEVELOPMENT

Final results from the activity will be soon available and will give a measure of the effectiveness of the chosen approach. Besides, a more general framework for the development of team-working skills is presented, based on the Eclipse and Jazz framework and tools. Such a test-bed can be considered as a learning tool itself and it can be re-used in many different applications, even different from the software engineering we have been referring to along the whole paper. Especially in Public Institutions and large Organizations, cooperating to reach a common objective can be a critical point to achieve results and meet the milestones. The availability of a framework enabling users to cooperate through advanced software systems can greatly empower such scenarios and interesting perspectives can be considered in many other fields. ETC experimentation has had positive results; students learned how to make good software working together. The collaboration has reduced the learning curve and startup times thanks to the teaching aids available on the ETC platform, the problems were resolved quickly through cooperative environment that has facilitated the interaction among the parties. Also, some difficulties in the installation and in the set up of various complex applications have been overtaken. In future experimentations, learning time will be faster thanks to the work already done by the pioneers of the experimentation reported in the previous paragraphs.

ACKNOWLEDGMENTS

The authors wish to thank IBM and the Italian ECLIPSE Community.

REFERENCES

[1] C. Ortiz, "Toward a Personalized Graduate Curriculum," MIT Faculty Newsletter, vol. XXII, no. 3, 2010.

[2] P. Dillenbourg, Collaborative Learning: Cognitive and Computational Approaches. Advances in Learning and Instruction Series, Elsevier Science, NY, 1999.

[3] IBM, Smarter planet Jam, available at the address: http://www.ibm.com/developerworks/university/smartplanet_jam/index.html (Accessed Jan. 2011)

[4] B.G. Silverman, "Computer Supported Collaborative Learning," *Computers Education,* vol. 25, no. 3, pp. 81-91, 1995.

[5] R.E. Slavin, *Cooperative Learning: Theory, Research, and Practice.* Englewood Cliffs, NJ: Prentice Hall, 1989.

[6] M. Hamm and D. Adams, *The Collaborative Dimensions of Learning.* Norwood, NJ: Ablex, 1992.

[7] S. Wiliams and T.S. Roberts, "Computer Supported Collaborative Learning: Strengths and Weaknesses," *Proc. Int. on Computers in Education*, pp. 328-331, Dec. 2002.

[8] A. Bandura, *Social Learning Theory.* Englewood Cliffs, NJ: Prentice Hall, 1997.

[9] M. Resnick, "Distributed Construction," *Proc. Int. Conf. on the Learning Science*, 1997.

[10] G. Booch and A. Brown, "Collaborative Development Environments," *Advances in Computers,* vol. 59, Academic Press, Aug. 2003.

[11] C. Ghezzi, M. Jazayeri, and D. Mandrioli, *Fundamentals of Sotware Engineering.* Englewood Cliffs, NJ: Prentice Hall, 2002.

[12] I. Jacobson, G. Booch, and J. Rumbaugh, "The Unified Process," *IEEE Software*, vol. 16, no. 3, pp. 96-102, May/June 1999.

[13] OMG - Object Management Group, "The Unified Modeling Language," http://www.uml.org, 2010.

[14] B. Boehm, "A Spiral Model of Software Development and Enhancement," *Software Engineering Notes,* vol. 11, no. 14, pp. 14-24, ACM, New York, 1986.

[15] IBM, "Eclipse," http://www.research.ibm.com/eclipse. (Accessed Jan. 2011)

[16] L. Cheng, S. Hupfer, S. Ross, and J. Patterson, "Jazzing up Eclise with Collaborative tools," *Proc. of the 2003 OOPSLA Workshop on Eclipse Technology Exchange*, pp. 45-49, Anaheim, CA, 2003.

[17] R. Frost, "Jazz and the Eclipse way of Collaboration," *IEEE Software*, vol. 24, no. 6, pp. 114-117, Nov./Dec. 2007.

[18] M. Coccoli, P. Maresca, and L. Stanganelli, "Enforcing Team Cooperation: an example of Computer Supported Collaborative Learning in Software Engineering", Proc. of the 16th Int. Conf. on Distributed Multimedia Systems, pp. 189-192, Chicago, IL, U.S.A., Oct. 14-16, 2010.