Applications of a Collaborative Open Object Oriented Learning Environment to Electronics, Computing and Mathematics Education

Colin B. Price
University College Worcester
Henwick Grove, Worcester, United Kingdom, WR2 6AJ
Tel 44-1905-855316, e-mail: c.price@worc.ac.uk

Abstract - A lightweight Java application suite has been developed and deployed allowing collaborative learning between students and tutors at remote locations. Students can engage in group activities online and also collaborate with tutors. A generic Java framework has been developed and applied to electronics, computing and mathematics education. The applications are respectively: A digital circuit simulator which allows students to collaborate in building simple or complex electronic circuits, a Java programming environment where the paradigm is behavioural-based robotics, and a differential equation solver useful in modelling of any complex and non-linear dynamic system. Each student sees a common shared window on which may be added text or graphical objects and which can then be shared online. A built-in chat room supports collaborative dialogue. Students can work either in collaborative groups or else in teams as directed by the tutor. This paper summarises the technical architecture of the system as well as the pedagogical implications of the suite. A report of student evaluation is also presented distilled from use over a period of twelve months. We intend this suite to facilitate learning between groups at one or many institutions and to facilitate international collaboration. We also intend to use the suite as a tool to research the establishment and behaviour of collaborative learning groups. We shall make our software freely available to interested researchers.

Index Terms – Open and Distance Learning, Simulation and Visualization, Programming in Java.

INTRODUCTION

The Web has clearly emerged as a ubiquitous medium for the delivery of information and learning in both the public and private sectors. The ease of use and uniformity of this platform enables all modern citizens access to services they require. This is especially true in education where students now expect both course content and management to have a significant Web-based component. Research has demonstrated that the Web is an effective learning medium, with student outcomes at least equivalent to those of traditional classroom-based students [8, 9, 13]. Also, as we enter an era where the Web is evolving from a medium dedicated to content display to one which is endowed with meaning [3] this new Semantic-Web will offer new opportunities to researchers and practitioners in education. At university College Worcester (UCW) we recognize one important use of this technology in Open and Distance Learning (ODL). There is much to offer here: Students dispersed over a wide geographical area can register to study in a virtual community which obviates the need for expensive travel and accommodation on a physical campus. But the use of ODL raises important pedagogical issues. Learning as a distributed community allows us to refer to the social component of learning as described by Vygotsky [18] as we consider how students can work together to create new knowledge collaboratively. Theoretical grounding is found in the notions of Lipman’s “communities of inquiry” [12] and Wenger’s “communities of practice” [19]. This paper reports on a long-term project to provide students with collaborative learning environments, and suggests that a content-driven approach (rather than a technology-driven) approach is useful in developing ODL systems. We have developed a flexible Java-based approach which can be deployed within a wide range of subject areas, and in this paper report on the development and student evaluation within three particular areas of learning: First the study of dynamic systems via the solution of Ordinary Differential Equations (ODEs) applicable to a wide range of disciplines from science to economics. Second, the learning of Digital Electronics, of use to many electronic engineering and computer science courses. Third, the learning and teaching of computer programming which has been the subject of much reflection, even heartache, and remains an important issue within computer science education. In the following sections of this paper we summarize contemporary thinking on ODL relevant to our project, outline the aims and objectives of our research, then discuss the implementation and student evaluation of these three areas. At UCW we have a tradition of working with small groups of students, and to engage students in group activities as well as teamwork based learning, which we have consistently found to be beneficial to both students and faculty. This motivated our attempts to research and develop applications which allow us to both deliver (and evaluate) collaborative group work.
**Approaches to Open and Distance Learning**

We take as our central pedagogical principle the Constructivist paradigm of learning, and accept that discussions on collaborative learning should start from this base [10,14]. Collaboration gives learners real-life experience of working in a group, learning from others and contributing their own understanding. Students are also encouraged to accept the responsibility for their contributions to the class learning process, which should lead to improved personal learning. Collaborative learning may be defined as "... a situation in which people learn or attempt to learn something together" [5].

There is a plethora of tools currently available to support electronic collaboration and a number of reports on these tools have appeared in the literature. One example is the Athabasca University Centre for Distance Education report which presents the results of an analysis if over 100 collaborative tools [1]. Trends identified in this analysis are the use of text-conferencing, audio-conferencing, video conferencing, the use of whiteboards, polling tools and entire course delivery tools. Within Europe the "Innovative Technology for Collaborative Learning and Technology Building" (ITCOLE) project was aimed to develop pedagogical models of collaborative knowledge building for European education. It also aimed to produce a modular knowledge-building environment to support collaborative learning [11]. The focus was clearly on the pedagogy, not the technology. Aimed at both primary and secondary schools, this initiative was designed to develop students' abilities to adopt, cultivate, create new and share knowledge with others. The guiding premise is that knowledge is not absolute nor static rather it is shared within social organizations, and that learning occurs within such communities. It proposed a move away from existing e-Learning environments which were designed to manage study materials, the students themselves, assessment, grading and basic cooperation, towards encouraging engagement in active learning and knowledge creation. The ITCOLE project has developed a number of significant tools. "Synergeia" is an extension of BSCW which needs no further comment, "Fle3" is a VLE which provides the usual access to shared resources, a structured "Knowledge Building" tool which is an interactive prompting database, and a "Jamming" tool which is a shared space for the construction of digital artefacts (video, pictures, text, etc.) [11]. The collection of tools is highly structured and contains many management facilities. At UCW we also have an established tradition of science education, especially for secondary school (K-12) teachers. Here we make a specific (perhaps even radical hypothesis): Science education may be viewed as an important and unique target for ODL since the natural process of scientific enquiry involves collaboration, peer review as well a experimentation and simulation, which complement the activities of theoreticians. A huge number of demonstration and simulation applets has appeared in recent years together with toolkits for building these applets. Pupils in secondary education (e.g. US K-12) whose teaching was traditionally based on acquisition of a corpus of scientific facts are now, via these simulations, able to participate directly and truthfully in the real scientific method, engaging in the interplay between theory, experimentation and simulation. The "CoVis" project at Northwestern University is a good example in science education where there are a range of collaborative tools including video teleconferencing, a shared software environment which includes visualisation tools [4].

Our hypothesis is that the domain of science education is a rich and potentially fruitful domain in which to conduct research into collaborative ODL. Since collaboration is a basic premise of science as an activity. Scientific research is grounded on the understanding that a priori models can be tested and falsified in the sense of Popper (and thereby refined), which in turn leads to the development, extension and integration of a theoretical corpus of knowledge. Objectivity is ensured by the process of peer-group review and publication. This structure of professional science can be directly mapped onto a collaborative ODL structure. Collaboration exposes the individual learner to alternative solutions and approaches to solving problems, from other group members. Collaborative experimentation and discussion implies a real-time peer review process, by the students themselves, on their own work. The role of the instructor is not an assessor, but rather a facilitator or guide. Working within a collaborative structure, students cannot side-step these issues, rather they become intimately involved in these meta-concepts which appear as real concerns as they work collaboratively in a concrete domain such as solving problems in digital electronic circuits, programming or linear systems.

Collaborative learning may be roughly classified as follows: Discussion Groups (email BSCW), systems for Data Collection & Organization, Sharing Documents (eg “SamePage” [6]), Synchronous Communication (such as online chat and video conferencing) as well as large-grain Online Courses or Workshops. We refer to these approaches as eliciting a “broad” form rather than “deep” form of collaboration. They all include some management functionality addressing the issues of access into a collaborative asynchronous or synchronous group. Research and development of these systems started from system perspective, rather than the content perspective. We suggest the contrary, that course content be restored to a position of prominence in pedagogy, and so intend to investigate how specific content can drive the construction and deployment of collaborative ODL initiatives. This is our definition of “deep” collaboration which is essentially content driven.

Starting from the subject domain contents we have engineered a number of distributed activities, where students form collaborative groups to synchronously work on one or more specific problems. Typically the software application provides them with a modelling window, where they may drag and drop graphical objects, write computer code, or compose differential equations, and a second visualization window, where they may view the results of their simulations. We have designed various modalities on this theme which are described below. Our intention in emphasizing the subject content is to
Aims and Objectives

Surprisingly, the catalyst for this research came from student requests to set up a software environment to support collaboration. We had been teaching Java programming using simulated robots for several semesters in small workshop classes of maximum size 20. Students noticed themselves that they were collaborating (verbally) on problem-solving activities, and also that the minimalist Java application we had built for them could easily be extended to support more formal collaboration. Initial trials showed that they were right!

Our aims crystallized quickly: We recognized the need to develop an environment which could be applied outside the arena of learning programming, it should aim to be generic. The objectives followed naturally: To produce a minimalistic environment which could be quickly changed (on a timescale of a day or two) or else dropped without the waste of excessive effort. To facilitate several forms of collaboration, and not just sharing of text. To provide a deep collaboration, grounded in domain expertise at UCW, which we also felt likely would produce an immersive pedagogical environment for our students to learn in. This research, involving deployment in classes, has been underway for two semesters, and has achieved the aims and objectives initially set out.

Ordinary Differential Equation Solver

The modelling of dynamic systems using ordinary differential equations (ODEs), despite its long history in science (over 200 years) remains an important paradigm. Alternative methodologies such as Discrete Event Simulation (DEVS), Finite State Machine, Markovian processes and Petri Nets, for example, have become important in modelling natural systems, business information systems and in computer science amongst others. Nevertheless ODE models remain a fundamental tool for modelling natural systems including population dynamics, economics, and biophysics. Experience gained and confidence in the use of the mathematics of ODEs is still valid and viable. To support collaborative ODL work using ODEs we have developed a Java tool to support the real-time interactive collaborative building of ODE models for dynamic systems. Realistic models such as those describing the behaviour of single or aggregates of biological cells, neural circuits or economies can take the form of a variety of systems of equations. Each system is open to the specification of a number of parameters, which may be inferred from expert knowledge of the domain, via hypothesis based on theory, or via experimental results. Clearly there is a need for simulation here, to critically analyse the results of proposed models against experimental results. This perspective has provided motivation for development of a collaborative ODL methodology specifically aimed at the development and analysis of systems of ODEs.

![Figure 1. Ordinary Differential Equation Solver GUI. Code from each group member appears as tabbed pane at the West. Selected ODE is solved and visualized in the graphs in the East. A chat room is provided at the South, the North toolbar contains transmit “Tx” and receive “Rx” buttons.](image)
The application “ColeODE” presents each student with two windows, one a text region where the ODE, together with parameters, initial conditions and scaling information may be entered in a simple Java-like syntax, (see Fig.1). The second window displays the graphical “orbits” of the ODEs when the text description is “compiled” and “run”. There is a quick turnaround between model or parameter adjustment and visualization of its effects, where the student may build up a family of solution orbits and hence get to understand the ODE system or underlying physical or mathematical model. This functionality is provided in a single-student “stand-alone” mode. But each student is two mouse-clicks away from joining a collaborative group, where the same application allows sharing of the ODE and parameters, so that each student in the group may examine other students’ equations, and also simulate them on their local machine. A vanilla (but integrated) chat room completes the application.

A typical session (and students’ evaluation) unfolds as follows: the class is presented with a given system of ODEs, e.g. for the “slip-stick” behaviour describing the movement of a violin bow over a violin string. They are required to investigate the effects of varying the multitude of parameters present in the ODE model, and to identify which are the key parameters in defining the behaviour of the violin. We have observed the behaviour of a class of ten students by monitoring and recording their activities (with the computer and with each other) during this task. The class was of mixed ability, but was motivated to learn some maths (content) rather than being rewarded for taking part in a research project. The typical behaviour we observed was an initial phase of solo activity interspersed with verbal communication, (this session grouped all students in close proximity within one teaching area). Individuals did not elect to form a group until they felt individually confident enough to present viable results to their peers. One relatively weak student found the activity quite distressing, and needed to work through a paper-based tutorial to gain this confidence. When questioned at the end of the sessions, this student reported that the collaborative methodology was a hindrance to his personal learning. Normally he would have teamed-up with a “buddy” and elicited verbal support during this task. He reported that the technology “got in the way” of his learning and suggested he would rather “read a book”.

The majority of students found the technology simple and useful. They reflected on its ease of use, compared with Web-CT (which has recently been deployed at UCW). The behaviour of these students was interesting: They self-organized into groups of two or three, sharing experiences and models using the application as we intended. They reported that the collaborative environment was “fruitful”, “fun” and all agreed that the approach was beneficial to their individual learning. No student seemed aware of any explicit or implicit “peer review” process present in this collaborative work. Many students were aware of our intentions to investigate modalities of collaborative group work, and favourably compared this work with other UCW initiatives such as formal collaboration to produce a software product and report. Several students mentioned that the activities “sucked them in”, (our objective of an immersive learning environment), where they could engage with difficult subject material, without weariness for a couple of hours without a break. This is truly encouraging!

A second task with this class involved the modelling of a radioactive decay process where students had to make preliminary research to uncover suitable models and parameters. Students reported that collaboration was a combination of verbal with some out-of-class emails in extremis. Getting down to the mathematical simulation, students reported that the collaborative environment felt artificial and unwarranted, though the “local” simulator was a vital tool in exploring the mathematical material. All students worked to some extent in collaboration, though they identified this was because of the class requirement for groups to give a presentation in a “plenary” at the end of the class. Clearly the perception of geographical closeness influenced the students experience, (unfortunately institutional constraints precluded a true ODL deployment of this task).

These tasks have clearly identified issues concerning both the usefulness and formation of learning groups. It seems inappropriate to engage with collaborative ODL technology when students are able to collaborate verbally. This may be as expected. But a more serious concern is the process of formation or nucleation of collaborative groups. It is clear that individual students need support in identifying potential groups of students with similar problems and needs. We propose to address this issue with the addition of an artificial agent layer which will identify potential group members and negotiate ODL group session times.

**DIGITAL CIRCUIT SIMULATOR**

The Digital Circuit simulator was targeted for first year (CS1) Computer Science courses and also Secondary (K-12) schools, based upon experiences using freeware logic simulators. It provides the ability to construct and simulate both combinatorial and sequential digital circuits of high complexity using logic gates, counters, shift registers, memory and ALU blocks. The click and drop interface allows rapid construction of a digital circuit, with varied input sources (switches, oscillators) and output devices (LEDs, oscilloscopes). The simulation is not executed at component level (e.g. pSpice) which is more suited to Electronic Engineering courses, e.g. element delays are not incorporated into the simulation, rather the focus is on combination of the standard logical blocks. Students build a circuit on a “workbench” canvas which may be directly simulated in the “Local” mode of operation. The student may then elect to “broadcast” this circuit to all other members of the group where it will appear as a tabbed panel indexed with the student’s chosen identification. Any student within the working group may elect to transmit his workbench to the group, to receive the deposited group workbenches, or to do both. Any deposited workbench may be immediately simulated by the receiving student, (see Fig.2).
Several tasks were given to classes of around 20 students who were distributed over several locations thus preventing verbal collaboration. In an introductory task driven by an on-line worksheet asking students to discover truth tables of elementary (and combinations of) logical gates, students elected to work independently. They reported that there was no “reason”, “advantage” or “purpose” to work collaboratively, that they “needed time” to assimilate knowledge in “their own time” in a “personal” mode of learning. A follow-on activity (within the same 4-hour session) required the students to solve a traffic-light problem, but they were encouraged to work in a collaborative mode, and to actively seek collaborative partners. We found that students needed help in negotiation the formation of collaborative groups, the tutor would move between locations to elicit the formation of suitable groups. Once established, students found that group work was beneficial: Weaker students reported that the support they received from their peers was “beneficial”, “useful in personal learning”, more “focused” than available from “tutor or worksheet assistance”. But again the main issue here appeared to be the mechanism of group nucleation which had to be motivated by direct tutor direction. A true collaborative ODL environment needs to incorporate this functionality, and again we suggest that agent technology may provide a viable solution.

**JAVA PROGRAMMING ENVIRONMENT**

The Computing Section at UCW has invested resources to develop an Integrated Learning Environment for learning programming (both Java and ‘C’) using autonomous (physical and simulated) robots as the vehicle for learning. The paradigm of this approach has been detailed elsewhere [17]. We have recently extended this approach to take advantage of pedagogy implied by the collaborative ODL paradigm [16]. The scenario is a “Robot World” where various active objects such as robots, rocks, trees and gremlins move around a shared space. Each student is asked to program the behaviour of his robot within this world to achieve a particular goal, such as seeking a light, or clearing up some rocks into piles. This approach requires the use of collaborative ODL, since the robot programmed by each student is entered into a common world, where each student observes the behaviour of all other robots. This use of a shared robot world enforces collaboration, in the sense that each student observes the performance of his robot control code and can peer-evaluate this within the given context.

Students reported that they were unaware of the need for explicit collaborative activities, since this was implied within the context of the activity. They appeared to be concerned with the details of programming tasks, how to elicit behaviour in order to solve the given problems, and how well their particular solutions performed and were in no way deflected by the collaboration technology. Clearly our intention of a deeply grounded activity succeeded. Where there was no artificial need for collaboration, students performed well, and even the weaker students were enthusiastic. They worked at an advanced tempo, (perhaps more motivated by competition rather than collaboration), but unanimously reported that the experience was personally valid. They felt that they had a rewarding and useful learning experience. Many students admitted later that they had been motivated to conduct further Web-based research into the concepts suggested in the “lecture” component of these sessions.

Our conclusion here is simple: When there is no “artificial” requirement for collaboration, then collaboration works. The development of collaborative ODL environments must be motivated by the subject content rather than a pedagogically justified technology.

**SOFTWARE ARCHITECTURE**

Java is used to provide a lightweight accessible Client-Server architecture where each student’s Java application is configured as a client. Since we use dynamic applications, such as simulation, the locus of the processing power is a concern. To distribute processing required, all computational engines are placed within the clients. Since the actual objects of simulation may be quite complex (e.g. involve graphical information, e.g. circuit elements for the Digital Simulator, and Robots and other Robot World object), it is highly desirable to transfer the “minimum number of bytes” between

![Digital Electronic Workbench GUI](image-url)
client and clients via the server. We employ judicious mapping of environment onto factored parameters, using the Java Serialization interface, with Remote Method Invocation (RMI) as a generic communication methodology. The software is designed so that each simulated object communicates a minimal serializable “footprint” to the server. For example we never transfer images or icons, but maintain all “large” objects as local resources. It is the responsibility of the clients to reassemble the object from the received footprint and to execute the simulation. This software structure ensures a most efficient distribution of processing and a minimal requirement on Internet bandwidth. Firewall transparency can be assured by use of Servlet technology layered on top of RMI.

**CONCLUSIONS**

The management of this project has turned out to be quite exciting. Students involved have been cooperative and critically supportive. Faculty have been motivated to contribute to software development. Indeed faculty within other areas of study such as art and humanities have expressed interest in the development of similar resources. Despite this positive aura, there are clearly pedagogical and technical issues which need addressing as a matter of priority. These relate to the formation and management of learning groups which our naïve technology has failed (nor perhaps could ever hope to) address. Our current thinking is to invoke agent technology to negotiate the formation of learning groups and to execute the simulation. This software structure ensures a most efficient distribution of processing and a minimal requirement on Internet bandwidth. Firewall transparency can be assured by use of Servlet technology layered on top of RMI.

**REFERENCES**

[1] Athabasca University Centre for Distance Education. 2004 URL http://cde.athabascau.ca/sotfeval/reports.htm


