

Evolution of the Remote Collaboration Tool:

A Continuing Process

T.P. Amsler, R.V. Jain, R.F. Walters

Introduction

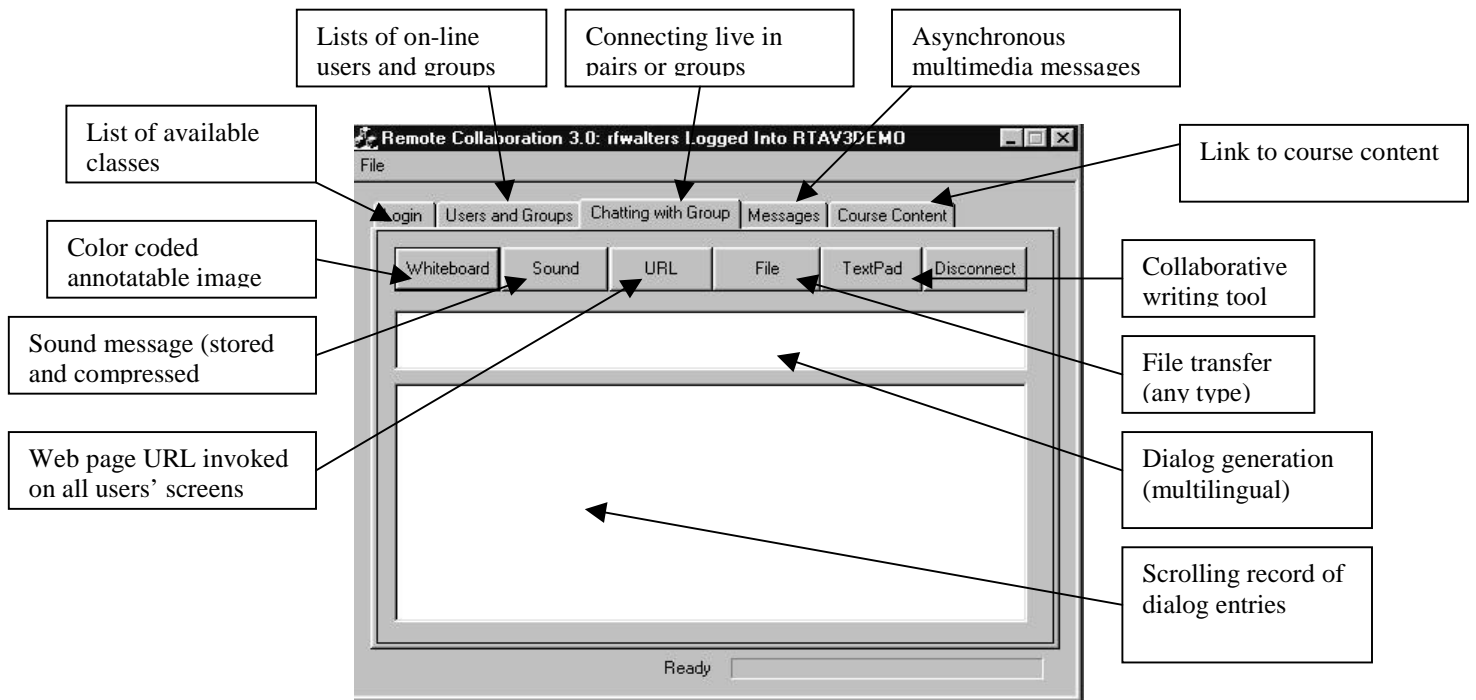
Over the past six years, we have been developing a tool that can be used to enhance collaboration between persons who are not co-located in time and/or space. The package, now named Remote Collaboration Tool, was originally designed to enhance on-line learning. It has been used to augment learning opportunities in several different learning domains, including second language acquisition, introductory computer science, public health administration, and medical informatics. As the product became more widely used, it attracted the attention of people whose interests lay more in collaboration between researchers. They saw its potential in facilitating collaborative research between colleagues who were located in several different states or even countries. These users recognized its potential, but they also saw the value of adding some new features to the package to make it even more valuable. At the same time, its increased use at locations away from the originating campus (the University of California, Davis) led to a recognition that the package required substantial rewriting.

This paper describes the steps that have been taken over the past year and a half to upgrade RCT, and plans for its continued evolution. The techniques used will, we believe serve as a model for development of other tools created through Open Source Software technology.

Earlier Versions

Beginning in 1997, faculty at the University of California, Davis, started using RCT for second language acquisition courses. The package was designed using client-server architecture, with a server running on UNIX platforms and clients available for UNIX, LINUX, PC, and Macintosh workstations. The tool was found especially effective in paired student collaboration assignments, such as password games (guessing the word in the target language described by the partner), matching profiles assigned to each student (e.g., apartment hunting in a foreign city), and other tasks (Blake, 2000).

The following illustration summarizes graphically the options available to users of RCT:



On-line Chat Window

As illustrated above, three modes of interaction are possible: synchronous chat (the main window open), asynchronous messaging, and links to course content (Web URLs, CD files, etc.). Users can connect in pairs or groups and can save interactions on their own systems. The feature called Textpad requires some explanation. It was requested by second language instructors as a vehicle that could be used by pairs of students writing a joint summary of projects undertaken using RCT. One user at a time has control of the screen, and can write a component of what will be the joint report. Control is passed using a queue based (at present) on order of request for control. In a group use, the leader can take control. (we will add a baton-passing feature under leader control .) The text can be multilingual, and the file can be saved on a local computer, edited offline using standard ASCII text editors, then retrieved into a subsequent Textpad exercise.



This version served second language acquisition courses very effectively. It is being used on the Davis campus for teaching French, Japanese, and Spanish, and on several other campuses for teaching other languages, including German and Italian. It is also being used for virtual office hours in a database class at Davis, and it is being investigated for use on a number of other campuses, principally for second language courses. The visibility gained from these uses has resulted in an increasing number of inquiries from institutions in the USA and internationally, and some collaborative projects for its evolution are developing at this time, including one with the Federal University of Brazil in Porto Alegre.

In the collaborative research domain, pathologists discussing images from a very large shared databank have used RCT on a test basis. This initial work has led to a new collaboration in which tools developed by these researchers will be merged with new revisions of RCT described later in this report.

Early uses of RCT led to a realization that substantial revisions of the earlier versions were needed. In addition to improving the robustness of the package to minimize frustrations arising from technical problems during its use, we identified several features that would significantly enhance the package if they could be incorporated. Accordingly we embarked on a redesign that is described below.

Redesign Considerations

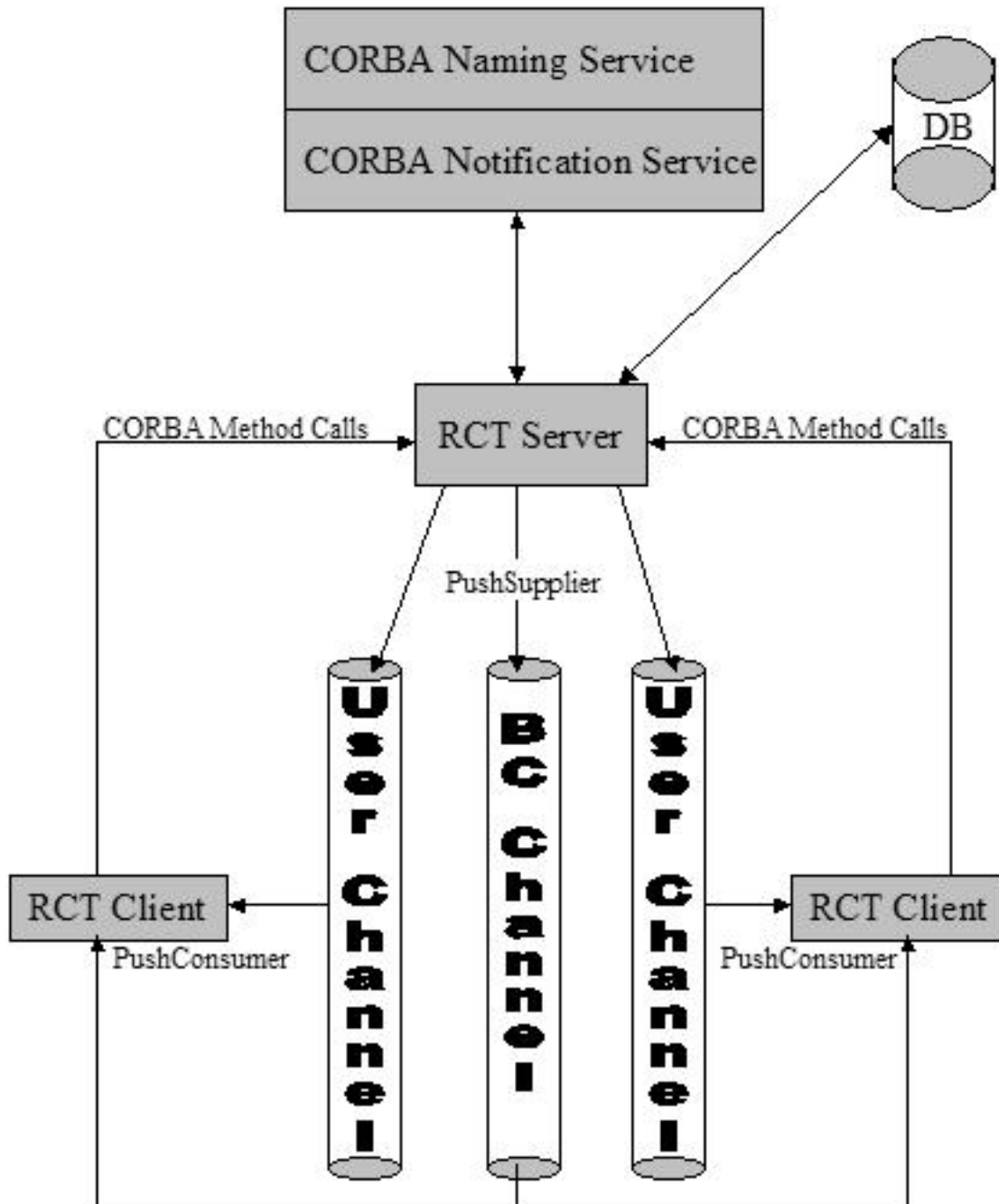
The redesign effort involved improvements, additions, and correction of minor but annoying technical problems. Taken together, it became clear that the package would require substantial rewriting. Legacy C++ code had been written by students at Davis (both undergraduate as well as graduate), and by staff who had worked as students prior to be hired to work on the project. Version control was limited. Some features had been implemented on an *ad hoc* basis but remained in the operating version.

We also realized that continued funding for the project would depend on having available a reasonably complete version before additional funds could be secured. Attempts to market a copyright version of RCT were unsuccessful. These factors led us to make some fundamental changes in our approach to the package. It had always been our goal to make this product available to educational institutions without charge, but to license it to corporations and non-educational institutions as the opportunities arose. With the lack of success in marketing, and a realization that the tool would probably evolve for a number of years – well beyond the likely commitment of the research group at Davis, we concluded that other options for development would be necessary.

The first major design change was to settle on Open Source Software (OSS) technology for the new version. This approach would create opportunities for collaborative development that would not be as easily realized using the copyright, proprietary approach. It also meant that we could take advantage of a large number of tools and utilities already developed by OSS developers elsewhere. This fundamental change was consistent with our long-term goals for the use of RCT in coming years, and it was accepted by the University as a reasonable solution to the growth of a potentially powerful tool.

We also utilized a source code management tool, CVS, which is based on the Revision Control System, RCS. This tool allows us to have multiple developers working remotely and at the same time on the same source code base. In addition, it makes the source code readily accessible over the Internet.

With the CORBA approach to communication, it was possible for us to envisage a structure in which the RCT database could be interfaced with CORBA technology to permit secure data associated with use and management of RCT to be shared using standard CORBA services such as the Notification Service. The overall design is illustrated in the following diagram.



It took over a year to develop the infrastructure that would allow us to begin work on enhanced features that we wished to incorporate, but the resulting structure has proven to be highly reliable, providing fast response and offering upwards scalability that would have been difficult to achieve with the older version or other

approaches using only in-house developed software tools. While we did encounter some problems in view of the fact that the functionality required in the new version of RCT pushed requirements for new OSS tools that were in fact developing as we worked on our package, we were very satisfied with the willingness of other contributors around the world to share their insights as well as their evolving code to assist us in this development.

OSS and CORBA have allowed us to rely on infrastructure tools built by others, thereby both accelerating development time and increasing the robustness of the end product, since the tools provided had already undergone significant testing in the OSS community. Built-in encryption eliminated the need for us to write that component ourselves. Multilingual tools available for working with UNICODE characters greatly expanded our flexibility in terms of languages that RCT could accommodate. The net result was that the OSS infrastructure is highly reliable, fast, flexible, modular, and easily maintained.

Enhanced Functionality

The features of RCT illustrated earlier created a highly useful tool for a variety of different types of communication between remote users. However, as might be expected, users offered a number of suggestions for further enhancement, many of which were incorporated into the design of the OSS version. We also realized that, no matter how many features were added there would always remain a need both for incorporation of new features and for customization of the OSS version, modifying or eliminating features for certain applications. In this section, we describe a few of the most important additions to the OSS version.

Seamless Link between Synchronous and Asynchronous Communication: Many applications, both in learning as well as research environments call for collaborators to communicate over an extended period of time. The ability to interact effectively will often require both synchronous and asynchronous communication. We wanted to extend RCT so that the transition between live and message modes of communication was seamless for pre-defined “teams.” The Team concept also requires that previous interactions be preserved so that it can be reviewed and edited by all members of the Team. This requirement in turn necessitates development of an archiving component that can be retrieved by Team members and updated either synchronously or through messages. Adding the Team feature means that projects (research or learning) can extend over long periods of time, provided that appropriate editing and archiving features are included.

Enhanced Imaging: The whiteboard feature of RCT is powerful, but in its current form, the images are pixel representations of both the original image and annotations added to the image. It would be necessary in many applications (e.g., the pathology example cited earlier) to preserve the original image and separately to store layers of annotations, time stamped and identified by its author. We also realized that the assignable control of image annotation was a potentially important feature both in learning and research environments. In some cases, control of whiteboard annotation may need to be controlled by a baton passing functionality such as described in the Textpad feature.

Incorporation of Enhanced Confidentiality and Security: These issues are more complex in the RCT environment, since there are multiple levels of control required. Data must be transmitted securely (CORBA and SSL provide this feature with low-level communication processes). Information must be capable of being optionally hidden from users under the control of the instructor. Different access levels should be established, e.g., for system administrators, lead course instructors, teaching assistants and students. The instructor in charge should be able to assign these control levels. Some forms of user authentication are needed to verify

that a user is indeed the person using a given name and password. All these considerations should be included in RCT.

Shared Execution of Applications such as Models (with assignable control): Many applications in the learning and research environments would benefit from the ability to run an application on one computer but control it from another one, sharing the output with a group. The control should be transferable under leader control to different members of the group. (Here again, the baton passing functionality developed for Textpad comes into play.) While some proprietary products exist, such as Sun Microsystem's SunForum and Microsoft's Net Meeting, there is no OSS package offering these features. The T.120 protocols (especially T.128) are suitable for this type of application, and we intend to develop this functionality in RCT. We anticipate that this development will also require further research into interface of application sharing within the CORBA environment, but we are confident that it can be accomplished. (In this case, as with several other components of the OSS version of RCT, our design goals are pushing the current development status of OSS functions.)

Incorporation of Foreign Languages including Arabic and Hebrew: Display of foreign character sets is provided by OSS Graphical Tool Kits. In addition, CORBA provides UNICODE support in its services. It remains to incorporate language input methods that are both user friendly and flexible with respect to their use with specific target languages, including keyboard mappings that are consistent with various localization practices.

Enhanced Administrative Support for Content Editing: RCT Administrative control includes several different types of functionality. A course manager should have the ability to create new classes, teams, and groups with appropriate user code and passwords, define links to course content with dynamic control over the current list of resources available, and view and edit interactive data generated during RCT use.

Current Status

At the time of writing this report (March 1, 2002) we are close to having initial alpha release of our new RCT UNIX server and client running on the Linux platform. This initial release includes the Class, Team, and Group concepts as well as the Chat Module. The server comes with a new administration tool, which is written in PHP. The administration tool depends on an Apache web server, a PostgreSQL database, as well as PHP.

Currently, we are also working on a Java client version, which will be platform independent, running on Unix, Windows, and Mac OS X. The only requirement is to have a working Java installation of version 1.3.01 and above. This version will also include internationalization, I18N, which allows the application to adopt itself to any supported language. All the application specific strings, numbers, and symbols will be shown in the predefined language.

Since our work is Open Source Software, we anticipate that, as OSS/RCT becomes more widely known, it will be possible to share development of the tool with others. We are particularly pleased that Professor Liane Tarouco, of the University Federal of Rio Grande Sul in Porto Alegre, Brazil, has agreed to share in the continued development of this tool. We hope that many others will join in this effort.

Availability

The UNIX alpha release should be ready within a few weeks, downloadable from our web site. There will be detailed instructions on how to install all the necessary software packages. This process is lengthy since both the server as well the client depend on cutting edge software packages such as Gtk+-1.3.x as well as omniORB, PostgreSQL, Apache, and PHP. We will run a dedicated test server here at UCD, which people can access so that they don't need to install all the server related software packages. We may also provide precompiled Linux RH7.2 Clients.

We welcome suggestions for specific applications and new features. We believe that the tool will have wide applicability, and we hope that other institutions will join us in using and improving the product.

Acknowledgments

Support for development of RCT to date has been provided by the University of California, The US Department of Education's Fund for Improvement of PostSecondary Education, the US Army's Defense Language Institute, and the Apple Corporation. To all of these institutions, we express our profound thanks.

References

Blake, R. (2000), "Computer mediated communication: A window on L2 Spanish interlanguage," *Language Learning and Technology* Vol . 4:1, pp. 120-136.

Walters, R.F., BB. Douglas, R.J. Blake and D.W. Fahy (2000), "Interactive Tools and Language Acquisition," *Multilingual Computing*, Vol. 11:1, pp. 35-39.

For Further Information, send email to: walters@cs.ucdavis.edu

Or see Web Page: <http://davinci.cs.ucdavis.edu>