ChaRLeS: An Open-Source Chat Room Learning System

Edoardo BONTÀ, Giovanni TORRISI, Marco BERNARDO

University of Urbino, Italy

Abstract

This work presents ChaRLeS – Chat Room Learning System, a virtual classroom recreating the environment of a physical university classroom within a learning management system. ChaRLeS has been developed to enhance the usability of the textual chat of Moodle, by taking inspiration from the one of Land of Learning.
In order to guarantee a good level of portability and performance, ChaRLeS has been designed as a learning tools interoperability (LTI) provider, to be interfaced with Moodle and other LTI consumers, and has been implemented as a real-time web application.

Keywords
chat, LTI, e-learning, real-time web application, Moodle.

Introduction

ChaRLeS is a multilayered textual chat realized as a real-time web application, which overcomes the classical limitations of a Computer Mediated Communication (CMC) against Face-To-Face Communication (FTFC). In particular, ChaRLeS addresses the transactional and geographical distance, which tends to hinder the creation of a learning community and consequently the effectiveness of the learning process.

The main driving factor for the development of ChaRLeS was the improvement of the native textual chat of the learning management system Moodle. However, there are several reasons that motivated its implementation as an autonomous web application. First, an external application is less intrusive from the point of view of the security and of the stability of the application to be integrated with. Second, the external app can be hosted on a different server, without subtracting resources to the main application. Third, it can be used as a stand-alone application or interfaced with other platforms and learning management systems. Fourth, it can be developed using the preferred programming language and the relative frameworks – e.g., the libraries for handling real-time communications. The only constraints to be taken into account are the data interchange and the authentication/authorization protocols to use with the target system. Fifth and finally, there exists a sound and well-defined specification for interfacing an external tool with Moodle and with other learning management systems: the *Learning Tools Interoperability* (LTI).

In addition to the web application ChaRLeS, there is another module, which is the plugin *mod_charles* for Moodle. This plugin has been developed to facilitate the interaction between Moodle and ChaRLeS, rather than using the generic plugin "External Tools", in which only raw parameters can be passed to the web application. Using the plugin, the Instructor (i.e., the name for a teacher/lecturer in LTI jargon) is able to configure the chat room instances of ChaRLeS from within Moodle in a way very similar to that of the native chat, also including the automatic publication of the events in the calendar.

Both modules are open-source. In particular, the web application ChaRLeS is under the MIT License, the same license as the libraries and frameworks used by the application itself. The plugin *mod_charles*, instead, is under the GNU GPL License, as required by Moodle for its official plugins.

State of the Art

Moodle is one of the most popular learning management systems. The platform is well-documented and has a huge number of internal and external

plugins, most of which developed and maintained by its large open-source community. Among these plugins is its native "Chat", whose usability and performance are not satisfactory.

Typical weaknesses observed by the users of our university are: the narrow and single-lined text area for writing messages, the vertical scroll of the page at the end of each polling interval (polling technique is used for reading messages from the server), and the high fragmentation of the chat sessions into which messages are collected. The use of the option "chat server daemon", in Moodle administrator settings, can alleviate the performance issues due to the polling technique, but does not solve the other problems.

On the other hand, there are some chat applications having a good usability and a reasonable performance, as *Land of Learning*, or *LoL*, *Classroom* (Pigliapoco et al., 2008, pp. 1-9), which inspired our work. Nevertheless, many of these applications are based on obsolete/deprecated technologies (as Java applets or Flash/Silverlight plugins) or are prone to security vulnerabilities and to firewall problems. Others require the installation of specific client application/browser plugins to be comfortably used in PC or mobiles, as *WhatsApp*, *Skype* or *Hangout* do, for example.

ChaRLeS, the web application we present, has been designed to overcome the weaknesses of the native Moodle chat, by adding several functionalities that result in an increased usability. At the same time, it is compliant with the standard LTI and with some recent HTML5 features (i.e., *WebSockets*). Thus, the application can be used as an external tool not only by Moodle, but also by other learning management systems. Moreover, ChaRLeS is firewall traversal and accessible from any device (PC, smartphone, tablet) and from any updated browser enabling JavaScript. It has an excellent mechanism for sending/receiving messages with very low latency and bandwidth consumption.

Methodology

This section is organized into the following subsections: *the underlying chat model* and *the design of the web application architecture*.

- **The underlying chat model**

Following a constructivist approach (Crotty, 1994; Stewart, 2013), ChaRLeS creates a multilayered text-based learning environment that overcomes the classical limitations of Computer Mediated Communication (CMC) against Face-To-Face Communication (FTFC). In particular, ChaRLeS tackles the transactional and geographical distance, which usually hinders the creation of a learning community hence the effectiveness of the learning process.

The pedagogical model on which ChaRLeS is based addresses the traditional text chat room inefficiencies (Vronay, 1999) derived from organization and semantic limitations, specifically: (1) lack of recognition, due to the massive use of nicknames; (2) lack of intention indicators, which prevents participants from clearly understanding who a message is addressed to; (3) lack of control over turn positioning, which may provoke confusion, disorganization, and overlapping in participants' contributions in the chat session; (4) typing inefficiency, which limits the communication rated by the users' typing skills; (5) lack of context, caused by the fact that all communication, regardless of its semantic context, is written inside the same text chat; (6) low signal-to-noise ratio derived from the fact that the important pieces of didactic information are mixed up with socialization or service information, which are very useful for the creation of a learning community in a synchronous interaction context, but do not need to be recorded; (7) general uselessness of the chat history, since many chatting room systems suffer from limited readability because of the lack of coherence, the fragmentation of contributions, and the interruptions caused by incoming messages.

ChaRLeS deals with the organizational and semantic inefficiencies described above by creating an innovative and highly efficient 4-channel layered interaction system:

(1) The lack of recognition is resolved due to the spotless LTI integration with Moodle (or with other LTI compliant platforms). All chat participants in ChaRLeS are immediately and transparently recognized in their respective roles (Instructors or Learners). No use of unintelligible nickname is permitted.

(2) Intention confusion is avoided thanks to the use of multiple channels with different communication, i.e., *Question*, *Lecture*, and *Service* channels. Moreover, ChaRLeS allows users to send private messages to selected addressees through a *Private* channel.

(3) The lack of control over turn positioning is solved thanks to the question-handling mechanism that allows all participants to prepare and post their contributions simultaneously. The instructor/moderator will decide when posting a specific contribution from the *Question* channel creating in this way a coherent chat discourse.

(4) Typing inefficiency is addressed by the automatic question-handling mechanism that allows all users to prepare the contributions at their own pace and post them simultaneously, without the risk of creating communication noise.

(5) Lack of context is eluded since all contributions are coherently organized in the chat thread.

(6) Signal-to-noise ratio is enhanced by strictly dividing the different communication flows and keeping away from the *Lecture* and *Question*

channels (which are recorded) the communication noise usually caused by service, socialization, and private interactions.

(7) The usability of the chat history is settled by the real-time production of a HTML log file that includes just the relevant interactions from the *Lecture* and *Question* channels, filtering out the irrelevant communications of the *Service* and *Private* channels.

Once the typical limitations of the CMC text-based interaction systems have been answered, ChaRLeS can better express its potential even against more friendly and modern videoconferencing solutions. In particular: (i) text uses very low bandwidth, thus ChaRLeS reduces digital divide and can be easily used from any kind of network; (ii) the texts resulting from the synchronous interactions are automatically produced, stored, and easily accessed (and hopefully studied) from the integrated Moodle platform; (iii) ChaRLeS can transparently interface with most input/output devices, also on mobile systems (i.e., screen readers, voice input devices, etc.) allowing also disabled people to fully participate in online classes.

- **The design of the web application architecture**

As already mentioned, ChaRLeS is a real-time web app (Lengstorf et al., 2013), i.e., a web application that uses *WebSockets* (Wang et al., 2013) or alternative techniques, also known as *Comet* (Crane et al., 2008), for pushing messages from a server to subscribed clients in broadcast or selective way.

The server push notification is desirable for realizing a modern web chat application, but it is complicated for a developer to use directly the *WebSockets* or other pushing techniques. Fortunately, for languages like C# and Java on the server side, as well as for JavaScript on the client side, there are some frameworks that handle the real-time communications in a way that is transparent and easy to use for the software developer.

Since we used C# to develop ChaRLeS on the server side, the library we chose for handling real-time communications with this language is *SignalR* (ASP.NET SignalR). The library can be used within the new open-source framework .NET Core (ASP.NET Core) for porting stand-alone and self-hosted web applications to different platforms. This means that the same web application can be hosted in operating systems such as Windows, Linux, or MacOSX.

The framework .NET Core, together with many libraries used by the web application, is licensed under the MIT open-source license. This is the main reason why we have adopted the same license for our web application.

The diagram in Fig. 1 shows the whole architecture of the application ChaRLeS. As can be noted, the service is strongly decoupled from the client application by means of a well-defined set of web API and real-time methods (custom services). The client application is written in pure HTML and in

JavaScript, so no knowledge about .NET/C# is required to maintain the user interface, i.e., the code executed by the browser on the client side.

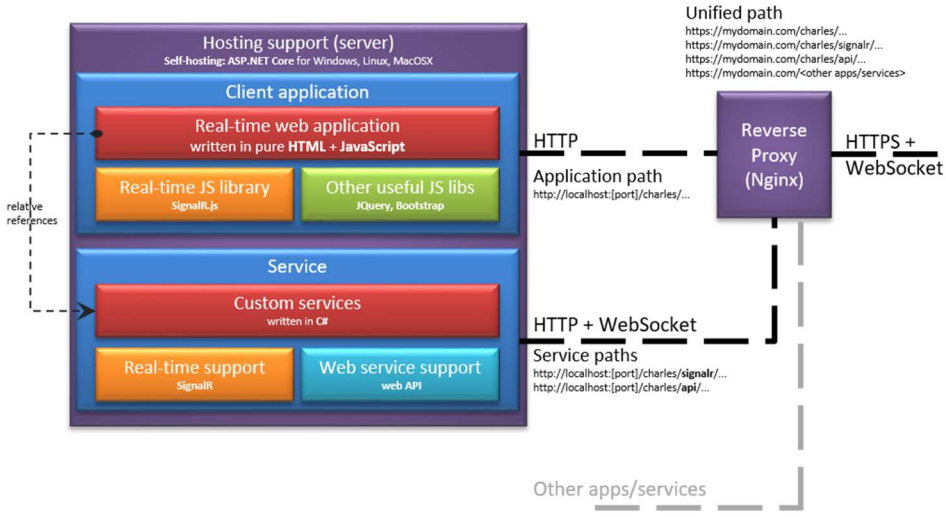

Figure 1 – Architecture of the web application ChaRLeS

Another important thing to note is that also the real-time library *SignalR* is composed of two different parts: the client module, which is a JavaScript library, and the server module, which is a .NET library.

Further, on the client side there are other useful libraries, in particular *jQuery*, that offers general purpose JavaScript functions, and *Bootstrap* for handling the style and the layout of the user interface and for guaranteeing a simple portability of the web pages in mobile devices.

Finally, it is worth to mention the optional use of a reverse proxy server for unifying different paths and different web applications in a single HTTP port, and for allowing communications through the secure protocol HTTPS by means of a (single) SSL certificate. We use Nginx as a reverse proxy, thanks to its good support to the *WebSockets* and to the persistent HTTP(S) connections.

Results and Discussion

The outcome of the chat model and of the design/implementation of the architecture presented before is the web application ChaRLeS.

Fig. 2 depicts a typical usage scenario of a chat room instance of ChaRLeS, where a talk is taking place between two participants. One of them, Steve Smith, is a Learner, while John Doe is an Instructor. The point of view of the figure is that of the Learner, which in this example uses only *Question* and *Private* channels. The Instructor, instead, writes a *Service* message for all the participants, then submits a question from the Learner, and subsequently answers the question through the *Lecture* channel (which has been disabled for the Learner, and does not appear in its own list-box "channel"). Afterwards, the Instructor and the Learner continue their conversation by using the *Private* channel to communicate with each other.

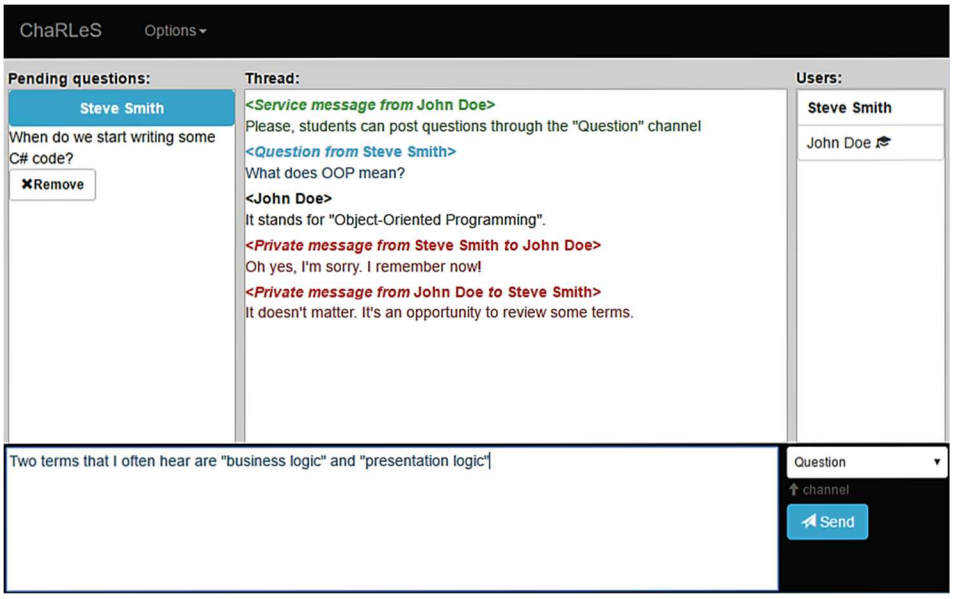

Figure 2 – The chat room instance page of ChaRLeS

As far as the history of the chat session is concerned, only the submitted *Questions* and the *Lecture* messages (e.g., the answers) are saved in the chat session log, or *chatlog* – unless the settings are changed from their default values. The *Service* and the *Private* messages, instead, are preserved in a server cache that will be cleared after a certain period of inactivity of all the participants to the same chat room – 20 mins, by default.

A new participant entering the chat room will be immediately updated with all the pending and submitted *Questions* and the *Lecture* messages, together with the *Service* messages stored in the cache, but will not see the *Private* messages addressed to other participants. However, when the whole message his-

tory is larger than a given amount of memory – 2 MB, by default – the participant will receive only the last messages (< 2 MB). In any case, the entire history of the session will be available to all authorized users on the *viewsession* page, from which the *chatlog* can be visualized or downloaded.

To facilitate the configuration of a ChaRLeS chat room instance from within Moodle, a specific plugin called *mod_charles* has been implemented in language PHP. The new plugin is preferable rather than the generic plugin "External Tools" (also known as *mod_lti*), in which raw string parameters must be set manually by the user for being passed to the web application ChaRLeS as HTTP POST parameters. Nevertheless, the plugin *mod_charles* depends on the native plugin *mod_lti*, but does not require the enabling of the native plugin to work correctly.

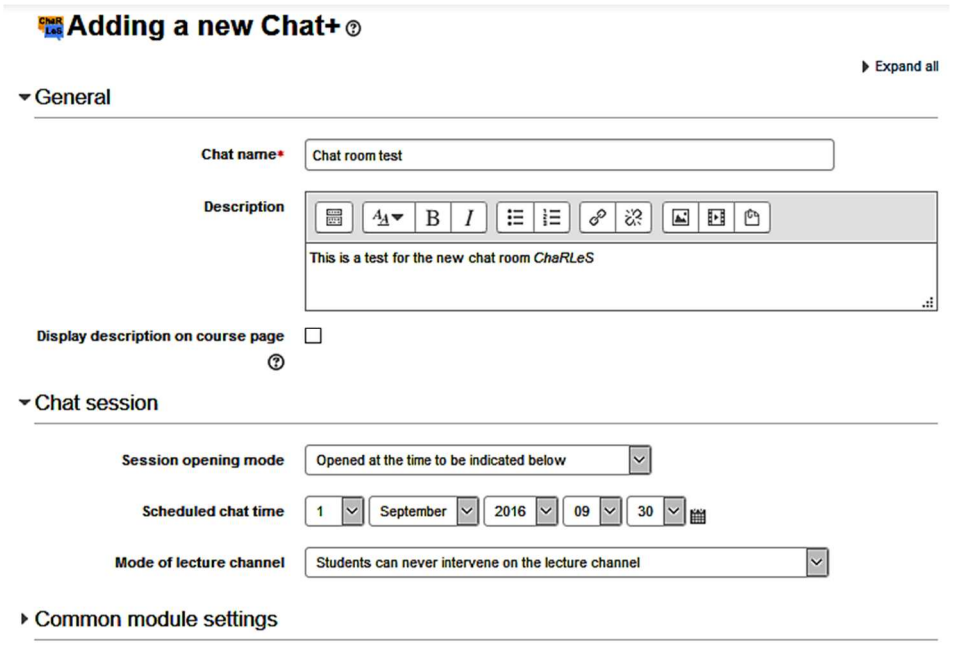

Figure 3 – The configuration page of the Moodle plugin *mod_charles*

By means of the configuration page of the plugin *mod_charles*, shown in Fig. 3, an Instructor can set a few, but relevant, parameters. In particular, the user can specify the "session opening mode" as "immediately open (no publication on the calendar)" or as opened at a given time, i.e., "opened at the time to be indicated below". By choosing the second option, a visual control for setting the start time of the chat room instance will be enabled.

Another configuration parameter is the "mode of lecture channel". The specified item can be "students can never intervene on the lecture channel", or "students can intervene on lecture channel in presence of the lecturer", or "students can always intervene on the lecture channel". In any case, for a greater flexibility, the *Lecture* channel can also be enabled/disabled dynamically by the Instructor through a checkbox contained in the menu of the chat room instance page.

Conclusion

Textual chat rooms can be effectively applied in academic e-learning environments since they provide benefits in terms of student-to-instructor interactivity, interaction quality, cross-platform compatibility, and accessibility. However, chat rooms are traditionally affected by organizational and semantic deficiencies that limit their actual usability. In this paper, we have started from these premises for motivating the development of a new tool, called ChaRLeS, aimed at exploiting the advantages of text-based interactions while overcoming the limitations of traditional chats. The tool creates an academic constructivist environment that provides support for formative evaluation and active, context-specific, and social learning.

The real-time web application ChaRLeS can now be used as an improved alternative to the native chat of the platform Moodle. The same application can also be integrated with other platforms supporting the LTI specification as consumers, or can be used as a stand-alone chat service. The architecture of ChaRLeS has been designed by taking into account quality requirements as functionality, usability, efficiency, and portability both on the server and on the client side.

For the future, we plan to include additional functionalities, as an equation editor based on LaTeX/Mathematics and on the JavaScript library MathJax. Also, we plan to introduce a virtual whiteboard relying on Scalable Vector Graphics (SVG) technology, and some new communication channels, as the *Instant poll* for the Instructor, for receiving immediate feedbacks from participants.

References

ASP.NET Core. ***Welcome to ASP.NET Core!*** **http://www.asp.net/core/**

ASP.NET SignalR. ***Incredibly simple real-time web for .NET.*** **http://signalr.net/**

Crane, D., & McCarthy, P. (2008). ***Comet and Reverse Ajax: The Next-Generation Ajax 2.0.*** **Apress. ISBN 978-1-59059-998-3.**

Crotty, T., (1994). *Integrating Distance Learning Activities to Enhance Teacher Education Toward the Constructivist Paradigm of Teaching and Learning*. Proc. of the Distance Learning Research Conference, Texas.

Lengstorf, J., & Leggetter, P. (2013). *Realtime Web Apps: With HTML5 WebSocket, PHP, and jQuery*. Apress. ISBN13: 978-1-4302-4620-6.

LTI. *Learning Tools Interoperability*. https://www.imsglobal.org/

MathJax. *Beautiful math in all browsers*. https://www.mathjax.org/

Pigliapoco, E., & Torrisi, G., & Messina, M., & Bogliolo, A. (2008). *LoL Classroom: A Virtual University Classroom Based on Enhanced Chats*. European Journal of Open and Distance Learning, (EURODL), pages 1-9.

Stewart, K. (2013). *Facilitating a Constructivist Learning Environment Through Chat Room Dialogue*. Proc. of the International Conference on e-Learning, 532.

Vronay, D., & Smith, M., & Drucker, S. (1999). *Alternative Interfaces for Chat*. Proc. of the 12th annual ACM symposium on User interface software and technology, Asheville, NC.

Wang, V., & Salim, F., & Moskovits, P. (2013). *The Definitive Guide to HTML5 WebSocket*. Apress. ISBN 978-1-4302-4740-1.