

samePage

Research In Real Time Communication

Alexander J. Faaborg
Human Computer Interaction Group
Cornell University
May 10, 2001

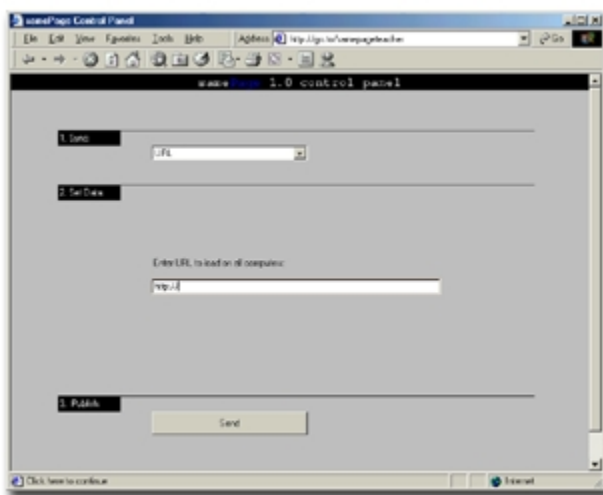
Abstract

Utilizing a network of client and server side applications communicating with http tunneling, samePage is able to facilitate real time push content and collaborative applications for use in the educational environment.

Section I - User Documentation and Technical Summary

Push Content

In its current implementation samePage allows an instructor to have complete control over what is displayed on a group of computers, from web sites to interactive applications. When a student loads samePage by entering **go.to/samepage** into their web browser, samePage will load full screen and immediately display the current content being broadcast. Using the samePage control panel (accessible by going to **go.to/samepageteacher**), an instructor can then control what is displayed over the system. Once the teacher specifies and publishes a piece of content, it will immediately appear on every computer for the student to view or interact with.



Teacher's control Panel
go.to/samepageteacher



Student's Computer
go.to/sampage

Content

Current types of content include:

- Web site
- File - a download dialog box will appear on each computer
- Multiple choice question - results will appear in real time without a refresh

The technical architecture of samePage allows for new interactive applications like the multiple choice question to be easily created, utilizing much of the same communications code that is already place. These applications can communicate in real time with the samePage server, allowing for allowing for synchronous communication between applications.

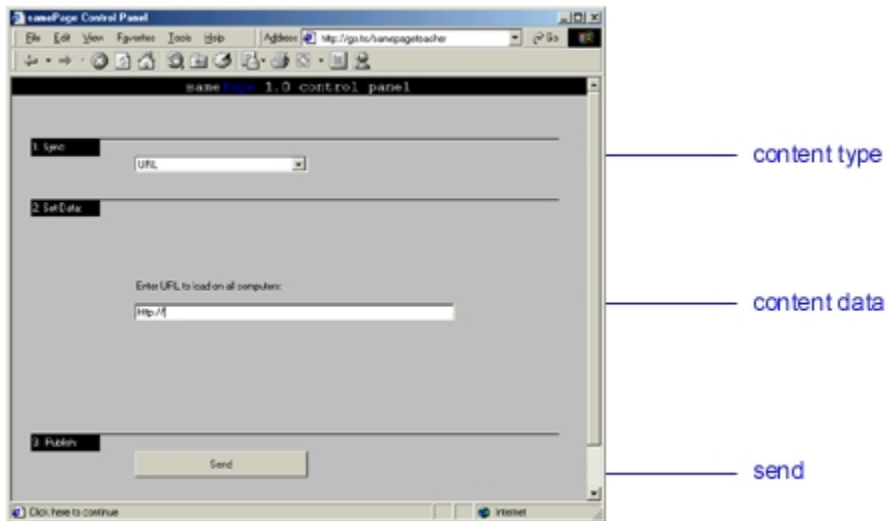
Student's Interface: Full Screen IE

Essentially there is no user interface for the student. Running full screen, the content published over samePage is the only thing a student can view. Interactive applications like the multiple choice question may appear, however there is no way to interact with samePage itself aside from loading and closing it.

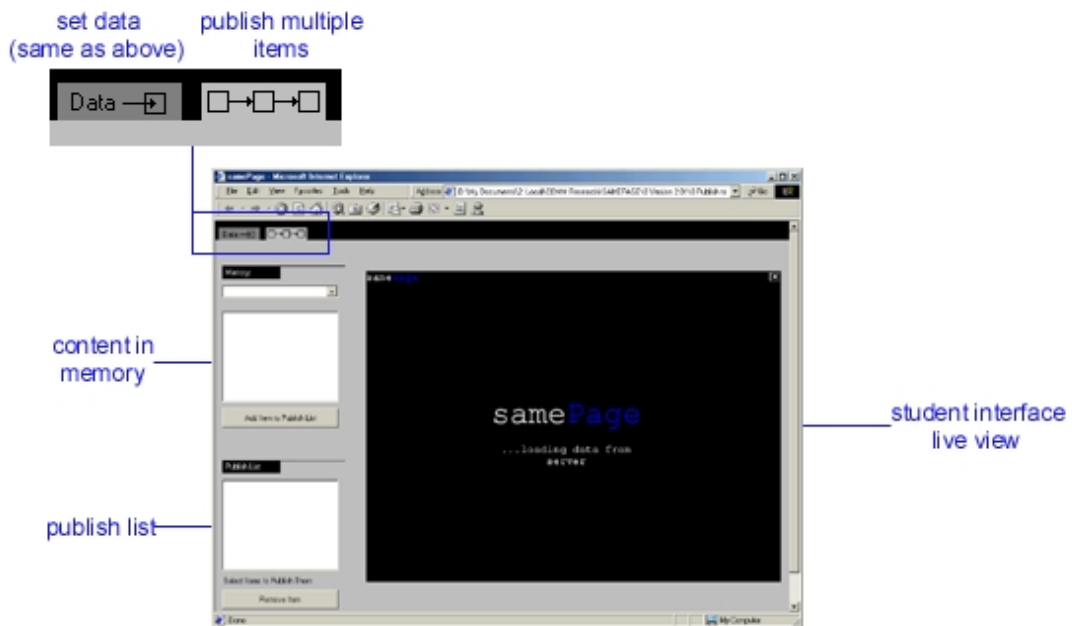


Professor's Interface: The Control Panel

Publishing content is achieved with the control panel.



A new version of samePage will allow a professor to specify a queue of items to publish before the lecture begins. Due to still unresolved technical problems, a stable version of this system has not been loaded on create.hci.cornell.edu. However, here is how the interface currently works in the beta version:

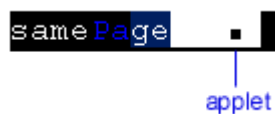


Technical Overview

samePage uses http tunneling between java servlets and client side java applets to allow real time communication. Because objects holding data are sent in this way, samePage can be viewed as one large distributed application with the internal code spanning over a wireless network.



When a teacher publishes content, like the URL to a web site, this information is sent to the server and then tagged in memory as the piece of content currently being displayed. A small java applet on the students computer checks in with the server every second to monitor for new content. Once new content is detected, the applet loads the new content in the viewing area. You can see this small applet by highlight the text in the title bar near the samePage logo, it is 5 pixels high by 5 pixels wide.



This communications network can also be used for applications that need to synchronize other forms of data, like the multiple-choice question. In the case of the multiple choice question, the client side applet requests the question itself and then responds to the server with an answer when the student provides one. The server continuously sends out the results of the question, allowing real time updates on the client side. When an interactive applet like this appears, it conducts it's own communications with the server, and acts independently from the small applet in the title bar.

Installing

There is no installation for the student or the professor. In both situations the client side java applets load automatically.

Configuring the server consists entirely of setting up one java servlet and adding the web content to the public HTML directory. samePage should theoretically work with any web server that can handle java servlets, but it has been developed and tested on Allaire's JRun.

These are the only steps needed to install samePage using JRun:

- Install the demo version of **JRun 3.0 Service Pack 2a** from <http://www.allaire.com/Products/JRun/>
- Place the servlet class files in the servlet directory: **Jrun/servlets**
- Place the web content in the "default-app" directory: **Jrun/servers/default/default-app**

Development Time Line

Using terminal services in Windows 2000, new versions of samePage were loaded onto create.hci.cornell.edu remotely during the course of the semester. This way, whenever

samePage was accessed, the most up to date version would be available. A development server in my apartment was used to create and test beta copies of samePage.

This table summarizes key events in the development process.

Date	Event
Fall Semester	Learned how to use http tunneling to create distributed applications.
1/28/01 to 2/13/01	An early version of the samePage communications architecture is tested with the multiple choice question application.
2/22/01	The communications architecture and the multiple choice question application are simultaneously improved and used to test each other.
2/22/01 to 3/01/01	The communications architecture is expanded to handle multiple types of content. The student user interface is redesigned to run full screen.
3/01/01	The new architecture is tested in the HCI lab in front of professor Gay and it unexpectedly crashes when two separate applets cross threads and start to control each other inside the virtual machine.
3/05/01	A bug is found in the Java Web Server, a product by Sun Microsystems (sun recently canceled this product). The development web server is switched to Allaire's JRun, and JRun is also loaded on create.hci.cornell.edu. An extensive amount of time is spent changing and configuring servers.
3/05/01 to 3/14/01	The internal design of the control panel is changed so that when new applications are added they can be easily integrated in the graphical user interface.
3/14/01 to 4/9/01	Internal changes are made to the communications architecture to make the system more efficient and dependable. Several minor interface changes are made to the control panel including an animated graphic when data is published.
4/9/01 to 5/1/01	A new version of the control panel is created to allow a professor to create and save multiple content items before starting class. This is finished, but the server side code to hold and advance through the data items remains unstable after multiple revisions. The beta copy is never loaded on create.hci.cornell.edu because it is less dependable than the 4/9/01 version.

Review of Testing

Three independent technical tests were completed in the classroom setting during development. Each test was completed with only minor technical errors. However, samePage was apparently used in the classroom only during these tests.

After asking Jennifer Williams on four occasions “if a situation had arisen where samePage could be useful” she asserted each time that it simply had not. She also only wanted the system to be used only when I was present, imply usability was a clear barrier to entry. samePage also was not used because she felt that the idea of push content was just too pushy, stating it was too “big brother like.” Fundamentally she saw the samePage system as pointless, commenting: “when we need to go to a web site, we just tell the class and they all go there.”

This brings up an interesting balance between a total lack of control, where wireless computers are simply a tool for instant messaging with boyfriends and day trading in class, to complete control where wireless computers are used to display simply a static extension of what is appearing on the projector in the front of the room.

If a positive use of wireless networks in the classroom is to be found, it likely exists somewhere in the middle of this spectrum; an application that manages the learning environment and does not allow distractions, but also takes advantage of the samePage network for collaboration and interaction, not simply control. As demonstrated in the rather simplistic multiple choice question example, the samePage architecture can be utilized to create such an environment.

Section II - Research in Collaborative software

We will involve the students themselves in addressing and solving the problems of nomadic collaboration – Nomad Project Proposal to Intel.

The Collaborative Killer Application

While the current application of samePage is to push content onto the student's computer, this is hardly the extent of what the system can be used for. The samePage architecture has been created so that information can flow in real time not just from the teacher to the class, but from the student to the teacher, or between groups of students. Because of how fast this information travels and updates, samePage may be able to significantly change the way teachers educate and the way classes operate.

The yet unanswered question is exactly what type of real time collaborative application will be useful enough to change the format in which students learn in a classroom setting. With the samePage architecture already in place, I feel this is a question worth exploring.