Vision Statement

Team name: NP Compete

Project name: p2pcast

Members: Daniel Vicory (lead), Nicole Theokari, Jerry Medina, Omar Masri, Justin Liang

Background

Currently, there is no easy or decentralized way to do video broadcasting. Existing solutions to video broadcasting require OS-centric programs or browser plugins. Additionally, they require the use of significant centralized server resources. This means that offering video broadcasting services is neither cheap nor convenient for users to decide on a whim to livestream a broadcast. Two very popular approaches to livestreaming video are Skype and twitch.tv. However, Skype is a program which a user and all viewers must install - and is not particularly suited for broadcast, although it is peer-to-peer. Twitch.tv is another approach, though it is centralized, and requires their own program to be a stream. Both are proprietary formats and users are not in control of their own broadcast streams.

The uses of video broadcasting for the masses are endless. For example, a user could want to share their professor’s lecture. Another interesting use case would be to share, say, a local concert with their friends. In these scenarios, directly streaming to all the viewers would be impossible. You, as a user, will unlikely have the CPU or bandwidth resources to support more than a few viewers. If you want to support more than a few, then you would have to make use of a service to rebroadcast your stream to all your viewers. Such a service will have restrictions on what you can stream, to how many you could stream it to, and likely have costs involved.

Our Project

The outcome of the project would provide a solid solution to these problems. Our project is to implement peer-to-peer video broadcasting within the browser using WebRTC. WebRTC is browser standard under development by the W3C to allow for the transmission of general data/files, video, and audio between browsers (or peers). As WebRTC requires external signaling to facilitate connections between peers, the directory will have at least some centralized component (though federation is possible). Additionally, because WebRTC is an emerging standard, there lacks poor cross-browser support, so our project will only focus on developing for Google’s Chrome Canary web browser and as WebRTC matures support for other browsers will come.

Users will be able to select a video source within their web browser, which will be available in a single channel on a web site using our software backend. Viewers will then be able to go to that web site with that specific channel. The web server backend will facilitate discovery of other peers for the viewer so that another peer can rebroadcast the stream they are receiving to that new viewer. In this way, the original source for the broadcast is streamed to a few peers, who then rebroadcast it to other peers, and so on, allowing for peer-to-peer live video broadcasting.

Technologies our project will make use of include JavaScript, node.js, and the emerging WebRTC standard.

Process Model

We will be having daily scrum meetings to discuss what each member is working on and what they will be working on. To manage ours tasks we will be using Trello and Google Drive and employ agile development techniques. We have also established weekly meetings with our mentors and weekly sprints to ensure that communication and work with our mentors is effective and productive.

Milestones

Feb 11, 2014: Requirements specification complete
Mid 1st Quarter: Basic streaming from a broadcaster to a single viewer, beginning of channel concept, very rough UI
March 4, 2014: Design specification complete and updated SRS
End 1st Quarter: Streaming to multiple viewers, channel concept about complete, all signalling handled centrally, UI begins to shapen up, prototype demonstration
Mid 2nd Quarter: Decentralize some of the signalling, handle failures more gracefully, begin to polish UI
End 2nd Quarter: More robust decentralization, be aware of differences in peer bandwidth/latency to pair peers better, very polished UI