Private Overlay of Enterprise Social Data and Interactions in the Public Web Context

Kundan Singh[1] and Venkatesh Krishnaswamy[2]
IP Communications Department, Avaya Labs
[1]Santa Clara and [2]Basking Ridge, USA
{singh173,venky}@avaya.com

***Abstract*—We describe our project, *living-content*, that creates a private overlay of enterprise social data and interactions on public websites via a browser extension and HTML5. It enables starting collaboration from and storing private interactions in the context of web pages. It addresses issues such as lack of adoption, privacy concerns and fragmented collaboration seen in enterprise social networks. In a data-centric approach, we show application scenarios for virtual presence, web annotations, interactions and mash-ups such as showing a user's presence on linked-in pages or embedding a social wall in corporate directory without help from those websites. The project enables new group interactions not found in existing social networks.**

Keywords-Social network; virtual presence; web annotations; enterprise communication; mash-up; HTML5; WebRTC

I. Introduction

Enterprise social networks aim to provide an informal and flexible way to interact and share information within the rigid enterprise policies. They suffer from several issues as follows.

Poor adoption: Employees are reluctant to participate for many reasons, e.g., no training of the new software, lack of common purpose to contribute, or unclear about how the social data may be used – for performance evaluations or legal consequences.

Privacy threat: The IT (information technology) department worries about leaking proprietary data outside the company. Also, people like to separate their private and professional data.

No persistence: A social application often stores and controls the user data, and it is unclear what happens when people leave or a new social application replaces the existing one.

Fragmentation: A new social channel often fragments existing information by creating another place to search, and does not integrate well with directory or unified communication systems

Existing common social practices do not always fit in an enterprise. For instance, social networks use bottom up data sharing with individual contributions and define data popularity by participation, unlike enterprises where department heads dictate and control what data is important in a top-down manner. Instead of focusing on social relations, enterprise software should assist in what a user does at work. We propose the following high-level software requirements to solve the enterprise social software issues listed above.

1) Integration of existing behavior: Instead of creating another collaboration space, it should allow interaction in what you are doing, e.g., writing document, browsing or checking email.

2) Separation of data and application: Allow managing and controlling the user data independent of any one application, so that the enterprise IT can keep the data private and searchable, and new applications do not fragment the data.

3) User in control of her data: Although the data is private within an enterprise, a user should control her own pieces, e.g., who sees her post or where else is it used, and get notified when someone views her data.

Our project named *living-content* uses these to show many interaction scenarios in an enterprise. We apply the known topics of virtual presence and web annotations to enterprises with two motivations: change the collaboration behavior from “go to a place” to “wherever you are”, and use public web pages as contexts of interactions within an enterprise. The collaboration applications are initiated and controlled by the user by installing our living-content browser extension. Besides collaboration, it also personalizes web browsing for the user by selectively modifying certain web pages such as corporate or social directory with users’ enterprise data.

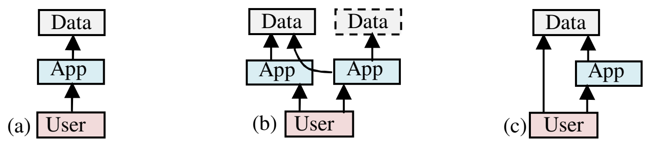


Figure 1. (a) A single application controls the user data, (b) an application allows another to access its data, (c) the user controls her data/approves apps.

Our goal is to separate the enterprise social data away from any single application and make them available as properties to any authorized web application. As shown in Fig.1(a) and (b), social data such as user profiles, interactions and shared notes are often controlled by a web application which requires pair-wise integration with another application that wants to access them. Separating the data in Fig.1(c) prevents its fragmentation and allows the enterprise policies such as access control, privacy and backup to be applied to the data independent of specific social websites and by the entity that owns the data.

After showing the motivating examples and related work in Sections II and III, respectively, we describe the details of the architecture and implementation of our project in Section IV. Section V discusses more user scenarios enabled by our project and their challenges such as access control policies. Finally, our conclusions and future directions are in Section VI. This paper presents the details of design and implementation, and the ramifications of the idea. Performance evaluation and a real deployment experience are for further study.

II. Overview and Motivating Examples

A common enterprise practice is to go to an online meeting system and then import a document to share. Often the document provides the context for the meeting. The meeting system stores the interactions that become unavailable when the document is used outside the system. The edits made by the participants may become unmanageable with many versions, or become unavailable to others outside the meeting system.

Instead of the meeting system, the document could become the starting point of collaboration, as demonstrated in this paper. We define *living-content* as a web page (document) that initiates and provides a context for collaboration and allows storing the annotations and interactions by the viewers within this context. Any web page can become a living-content using our browser extension. The browser extension enables several collaboration applications such as virtual presence, web annotations and co-browsing within the context of a web page or website. These applications store the user data independent of a single website, are authorized by the user to access her data, and can operate on any website while keeping the data private within the enterprise as shown in Fig.2.

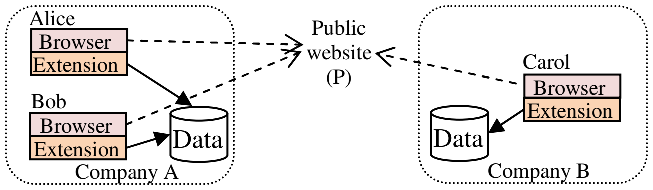


Figure 2. Alice and Bob see each other's data (annotations and interactions) on the public website, but Carol of another company cannot see their data.

Our project enables several enterprise use cases. We classify these scenarios into four categories as follows.

A. *Web Annotations: Private Social Data on Public Web*

Many use cases are enabled if employees can share data in the context of public web pages while keeping the data private within the enterprise. For instance, a sales team may put notes on a customer website, e.g., who to contact or what features are desired for this customer. An employee may post a comment on a web article about the company's product and, two years later, another employee may learn about that past discussion on that article. When a company launches a new advertisement campaign on YouTube, its employees can discuss it on the page itself instead of having to create yet another forum without making their comments public to others outside the company. The job posting web page on craigslist or Linked-In may be used by the hiring manager, human resource and interviewers to co-ordinate private notes containing screening questions and candidates' progress, instead of tracking chains of emails. Enterprise IT department may proactively put technical knowledge on StackOverflow pages in the form of private notes to the employees, because an employee is likely to see that website when looking for a solution to a computer problem.

The web annotation application in living content allows a user to leave notes in a text box, drawing or highlighted text, or to edit a web page and share it with others in the enterprise.

B. *Virtual Presence: Connected Browsing on Any Web Page*

Several use cases are enabled when employees can quickly and easily initiate a conversation from within any web page. With living content, any web page can be used as a readymade context to initiate collaboration from, instead of having to setup, invite and co-ordinate participants on a separate meeting system. For instance, a customer support person may notice that a customer is stuck on the products page and may attempt to guide her using co-browsing, video call and real-time drawings. Team members may visit their team page at a scheduled time for their weekly video meeting or to view the past minutes. A candidate walks in online to a job posting page, is placed in a queue, and later picked up by an interviewer for a video interview. Furthermore, to facilitate programming questions, a shared notepad may be launched on the same third-party web page during the interview without help from that third-party website.

In addition to traditional view of virtual presence where people visit the page to communicate at any instant, the application in living content also supports offline subscription of a page. For example, the author or owner may subscribe to her web page, and when a visitor posts a note on that page, it gets immediately delivered to the subscriber on her instant messenger or email, thus linking between synchronous and asynchronous views of collaboration with virtual presence.

C. *Client Mash-ups: Enhanced Third-Party Websites*

The user experience can be improved on many existing websites if the existing user or enterprise data can be used to modify the content on that websites. The living content extension can embed social and media data, e.g., live video presence on a person's home page, or lab's webcam feed on a department's page, completely using client-side changes and without help from those websites. Such enhancements improve adoption of the collaboration tools by reducing the number of steps needed to initiate conversation in whatever the enterprise user is already doing.

The living content extension can enable impromptu interactions among the developers and testers directly on the popular enterprise applications such as webmail or web pages of subversion or jira without you having to look up a phone number or add a contact in the instant messenger to initiate a conversation. The extension adds presence and click-to-call buttons on corporate directory pages, so that when a user visits a colleague’s directory profile, he can see her presence status and can quickly initiate conversation. Furthermore, the extension can modify the directory listing page on the client side to include the visited user’s social profile or mutual past conversations in addition to her administrative data of the directory, without changes to the directory website or database.

These are the examples of application mash-ups at the data level, and are possible because of the data-centric design in living content which decouples the data producer from the consumer application. Instead of an application asking another application for permission to use the user data, the end user directly grants permission to individual applications to use her data using our generic and consistent data access interface.

D. *Social Presence and Data-Centric Applications*

The data-level mash-ups are further illustrated by sharing social presence among existing applications. The living content project includes collaborative editing and social wall applications, which improve user experience by sharing data with the instant messenger application. A user may edit a web page by creating its local view, and share or merge with her colleagues or instant messenger contacts. We are building a per-user social wall to show different content based on who is viewing and from where, and to show contextual data from external channels like emails, phone calls and instant messages.

The social wall can be embedded client-side by the living content extension in the corporate directory listing page or third-party social websites, but visible only from the enterprise network. The social wall gives a social presence of the user in an enterprise, allows message posting or sharing of calendar events, business cards or files. It can change appearance when overlaid on other web pages such as her Linked-In profile. A user may share a custom view of her social profile and interactions to specific users or groups of users. The social wall may contain contextual data such as the attachments from the recent emails they shared or imminent calendar events [1].

III. Related Work

Unlike the earlier notion of *living content* as a dynamic web page [2], we define it as allowing interaction by its viewers in both synchronous and asynchronous modes. We apply the concepts from virtual presence, web-based communication and data-centric design to enterprise use cases.

Virtual presence [3-4] makes people aware of others browsing the same web page and allows them to interact. It maps the visited URL (uniform resource locator) to a room identifier and a chat server location, and automatically joins the chat room. Early systems [5-9] used web servers or proxies for information gathering and focused on intelligent group creation and proactive information delivery, e.g., based on current topic, past browsing history or link distance. Projects such as WebRogue [10], Cheerz, Weblin, RocketOn, Googlin, BumpIn, Zspeech, Gabbly, SamePlace, WebTalk [11] and Open Virtual World, many of which are not active anymore, demonstrated several concepts, e.g., avatars on the web pages, virtually imitating whisper, scream, handshake and follow gestures, correlating synchronous and asynchronous communication, and a browser plugin or extension to interact on a website without control from that web server.

Custom browsers [12-13] can integrate browsing with social and media sites such as Facebook, Twitter, YouTube and Flickr. Some [14-15] allow editing HTML, CSS (cascading style sheets), XML (extensible markup language) and SVG (scalable vector graphics). Systems to annotate or co-edit web pages [16-18] such as ComMentor, CoWeb, Annotator, ThirdVoice, CritLink, CoNote, Diigo, MyStickies, Open Annotation, W3C's Annotea and Google's Sidewiki [17][19-29] have emerged over more than fifteen years. Although, many of them have disappeared in favor of website controlled sharing and commenting, they did demonstrate several concepts, e.g., using browser extension vs. server assistance, ability to leave a note anywhere on the page, at the end or only in the designated areas, security level to delete or move, personal versus public access to notes, searchable text, and finding time dependent notes.

Most recent efforts are focused on web as a global system of scientific and scholarly notes [30], but not on improving the enterprise collaboration experience. We focus more on utilizing web annotations with other contextual and communication data to improve the short-term and long term enterprise collaboration experience, thus making our work complementary to earlier and ongoing web annotations research. Earlier systems also exposed a threat to websites where the visitors could publicly deface a web page bypassing the owner's control over the content [21][29]. Our work avoids the problem by keeping the annotations and interactions as private data within an enterprise but not visible from the outside public Internet.

Web based multimedia communication is traditionally done using browser plugins such as Flash Player [31] to enable voice and video. Companies have started to embrace WebRTC (Web Real-Time Communications) [32] to enable plugin free media path where one or both ends are in the browser, e.g., many hosted telephony services have added browser as yet another client. Social businesses can bring live web-based interaction in their existing applications such as bug tracking, enterprise wiki, or customer support [33]. Many such systems enable use of a browser to connect to an existing service or application, whereas we focus on building a generic framework for WebRTC independent of a specific application so as to do mash-ups of web applications at the data level.

Plethora of social networking sites have emerged in the last few years and many of them for enterprises, e.g., Yammer, Beehive, Chatter, GoInstant or NextPlane. A trend among these is to create yet another visible fragmentation causing the issues we discussed earlier. Islands of these "data hiding applications" require pair-wise application-specific integration that do not scale for mash-ups as the number of applications grows. As mentioned earlier, a user data-centric model [34] separates the user data from the application accessing them, so that an application can use the data upon user's approval rather than from another application. Recent proliferation of social web applications has amplified the need for such architectures [35-36]. Some community projects address the social data's privatization problem [37-39]. Menagerie [40] allows sharing user's data across web applications using hierarchical naming. BStore [41] has a global web-accessible storage where the user controls which applications get access to her data. Any of these data separating systems can potentially be modified to work in our project.

The main differences in our work are as follows.

1. We focus on enterprise use cases to relax the scalability concerns, but strengthen the security and access control policy needs of the private interactions.
2. We show how to interoperate at the data level without control from another application, e.g., another application may create the annotations to display in living-content, or a user may receive posted annotations on her instant messenger.

3. We use HTML5 technologies such as WebSocket [43] and WebRTC [32] to enable generic real-time interactions in rich Internet applications.
4. Most importantly, we identify many useful enterprise use cases that are enabled by a few simple architectural concepts.

Next, we describe the architecture and implementation of our project that explores enterprise social interactions by building a private overlay of data on top of the existing web.

IV. Architecture and Implementation

A living content use case has three elements: (1) the website or web page that provides the *context* and is visited from the user's browser, (2) the *application* such as web annotation or collaboration that provides the application logic in the form of JavaScript code independent of the website and is enabled by the living-content browser extension, and (3) the *data* generated and used by this application. A user may create different types of data in the context of a web page based on the application being used, e.g., notes or drawings, page edits, chat messages, call logs and recordings, and list of people who participated. As shown in Fig.2, these pieces of data and their contexts are stored in private data storage within the enterprise and a browser extension is used to access them.

A. *Resource-based Application Model*

We use the resource-based application model from our prior research [42], which runs the application logic entirely in the browser while using a resource server as the data storage and event notification system. A resource server is a generic data store with no application specific logic, e.g., for annotations vs. chat history. The resource data is kept private where the user controls her data. An application can define its own data model or use the one from another application, but it does not control the data.

The resource server and the browser extension constitute a generic framework to enable presence and collaboration among people visiting a web page. On this framework, we have built specific collaboration applications such as web annotation, impromptu conversation, shared notepad, white-board and co-browsing. Any other application such as email or social wall can read or write the data upon approval from the user.

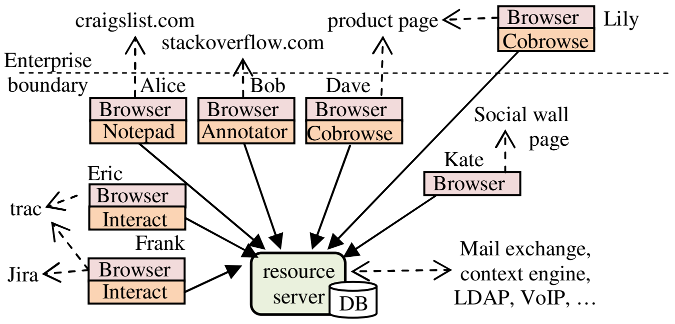


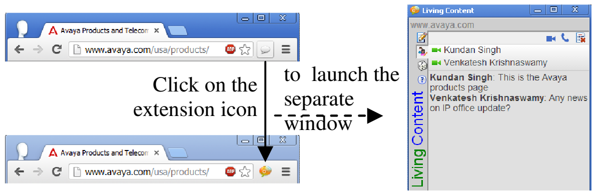
Figure 3. The resource server decouples the data and the applications enabled by the living-content browser extension.

Fig.3 shows the various applications on various websites across the enterprise boundary all using the same resource server within the enterprise network. These applications create and use resources on the resource server to store specific data items, e.g., a note text or a presence status.

Resources are hierarchical data similar to files but with structured representation, e.g., in JSON (JavaScript Object Notation). They are identified by relative paths on the server, e.g., /room/1234/notes. An application accesses them over a WebSocket [43] connection using a request-response protocol [42]. The protocol and its JavaScript API support the standard CRUD (create, read, update, delete) methods on resources. They also allow publish-subscribe on resources, e.g., an application that subscribes to /room/1234/notes is notified when this or its immediate child resource is created, updated or deleted. A file-style access control with permission flags to read, write, append, traverse and send-event are used.

B. *Browser Extension Supporting Pluggable Apps*

The living-content browser extension hosts applications such as web annotation and conversation. The framework to interact among the page visitors is in the extension, but the application specific code is downloaded on demand from an external web server, which is currently co-located with the resource server. This software design of pluggable applications allows adding more applications without changing the framework or redistributing a new version of the browser extension.


Avaya Products and Telecom
www.avaya.com/usa/products/
Click on the extension icon
to launch the separate window
Avaya Products and Telecom
www.avaya.com/usa/products/
Living Content
www.avaya.com
Kundan Singh
Venkatesh Krishnaswamy
Living Content

Kundan Singh: This is the Avaya products page
Venkatesh Krishnaswamy: Any news on IP office update?

Figure 4. The living-content browser extension and the separate window hosting the various applications tied to the browser tab.

Fig.4 shows the living-content icon that appears in the user's browser. When the user clicks on the icon, the living content extension is enabled for this browser tab, a *separate window* is opened and is tied to the user's current browser tab. The icon on the tab changes appearance to indicate that the extension is enabled. The separate window shows the user interfaces of the living content applications in the context of that tab. The example screenshot shows the virtual presence and conversation application, which allows a user to see and interact with others who have also enabled the extension on that visited web page.

As the user navigates from one page to another in that tab, the extension delivers the tab's location URL to the separate window, which creates a resource path, e.g., using one-way hash (H) on the domain portion or the full URL. This root resource path is given to the applications within the separate window to create more resources underneath, e.g., chat messages. Thus, all the collaboration resources within the context are created in the resource sub-tree of the root resource path. An application can decide whether it creates root resource path in the context of a website domain or its individual web page URLs, e.g., annotations may be stored for web pages, but virtual presence for a website so that one can see anyone else visiting the same website domain even if on another web page. Fig.5 shows an example resource tree for presence and annotation. For popular websites such as YouTube or Amazon, instead of using the domain, we extract certain parts from the URL to construct the context's resource root so that the same resource path is used for people visiting the mirrored websites or alternate links for a particular video, a user profile or a product page.

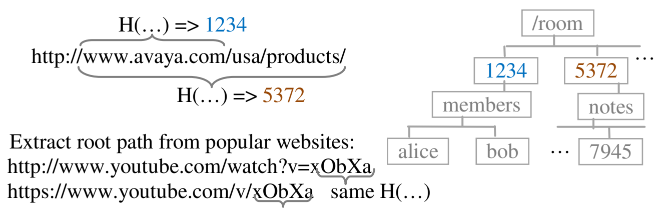

H(…) => 1234
http://www.avaya.com/usa/products/
H(…) => 5372
Extract root path from popular websites:
http://www.youtube.com/watch?v=xObXa
https://www.youtube.com/v/xObXa
same H(…)
/room
1234
5372
…
members
notes
alice
bob
…
7945

Figure 5. Example resources: resource path derived from hash (H) of website domain or full web page URL for presence vs. annotation, respectively. Regular expression is used to extract context from popular website URLs.

The separate window hosting the applications has three tabs: conversation, editing and applications. The virtual presence and web annotation applications are extracted out in the first two tabs, whereas other pluggable applications such as shared notepad and co-browsing are available in the third. These applications are further described below.

C. Virtual Presence and Multimedia Conversation

The conversation tab (Fig.6a) shows a list of users who are on the same website or web page and allows interaction among them. It assumes a virtual chat room resource under the root resource path, and keeps membership and chat messages in that resource tree, e.g., /room/1234/members, /room/1234/messages, respectively. It allows real-time interaction among the viewers via audio, video and text. File sharing is done by including a data URL with the content of the file in an HTML chat message resource. Since the browser imposes limit on the size of the clickable data URL, it is converted to a JavaScript Blob and a clickable object URL of the blob is displayed.

The chat room membership is a transient resource which is automatically removed at the resource server when the client connection terminates. The chat messages are persistent and stay until deleted, so that users can view the past interactions in the context of a webpage. The application may restrict and show only the most recent twenty chat messages, for example.

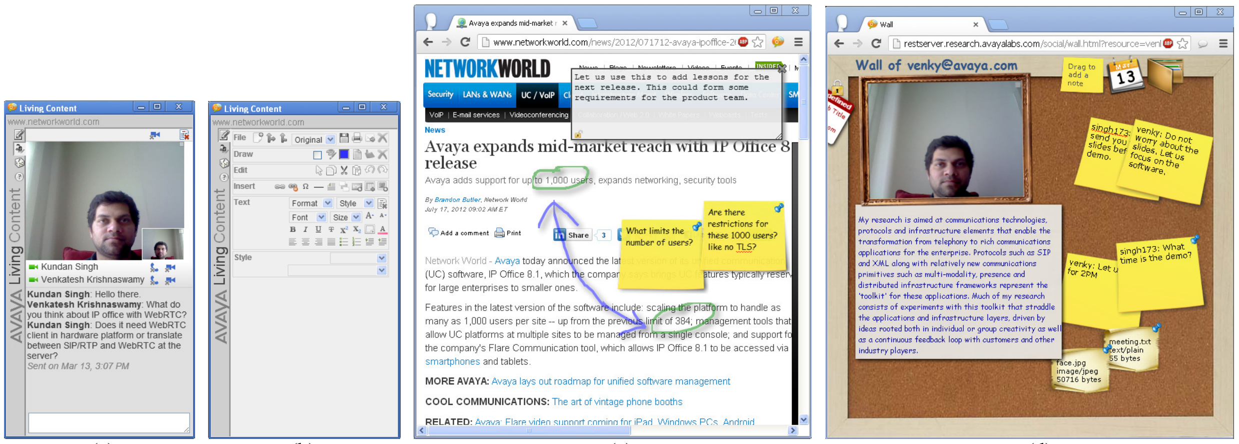
We use our aRtisy developer platform [42] and its communication widgets to implement the conversation panel. Widgets run the application logic in the browser while using only the resource server at the backend. These communication widgets use WebRTC [32] to enable real-time voice and video among the chat room members. A client-server WebSocket [43] connection to the resource server is used to send end-to-end session negotiation messages on the membership resource, e.g., the ones shown in Fig.5 for alice and bob. Currently, it creates a full-mesh media path among the members, but may be modified in future to use a centralized media server for efficiency. A flag in the chat room resource indicates voice or video session. Any member can start or stop multimedia by writing this flag, and a new member can learn about an ongoing session by reading it. The conversation application has other common communicator features such as the indication when someone is typing, or the ability to send emoticons, or to mute audio/video of individual participants.

D. Web Annotations and Page Edits

The editing tab (Fig.6b) allows web annotations and editing of the web page that the user is visiting. Additionally, it allows creating personal views to store the edits. A view can be shared with another user or merged with another view. The difference between annotation and edit is that an annotation is layered on top of an existing web page content to highlight certain parts, whereas the edit mode allows modifying the web page content using the HTML5 contenteditable attribute.

Web annotation is a well-researched topic. Here, we give details of our implementation. We support several types of annotations including an inline text highlight, post-it style text box and freehand drawing (Fig.6c). The drawings are done via SVG. Other widgets such as shared notepad, white-board, or video elements can also be used as overlaid dynamic annotations. Additionally, a shared pointer widget allows a user to share her mouse pointer position with other visitors of the page, e.g., to bring to attention certain areas on the page during a live conversation.

The difference between post-it style annotations vs. shared



Kundan Singh
Venkatesh Krishnaswamy
Kundan Singh: Hello there.
Venkatesh Krishnaswamy: What do you think about IP office with WebRTC?
Kundan Singh: Does it need WebRTC client in hardware platform or translate between SIP/RTP and WebRTC at the server?
Sent on Mar 13, 3:07 PM

```
Let us use this to add lessons for the
next release. This could form some
requirements for the product team.
```

Avaya expands mid-market reach with IP Office 8 release

Avaya adds support for up to 1,000 users, expands networking, security tools

By Brandon Butler, Network World
July 17, 2012 09:02 AM ET

What limits the number of users?

Are there restrictions for these 1000 users? like no TLS?

Network World - Avaya today announced the lat… (UC) software, IP Office 8.1, which the compan… …atures typically reser… for large enterprises to smaller ones.

Features in the latest version of the software include: scaling the platform to handle as many as 1,000 users per site -- up from the previous limit of 384; management tools that allow UC platforms at multiple sites to be managed from a single console; and support fo… the company's Flare Communication tool, which allows IP Office 8.1 to be accessed via smartphones and tablets.

MORE AVAYA: Avaya lays out roadmap for unified software management

COOL COMMUNICATIONS: The art of vintage phone booths

RELATED: Avaya: Flare video support coming for iPad, Windows PCs, Android

Wall of venky@avaya.com

My research is aimed at communications technologies, protocols and infrastructure elements that enable the transformation from telephony to rich communications applications for the enterprise. Protocols such as SIP and XML along with relatively new communications primitives such as multi-modality, presence and distributed infrastructure frameworks represent the 'toolkit' for these applications. Much of my research consists of experiments with this toolkit that straddle the applications and infrastructure layers, driven by ideas rooted both in individual or group creativity as well as a continuous feedback loop with customers and other industry players.

singh173: send you slides bef demo.

venky: Do not worry about the slides. Let us focus on the software.

venky: Let u for 2PM

singh173: What time is the demo?

meeting.txt
text/plain
55 bytes

face.jpg
image/jpeg
50716 bytes

Figure 6. Screenshots: (a) living-content's conversation tab showing a two-party video call and text chat, (b) living-content's editing tab showing various editing and annotation controls, (c) an annotated webpage with three types of annotations – graphic drawing, post-it styles notes and embedded shared notepad, (d) personal wall of a user showing walls posts, shared files, user profile, live video presence and links to calendar and address book.

notepad widget is that while the former has static text, the latter shows real-time typing and allows interaction within the shared widget. Although the annotations are updated in real-time so that the other viewers see the new notes without having to refresh the web page, they persist even after the end of the conversation when the creator is no longer viewing the page.

The annotations are stored in the resource server separate from the web pages on which they apply. An annotation resource contains both the data and type of the annotation. The type determines how to show the annotation when another user loads the page and enables the living content extension. The data is type-specific, e.g., text string and its position on the web page for a post-it note, or the (x, y) position of the control points of a freehand drawing that can recreate the SVG. Since these annotation resources are owned by the user and not the annotator application, they can be used in other applications, e.g., to search for all the notes created by a particular user or to get a list of all the annotators on a particular web page.

While most of the annotations are positioned in the overlay, the text highlights must be in line with the text content of the underlying visited web page. Thus, dynamic anchoring of such annotations is important when the web page is changed on that third-party website. We describe challenges related to access control, cold start and dynamic anchoring of annotations in Section V.

E. Pluggable Application Framework

The applications tab in the separate window shows a list of other applications built on top of the basic framework. These fall under two categories: intrusive items such as co-browsing and drawing that access and/or modify the underlying web page content, and non-intrusive items such as shared notepad or white-board that are launched in a separate browser tab and do not interfere with the visited web page. More applications are easily included without changing the basic framework.

The locked browsing application, when enabled, facilitates co-browsing by locking the browser tabs of the viewers, i.e., it captures any change in the location URL and tells all the other viewers' browsers to visit the same URL. The synchronization messages are sent via the resource server.

The shared notepad allows text editing and delivers the edits to all the visitors in real-time. To avoid simultaneous edits by two or more visitors, we use a shared resource as a mutual exclusion lock to allow only the lock owner to edit at any time. Similarly, a visitor may launch the shared whiteboard application attached to the visited page, see real-time as well as past edits on the board, and improve the collaboration experience.

F. Click-to-Call from Corporate or Social Directory

The ability to initiate conversation on any website is at the core of the living content project. Virtual presence is one way to achieve this. Another way is to modify the visited web page to inject communication elements. The application logic to do such modifications run in the client browser without help from or changes to the visited website, but is tailored towards specific websites. For example, it could identify a web page at http://mycompany/users/{user} as an employee's homepage and insert a click-to-call button on the displayed web page using the browser extension. Clicking such a button initiates multimedia call with that target user whose home page is viewed.

Our living-content extension modifies web pages on our internal corporate directory, bug tracker, wiki and public Linked-In to add a clickable presence icon next to user identity. Thus, the user's presence status is displayed on these web pages wherever the user's name appears (Fig.7). The icon's appearance is bound to the presence resource of the target user, e.g., green for available, grey for offline, red for busy and orange for away.

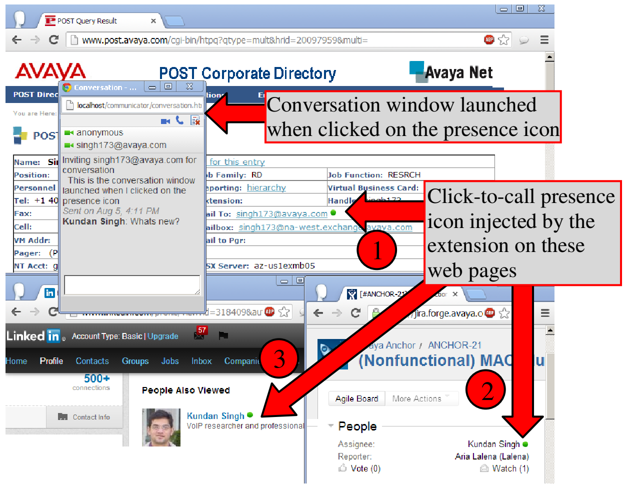

Conversation window launched when clicked on the presence icon
Click-to-call presence icon injected by the extension on these web pages
1
2
3
AVAYA
POST Corporate Directory
Avaya Net
Conversation
anonymous
singh173@avaya.com
Inviting singh173@avaya.com for conversation
This is the conversation window launched when I clicked on the presence icon
Sent on Aug 5, 4:11 PM
Kundan Singh: Whats new?
Linked in
People Also Viewed
Kundan Singh
VoIP researcher and professional
(Nonfunctional) MAC
People
Assignee:
Reporter:
Kundan Singh
Aria Lalena (Lalena)

Figure 7. Screenshots of click-to-call presence icon injected by the living content browser extension on three different websites – (1) corporate directory listing, (2) Jira bug tracker and (3) Linked In profile.

Any application can set the presence resource. For example, communicator is a full featured instant messenger with contact list, presence, text chat, voice, video, and file sharing written in HTML5 using the resource model where all the application logic runs in the browser. It allows a user to set her presence resource corresponding to her enterprise identity. Additionally, it allows attaching the enterprise identity to her Linked-In account so that her presence status appears on Linked-In pages next to her name.

When a visitor clicks on the presence icon, the extension launches a conversation window to enable multimedia chat with the target user (Fig.7). This window uses the same set of communication widgets and resources as the living content conversation tab of the separate window. These examples show the application mash-ups at the data level where the presence and conversation resources are produced and consumed by different applications.

G. Context Sensitive Personal Wall

We gave an overview of the enterprise social wall in Section II D. Our personal wall application shown in Fig.6d is a web-based enterprise social network that allows a user to manage work related discussions, interactions and documents within an enterprise. It differs from existing social walls in two aspects: (1) the content of the wall (and its appearance) changes based on who is viewing and from where and when, and (2) the wall displays resources which may have been created outside the wall application, i.e., it decouples the social data of profiles, connections or wall posts from the wall's application logic. The first aspect means that the wall could display a context-sensitive summary of the visitor's past interactions with the wall owner that happened outside the website via email, outlook calendar, instant messenger or phone call. Fig.6d shows the live video feed or video presence enabled by the wall owner from his webcam, and viewed by the visitor when he lands on the wall page.

1) Automatic social profile and resource connectors

To improve the adoption of this application, the existing enterprise data may be used to bootstrap the social resources in the resource server. For example, the user profile could use the information from the corporate directory, group or department membership, profiles on existing social networks such as Linked-In, and/or collaboration history with others within the company based on past emails, shared calendar events and phone calls. Secondly, making the user profile dynamic that changes over a period of time based on more recent information further improves its usefulness. Finally, it is crucial to present the information in a concise and appealing visual layout without being too verbose to lose the visitor's interest. Hence, the profile content can further be filtered based on who is viewing and from where.

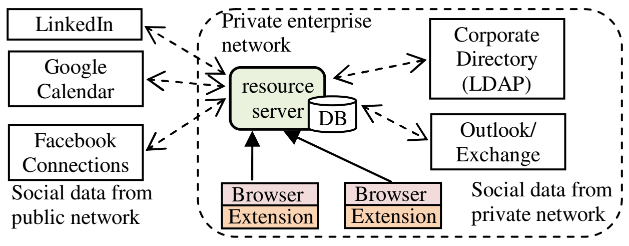

LinkedIn
Google Calendar
Facebook Connections
Social data from public network
Private enterprise network
resource server
DB
Corporate Directory (LDAP)
Outlook/ Exchange
Browser Extension
Browser Extension
Social data from private network

Figure 8. Connectors to resource server for third-party websites and apps.

In our software architecture, connectors are used in the resource server to delegate part of a resource tree to third-party social websites or enterprise applications as shown in Fig.8. It allows the web applications in the browser to transparently access the social and enterprise data upon user's approval without knowing whether the data is stored in the local resource server or dynamically accessed from third-party. Additionally, data from existing enterprise applications such as outlook calendar, address book or directory can be connected to the resource server. Similar to a file system mounting, a resource path could be mounted for specific social sites, e.g., /dev/linkedin could allow an application to access Linked In APIs via resource access or update /dev/linkedin/56854/status to update the user's status on Linked In.

The automatic profile creation deals with these specific questions: What kinds of user information constitute a user profile? Where are these pieces of information available? How often do they change or need to be recomputed? How do we extract such information from existing tools and websites, and use it in our enterprise social network? How do we deliver changes made back to these external tools and websites?

2) Context from embedding web page and viewer

A common problem with social walls is that there is too much useless information for many visitors. This often requires the visitor to manually skim through everything and filter out useless information. The problem is more prominent in enterprise social networks where the viewer does not have enough time, often loses interest and stops using the system. In such cases, changing the view of the social wall based on the contextual relationship between the wall owner and the visitor is desired. For example, if the visitor is in the same department as the wall owner, then departmental calendar events and shared documents may be automatically displayed. For a large number of posts on a wall, we use a relevance score of the post based on who is viewing whose wall to sort the wall posts, e.g., the displayed files, messages or message threads.

It should be possible to embed or link the user's social profile from other websites, e.g., blog, wiki, bug tracker, etc., within the company and explore different behavior based on the identity of the viewer and the context supplied from those websites. Our browser extension allows embedding a subset of the social wall in other websites completely from the client browser, so that the social profile and her administrative enterprise data can be found in one place. The embedded social wall also enables a subset of data sharing, interactions and video presence on those websites that do not already have these features. The wall's appearance is custom defined based on the web page that embeds it, e.g., corporate directory vs. Linked In. Furthermore, it is possible to search the social profile data by keywords or associations (contacts) independent of a specific wall application.

A context sensitive wall deals with these questions: What kinds of relationships between the wall owner and the visitor can be derived from the wall's context? What other relationship information can be imported from other sources? How can the relationship be used to filter and rank the pieces of information in the social wall? How can it visually layout the information within the available constraints (e.g., screen space) without being too verbose?

3) Custom boards and sharing a subset of social profile

The wall application also enables collaboration and sharing in smaller groups. It allows a user to create a custom board with dynamic content to share with others in the group. For example, a user may create a board, attach project specific data such as documents, emails, meeting notes, and share the board with the project collaborators. The dynamic element in the board such as people's profile, message thread or conversation widgets are automatically updated when new information is available, instead of having to re-share the board. For example, if a message thread has a dynamic email filter, a new received email that matches the filter automatically appears on the board. This avoids redundancy by copying the content by reference instead of by value on the custom wall. Such custom boards can also be embedded on third-party websites such as project wiki, where different people may see different views based on their relationship with the board owner.

H. Specifics of Our Implementation

We have already implemented several core pieces including the web annotations, virtual presence, multimedia conversation, locked browsing, shared notepad, whiteboard, personal wall and custom shared boards. Our browser extension is written for Google Chrome that supports HTML5. The extension is easy to port to another browser such as Firefox that supports browser extensions and HTML5. The resource server is written in Python, and uses PostgreSQL database to store the resources and pywebsocket [44] as a WebSocket server. The applications are in HTML, CSS and JavaScript, run in the browser and have no server-side application logic beyond the resource access and event notification of the resource server. Many HTML5 features are used such as WebRTC, WebSocket, LocalStorage, audio/video elements, drag-and-drop, file API, postMessage, and SVG.

V. More Scenarios and Challenges

This section describes some challenges in the new scenarios enabled by living content.

A. User Interface Layout

If a browser extension is not desired, a web proxy could inject code for web annotations and virtual presence without control from the website [7][9]. On mobile platforms that disallow browser extensions, a native application could replace the living-content functions.

It may be desirable to embed the conversation and editing features in the browsing tab as a sidebar instead of launching a separate window. Firefox allows a sidebar extension, whereas Chrome does not. If we inject a separate frame in the webpage to host the extension, the content of the frame reloads and loses state when the user goes from one page to another.

The annotations on websites with dynamic content such as blogs or news become irrelevant when the page changes, e.g., the annotation posted yesterday on a news site may no longer be correct today. To solve this, the annotations could be time stamped along with storing the page content. This also allows showing history or timeline of annotations.

The position of a drawing or text highlight is important but less so of a text note. An absolute (x, y) position of the annotation does not work if the page layout changes when the browser window is resized. The layout of the page is sometimes also dependent on the browser or platform, in which case a note created on one browser may not correctly appear on another. To solve this, the annotation could use the position relative to some element on the webpage.

An annotation has four pieces of information: style, state, selector and scope [26]. The style defines the presentation of the annotation, e.g., using CSS. The state identifies the web page when the annotation was created, e.g., via timestamp. The selector determines the specific portion of the web page to annotate. The scope contains the context behind the creation of the annotation, e.g., what the user was viewing at that time, especially for dynamic content. We have explored anchoring of annotations to semantic structure of the dynamic web pages so that when a page has minor edits the annotations can still be applied by recalculating the selector.

The user interface could use different styles for temporary vs. permanent annotations. For example, while reviewing an online article, one could put temporary notes and markups, and turn some of them into permanent at the end. Alternatively, the permanency level could be determined by social feedback, e.g., more "likes" increases the age of the annotation.

B. Access Control and Groups

With more people annotating a popular web page, it quickly becomes crowded with notes irrelevant to many visitors. One could filter the notes based on the visitor's context, e.g., her contact list, group membership or other preferences. It could show only the recent notes, but allow search on all.

User groups could be automatically created using contact list or corporate directory. The user's browsing history could determine groups of users who often visit similar web pages or search for similar terms, suggesting a common interest. Unlike simple permissions of public vs. limited, fine grained controls like the Linux file ACL (access control list) could be used.

The concept of views is borrowed from version control systems, and allows creating private and separate views of the annotations and edits on the same webpage. This enables multiple groups of people to edit and collaborate on the same webpage without interfering with each other.

Moderator control of annotations is important, e.g., ability to delete inappropriate notes or modify them after the creator is no longer available. The problem is similar to that of the shared repositories in enterprises with many people contributing to file changes, and can be resolved similarly via peer or supervisory reviews of the changes.

Moderator control of voice and video chat is also important. The conversation application can use the corporate hierarchy or page's activities history to automatically pick a moderator. A user should also be able to block media from a misbehaving member, similar to blocking a rogue chat participant.

The system should smoothly transition a person who moves from one group to another or leaves the system, e.g., the group permissions should persist but the ownership of the previously created resources may be reassigned.

C. Enterprise Policies to Social Data

Enterprise policies often dictate how an enterprise social software behaves, e.g., periodic backups of data, access to other data such as corporate directory, or isolation from the public Internet to prevent information leak. The access control policies may become complex and are often not implemented in existing enterprise social software, e.g., allow access to a document within my group only when the visitor accesses it from the corporate network, but not from a mobile device, or allow posting comments on my article from only those with whom I have exchanged emails in the past month. Such complex access control policies are hard to model based on the simple file system style access control because the user space is not known in advance and the access policy needs external data not typically delivered during user authentication.

We are creating an identity and policy assertion layer in our architecture. A separate identity and policy server maps the user's enterprise identity with her other social presence such as email or Linked In identity, and allows the resource access based on any identity. For example, a user could continue the personal wall conversation on her Linked In page using dynamic embedding and unified identity, or a user could interact with her Linked In contacts on her enterprise personal wall or her social wall embedded in the enterprise corporate directory.

The server delivers various primitives for constructing the access policy, e.g., user groups or domains, connected device characteristics, summary of past interactions, etc. It allows access to the resources from outside the enterprise network when properly authenticated. It supports dynamic group creation and membership, e.g., group of people I have talked to in the past year, or those with whom my collaboration score is high. A collaboration score between two people is a multi-dimensional value of past and recent interactions.

D. *Web of Annotations and Interactions*

The overlay of annotations and interactions creates another web (or graph) that can be navigated or searched on. For instance, an employee visiting a webpage notices an annotation from a co-worker and clicks on it to bring up other relevant annotations that link to other web pages. This content and social graph allows the enterprise to analyze it from another angle, e.g., identify pages on public web related to a topic that are important to its employees. The hyperlinks in annotations provide another way to share related pages and add graph links.

Users can search the overlay data on the resource server independent of individual applications. The search can include social aspects, e.g., ranking based on the visitor's relationship with the resource owner. Other factors such as resource age and popularity of attached webpage may also affect the ranking.

The web of annotations can provide input to the policies, e.g., share with others who have annotated on my authored articles, or give relevance score of people or documents based on the links in the annotation graph.

E. *New Ways to Interact*

Combining multimedia with annotations can be done in two ways: (1) annotate multimedia content such as an audio or video file so that the annotation appears when the media player reaches the specific time or content selector, or (2) annotate with multimedia content so that one can post an image or a brief audio or video clip in the notes. The latter is readily enabled by our widgets that host the multimedia content and live streams, e.g., to put live webcam stream from the lab on the department's page to monitor the lab's activity. Other widgets, e.g., text chat and image drawing box as annotations could provide more ways to interact within the web page.

Our enterprise personal wall enables new ways to interact, e.g., client initiated federation with social presence on Linked-In, or shareable and interactive wall embedded on social business tools such as sharepoint, subversion or jira. A user could drop her digital visiting card (a .vcf file) on a wall to request connection with the wall owner, or drop a calendar event (.ics) to invite to a meeting. To mash-up at the data level, the address book and calendar data from the mail exchange could be exposed as resources used by the personal wall (Fig.8).

These scenarios present new challenges, e.g., ability to synchronize the profile resource when the user changes her profile on Linked-In. Similarly, if the user links to her Google calendar or Yahoo address book in the personal wall, it should get the changes via efficient push instead of periodic pull from those websites. Embedding of click-to-call or custom boards on third-party websites is prone to change on those websites.

F. *Interoperate With Existing Documents*

Our project relies on HTML5 for creating and maintaining documents. Enterprises often use office tools such as Powerpoint and Word to create documents. It is possible to sometimes, but not always, convert such documents to HTML5 to work with living-content. Word and PDF file formats use their own way of annotations (highlights) and comments within the document. The pdf.js [45] project may be used to render PDF files in HTML5 and apply the web annotations.

The adoption of enterprise social software often suffers from a cold start, i.e., people will annotate only if others have already annotated before them. This results in very sparse annotations or social interactions on most web pages, but very heavy debate on a few websites. One can use the enterprise data from outside the social wall, e.g., email, instant messaging conversations, phone calls, and calendar events, to pre-populate the user's social profile within the wall. Similarly, even showing unrelated annotations from people in their group or related annotations from other web pages based on similarity (keyword matching) could further motivate them to contribute.

VI. CONCLUSIONS AND FUTURE WORK

Enterprise social networks should be a reflection of the real interactions within an enterprise instead of creating yet another way for online connections. There are six core functions in a social network – search, grouping, authoring, tags, signals and recommendations [46]. We present a data and user-centric software architecture for building pieces of these core functions at the enterprise-wide data level instead of application specific fragments. Our resource server decouples the enterprise social applications from the user data, so that the data can be managed and controlled independent of individual applications.

We have presented web annotations, virtual presence and personal wall as use cases of this resource application model. The application logic runs in the client browser, and the user identity is linked to the corporate directory. In addition to the telephony-style applications such as phone calls, conferences or instant messages, our architecture enables newer applications for synchronous and asynchronous interactions. The end user is in control of her data and applications, whereas the enterprise IT can manage the overlay data, e.g., take backups or protect inside the enterprise boundary. Since custom configured browsers are common in enterprises, the living-content extension could be pre-installed to improve adoption among employees.

We take a clean slate approach ignoring the legacy communication protocols while relying only on HTML5. Our project is an initial attempt to look at the enterprise social interactions from a new angle. Although we have an initial implementation, a few scenarios mentioned in our paper including the ideas discussed in the section V still need to be implemented. We are working on a loosely coupled integration of resource server with existing social networks via connectors. We are also exploring website controlled annotations and collaboration by injecting the living-content applications from the website instead of a client side browser extension. Our future work on web-based collaboration will likely be guided by how WebRTC and related technologies are adopted by browser vendors and enterprises.

ACKNOWLEDGEMENTS

We thank Dilpreet Singh Chahal for implementing shared notepad and SVG-based drawing annotations, Sireesha Pilaka for the custom wall boards and message threads, and Parth Joshi for text highlighter and shared mouse pointer. John Buford helped in integrating living content with connected spaces, a hosted meeting and document collaboration system. Krishna Kishore Dhara is helping in utilizing contextual information in annotations. Various members of our research group provided valuable feedback on our project and its initial implementation.

REFERENCES

[1] K.K.Dhara et al., "Reconsidering social networks for enterprise communication services", IEEE Globecom, Florida, Dec 2010.

[2] N.Usborne, "Living content: it's what people want", Blog article, Jul 2010, http://searchengineland.com/living-content-its-what-people-want-46006

[3] H.Wolf, "An introduction to virtual presence", Technical note, Jul 2007, http://www.virtual-presence.org/notes/VPTN-1.txt

[4] H.Wolf, "Extension to Jabber group chat for virtual presence", XEP-0151 (deferred), Jul 2005

[5] Y.Mass, "Virtual places – adding people to the web", Internation world-wide-web conference, 1995.

[6] P.P.Maglio and R.Barrett, "WebPlaces: adding people to the web", poster, Internation World Wide Web conference (WWW), Toronto, Canada, 1999.

[7] R.Barrett et al., "How to personalize the web", ACM conference on human factors in computing systems (CHI), Atlanta, GA, Mar 1997, http://www.psrg.lcs.mit.edu/projects/inforadar/p75-barrett.pdf

[8] G.Sidler et al., "Collaborative browsing in the world wide web", Joint European networking conference, May 1997.

[9] T.Erickson et al., "A sociotechnical approch to design: social proxies, persistent conversations and the design of Babble", ACM human factors in computing systems (CHI), 1999.

[10] A.Soro et al., "WebRogue: meet web people", International conference on web based communities, ISBN: 972-99353-7-8, pp.267-271, 2005.

[11] WebTalk: chat with others browsing the same website as you, open source project, Jun 2010, http://code.google.com/p/webtalk-project/

[12] Flock, 2011, http://en.wikipedia.org/wiki/Flock_(web_browser)

[13] Rockmelt, A socially connected browsing, 2012, http://en.wikipedia.org/wiki/Rockmelt

[14] Amaya, the W3C's web editor, 1994-2012, http://www.w3.org/Amaya/

[15] BlueGriffon, the next-generation web editor based on the rendering engine of FireFox, http://www.bluegriffon.org/

[16] V.Vasudevan and M.Palmer, "On web annotations: promises and pitfalls of current web infrastructure", International conference on system sciences, Hawaii, 1999.

[17] S.Jacobs et al., "Filling HTML forms simultaneously: CoWeb – architecture and functionality", Computer networks and ISDN systems, Vol 28, issues 7-11, p.1385.

[18] Open cooperative web framework: JavaScript enablement of concurrent real-time interactions, Jan 2011, http://opencoweb.org/

[19] M. Röscheisen et al., "Shared web annotations as a platform for third-party value added information providers: architecture, protocols and usage examples", Standford University Technical Report, 1994.

[20] I.Ovsiannikov et al., "Annotation software system design", Annotation technology, 1998

[21] ThirdVoice: a web annotation browser plugin for IE, 1999, http://en.wikipedia.org/wiki/Third_Voice

[22] K.P.Yee, "CritLink: advanced hyperlinks enable public annotation on the web", 2002, CiteSeerX:10.1.1.5.5050

[23] G.Gay et al., "Document centered peer collaborations: an exploration of the educations use of networked communication technologies", Journal of computer mediated communication, Vol.4, issue 3, 1999.

[24] Diigo: web highlighter and sticky notes, https://www.diigo.com/

[25] MyStickies: sticky notes for the web, http://www.mystickies.com/

[26] R.Sanderson and H.V. de Sompel., "Making web annotations persistent", ACM Joint conference on digital libraries, pp. 1-10, 2010, arXiv:1003:2643 [cs.DL]

[27] M.Koivunen, "The Annotea Project", W3C, http://www.w3.org/2001/Annotea/

[28] Google Sidewiki: a web annotation tool, 2009-2011, http://en.wikipedia.org/wiki/Google_Sidewiki

[29] C.Irvine, "Google Sidewiki: new tool lets anyone comment on webpages", The telegraph, Sep 2009.

[30] T.Carpenter, iAnnotate – whatever happened to the web as an annotation system, blog, http://scholarlykitchen.sspnet.org/2013/04/30/, Apr 2013.

[31] K.Singh and C.Davids, "Flash-based audio and video communication in the cloud", Technical report, 2011, arXiv:1107:0011 [cs.NI]

[32] WebRTC 1.0: Real-Time Communication Between Browsers, W3C Working Draft, Aug 2012, http://www.w3.org/TR/webrtc/

[33] A.Lepofsky, "Social business 2013: less talking. More doing." Dec 2012, http://www.constellationrg.com/blog/2012/12/social-business-2013-less-talking-more-doing

[34] R.Joshi, "Data-oriented architecture: a loosely coupled real-time SOA", Whitepaper, Aug 2004, http://www.rti.com

[35] T.Berners-Lee, "Socially Aware Cloud Storage", Notes on web design, Aug 2009, http://www.w3.org/DesignIssues/CloudStorage.html

[36] E.Naone, "Who owns your friends?", MIT Technology Review Magazine, Jul/Aug 2008.

[37] The DataPortability project to connect, control, share and remix, 2007-2009, http://dataportability.org

[38] Unhosted web apps: freedom from web 2.0's monopoly platforms, http://unhosted.org

[39] The diaspora project: a privacy aware, personally controlled, distributed open source social network, 2010, http://diasporaproject.org

[40] R.Geambasu et al., "The organization and sharing of web service objects with menagerie", World Wide Web Conference (WWW), 2008.

[41] R.Chandra, P.Gupta and N.Zeldovich, "Separating web applications from user data storage with BStore", USENIX Conference on Web Application Development (WebApps), Boston, MA, Jun 2010.

[42] K.Singh and V.Krishnaswamy, "Building communicating web applications leveraging endpoints and cloud resource service", 6th international conference on cloud computing (IEEE Cloud), Santa Clara, CA, Jun-Jul 2013

[43] The WebSocket API, W3C candidate recommendation, Sep 2012, http://www.w3.org/TR/websockets/

[44] Pywebsocket project: WebSocket server and extension for Apache HTTP Server, http://code.google.com/p/pywebsocket/

[45] PDF viewer built with HTML5, project page, https://github.com/mozilla/pdf.js

[46] A.McAfee, "Enterprise 2.0: the dawn of emergent collaboration", MIT Sloan Management Review, Vol.47, No.3, Apr 2006.