ABSTRACT
Collaborative navigation systems provide a useful way for virtual groups to share information through the web. However, the common set of features of these tools is not enough to offer a more face-to-face-like browsing experience. To fill this gap, this paper presents a novel collaborative web browsing proposal, which aims at integrating notably flexible management of sessions, the maintenance of shared production spaces and efficient communication facilities. Also, the proposal relies on a collaboration ontology that provides a well defined conceptualization and a common vocabulary. In order to demonstrate the feasibility of our approach, OCEAN prototype was developed and tested.

Keywords: Collaboration, web browsing, ontology, groupware, tutoring, CSCW, e-learning.

INTRODUCTION
The variety of ways by which people interact have dramatically changed in the recent years. The conception of new teamwork paradigms has encouraged the creation of innovative collaboration technologies. This fact has led to the dynamic collaboration scenarios faced nowadays. Amongst these recent paradigms, there is one called collaborative web browsing (co-browsing) paradigm. Such paradigm consists in allowing distant users to jointly browse Internet content, for instance, web pages.

Several application areas can take advantage of the co-browsing paradigm. For example, web search, which is one of the most common online activities, is often undertaken in shared-computer context (Amershi & Morris, 2008). Educators have also verified the added benefit of co-browsing for teaching as it fits nicely into the theory of constructivism, allowing students to learn by exploring and sharing their own ideas and knowledge (Aneiros & Estivill-Castro, 2005). In fact, in different domains co-browsing systems have been commonly used as: (i) e-learning systems, to handle online lectures and presentations (Santos et al., 2009) (Brooks, Hansen, & Greer, 2006)(Gerosa et al., 2004); (ii) helpdesk applications, to support users in guiding others through desired tasks (Dieberger et al., 2000); (iii) e-commerce environments, enabling users to recommend products or to negotiate purchases (Sosign Interactif, 2008)(Gerosa et al., 2004); (iv) lightweight alternative for desktop sharing tools, to enable sharing of web-based content (Hoyos-Rivera et al., 2006)(Esenther, 2002); and recently (v) feeding social networks with browsing recommendations (Santos, Sana, & Oliveira, 2009)(Maintainers, 2008a).

This paper introduces a co-browser named OCEAN (Oliveira Santos, 2010) that, besides supporting basic co-browsing functions, offers additional important functionalities, mainly related to
a lightweight architecture for users communication and the maintenance of collaborative session awareness. It is worth to mention that one of the main advantages of this approach concerns the introduction of document annotation facilities, useful for expressing thoughts and getting the focus of attendees on what the presenter wants to highlight. For example, while accessing online lecture materials, text notes and draws over the content can help students to remember the key points of that lecture, or even favor new discussions (Chong & Sakauchi, 2001b).

The conceptualization behind OCEAN is captured by an ontology regarding the collaboration domain, proposed in (F. F. Oliveira et al., 2007). This ontology provides reusable domain knowledge and a common vocabulary (Guarino, 1998) used to optimize the OCEAN conceptualization process, decreasing the overhead on domain knowledge acquisition.

The remaining of this paper is structured according to the following organization: Section 2 presents the OCEAN proposal. Section 3 formalizes such proposal, presenting conceptual models. Section 4 describes how this proposal is designed for supporting its features. Section 5 presents the OCEAN prototype. Section 6 discuss the approach adopted in OCEAN in comparison with related work. Finally, section 7 concludes this paper, presenting some outlines of our future work.

**THE PROPOSAL: CHARACTERIZING MAIN FEATURES**

Web browsing is traditionally an individual activity, where a person uses a web browser (e.g. Mozilla Firefox, Microsoft Internet Explorer) for accessing published hypertext documents. However, this browsing can actually be seen as a social event (Gerosa et al., 2004), where users share their browsing activity (Hoyos-Rivera et al., 2006). In this scenario we introduce OCEAN.

In order to properly organize OCEAN conceptualization process, this last was mainly founded on the 3C Model. Firstly proposed by Ellis et.al (Ellis, Gibbs, & Rein, 1991), this model classifies collaborative application, also known as groupware systems, through three characteristics: coordination, cooperation and communication.

**Coordination**

In the context of collaboration sessions, coordination appears to be a major aspect for keeping controlling or management. Commonly, this management is made by imposing constraints in order to control participants’ activity. On one hand, massive set of coordination rules turns the service bureaucratic, making the collaboration a daunting task. Usually such systems establish exclusive roles to participants, limiting his/her actions to it in the entire session. On the other hand, not imposing any restriction could also be harmful for the quality of the session, since it could cause rework and loss of information. Therefore, the point is to find an equilibrium, allowing users to collaborate in a certain manageable freedom state (W. Wang & Haake, 1998)(Schmidt & Simone, 1996).

In this regard, OCEAN introduces a flexible mechanism for managing co-browsing sessions. This mechanism is divided in two coordination levels. The first deals with subgroups of participants, where each of these subgroups is independent to browse. Following, the second coordination level deals with subgroups’ internal coordination rules. Internally, there is only one partici-
participant (presenter) with browsing privileges in the same subgroup, while all the others in the subgroup are attendees. In particular, the presenter role can be assigned to any other subgroup’s participant, for instance through token passing mechanisms (Borghoff & Schlichter, 2000).

Therefore, OCEAN coordination mechanism enables the formation of specialized collaboration sub-session allowing groups of users to work on their specific goals, but still participating on a main co-browsing session, for a major shared goal. Figure 1 depicts an arbitrary scenario of a collaborative browsing session, focusing on the coordination levels.

![Co-Browsing Session - Coordination Structure Snapshot](image)

Figure 1. Coordination levels

Moreover, the OCEAN’s coordination mechanism is quite simple to manage. It provides coordination primitives in order to easily re-configure the session, without overcharging the user with too many commands or workflows for example. The primitives are the specific coordination actions supported by the mechanism, and are associated with the related task level. Considering the groups management, on coordination level 1, the primitives are: create thread, join thread and leave thread. Using these primitives, any user can respectively, create a new co-browsing thread, join an existing co-browsing thread or leave the co-browsing thread he/she is currently participating in. Likewise, the coordination level 2 provides: privilege grant and privilege revoke, used for managing privileges for user’s action and information access. For instance, these primitives are used for exchanging the presenter role between two participants, reconfiguring their browsing privileges.

**Cooperation**

According to the 3C-model, cooperation is the production resulting from the collaboration activity that occurs inside shared workspace (Fuks et al., 2007)(Fuks et al., 2005), either for real or virtual spaces. Actually, the cooperation can be materialized by the set of collaboration artifacts produced in a collaboration session. In face of that, cooperation in the collaborative web browsing paradigm lies on the participants’ browsing activity. Thus, the main artifacts produced in a co-browsing session are the web contents browsed by the group, where the cooperation can be materialized through a Co-Browsing History.

Several co-browsing systems do not provide any session history recording, like in (Maintainers, 2008b) and (Gerosa et al., 2004). Consequently, some collaboration information may be lost.
during session execution. This is an important drawback regarding, for instance, the support for session latecomers. It is known that Colab (Hoyos-Rivera et al., 2006) and Clavardon (Sosign Interactif, 2008) keep a history of browsed URLs, while the system proposed by (Gerosa et al., 2004) creates a log of users’ actions. However, logging actions and recording URLs may not be enough for allowing users to become contextualized with the co-browsing session. For instance, when participants join an ongoing session, they may want to review previously presented information (e.g. URLs, annotations and coordination decisions). In addition to that, making such information available when the session is over is an important feature, allowing the creation of a knowledge repository for later reference. It may be useful for users who did not take part in the session to be able to acknowledge what has been discussed in it.

Due to the above facts, the co-browsing history proposed here is a registry that, different from other co-browsing environments, stores all events occurred in a co-browsing session. Such history is a structure organized as parallel threads, each thread for one coordination subgroup.

**Communication**

According to Fuks et al. (Fuks et al., 2007), communication is related to messages exchanging and negotiations by participants. Regarding co-browsing scope, the collaboration essentially lies on the web content recommendations. Communication can be really relevant for supporting this co-browsing activity, for example for promoting knowledge dissemination or for providing means of negotiating content relevance and leadership privileges during presentations.

General purpose communication features like chat rooms and conferences are quite useful on co-browsing sessions. However, providing general communication functionalities might not contribute to the system’s specification. General communication tools are still valuable for free collaboration, and should not be ignored. Accordingly, instead of defining them as part of OCEAN’s specification and re-implementing them, a good strategy for using such features is to provide means for integrating existent communication tools to the co-browsing system. Such strategy was adopted by Lima et al. (Lima et al., 2007), showing the integration of a co-browser with an audio conference system. We know that this type of feature is important but choose to support annotations first due to the fact that is a brand new concept, letting the chat communication to further implementation.

As aforementioned, in OCEAN, the choice was for the Annotation feature, a subset of the digital ink concept (Provensi, Costa, & Sacramento, 2008), to mainly represent the communication aspect of our proposal. Generally, an annotation feature stands for giving to a user the ability to make notes and marks over a visualized content, with the intention of pointing some specific part of this content, so calling other viewers’ attention. Such facility is quite useful on reviewing tasks, for instance. This feature is usually encountered on collaborative document manipulation systems, for instance, Microsoft Word, Microsoft Visio and Adobe Acrobat. Contrary to traditional browsing paradigm, where users are isolated document readers, in the collaborative browsing, they are just sharing the act of reading these documents. In such scenario, an annotation feature provides the ability to dynamically review the shared document. For example, while accessing online lecture materials, such annotations can help students to focus on the key points of a lecture. Also, it can enable students to engage in a discussion (Chong & Sakauchi, 2001a).
Annotation feature is supported in Clavardon (Sosign Interactif, 2008) and WebAn (Marshall & Brush, 2004). This is an important resource, allowing the participant to share information. However, the way it is implemented in such tools is not expressive enough since users are just allowed to highlight parts of texts. Note that co-browsing deals with web-pages, which are usually composed by texts, images and embedded media. In OCEAN, annotations are allowed over all the shared content. This feature makes the collaborative browsing more powerful as it reduces the need for additional collaboration tools, enabling participants to share contextualized comments on each other’s content. For instance, Figure 2 shows an example of an annotated web page (www.ufes.br), where a user has painted an ellipse, a rectangle, two arrows and has written one text note.

![Figure 2. Annotations Example](a) Original Web Page  
(b) Annotated Web Page

Another important communication facility is the negotiation support. Actually, negotiation can be viewed as a specific case of communication in collaborative systems, mainly related with the communication process necessary to take decisions in group. Such negotiation tools include invites for external users get an specific session, while the second (Privilege Request) is used for changing participant’s roles inside a session subgroup.

**CONCEPTUAL MODELING**

OCEAN’s development followed an ontology-based approach, similar to the ones proposed by (F. F. de Oliveira, 2009). The Collaboration Ontology (F. F. Oliveira, Antunes, & Guizzardi, 2007) has been a source for the analysis development phase of OCEAN. This domain ontology has been proposed preliminary in (F. F. Oliveira, Antunes, & Guizzardi, 2007), and is elaborated further in (F. F. de Oliveira, 2009). We build the OCEAN conceptual models upon it, by extending concepts and relations that are present in particular within the collaborative web browsing application domain (Gaaevic et al., 2006)(X. Wang, Chan, & Hamilton, 2002). The main benefits of such ontology-based domain modeling are (i) a domain ontology is supposed to be a strongly-axiomatized domain specification. As such, it is capable of restricting what can be said in specific applications within the ontology’s universe of discourse, (ii) domain (ontology) modeling can be an effective means for enabling the rapid prototyping of applications (Guarino, 1998), (iii) it has already been defined in accordance with the 3C-model.

Fragments of the Collaboration Ontology, containing concepts related to the OCEAN context, are shown on Figure 3. Since this ontology is based on the 3C-model, it is also distinguished on
coordination, cooperation and communication sub-ontologies, and its they are respectively presented on Figures 3(a), 3(b) and 3(c).

![Diagram of Collaboration Ontology](image)

**Figure 3. A Fragment of the Collaboration Ontology**

As aforementioned the Collaboration Ontology provides a conceptualization in an effort to cover the whole collaboration domain. Due to that, concepts, relations and attributes of the OCEAN conceptual models are mapped from concepts of this ontology. These concepts in most cases are specialized in order to achieve the specific application sub-domain needs. Such mappings are depicted by the concepts in gray in Figures 4(a), 4(b) and 4(c). The remain concepts (in white) remain to specialized characteristics of OCEAN’s conceptualization. An important concept, *Cobroswing Session* (a specialized concept of a *Collaboration Session*), that represents the event that participants interacts given theirs individuals *Contributions* in order to collaborate. Moreover, this cooperation can be recorded by outcomes realized by the *Participant Contributor*, model as the *Cobrowser Resource History*. In this way the main artifacts produced during a *Cobrowsing Session* are the pages and theirs *Contributions*. 
Figure 4. OCEAN Conceptual Model

DESIGN ISSUES

Distributed Architecture

Most co-browsing systems are based on a specialized proxy-server, that intermediates the transmission of all recommended web pages, handling the content to all session members, e.g.: (Maintainers, 2008b)(PageShare Technologies Inc., 2008)(Maintainers, 2008a) and (Maly, Zubair & Li, 2001). In this case, the proxy-based architecture can lead to a potential performance problem, since the proxy server can become a bottleneck due to the translation and redistribution of the whole shared web pages for all clients.

An alternative to the use of proxies is to enhancing a standard browser. This approach consists to insert embedded applications in a standard web browser. Common alternatives are using Java applets or specific browser extensions. Since users do not need a proxy to handle the web page content, they are independent to download shared web pages directly from the source (web servers). Therefore, users can leverage their own bandwidth, no longer depending on the connection to a specific intermediary proxy server.
So, in summary, comparing the performance of OCEAN application server with a common proxy-server, OCEAN has two main advantages. The first concerns server resources saving, as it must process a much smaller amount of data not having to translate any web content. Secondly, network connections consume less bandwidth, since OCEAN server only handles transmissions of small session events.

As a result, OCEAN was designed to be a lightweight system, dealing with three distributed entities, the application server, the client standard browser and the web servers which hold the co-browsed web pages. The application server acts as a synchronization point, dealing with a minimal workload to accomplish its task. On the other side, the client has greater responsibilities, in order to save server resources.

As showed in figure 5, the application server is essentially composed by the notification service. This service is responsible for handling all the created contributions, forwarding them to the concerned participants. Such contribution routing process is coordinated by the notification protocol, explained in the next section. In addition, the application server is composed by the session manager, that controls all co-browsing sessions in the system, managing for instance, the participants privileges and the state of all sub-groups in the session. At last, the co-browsing history repository is used for storing all the information about all co-browsing sessions, composing the co-browsing histories. Such histories are a persistent registry of the session contributions, important for maintaining a common workspace for all participants, even when working on separated co-browsing threads.

At the client side, the participant uses a standard browser accessing the OCEAN client application that in turn encapsulates the shared contents (web pages and other contributions). Due to standard browsers security constraints, this architecture relies on the browser extension compo-
component, working as a access bridge to shared content relevant information. Nevertheless, the browser extension is not a mandatory requirement for using OCEAN. For example, if a user does not want to install it, thus he simply could not act as a presenter. Even though, he/she still would be able to participate of a group as an attendee.

It is important to mention that, allowed by the use of the browser extension, OCEAN can share HTTP cookies. Cookies are parcels of text sent by a web server to a client (usually a browser) and then sent back unchanged by the client each time it accesses that server. They are usually used for authenticating, session tracking (state maintenance), and maintaining specific information about users, such as site preferences or the contents of their electronic shopping carts. In a co-browsing session such cookie sharing feature is valuable for maintaining all clients in the same state with the shared content’s web server, as if they were only one user. However, sharing cookies implies on a privacy discussion (Maintainers, s. d.). In this matter, OCEAN actually only warns the group presenter every time the application client is sharing his or her cookies. In future works we intend to offer some interactivity or manageability for this cookie sharing mechanism.

**Notification Protocol**

The messages handled by our system consist of small amount of data describing the supported informative contributions, which regards to browsing, annotation, negotiation and management actions. For example, a web browsing contribution is described by a message containing only the URL of browsed content. Hence, in a nutshell, the communication protocol consists of small messages distribution over the participants of a session. As the notification protocol is responsible for notifying, or publishing, some eligible consumers about produced contributions, it has been specified as a specialization of the Message Queues (Eugster et al., 2003) and Publish/Subscribe (Eugster et al., 2003) communication paradigms.

This protocol manages each informative contribution type in order to control how these contributions should behave in the system. Such classification is made considering (i) scope, the information could be published for the entire session or just inside a co-browsing thread; (ii) contribution type, different contributions could need a different publishing process; and (iii) participants privileges, for Instance only the presenter of a thread can publish a URL at this thread. Tables 1 and 2 present the configuration of the notification protocol.

**Table 1. Session Scope Privileges**

<table>
<thead>
<tr>
<th>Contribution Types</th>
<th>Publish Privilege</th>
<th>Subscribe Privilege</th>
</tr>
</thead>
<tbody>
<tr>
<td>Login</td>
<td>revoked</td>
<td>granted</td>
</tr>
<tr>
<td>Logout</td>
<td>granted</td>
<td>granted</td>
</tr>
<tr>
<td>Create Thread</td>
<td>granted</td>
<td>granted</td>
</tr>
<tr>
<td>Join Thread</td>
<td>granted</td>
<td>granted</td>
</tr>
<tr>
<td>Leave Thread</td>
<td>granted</td>
<td>granted</td>
</tr>
<tr>
<td>Join Invitation</td>
<td>granted</td>
<td>granted</td>
</tr>
</tbody>
</table>

**Table 2. Thread Scope Privileges**
Illustrating the notification process, Figure 6 presents a common scenario which involves the broadcasting of a navigation (URL) from the navigation leader to the attendees. The main events composing this scenario can be explained as follows: (1) the presenter notifies a new URL; (2) the presenter starts to download the web page content; (3) the OCEAN server notifies all the attendees with the URL; (4) each attendee downloads the web page content by their own; (5) finishing the content download, the client informs the system OCEAN application server that it is synchronized; (6) finally, this server redirects such acknowledgment messages to the leader, keeping him aware of the attendees’ synchronization state.

The notification protocol provides a lightweight approach for OCEAN. Most of the data traffic is composed of small messages, not overloading the application server. This way, all the downloaded web content is exclusively dependent on the participant’s network bandwidth, unlike proxy co-browsing solutions. Therefore, the OCEAN server does not become a navigation bottleneck for user downloads. Moreover, such synchronization process provides awareness information about users’ download state, without restricting their actions even during the asynchrony period.

**PROOF OF CONCEPT PROTOTYPE**

Aiming at demonstrating the feasibility of our proposal, a proof-of-concept software prototype for OCEAN has been developed. This implementation was built on top of the GWT (Google Web Toolkit), an open source framework for web applications development (Google Inc., 2008).

In order to create a co-browsing session, users must inform Name and an e-mail address at the login form. It is also possible to get in an existing session informing the session ID. Right after a successful user registration, a login informative contribution is published and the main co-brows-

<table>
<thead>
<tr>
<th>Contribution Types</th>
<th>Presenter Role Privileges</th>
<th>Attendee Role Privileges</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Publish</td>
<td>Subscribe</td>
</tr>
<tr>
<td>Browse</td>
<td>granted</td>
<td>revoked</td>
</tr>
<tr>
<td>Annotation</td>
<td>granted</td>
<td>revoked</td>
</tr>
<tr>
<td>Cookie</td>
<td>granted</td>
<td>revoked</td>
</tr>
<tr>
<td>Presenter Privilege Request</td>
<td>revoked</td>
<td>granted</td>
</tr>
<tr>
<td>Download Ack</td>
<td>revoked</td>
<td>granted</td>
</tr>
<tr>
<td>Presenter Token Pass</td>
<td>granted</td>
<td>revoked</td>
</tr>
<tr>
<td>History Revisit</td>
<td>granted</td>
<td>revoked</td>
</tr>
</tbody>
</table>

*Figure 6. URL Broadcasting Scenario*
ing window (illustrated in Figure 7) is presented. From this window the user can access all the main features that he/she can perform in a co-browsing session. On the top part, there is the OCEAN Toolbar, while in the middle of the co-browsing window, there is the Content Panel, where the shared web contents are presented. The OCEAN Toolbar is organized in two groups of panels.

Figure 7. An overview of OCEAN user interface

At first there is the session scope toolbar. The Session Scope Toolbar holds the interaction mechanisms used for publishing contributions in the whole session scope. More important, a key feature of OCEAN proposal is presented in this toolbar, the tab controls. The group coordination primitives are available through the tab panel controls, located in the toolbar’s second line. In this approach, the selected tab stands for the thread group that the user is member at a given time. Whereas, the other unselected tabs stand for the remaining thread groups, available in the session. Thus, the act of selecting a different tab is interpreted as: the users is leaving the current thread group (leave thread primitive) and joining the thread group related to the selected tab (join thread primitive). At last, the create thread primitives accessible through the "plus" button, located next the last tab.

This tab-based implementation approach was chosen due to two main reasons. Firstly, there was an effort to provide an environment as most similar to a standard browser as possible. These standard browsers usually implements a browser tab facility4 aiming at offering for their users a nice way for organizing their parallel navigation contexts. In doing so, we hope that OCEAN users feel more comfortable when using our application. The second reason is related to providing an easier mechanism for coordinating users while they are participating in different co-browsing threads. Once a user has granted a privilege to join a thread group, any extra authorization request for performing this task is unnecessary. Hence, users can easily join a thread and check out what other groups are doing, without being annoyed with authorization requests and responses. As a remark, CoLab (Hoyos-Rivera et al., 2006), the OCEAN’s most similar co-browsing system, is based on authorization messages for coordinating such groups. However, the
CoLab coordination mechanism focuses on social commitments between participants pairs, having groups just as a consequence of such commitments (see Section 3.1.1).

Inside any tab, there is the thread scope toolbar. This toolbar gathers the interaction panels that are tightly related to the specific co-browsing thread the user is participation. Certainly, the most important is the Address Bar. Similar to any standard browser, this panel allows the presenter to input the URL he wants to navigate to. Also, it displays to all participants the URL of the current shared web page, in order to promote awareness. Moreover, this component is a combo box, and stores in its drop-down list, a history of all browsed URLs in this thread. Such history enables the presenter to easily revisit any co-browsed page.

The co-browsing thread toolbar also includes the list of participants of this group, the annotation toolbar with tools for drawing over the documents. Finally, there is also the history panel, containing what happened in this thread.

In accordance with the OCEAN architectural design, the system relies on a browser extension in order to allow OCEAN’s scripts to access relevant information from the shared content, when such content comes from a different domain. Due to the vast documentation support and available open source extensions, we have chosen to implement an extension for the Mozilla Firefox standard browser. This implemented extension is essentially composed by a set of scripts running in background, not demanding any user interference. So, the extension is meant to accomplish two main tasks: observing browsing events and managing shared cookies.

In summary, this software prototype is the direct result of our proposal. Its data models are implemented exactly following the ontology-based conceptual models, and based on our development experience, we could experiment that this domain modeling was an effective method for enabling the rapid prototyping of applications. Whereas its distributed architecture, service logic and business rules follow design level specific definitions, which have favor a lot our development effort through the use of the same standardized notification protocol as the basis of all prototype procedures.

**DISCUSSION**

The OCEAN proposal gathers in the same environment a set of features that together make an effective collaboration environment either in terms of quality of collaboration sessions or the application performance. Regarding the available facilities, OCEAN introduces a new way of collaborative web browsing, allowing users to share their browsing experiences in flexible and intuitive sessions.

In comparison with other co-browsing systems, the great advantage of OCEAN is that it was idealized considering the three aspects of collaboration domain, defined by the 3C-Model. Most of works in this area are commonly focused on just one of these aspects. For example, CWB (Esenther, 2002), CoLab(Hoyos-Rivera et al., 2006)(Hoyos-Rivera, 2005) and IMMEX (Gerosa et al., 2004) are coordination oriented, while Browzmi (Maintainers, 2008a) and PageShare (PageShare Technologies Inc., 2008) get focus on communication facilities, and at last, GUH (Aneiros & Estivill-Castro, 2005) and Kiobo (Kiobo, 2008) are designed for attending cooperation needs.
All these works propose interesting mechanisms in each of 3C aspects, however not enough for supporting a complete collaboration session.

An important contribution of this work is the proposed coordination model. The idea of forming groups inside the session, firstly proposed on Hoyos-Rivera et al (Hoyos-Rivera, 2005), turns the co-browsing sessions more dynamic and productive. However, OCEAN innovates organizing these groups as sets of users sharing browser tabs. This tab-based implementation approach was chosen due to two main reasons. Firstly, there was an effort to provide an environment as most similar to a standard browser as possible. These standard browsers usually implements a browser tab facility aiming at offering for their users a nice way for organizing their parallel navigation contexts. The second reason is related to providing an easier mechanism for coordinating users while they are participating in different co-browsing threads. Regarding other features, OCEAN also innovates with the usage of annotations in co-browsing sessions and also the management of session history, poorly explored by the most of compared systems.

In terms of performance, OCEAN’s architecture and the notification protocol were designed for not imposing great delays for the users, mainly avoiding the aforementioned problems of

Figure 8. Proportional Delay: a report of medium notification protocol delay and longest download times observed in experiments, that have focused on the effect of number of users raising over such measures.
proxy-based solutions. In order to verify the performance of our proposal, a set of experiments were conducted aiming at quantify such delay. In other words, we have measured how long OCEAN users spend within the notification protocol.

Figure 8(a) presents a proportional comparison between the observed downloading and delay. Although we can observe a variability of the delay, it tends to be proportionally lower as the number of users increases. A second bulk of experiments was conducted in a heterogeneous Internet scenario, with users distributed through worldwide locations (e.g. Brazil, USA and Germany). In this scenario, the users’ Internet connection bandwidth varies from 100Kbps to 10Mbps. Additionally, the user’s network path to the OCEAN server has around 15,10hops, with an average network delay to the OCEAN server of 230±150msec. Figure8(b) depicts the average results for this distributed scenario, in which the notification protocol delay had an average value of 559±163msec.

CONCLUSION

The main goal of this work can be summarized as: “Propose an environment that fits users needs for collaboratively browsing the web with arbitrary purposes”. We expect to have fulfilled this objective by proposing OCEAN. Its features were specified in an effort to cover all the collaboration aspects (cooperation, coordination and communication) in order to offer solutions for the different activities related to a general co-browsing teamwork. In addition to that, OCEAN’s design and implementation have considered performance (responsiveness and scalability) and usability (user interface) requirements. As a consequence, the obtained results prove OCEAN to fulfill “users needs”, given that it provides adequate tools (proposed features) with an efficient usage (design and implementation) for “collaboratively browsing the web with arbitrary purposes”.

Comparing with the most adopted proxy-based approach, OCEAN’s architecture avoids bottlenecks on the server-side, taking advantage of each participant’s bandwidth, in order to promote a better synchronous collaborative experience. Preliminary evaluation experiments showed satisfactory quantitative results on session synchronization. In the future, we intend to improve the evaluation procedure by including larger experiments and some qualitative methods in order to evaluate cooperation aspects and system usability.

Even more, the OCEAN conceptual model was built based on a Collaboration Ontology (F. F. de Oliveira, 2009), which saved time and improved quality in the system’s development process. Especially, this ontology-based conceptualization appears as an important contribution for the development of any groupware, since mirroring the Collaboration Ontology these systems could benefit of, for instance: avoiding mistakes in modeling decisions, rapid prototyping and easier interoperability with other collaborative systems. In face of that, we propose a first step towards a methodology for supporting the entire development process of groupware systems, where the first phase is using the Collaboration Ontology for conceptualization.

Several directions can be explored for future research to extend the contributions presented in this work, either overcoming imposed constraints during OCEAN’s development process, or extending its proposal and goals themselves. For instance, extending communication with audio calls certainly would lose quality if it were transmitted using the notification protocol. In this
matter, it becomes necessary to investigate the QoS (Quality of Service) requirements of each proposed feature, and when a new one is to be inserted, it must be investigated too. The first step of such possible evolution should lie on offering different service levels at the notification protocol itself, in the same way as in (Dyck et al., 2007), distinguishing communication channels by priority transmissions. Such options could preserve the benefits of notification protocol and the privilege grants.

Another point for future work is the evaluation process. In a mid-term research it is important to perform a study for evaluate how real users behave during these co-browsing sessions and how OCEAN helps them in this task.

In face of all obtained results we consider OCEAN a successful attempt on producing a comprehensive and objective collaborative web browsing environment. More important, this process has produced scientific and technological contributions to the CSCW research field.

REFERENCES


(Release: 1.4)


