Active Annotations of Web Pages

Jakob Hummes, Alain Karsenty, Bernard Merialdo
EURECOM

Abstract

The World Wide Web (WWW) incorporates a huge potential for collaborative working. This paper focuses on Web page annotations that can enhance group–working on the Web drastically.

A model of Web page annotations is introduced that defines relations between the users, the Web pages and their annotations. It is taken together with technological restrictions of existing approaches to form a taxonomy and an overview of the state of the art.

Our « active annotations » approach results from the observation that in current implementations the temporal relations are left out. Active annotations are focused on synchronous notification mechanisms within the Web and are implemented in the platform–independent language Java. Their purpose is to support collaborative work within small groups.

Keywords :WWW, CSCW, Web page annotation, model, taxonomy, active annotation, synchronous notification, awareness

Introduction

Since its beginning the World Wide Web (WWW) is a fast growing repository of all kinds of information. The Web offers an easy access to the information stored in Web pages and is supported by a new generation of graphical browsers that are available on most platforms. Web pages are written in the Hypertext Markup Language (HTML), which provides linking between pages and a rich set of presentation information and allows to include different media formats.

CSCW researcher are currently paying great attention to the Web, because the vast majority of users already know the WWW and use it for their work. The graphical browsers become more and more a uniform interface for information access. Today, the Web already supports collaboration through information sharing and retrieval.

However, the capabilities for using the Web for cooperation are limited. The information flow on the Web is unidirectional ; Web pages are in their current form read–only data presentation. Feedback and discussions about the contents take place outside the Web as in email or Usenet News. This lack of integration leads to a lack of

close relation between the Web pages and potential comments. Users in the Web are also not aware about others and thus cannot easily get in touch to share experiences.

Due to the enormous size of the Web and its anarchic structure, it is difficult to find the information one is interested in. Although search engines help the users, a query often reports a huge numbers of links, whereas only a few are from interest.

Annotations of Web pages address these problems. Annotations can be used for feedback, provide interesting links to other pages inserted by different users and may act as rating mechanism to overcome the information overflow. However, maybe they are most valuable for groups to provide a platform independent collaboration facility.

In small collaborative work–groups, annotations may be used to improve the quality of documents cooperatively. They can be used to discuss the contents of documents, but also to point to related work, or to submit a change proposal to a specific phrase within the document, which is accessible via the WWW.

In recent work, Web page annotations are defined differently. In this paper, we define :*An annotation of a Web page is any object, which is displayed within or accessible from the original Web page by accessing the original.* This definition is viewer–centric and does not define, how annotations are created, managed or stored. However, it excludes explicitly the view of an annotations as an object, which manipulates the original document by out–filtering some information ; our definition allows only to add information to a document. The definition does also not include shared bookmarks, because they are not visible by accessing a Web page.

After a motivation for using annotations for cooperative work, this paper will introduce a model to characterize annotation services by identifying relations between the addressed users, the Web pages and their annotations. Then a taxonomy of existing approaches will be given, which takes the model as basis, but includes also restrictions of the technologies that currently dominate the market. The model and taxonomy will lead to the conclusion that also temporal relations must be considered, that are addressed by the new approach of active annotations that couples annotations and user awareness tightly.

Use Case Scenarios

Before we will give a more formal model of annotations, we present some scenarios to highlight the usefulness of annotations for different purposes.

Annotations for more Democracy on the Web

Public annotations are seen as a way to introduce more democracy to the Web[1]. The roots of this approach lay in the Usenet (News), where all posts can be commented by other readers. There is no way to prevent my postings from being commented. If I will advertise in a posting a new product, which functionality is beaten by another product, it is likely that a follow-up will point to this other product. The same is true, if I am distributing political arguments ; they can be disproved by everyone. Newsreaders that support threading display the original postings and their follow-ups together, thus that everyone can follow the discussion. The Web does not offer such a functionality today ; viewing a Web page, there are no links to other opinions, if not explicitly designed by the owner.

Annotations for Group–Work

A group that jointly edits some publications exchanges drafts, comments on these drafts and pointers to related work. If they do not work at the same place and at the same time, they have to exchange their work asynchronously. Today, email is used as favorite exchanging mechanism for this purpose. An annotation service allows the group members to annotate each word, sequence of words, paragraphs or the entire Web page, and thus is supporting the group task very efficiently. The annotations can consist of corrections and comments, but also of a discussion between the members about the best form of presentation. Annotations contain links to refer to related Web pages. Small icons are associated with these annotations to show their types and creators. Discussions about the whole document are added as threaded links at the end of the page. Annotated annotations are also displayed as threaded discussions. If two members are viewing the same page at the same time, they are aware of each other and the discussion becomes synchronously.

Teleteaching

A laboratory course for computer science students is being held remotely. All documents for this course are accessible as Web pages. The students are working at the same time and can ask the tutors by using an AV connection outside the Web. A tutor has encountered some similar questions to a part of the documents and decide to annotate this part with an example. After she has finished the example, she submits her annotation, which immediately appears as icon in the annotated part to all viewers of this Web page. Students, who are viewing other parts or Web pages are not disturbed by this notification and will see it instead, when they are reaching the annotated part. The tutor starts also an agent, which incrementally adds more informations via annotations during the course.

Model for Web Page Annotations

To capture different facets of annotations services, we introduce a model to describe the objects and their methods. The objects are members of the sets of users, Web pages and annotations, while their methods can be described as relations between the objects. A graphical overview of the model is shown in Figure 1.

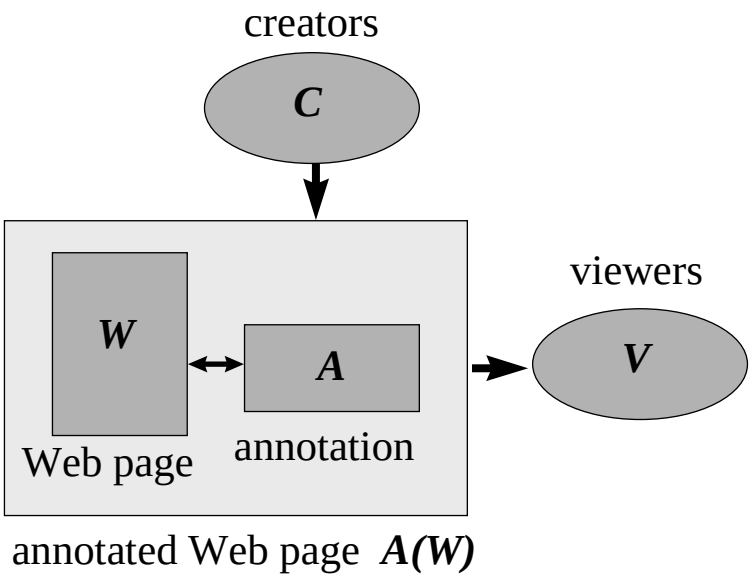

*Figure 1 : Graphical representation of a model for Web page annotations. Creators **C** add annotations **A** to a Web page W that are visible to viewers **V**. **A(W)** maps the annotations to the annotated Web page.*

The model is based on the observation that a set of users *C* may create annotations *A* for a Web page *W* ; other users *V* are allowed to view them. The pair *(A,W)* — annotations for a specific Web page — is tightly coupled thus that the abbreviation *A(W)* stands for an annotated Web page.

The following relations between the sets describe the structural behavior of Web page annotation services.

The users *C*, who may actually create annotations, and users *V*, who may view these annotations, define the **purpose** for annotations ; *C* is always a subset of *V* : $C \subseteq V$. If all users of the Web are allowed to annotate pages and view annotations, the annotations are called **public** ; we set $C=V$ and $|C|=|V|=n$ (n is the number of WWW users). **Private** annotations are defined through $C=V$ and $|C|=|V|=1$. **Group** annotations are defined through $C=V$ and the cardinality of the sets equals the group size, if the group is **closed**. If the annotations are visible also to non–members of the group, i.e. $|C|<|V|$, it is an **open** group. Other methods than create and view an annotation can be defined, e.g. for modifying and deleting them, and be added to this model through forming new relations. For simplicity we do not include them here.

The right to create an annotation is not only dependent from the rights of a user within a group, but also from the relation between a creator and the Web page, which is defined by the annotation service. Instead of having the right to annotate any page, it may allow only annotations for some special Web pages or for those that belong to the same group as the annotator. In our model this is expressed through the relation between the annotation creators, the annotation purpose, and the possibility to annotate a specific Web page ; we call it the **authorization**, which is given by the triple *(C,V,W)*.

The **granularity** defines a relation between a Web page and its annotations ; it denotes the finest object, which can be annotated. Examples are the whole **page**, words or characters of the **text**, **all**, and special objects defined by the annotation **system**.

The granularity is closely related to another relation between the annotated Web pages and their viewers : the **presentation** denotes, how annotations are presented to a viewer. The user may see them **in–line**, at the place they belong, either **directly** (i.e. the whole content) or **indirectly** as links. They can also be collected and displayed as **floating** annotations, e.g. at the bottom of the page. Annotations of annotations are **threaded**, if they appear in a logical arrangement.

The discussed sets and their relations until now include only the structural methods of an annotation service within the WWW. However, since annotating is an active process, there exists also dynamic methods between the objects, which form new relations between the identified sets.

Two relations deal explicitly with this constraint : The **lifecycle** denotes, how long an annotation is accessible, and the **notification** defines, who is when notified about a new annotation. The lifecycle is determined either **explicitly**, if the annotation must be removed manually, or **implicitly**, if it is controlled by the system. A notification is **synchronous**, if it is delivered immediately, otherwise **asynchronous**.

The purpose may change over time, e.g. private annotations may be made public. While the underlying sets for the authorization may change also over time and thus there will exist new methods dealing with keeping track of Web pages that may be annotated, the relation remains.

The relations and their associated methods are summarized in Table 1.

relation	structural methods	dynamic methods
(C,V)	purpose	transition between purposes
(C,V,W)	authorization	authorization
(A,W)=A(W)	granularity	lifecycle, « broken link problem »
(A(W),V)	presentation	notification

Table 1 : Relations and their respective methods in the model. The structural methods exist independently of the time, while the dynamic methods represent temporal relations.

Taxonomy of Existing Web Page Annotation Services

A vision for a general purpose Web annotation service would include to being able to annotate all objects of all formats on all accessible Web pages with the support of suitable formats for annotations and to set easily the access rights for the annotations. The annotations should also be creatable and viewable with all browsers and the annotation service should be at least as scalable as the WWW is.

However, this vision is not obtainable with existing browsers and servers ; it seems also unlikely that it can be developed without inventing new protocols or major changes to the existing HTTP protocol. Therefore, our taxonomy does not only compare the existing approaches on the basis of our model, but considers also technological issues.

To examine the technologies, a taxonomy of architectures for Web annotation services is introduced.

Architecture

The design issues determine the architecture for an annotation system. It must be decided, where the annotations are kept, how long an annotation is accessible and what is the largest group size one wants to support. If the group size is not limited, the architecture must ensure that the annotation service is at least as scalable as the WWW itself [2]. Each architecture has its own advantages and disadvantages that lead to restrictions for the user.

Storage

One of the major architectural decisions is, whether annotations are stored within the original, or being managed by a new service.

Annotations can be merged directly into and thus cause a modification of the original. That means they are stored within the original. This approach is only feasible, if each member of the group that may do annotations is known and considered trustworthy.

Managing the annotations by a separate server leads to the question, where and when they are merged into the original pages to display them together.

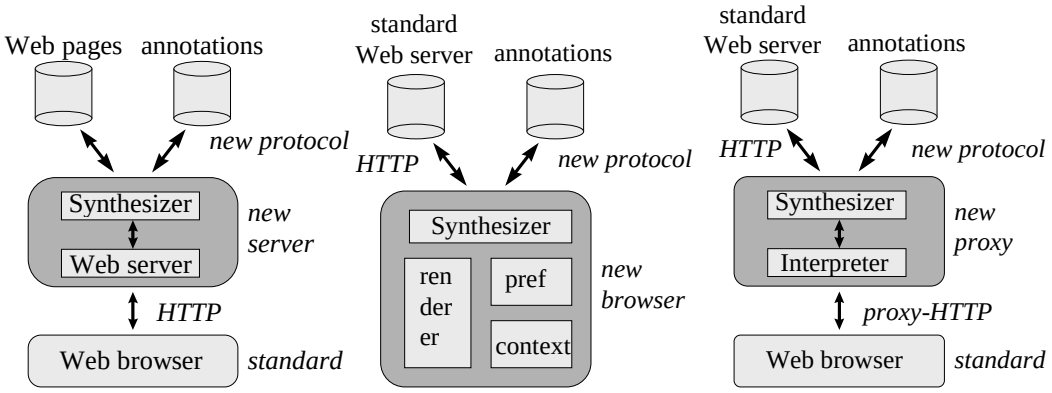

Figure 2 :Three different architectures to merge annotations with the viewed Web pages. The left architecture changes the web server, the middle architecture the browser to synthesize the annotations, which are kept in a separate database with the viewed pages. The proxy of the right architecture places a proxy in the HTTP stream that alters the original stream to combine it with the annotation ; the produced output is again conform with the HTTP protocol.

Three architectures have been proposed : Evolving the Web server thus that it can merge the Web pages with the related annotations and build an HTTP stream, which can be read by all standard browsers (s. Figure 2, left) ; enhancing a Web browser with the needed capabilities to synthesize the documents with the annotations and rendering them interactively (s. Figure 2, middle) ; and letting a proxy tap the HTTP stream from the server, synthesize it with the annotations and produce a new HTTP stream to the client (s. Figure 2, right).

If an annotation is accessible through the HTTP protocol it can be stored on an ordinary Web server. CGI scripts to handle annotations must be stored on the same server as the documents they have to access ; the same restriction holds for Java applets :

due to security restrictions by some browsers, applets are only allowed to call back a service that is located on the same server the applets are loaded from. These architectures must therefore keep its scripts and executables locally on the server.

Annotations can also be stored in a database ; then a new protocol must ensure that it is accessible by viewing the original. These architectures can keep the annotations remotely, i.e. their place is independent from the system's location.

Restrictions

The following list summarizes the restrictions that hold for specific architectures. Most restrictions depend from the current generation of browsers and servers for the Web.

- New protocol : If annotations are not stored on a Web server or can be accessed locally using the file access mechanisms of the operating system, they must be obtained from a different service via a new protocol. This new protocol must be handled by the application, which merges the annotations with the annotated Web pages.

- New server : This approach enhances a Web server with the functionality to synthesize Web pages with annotations. The architectural model is given in Figure 2, left.

- Server extended with standard extensions : Standard extensions to existing Web servers are CGI scripts or applications stored on the same location with the server, which are used to handle call backs (e.g. through forms or Java applets).

- New client : If the client is responsible to merge the Web page it gets via the HTTP protocol with its annotations, it must be customized, which leads to the architecture depicted in Figure 2, middle.

- Extended clients (e.g. plug–ins) : Some clients offer an interface to customize them (e.g. NCSA Mosaic or Netscape Navigator). However these extensions are in general platform and browser (and its version) dependent.

- State–of–the–art clients : Today's most used Web browsers — Netscape Navigator and Microsoft Internet Explorer — can send forms to a Web server (to execute a CGI script) and are Java enabled. Thus Java applets can be used for networking. However, due to their security management, these connections are restricted generally to connections between the client and the Web server, the applet is loaded from (cf. Netscape Navigator's security manager).

Existing Approaches, Prototypes, and Products

Various motivations let researchers create systems to support Web page annotation. In this section we will look closer on how annotations are supported by existing approaches.

System	Futplex	ComMentor	Hyper-G	OSF	Hypernews	CoNote	Active Annotations
Purpose	group	group(s)	group, all	group(s)	all	group	groups (all)
Authorization	group	all	Hyper-G	all	Hypernews	CoNote	AA
Granularity	HTML	plain text	Hyper-G	text	Hypernews	CoNote	text (word)
Presentation	in-line direct	in-line indirect	Hyper-G controlled	in-line indirect, floating	threaded indirect	in-line indirect	in-line indirect
Notification	no	no	no	no	no	no	synchronous
Lifecycle	explicit	?	?	explicit	explicit	explicit	selectable
Storage	within	local (server)	distributed (server)	remote	within	local	within
Restrictions	CGI	new browser	new server, browser	new proxy	CGI	CGI	Java, CGI, JavaScript
Description	collaborative workspace	value-add service	hypermedia information system	annotation, meta-information	discussion	teaching	tightly coupled CSCW, teaching

Table 2 : Taxonomy of existing systems. The purpose for annotations identifies, who may see them. Authorization demotes, who possesses the page, which can be annotated. The granularity defines the finest structure, which can be annotated ;system names mean that the original Web page has to be prepared to use within the system. The presentation denotes, where annotations appear within the annotated page. The lifecycle is explicit determined by the users or managed by the system. Storage denotes, where the annotations are stored relative to the system. Restrictions refer to, what is needed to install and to work with the system. The description highlights the primary goals that are addressed by the system.

Table 2 gives a taxonomy of the following systems that support annotations within the Web : Futplex [3] offers a collaborative environment for shared Web page editing ; ComMentor[4] integrates annotations that are designed for value–add services ; Hyper–G [5], [6] is a large–scale, multi–protocol, distributed, hypermedia information system that also offers an annotation service for users with native Hyper–G browsers ; the annotation support by the OSF [7] introduces a new proxy, which taps the HTTP stream to support all browsers and is designed for group collaboration ; Hypernews [8] brings threaded discussions like the UseNet to the WWW ; CoNote [9] addresses cooperative learning situations, where the courseware can be annotated to help the students ; our active annotations (see also the next section) are also designed to support remote teaching by directly informing a user about a new annotation, our second version addresses especially the needs of simultaneous collaborative work, as may be found in shared reviewing processes.

Figure 3 relates different architectures from selected annotation systems in the context of the potential number of annotation creators, their viewers and the reached Web pages.

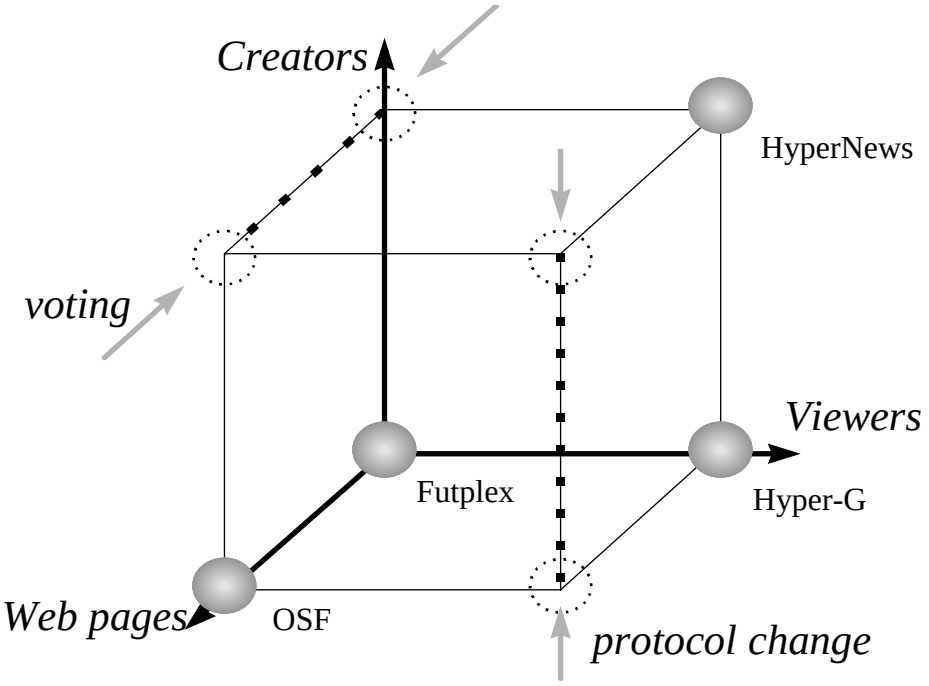

Figure 3 : 3D visualization of the target area of selected annotations systems The position relative to the number of potential annotation creators, their viewers and the annotatable Web pages are shown. Futplex is designed for groups and serve only local Web pages. Hyper-G lets only groups annotate, but the annotation are visible to all by its WWW gateway. Hypernews is open to all Web users, but only pages that are under its control may be annotated. The proxy by the research institute of OSF allows groups to annotate all Web pages. It si significant that four nodes are left empty. More annotation creators than viewers depicts more a voting scenario, while the possibility to create annotations for every Web page that are accessible by all Web users would require a major protocol change to the HTTP protocol to remain scalable.

Perhaps the system restrictions and purposes reflect best their usability for simply doing annotations. While almost all Web servers are capable to run CGI scripts and host Java applications, changes to the Web server itself are only usable for a small number of users. Thus the ComMentor system that requires changes to both, the server and the client, is from limited use. Although the Hyper–G servers are installed by different sites and offers a WWW gateway, annotations may be only made by using native Hyper–G browsers ; while the system supports well hypertext information services, the annotation service is only a small part of it.

Changes to the client are also critical, since most users are familiar with their favorite browser and do not want to change it ; plug–ins could be a solution, but they are platform dependent and have to support at least the major players in the browser market — Netscape Navigator and Microsoft Internet Explorer. The proxy approach of the OSF is the most general without changing the server or the client, also this approach can handle all accessible Web pages ; however it is not easy to install.

Active annotations focus on temporal relations and are designed for small groups working simultaneously thus that their implementation focuses on synchronous notification mechanisms. They need Java–enabled browsers and a server running on the same host as the Web server.

Other approaches that implement shared workspace systems on the Web are currently investigating annotation services. Examples are the BSCW project[10], [11] and the European research project Web4Groups[12], [13].

Active Annotations

Our approach focuses on the temporal relations in the introduced model. Active annotations differ from existing annotation approaches by including synchronous notification about new annotations within the annotation service. The user may choose that the currently viewed Web page is automatically updated, when a new annotation arrives. Thus the active annotation service forms also an awareness between the currently viewers of the Web page. As example, a small group of collaborative reviewers may see comments of the others immediately.

We combine the strengths of CGI programs to manipulate the documents on the server side, Java applets and applications to offer the awareness service, and JavaScript from Netscape to seamlessly include all components within the browser. The active annotations service allows users to annotate each word of an arbitrary HTML page that is read- and writeable by the server.

Design Goals

Our goal is to provide users with a facility to add and delete active annotations within their favorite Web–browser without needing to change their configuration or installing new helper applications. The development of the active annotation software should be platform independent, so we do not rely on system–dependent plug–ins.

After submitting an active annotation, it should be visible by all users, who are viewing the part of the Web page, which is annotated. Users, who are connected to the active annotation service, but are not viewing this page, should not be distracted by the arriving of a new annotation, but see it instead, when they will arrive to this part. The users should also able to choose, whether they want to see the updated page immediately or if they want to be only informed that the content may have changed, and reload it manually.

The active annotation service is designed for small to medium groups. While the design concentrates on synchronous notification about new annotations within the system, it is also useful for asynchronous annotations, e.g. when the group members are not viewing the page at the same time.

Presentation to the User

To support an intuitive user interface, the browser window is divided in three frames. Figure 4 shows a screenshot of the active annotation service.

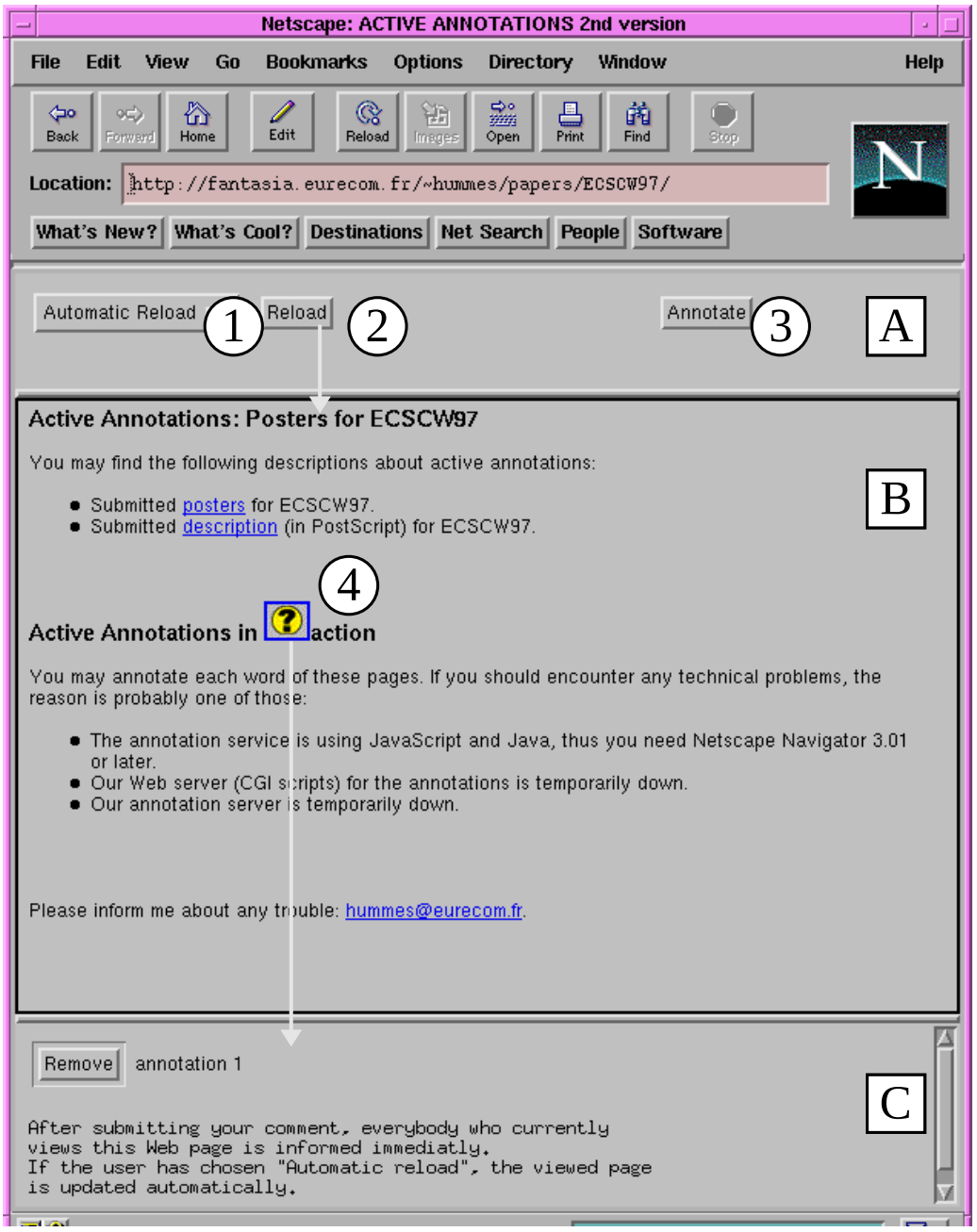

Figure 4 :Screenshot of the active annotation service. The Web page is divided in three frames ; A is the control frame of the annotations service and holds the controls for the selection of the notification strategy (1), a reload button (2), and the annotate button (3). Frame B incorporates the Web page that may be annotated. Annotations are displayed in frame C, when the user presses an annotation link (4) in the Web page.

Active annotations are included in a control frame (A) on the top, which stays during the visiting of the Web pages that are displayed in the main frame (B); the annotation frame (C) shows a selected annotation.

The control frame consists of a selection of the reload strategy (1), a button to reload the main frame (2), and the button to actually annotate the HTML page that is currently viewed (3). It holds also the applet that maintains the connection with the active annotation server. This applet triggers the action, which is performed, when a new annotation arrives. Also other actions than those that are currently implemented

(as described below) may be included easily by modifying the JavaScript sources of the control frame.

Creating an Annotation

Figure 5 explains step-by-step how an new annotation is created. To create a new annotation, the user presses the “Annotate” button in the control frame (1). This action triggers a CGI script that converts each word of the viewed HTML page in the main frame into a link; this output is displayed in a new browser window. The user chooses a word to annotate by clicking on it (2). A form handles the input of the new annotation text (3). By submitting this form, an HTML page containing the annotation is created on the server and an image that is linked to this page is inserted in the original in front of the annotated word (4). A JavaScript function of the control frame is informed about the new annotation and calls the communication applet to send the information to the active annotation server that broadcasts it to all connected browsers.

All users are informed about the arrival of the new a annotation by an applet; the viewed page is updated automatically if the user has selected this strategy in the control frame. Figure 6 shows the different action that is performed by the system for

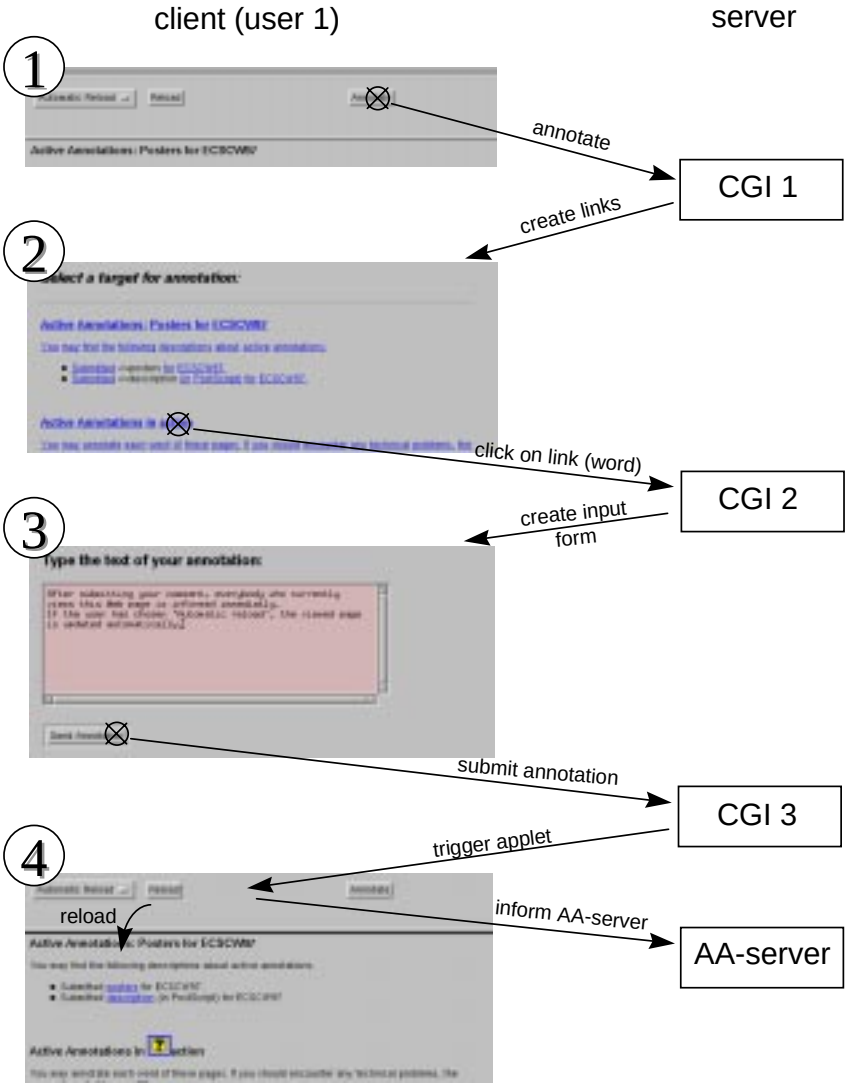

Figure 5 : Creating an annotation : step-by-step.

user 1, who has chosen to be only notified about new annotations, and user 2, whose page is automatically updated. The annotation may be viewed by clicking on its icon and is displayed in a separate third frame.

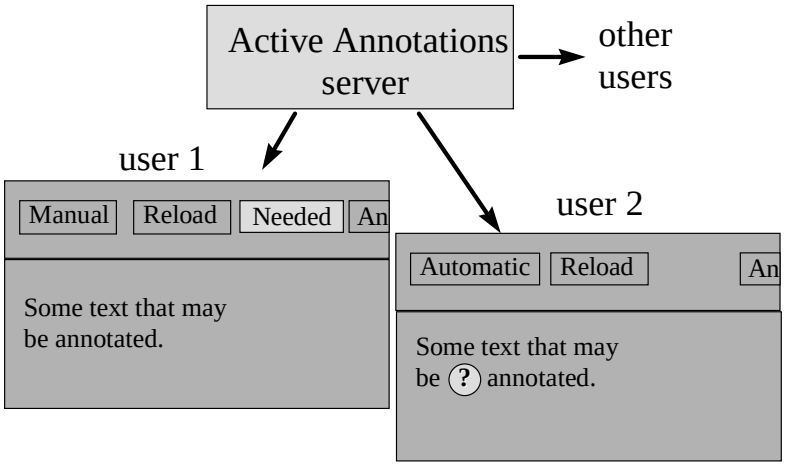

Figure 6 : User1 has chosen a manual reload strategy, while user2 wants to see updates immediately, after the server has broadcasted that a new annotation was created.

Architecture and Implementation

The HTTP protocol is a stateless protocol, which implements strictly the client–server principle : The client sends a request to the server and the server sends an answer back [14]. The server cannot initiate a transmission by itself. To not invent a new protocol, we have chosen to implement the needed peer–to–peer connection between the client and a server by Java applets, which are executed within the browsers and a server, which is implemented as Java application. Due to the implementation of the security managers in the browsers, the server must reside at the same place, where the Web pages are stored, which contain the Java applets to establish the connection.

The actice annotation service is designed for small to medium groups. However, we believe that access rights are orthogonal to the annotation service. The right to annotate depends therefore from the access rights to the Web pages and are not part of the annotation system itself. Thus the group for the active annotations is formed by all current viewers. Joining and leaving a group happen implicitly by selecting and leaving the pages that are under control of the annotation service.

The active annotation server implements a broadcast algorithm on top of TCP to communicate with the browsers. Each currently connected browser gets the same information over a primitive protocol; however the action taken can be configured.

The user may take any browser that supports Java and JavaScript to use the active annotation service. The applet maintaining the connection with the active annotations server is embedded in the HTML page for the control page (see Figure 7). No installation is required for the clients. An annotation is stored within the original HTML document. So, the active annotation service needs read and write rights for the pages that may be annotated.

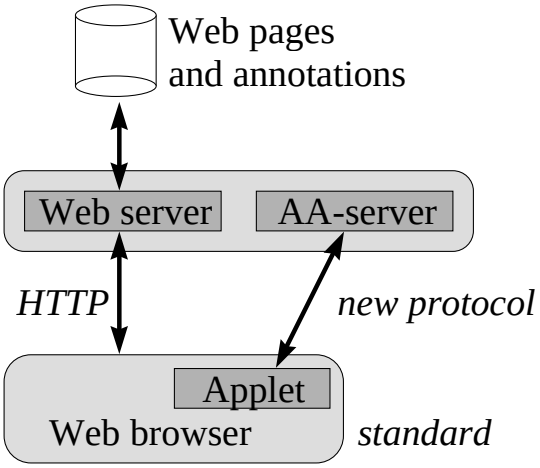

Figure 7 : Architecture of the active annotation service. The Web server is not altered, but accompanied by the active annotation server. The access to the server is handled by an applet thus that the browser does not need to be changed.

Ongoing Work

The current implementations use TCP connections between each applet and the server. Since a process may only maintain a small number of open connections at the same time, this restricts the number of clients. Using a multi-cast protocol instead of TCP should enlarge the potential number of simultaneous connected clients drastically.

The existing applets may be taken also to provide other behaviors. As example, a user could subscribe to different Web pages, to be informed of new annotations without rewriting the applets. Through the generic implementation the of active annotation service, only some JavaScript functions must be changed.

Conclusions

This paper introduced a model to describe annotation services on the basis of relations between the supported users, the Web pages and their annotations. The most important isolated structural relations were : purpose, authorization, granularity, and presentation. The temporal relations, lifecycle and notification, are not present in the existing annotation services, which were discussed to give an overview of related work. In the taxonomy of these systems, we included also their architectures. We pointed out the necessary changes of standard Web servers, protocols, and browsers, which were made to reach the functionality of the approaches. These changes lead to restrictions that are influencing the acceptance by the users.

We introduced a new service, called active annotations, which fills not only the missing temporal relations, but runs also within each Java–enabled browser. Active annotations are designed to support especially tightly coupled cooperative work at the same time, as it can be found in teaching environments and collaborative reviewing processes. The presented version allows Web page annotating on word level and sends a notification to all currently connected viewers. The users may choose between different notification and updating strategies.

The paper concentrated mainly on the needed infrastructure and the underlying architectures to create annotation systems. While very important, user-interface issues were only tangentially considered. Different icons to distinguish the type of annota-

tion or to identify the annotator could enhance each annotation system. Also access rights of the annotations and their respective Web pages were only mentioned, since these control mechanisms are orthogonal to the actual issue of annotating ; however an annotation system must be able to cooperate with an access control mechanism.

ACKNOWLEDGMENTS

The described work is part of the ACOST project, which is funded by the research institute CNET Lannion of France Telecom.

References

[1] **Glouberman, Misha**. 1996. *Adding Comments to the Web.* http://www.muchmusic.com/muchmusic/cyberfax/annot.html

[2] **LaLiberte, Daniel and Braverman, Alan**. 1995 (April). A Protocol for Scalable Group and Public Annotations, from *World Wide Web : technology, tools, and applications – Proceedings of the 3rd Int. WWW Conference.*, ed. Kroemker, D. Elsevier, Darmstadt, Germany. http://www.igd.fhg.de/www/www95/www95.html

[3] **Holtman, Koen**. 1996 (April). The Futplex System, from *CSCW and the Web – Proceedings of the 5th ERCIM/W4G Workshop. Arbeitspapiere der GMD 984.*, ed. Busbach, U. and Kerr, D. and Sikkel, K. GMD, Sankt Augustin, Germany. http://orgwis.gmd.de/projects/W4G/proc.html

[4] **Röscheisen, Martin and Mogensen, Christian and Winograd, Terry**. 1995 (April). Beyond Browsing : Shared Comments, SOAPs, Trails, and On–line Communities, from *World Wide Web : technology, tools, and applications – Proceedings of the 3rd Int. WWW Conference.*, ed. Kroemker, D. Elsevier, Darmstadt, Germany. http://www.igd.fhg.de/www/www95/www95.html

[5] **Kappe, Frank and Maurer, Hermann**. 1993 (June). *From Hypertext to Active Communication/Information Systems.* Technical report. IIG, Graz University of Technology. Graz, Austria. http://www.tu-graz.ac.at/

[6] **Andrews, Keith and Kappe, Frank and Maurer, Hermann**. 1995 (April). Serving Information to the Web with Hyper–G, from *World Wide Web : technology, tools, and applications – Proceedings of the 3rd Int. WWW Conference.*, ed. Kroemker, D. Elsevier, Darmstadt, Germany. http://www.igd.fhg.de/www/www95/www95.html

[7] **Schickler, Matthew A. and Mazer, Murray S. and Brooks, Charles**. 1996 (May). Pan–Browser support for annotations and other meta–information on the World Wide Web, from *Proceedings of the 5th Int. WWW Conference.* Elsevier, Paris, France. http://www5conf.inria.fr/

[8] Hypernews. http://union.ncsa.uiuc.edu/HyperNews/get/hypernews.html

[9] **Davis, James R. and Huttenlocher, Daniel P.** 1995. Shared Annotation for Cooperative Learning, from *Computer Support for Cooperative Learning–Proceedings.* http://dri.cornell.edu/pub/davis/Papers/Welcome.html

[10] **Bentley, Richard and Horstmann, Thilo and Sikkel, Klaas and Trevor, Jonathan**. 1995 (December). Supporting Collaborative Information Sharing with the World Wide Web : The BSCW Shared Workspace System, from *The Web revolution – Proceedings of the 4th Int. WWW Conference*. O'Reilly, Boston, MA. http://www.w3.org/pub/Conferences/WWW4

[11] *BSCW*. http://bscw.gmd.de

[12] **Alton–Scheidl, Roland**. 1996 (April). Web4Groups — A European initiative for setting up non–simultaneous group communication services for the WWW, from *CSCW and the Web — Proceedings of the 5th ERCIM/W4G Workshop. Arbeitspapiere der GMD 984.*, ed. Busbach, U. and Kerr, D. and Sikkel, K. GMD, Sankt Augustin, Germany. http://orgwis.gmd.de/projects/W4G/proc.html

[13] *Web4Groups*. http://www.soe.oeaw.ac.at/w4g/Web4Groups.html

[14] **Berners–Lee, Tim and Cailliau, Robert and Nielsen, Henrik Frystyk and Secret, Arthur**. 1994 (August). The world–wide web. *Communications of the ACM*, **37**(8), pages 76–82.