# WebBeholder: A Revolution in Tracking and Viewing Changes on The Web by Agent Community

Santi Saeyor  Mitsuru Ishizuka
Dept. of Information and Communication Engineering, Faculty of Engineering,
University of Tokyo. 7-3-1 Hongo, Bunkyo-ku, Tokyo 113-8656, JAPAN,
{santi,ishizuka}@miv.t.u-tokyo.ac.jp

**Abstract:** The WebBeholder is a cooperative agent community framework that provides open services on finding and displaying changes on the World Wide Web. Several agents and components in the community interact with one another to achieve the goals issued by users of the system. The system consists of a service provider agent that keeps watching and detecting changes on the Web, a number of personal mobile agents that represent each user, and a number of mediators to negotiate with the service provider agent for incoming personal agents. This paper describes the framework with an emphasis on evolution of the system, the interaction among agents and components, and our algorithm for generating comprehensive presentation of changes in structured context like HTML documents.

## 1. Introduction

The information in the WWW is supposed to be changed dynamically without any prior notification. Browsing through the sites for new updates is not only time consuming task but also vain in case that there is no change made on the sites once visited. We need some representatives to do such burdensome and tedious jobs for us. Furthermore, we would like to know when the changes occurred and how they look. That means not only tracking tools but notification and presentation issues are also taken into account.

Our research proposes this agent community framework in order to establish a more flexible and efficient approach to accomplish the changes detecting and displaying goals. The system features the flexibility and efficiency of using mobile information agents in constrained environments. The changes detection services in the community is provided in the way that the users can fully customize their agents to meet individual user model rather than posting all their preferences to be served by centralized server.

At the same time, the system focuses on presenting detected changes from the HTML Difference Engine. It implements our algorithm called Longest Common Tag Sequence (LOCTAGS) to determine meaningful changes in structured context like HTML document. This paper is divided into two main parts. The first main part is devoted to explain the evolution of the WebBeholder. The latter main part describes the formation and implementation of LOCTAGS algorithm.

## 2. Formation of the WebBeholder

A WebBeholder community is the community that consists of a service provider agent, a number of mediators, and a number of mobile agents that represent their users. The users customize their own agents to meet their preferences before dispatching them into the community. These agents are called personal agent. The WebBeholder community is designed to provide an environment in which various kinds of agents can interact with one another to achieve change detection and presentation on the Web.

### 2.1. Architecture

The environment of overall system for the WebBeholder community is shown in [Fig. 1]. The users of the community dispatch their own agents to the Mediator via the Internet. At the Mediator site, all personal agents are bound in a provided platform which the agents can execute their codes under a restricted control. There are three service modules within the Mediator. All service modules run independently. Each service module serves the personal agent in its own queue. The Request Broker is the module that negotiates with and posts the queries

to the Service Provider Agent for the personal agents.

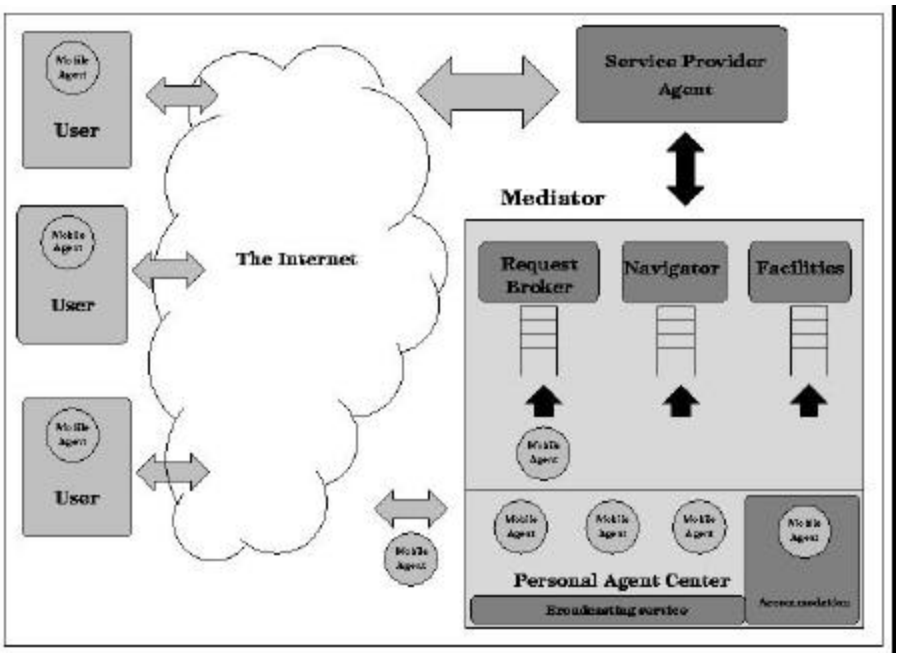


**Figure 1:** The WebBeholder Community.

The *Navigator* module tells the personal agents about locations of other WebBeholder communities. This service is provided in the case that personal agents could not find any information on the pages assigned by their users. The personal agent can query the Navigator to look further for some communities that have the desired information.

The *Facilities* module provides facilities for incoming personal agents. Since the mobile agents in the provided platform of the Mediator site have restricted access to the Internet and resource usage, the Facilities module offers these facilities under limited operations. The details of facilities are described in the topic Facilities in the Community

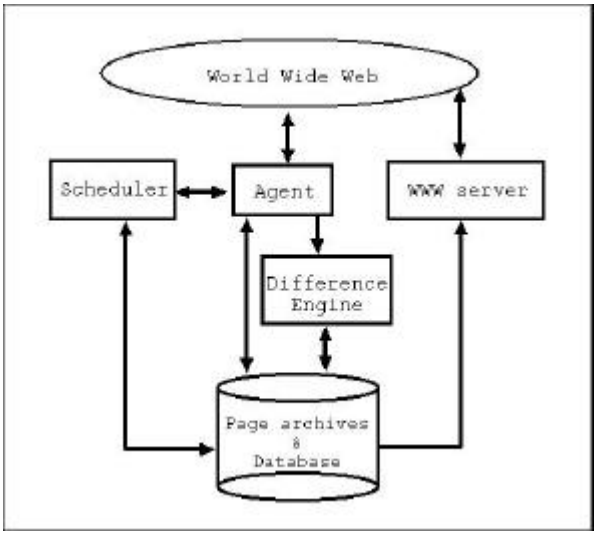


**Figure 2:** The building blocks of the service provider agent.

The main agent that offers services to the community is the *Service Provider Agent*. The architecture of the service provider agent is shown in [Fig. 2]. Its main modules can be listed as following:

- **Agent**: It is the heart of the service provider agent. It interacts with other modules in order to retrieve and

compare HTML documents.

- **Scheduler**: The scheduler will look up the pages registered for each user then makes a schedule of checking for the user. It constructs a timetable for the agent to make sure that each user will be served right in time.

- **Difference Engine**: The agent implement the Difference Engine in order to compare the content of updated pages and see whether there are significant changes in them. The old and new versions of HTML documents are compared by running the Difference Engine. The results from Difference Engine are very important for the agent to classify the changes. At the same time, it will summarize the updated information into another HTML document by innovative algorithm proposed in this research. The detail on Difference Engine is given in the Difference Engine section.
- **WWW server**: The page archives contain the old and new version of Web pages together with summary pages constructed by the HTML Difference Engine. When users are notified by their personal agents, they can view the changes with their browsers via the WWW server.

### 2.2. Facilities in the Community

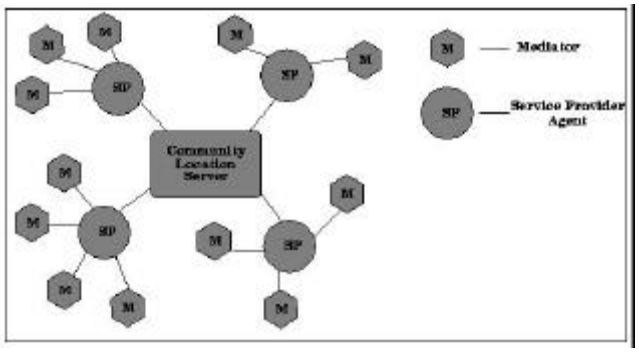


**Figure 3:** A number of WebBeholder Communities are linked together by a central Community Location Server.

The facilities in the community can be listed like the following:

- **Post Office**: The personal agents may have messages for their owners when they find something interesting or just for emergency cases. The message can be sent via the post office of the community.
- **Accommodation**: This provides accommodation for some personal agents that wait for some predictable events or could not go back to its user for a while.
- **Broadcasting service**: This facility allows broadcasting to all agents in the *Personal Agent Center*. This facility is also used to establish communication among personal agents.

### 2.3. Communication Among Communities

The WebBeholder communities are linked together as shown in [Fig. 3]. The *Community Location Server* is the center of all communities. Its holds the information about location of service provider agents, the Web pages they are responsible for, and their Mediator sites. The information may be asked from the Navigator modules in Mediator sites in order to dispatch some personal agents to where the desired information is already provided.

## 3. Difference Engine and Presentation of Updated Information

As the information retrieval module of the service provider agent gathers updated pages from the Web according to the schedule assigned by personal agents, the Difference Engine is activated to scrutinize updated information. The results from this investigation are divided into two categories. The first one is the result from the evaluation of updated information. The agent needs to know whether the updated parts in each page are significant enough to interest the user who posted the query. This result is also used to determine whether the

changes are worth informing to the user. The second category is the result for presentation. The Difference Engine produces a document page that shows the revision of the updated page so that the user can review and jump from change to change without difficulties.

The following sections describe the formation of the method for checking and displaying changes in arbitrary two revisions of a specific WWW page. We developed an algorithm called *Longest Common Tag Sequence (LOGTAGS)* to match tag sequences in old and new version of HTML documents. The algorithm was applied to help finding the right places for context comparison within a pair of HTML document. The differences are justified in a new HTML document conforms to its updated version's outlook in the way that the user can identify the differences at ease.

### 3.1. Longest Common Tag Sequence

HTML document consists of markup tags and context. The Difference Engine is designed to parse the HTML document based upon the basic that each tag is followed by context.

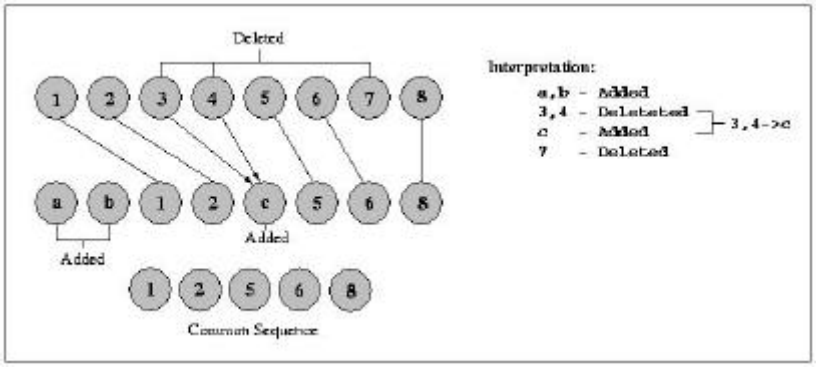


**Figure 4:** An example of presentation of updated sequence.

The merit of this method is that the HTML parser has no need to understand all the given tags described in HTML specification. In long run, the method is still valid for new markup tags introduced in later version or super set of HTML specification (to some extent, since the method is blind to the meaning of content-defining markup tags). However, the concept simplifies the parsing process remarkably and suits for processing large scale WWW pages comparison.

We applied the same concept of text comparing algorithm that implement the Longest Common Sequence (LCS) of characters in string. We view the HTML document as a string of markup tags and context. The algorithm treats the context and tag sequence separately but keeps the processing order in right sequence. The algorithm can compare the context to its pair at the right positions because the sequence of markup tags are checked and recognized. [Fig. 4] shows an example of interpretation based on the common sequence of both sequences.

### 3.2. HTML Difference Engine

The HTML Difference Engine was constructed to compare a pair of HTML documents. The output of the Difference Engine can be separated into two categories. The first one is the information of changes found when comparing. The second one is the HTML document that presents the changes. The code base of some links, images and JAVA applets are modified on the fly, so that we have no need to hold all images or applets' byte code locally in order to enable direct browsing when the user view the summary page. The longest common tag sequence algorithm is applied to construct the HTML Difference Engine, which is able to create smart presentation of changes detected in form of HTML document.

The architecture of the HTML Difference Engine is shown in [Fig. 5]. Old and new versions of Web pages are fed to the tag parser module in order to generate the tag sequence for Longest Common Tag Sequence Detector. Differentiator compares all tag streams in order to find additions, deletions, and corrections. The comparison process in the HTML Difference Engine detects the changes up to the level of one character. This

information will be used to generate the final HTML document that indicates the changes detected in form of merged presentation.

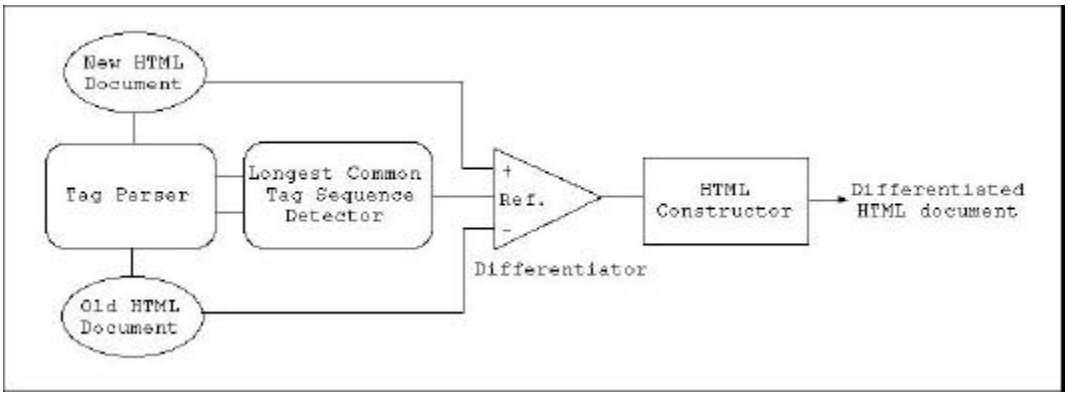


**Figure 5:** The building blocks of Longest Common Tag Sequence HTML Difference Engine.

### 3.3. Scoring of Changes

The service provider agent is also responsible for scoring the detected changes. The information is needed to determine significance of changes. The personal agent receives this information via request broker and performs evaluation base upon the preferences of its owner.

We use change scoring model to provide this information. The scores of change categories are listed in the [Tab. 1]. The scores are assigned to the categories according to their roles in HTML specification. The HTML Difference Engine accumulates the change scores associated with their categories. The total changes are considered significant if the score goes higher than a threshold value that is specified by each user.

**Table 1:** Change scoring for general HTML document

| Category | Score |
|---|---|
| URL in < A href=...> | 256 |
| Java Applet's Bytecode | 256 |
| Image in < IMG src=...> | 128 |
| Page's Title | 128 |
| Background Image | 64 |
| Background Color | 64 |
| Header < H1> ,< H2> | 64 |
| Header < H3> and smaller | 32 |
| Text (per character) | 1 |

## 4. Implementation

The agents and components in the WebBeholder are coded in Java. We implement the mobile agent package called *Aglet* that is provided by IBM Tokyo Research Lab. [Aglet 97].

The prototype of the WebBeholder has been tested locally in our laboratory. We have run two WebBeholder communities to serve some users that assign the service provider agents to keep eyes on approximately 200 pages on the Web. The users get notifications when change scores of observed pages become higher than thresholds provided for each page.

[Fig. 6] shows an example of change presentation. The deleted parts are displayed in stroked text. In the case that the deletion involves an URL link, a footprint icon is added to the tail of stroked text in order to indicate the deletion of the URL link. On the summary page, some implicit URLs that link to local pages on that site are modified during comparison process so that the users are able to click and surf the deleted link (if available). The deletion of a link does not imply the existence of that link on the World Wide Web. The addition parts are displayed in underlined bold text. In the same manner, a peg icon will be attached to the tail of any link that is

inserted to the Web page.

The LOCTAGS algorithm reveals its success in grouping common sequence of two HTML documents. The comparison is performed exactly where it should be done. Even the third row of the table in [Fig. 6] is completely deleted or a new row is inserted, the HTML Difference Engine knows how to group the common tag sequence and performs comparison correctly.


**Figure 6:** A result from HTML Difference Engine showing a presentation of changes in some elements of a table with the total change score of 949.

## 5. Conclusion

An alternative approach to detecting and displaying changes on the Web is proposed. The mobile agent community approach contributes its flexibility and efficiency to the system even in constrained environment. The approach enables open service system with less complexities and overheads. The LOCTAGS algorithm remarkably helps extraction of common tag sequence in a pair of HTML documents without complete knowledge of HTML's tag specification. As a result, the HTML Difference Engine which implements the LOCTAGS is able to present the changes in hierarchically structured text like HTML document correctly.

## 6. References

[Douglis 98] Fred Douglis, Thomas Ball, Yih-Farn Chen and Eleftherios Koutsofios (1998). The AT&T Internet Difference Engine: Tracking and viewing changes on the web, *World Wide Web, 1998* 1(1), 27-44.

[Aglet 97] Aglet-Workbench - Programming Mobile Agents in Java, *IBM Tokyo Research Lab.*, URL: http://www.trl.ibm.co.jp/aglets/.

[Bradshaw 97] Jeffrey M. Bradshaw (1997). *Software Agents*. AAAI Press/The MIT Press.

[Douglis 96] F. Douglis, T. Ball, Y. Chen, E. Koutsofios (1996). Webguide: Querying and Navigating Changes in Web Repositories, In Proceedings of the *Fifth International World Wide Web Conference*, Paris, France, May 1996, 1335-1344.

[Starr 96] Brian Starr, Mark S. Ackerman, Michael Pazzani (1996). Do-I-Care: A Collaborative Web Agent Proceeding of *ACM CHI'96*, April.