Artifact Centered Discouse支援する オンライン・ソフトウェアの設計と開発[1]

Toshiyuki Takeda
Center for Information and Media Studies
Kwansei Gakuin University
Uegahara 1-1-155
Nishinomiya, Hyogo, 6628501, JAPAN
takeda@kwansei.ac.jp

Daniel Suthers
Laboratory for Interactive Learning Technologies
Department of Information and Computer Sciences
University of Hawai'i at Manoa
1680 East West Road, POST 309
Honolulu, HI 96822, USA
1-808-956-3890
suthers@hawaii.edu

Abstract

PinkはArtifact Centered Discourseのコンセプトに基づき、文書やWebページなどの アーティファクトの一部または全体への注釈を維持しながら、スレッドをともなった議論を支援するためのシステムである。このシステムはWWWサーバーに付加的に提供されるサーバーサイドのスクリプトとして実装されており、標準的なWWWブラウザで利用することができる。本論文では、(1)インターネットのような分散環境における Artifact Centered Discourseの重要性 (2) システムのユーザーシナリオとインターフェース (3) 多様なアーティファクトをサポートについて論じる。

1. Introduction

In a temporally and spatially distributed environment such as the Internet, tools such as Email, Netnews and the WWW were created to facilitate the sharing of knowledge. However, these tools are sometimes less effective than direct communication such as face-to-face meetings because they don't facilitate context sharing. When participators talk about a document or web page (which we will call an artifact), email and other web-based tools are not always sufficient for such context sharing. Users only see the quoted part of the artifact directly in the usual quoting convention and it is too large to read when the whole document is attached to the message. Some implications of this deficiency of the tools for knowledge sharing are illustrated by the following examples.

With the expansion of the Internet, opportunities for security breaches by computer viruses have also expanded. Since system administrators cannot keep up with the new viruses that are always being developed, it is often the users who must collaborate to exchange knowledge on the vulnerabilities of networking devices and services. As Code Red and the Nimda Worm spread globally in 2001, the understanding of how to prevent such viruses was not sufficiently disseminated from system administrators to users. Although companies like Microsoft and CERT, as well as media like ZDNet, published a plethora of security information, many users did not take action. Myriad security advisories confused many users because the system environments they covered differed from those of the users. Here, the issue of context sharing comes into play, as many users cannot identify the context of the messages they read, or identify the portions of the messages that are relevant for their own context.

The seriousness of the situation is captured well by Microsoft's FAQ as follows: "Security patches are released to address specific security vulnerabilities. Many times, these vulnerabilities may not be applicable to your specific installation. You should carefully read each security bulletin to determine if the patch is applicable to your situation." However, even when users follow vendor instructions, problems may still arise. There is a pressing need for better tools with which users and system administrators can receive relevant information and obtain help in its application.

---

1 Designing and Implementing an Online Software Supporting Artifact Centered Discourse

Toshiyuki Takeda, Daniel Suthers

Kwansei Gakuin University, University of Hawaii at Manoa

Artifact-Centered Discourse

One solution to these problems is the creation of online tools to increase users' understanding of artifacts such as security advisories or open source code. These tools should enable users to share their knowledge about these artifacts. We call the type of discussion and argumentation that should be supported in the applications just described *Artifact Centered Discourse* [Suthers]. These discussions are also called *Anchored Discourse* [Guzdial] or *Contextualized Discussion* [TECFA].

Users need online tools that support discussions about artifacts (such as source code or security bulletins) by making it easy to refer to and annotate parts of the artifacts. This is not as simple as it sounds, since each user operates in a different system environment and has different knowledge. Some requirements of these tools can be considered as follows.

1. Discuss the contents of the artifacts. It is important to clarify and share assumptions, background knowledge and limits of applicability, which are not noted in the artifacts. Discussion in a shared context requires support of two functions:
   - Enable users who are reading and responding to a threaded discussion about an artifact to access and refer to the relevant passages of that artifact. Since artifacts such as security advisories are quite large, this function should highlight the portion of the document that is being discussed.
   - Enable users who are reading an artifact to access comments on a specific annotated region of an artifact. This allows users to share knowledge with someone who has the same interests.
2. Summarize and share as a new artifact the created knowledge that became clear to participants in a discussion.
3. Enable users of a variety of hardware and software platforms to access the community workspaces (including artifacts and discussions).

This paper describes the software system *Pink*, which was created to support Artifact Centered Discourse. Pink supports the understanding and creation of artifacts that reflect intellectual discussions among participants. Since Pink is client server software, it requires only a WWW browser for use.

2. Related Work

Previous research has shown that annotating text online enables participators to find relevant information more easily. Its central difference is that the documents provide a context for discussions.

The *Annotation Engine* [AnnotationEngine] is a kind of proxy server made by a set of Perl scripts and inspired by Crit Mediator. It is designed to help users read web pages with the use of annotations.

ComMentor [Roscheisen] is a research prototype for annotating web pages with a special customized browser.

CoNote [Davis] is a server side system that enables students to discuss homework assignments and web handouts. CoNote is a kind of proxy server that stores page contents. It provides a web-based front-end for annotations that can be anchored at pre-designated spots. CoNote annotations can only be placed in locations that the author of the document has designated as appropriate. Research has found that shared annotations of documents provide a richer communication forum than electronic media such as newsgroups, bulletin boards and email lists.

Cadiz [Cadiz] was a Microsoft Office 2000-based annotation system used by members of a development team (450 people created 9,000 shared annotations on about 1,250 documents over 10 months). Context based discussions are viewed as an improvement over previous work practices. Their case study is based on on-site experience.

CritLink Mediator [Yee] is an open source server side proxy software. A user enters a word or phrase to set an annotation point.

The *Journal of Interactive Media in Education* [JIME] is a polished and very successful system for collaborative and open peer review of journal articles. It associates discussion threads with articles and sections of articles in a frame-based web environment.

Kukakuka [SuthersXu] is under development in the same laboratory as Pink (the Laboratory for Interactive Learning Technologies at the University of Hawai'i at Manoa) and has many of the same objectives. A collection of Java servlets associates web pages with NNTP discussion groups and threads, presenting these together in a web client using frames. Kukakuka is being used to support discussions in a course on Human-Computer Interaction.

WikiWiki [WikiWikiWeb] is a collaborative software application, enabling web documents to be authored collectively using a simple markup scheme and without the content being reviewed prior to its acceptance. WikiWiki is not software for annotation, but users can make annotations to any document that is created in the system. Annotations are shown as embedded in the WikiWiki document.

Pink differs from the above software in several ways. Various kinds of documents can be used as artifacts in

Pink. Many of the systems described above support only annotations to web pages, whereas Pink supports not only web page annotation but also embedded annotation to documents, such as WikiWiki documents, created within the system. The current version of Pink does not contain a function for setting an annotation on part of a web page. This issue is described in the "Future Work" section. Pink also enables collaborative writing with WikiWiki functionality. Furthermore, users can make annotations to a real world document, such as a book or a journal article, by discussing excerpts posted by users.

3. A Sample User Scenario

This section presents a scenario of Pink in action. First we provide some necessary background knowledge concerning Pink's objects and user roles. Then we show how a manager would create a new workspace and how users would read and contribute messages in the context of artifacts.

User Objects

The user manipulates four types of objects. They are *Workspace*, *Artifact*, *Reference*, and *Note*. An *artifact* is the document under discussion. An annotation point can be attached to an artifact, and a *reference* connects an annotated point in the artifact with a comment chain composed by *notes*. Notes are threaded by following a comment chain. A *workspace* is created and opened for a certain goal, such as understanding the Nimda Internet Worm. Workspaces have artifacts, references, and notes as their constituents.

Three objects exist out of the workspace yet are dependent on this workspace. An event signal and relevant information is sent from the workspace to dependent objects if an object in the workspace is changed. The repository is essentially to Pink, as it provides a persistent storage for the workspace and its constituents. The Logger and Mailer are service objects for administration and management.

User Roles

The possible user roles in the present version of Pink are Administrator, Manager, and User. Administrator has the role of maintaining the entire Pink system, including user management. Manager and User are end user roles. A User can set an annotation to the artifact, make a new threaded discussion and append an argument to the threaded discussion. Since authentication and authorization functionality is not implemented in the present version, every user has Manager and User roles. As with a WikiWiki, all users have the same authority.

User Interface

One screen of the user interface is shown in Figure 1. On the left is an artifact, and on the red-highlighted "Concept Virus" is an annotated point (#1). Headers from the threaded discussion are displayed in the top right. The first header links to the front page of the group, and the remaining three the top-level headers (2, 3, 4) correspond to different documents (security bulletins) under discussion. The user is presently viewing document #4, CA-2001-26, and its threaded discussion. The thread corresponding to annotation point [1] is visible in this threaded discussion. The user is reading a message in the lower right frame. The following three subsections provide example user scenarios.

Managing a Workspace

This section describes how a manager creates and populates a workspace.

First, an administrator of Pink creates and opens a new workspace according to a manager's request. The workspace is created on the repository, and basic files and directories are generated. Access to the workspace of the manager and user is granted.

Then the manager opens the workspace for the first time, and a page called FrontPage is displayed. This document consists of two elements. One describes the workspace and its contents. Another is an empty list of artifacts in the workspace. When artifacts are added to the workspace, they will be listed with author name and other information such as creation date.

Next, the manager appends an artifact. The manager clicks a button on the FrontPage and moves to an administrator page. The following types of artifact can be used in the current version of Pink.

- Plain text: A plain text
- WikiWiki : An editable plain text with special tags to show the structure of this text
- Web page: A pointer to a certain web page.
- Book: Reference information about a book.
- Article: Reference information about a journal article.

The method for adding an artifact depends on its type and location. The manager either looks up a document that already exists in the repository; uploads a document from a client; writes a new document directly in Pink with the WikiWiki function; or points to an existing web page. The manager can also create a citation to a paper-based or other external artifact such as a book or journal article.

The next two sections show how users read and write in the workspace.

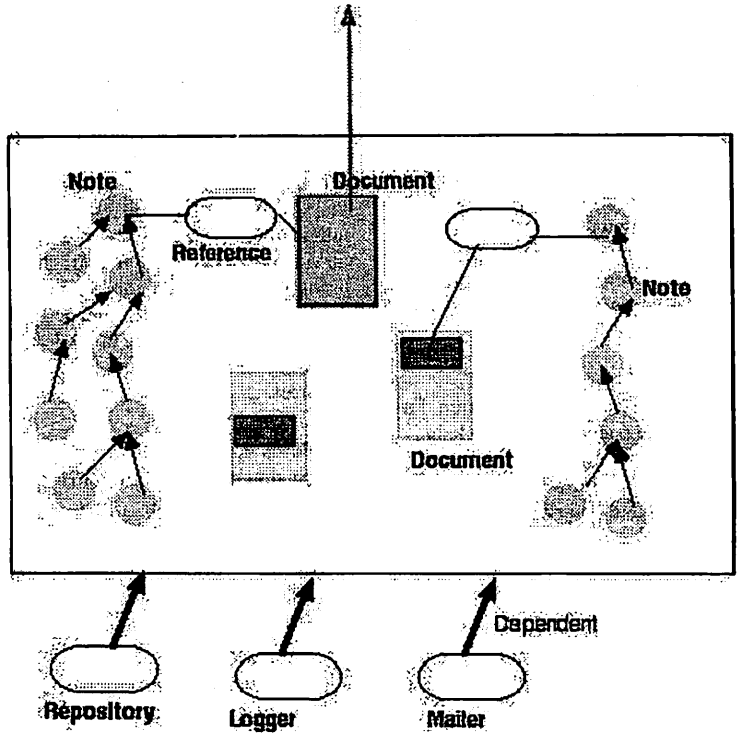

Figure 1. Objects of Pink System

Reading and Annotating Artifacts

When a user enters the workspace, the FrontPage is shown, listing both the artifacts and the threads in the workspace. Thus, a user can begin either by selecting and reading a document artifact or by selecting and reading topic threads of interest. References are displayed as links in both directions. When the user is reading a document and encounters an annotation, the user can simply click on the annotation's reference number in the artifact to display and read the related discussion. Conversely, a user who is reading a discussion can click on the numbered references (e.g., "[1]" in Figure 3) to view the referenced artifact. (The referenced artifact may be the document itself, or a link or citation to the artifact in the case of web pages or external documents, respectively.)

A user creates a reference to a portion of a plain text or WikiWiki artifact as follows. The user shifts to a page for inserting a special tag when the user clicks a button ("Annotate," Figure 2) in the artifact-browsing page. The user then inserts tags to indicate the extent of the reference, and clicks submit. A reference is then created in the repository and threaded discussion menu page.

Although an annotation point cannot be created for other artifacts such as books, journal articles, or external web pages, a similar effect can be achieved as follows. Pink provides a place to enter the relevant textual excerpt to be discussed. The excerpt is then stored in a plain text or WikiWiki artifact associated with the citation to the external document. This associated artifact can then be annotated in the manner already described as a proxy for the actual document.

Once a reference has been created, it shows up as a new thread in the discussion thread view (e.g., "CA-20011-26 Nimda Worm", Figure 3). The user then adds a Note to the Reference by clicking on "[Write Comment]," which loads a simple form for entering the Note.

Reading and Writing a Comment

The subject of the created Note is displayed in the discussion thread view, for example, "What does "concept virus" mean," Figure 3. Subsequently, users can reply to the Notes as in a normal threaded discussion.

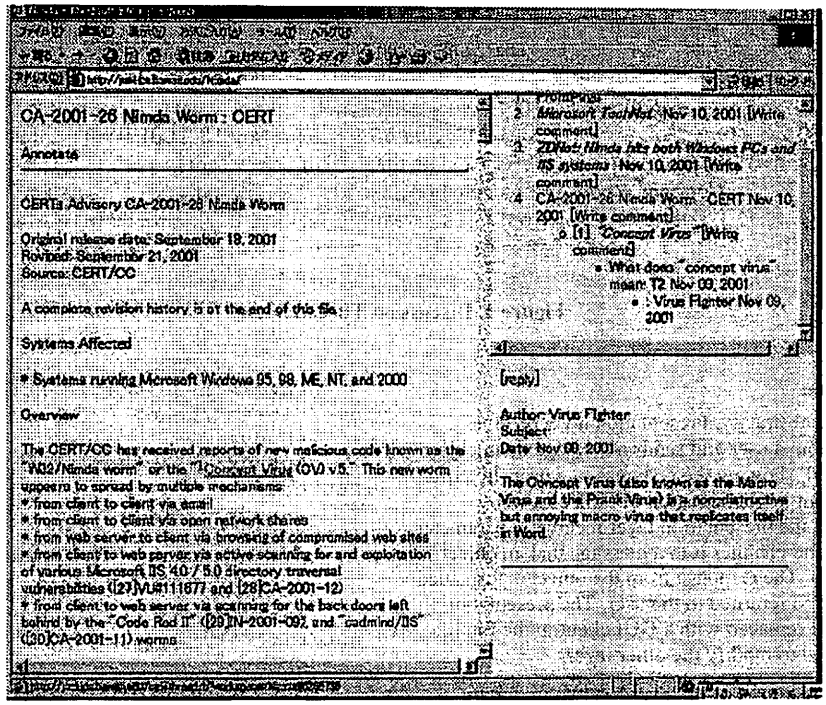

Figure 2. Sample Screen Shot

The Reference by which the markup was carried out is shown as an anchor in the document with the number of the reference. If the location is pointed at with a mouse, it will be highlighted. If a user clicks this part, the corresponding part in the threaded discussion menu is shown and a user can read notes attached to the annotation point. Thus, the user obtains relevant information without wading through irrelevant threads.

Conversely, if a user clicks the number of a reference on a discussion thread page, the part of the artifact that includes the annotation point is shown. Thus, the user can see the context of the annotation.

4. Software Design

Software Requirements

The current version of Pink is designed and implemented as a CGI script collaborating with a WWW server and is written in the Ruby programming language. Ruby is a pure object oriented script language that runs in an interpreter. No other special library on the web server is required for running Pink.

A W3C standard based web browser can be used as a Pink client. No special plug-in software is required to use Pink. Pink uses CSS and JavaScript for usability. However it is not required for the client browser.

A frame is used for the user interface of a Pink client. Although there are some usability issues associated with frames, frames enable artifacts and notes to be scrolled independently of each other while also being viewed simultaneously. This independence is important for sharing contexts in the Artifact Centered Discourse approach. However, browsers without frames, such as Lynx, can still access Pink, since it is possible to display and operate each frame (artifact, threaded discussion menu and note) as a separate page.

Architecture

Pink has 3-tier architecture consisting of Presentation, Model and Repository. Each part is exchangeable and extensible. Each object in Pink has a unique identification number, creation date, modified date and access date used for object management. Each object also has slots, such as "author" or "subject" for every object class. A certain object slot, such as "inreplyto," points to other objects.

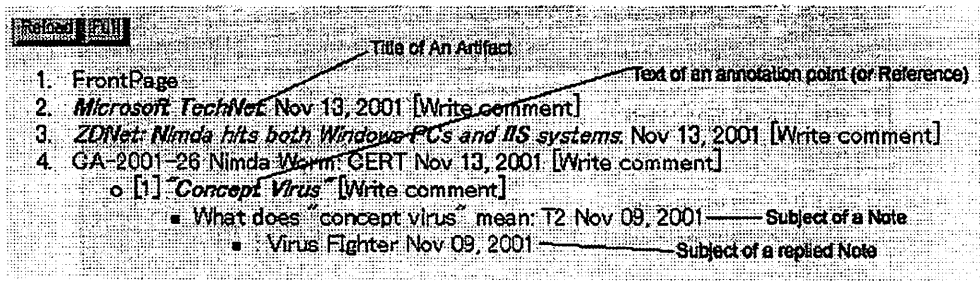

Figure 3. Discussion Thread View

Presentation

Considered abstractly, Presentation layer receives a directive from a user and sends a message to an object in the model, showing the response to a user. In terms of the current implementation of Pink as a web application, the presentation layer receives a parameter from CGI and changes it to a message that an object understands. The response from the object is formatted as HTML and returned to the user. The presentation layer can be replaced with a GUI client in the future without need to modify the other layers.

The Presentation layer has two important objects, WebComponent and WebTemplate. A WebComponent is an adapter object to connect a model and a client. A WebComponent wraps an object in the Model layer and sends it messages translated from CGI parameters received from clients. Before a WebComponet is evaluated, a WebTemplate is bound to it. The WebComponent generates the HTML to return to a client by evaluating messages contained in the WebTemplate. A WebTemplate is written in standard html format and can include another WebComponent. List 1 shows a sample WebTemplate for Note. Each element in #{} (such as "author" or "subject") is a message that can be understood by a Note object and evaluated with a Note object bound to a WebComponent. For example, List 2 shows the result of evaluating the WebTemplate of List 1 with author "Incident Team" and "Concept Virus."

```
<html><head><title>Note
view</title></head>
<body>
Author: #{author} Subject:
#{subject}
</body></html>
```

List 1. A sample of WebTemplate

```
<html><head><title>Note
view</title></head>
<body>
Author: Incident Team Subject:
Concept Virus
</body></html>
```

List 2. A sample of WebTemplate

In List 3, attribute class "WebComponent" is used. It is used in tag "span" or "div," and tells the system to create a WebComponent. A system binds this WebComponent and another WebTemplate specified by CGI parameter or system script. When a WebComponent in a WebTemplate is evaluated, the inner body part is shown. For example, List 4 shows a result when a dummy WebComponent associated a WebTemplate in List 3 and a WebComonent bound to a note with author "Incident Team" and "Concept Virus" is evaluated.

```
<html><head><title>Note
view</title></head>
<body>
<p>Reply to <span
class="WebComponent"
id="note"></span>
</p>
</body></html>
```

List 3. A WebTemplate including the other WebComponent

```
      <html><head><title>Note
view</title></head>
      <body>
      <p>Reply to <span
class="WebComponent" id="note">
      Author: Incident Team Subject:
Concept Virus</span>
      </p>
      </body></html>
```

List 4. A result of evaluation for WebTemplate including the other WebComponent

Repository

The Repository is a persistent database in which to store objects. It is designed to use simple interfaces like aspace-based repository, TupleSpace. [Gelernter]. In Pink, system processes perform simple operations to write new objects into a Repository, take objects from a Repository, or read (make a copy of) objects in a Repository.

Because the system has simple interfaces and the objects have a simple structure, it is easy to implement a Repository in many ways. In the current version, the Repository is implemented in two ways: Ruby's simple persistent object storage Pstore, and a plain text file. We plan a future repository implementation using a relational database and NNTP or IMAP for scalability and stability.

Model

The Model is a key component of the Pink system. Pink has flexibility and extensibility because every artifact is represented as an object in the Model layer implementing the abstract interface. A Model consists of a Workspace and Artifact, Note and Reference objects derived from the Workspace. There are also objects related to users and access control.

Workspace

The Workspace is a place to share and exchange knowledge by setting an annotation on an artifact and writing a note to a threaded discussion. A user can add or create an artifact, annotate it, and write a comment about the annotated part of an artifact.

Artifact

The Artifact object is used for representing a text document, web page, a book or journal article, etc. There are two kinds of Artifacts: structured and unstructured. The structured Artifact is composed of parts like W3C DOM. An annotation point can be set on each part of an Artifact. Sometimes a part is divided into small parts to set an annotation point. In DOM terms, when a user sets an annotation point, a newly created Reference object derived from "Element" is inserted into "Document" instead of the Element that is to be annotated. The annotated Element becomes a child node of the Reference node.

For unstructured artifacts, such as books or articles, it is impossible to use an annotation point directly. However, this is not a concern, because nearly the same effect can be obtained if the user enters text to cite into a reference object, where it can be seen in the threaded discussion menu.

Reference is a pointer object that refers to either a part or whole of an artifact object and is often a root point of a discussion thread.

A Note object includes a user's writings about something in a workspace and usually has a pointer to either a reference object or another note object. This pointer is called "inreplyto," a term from email and netnews. A Note object also has "references", a list of objects that reference the Note via their own "inreplyto."

A workspace knows what objects it has and can respond to a user query request, such as "all notes associated with this document." A workspace can generate and return a threaded discussion by gathering all Note objects that reply to reference objects associated with a certain artifact.

5. Future Work

The current version of Pink allows the user to set an annotation point on the entire external web page. Annotation of parts requires a filtering object that takes in an external page and changes it into a model object. This is a kind of proxy server function, the way CoNote and CritLink Mediator work. Pink already has similar functionality, but there are two reasons not to deploy this function as a service now.

One is the problem of intellectual property rights. Rewriting a certain page and showing this revised page to a user may not allowed in some cases, even though Pink just appends tags and doesn't change any contents of an artifact. For the Internet community to share knowledge, flexible usage should allow at least proper quotation.

Another reason is the problem of version management. The Web page used as a target artifact should embed a tag referred in the structure of artifact that is not changed. However, Internet documents on active sites do change, especially security advisories. The new version of the artifact may not include all the contents mentioned in the past discussion. When the system detects changes in an external web page or finds that a discussion thread is connected to obsolete content, the simplest approach would be to treat the old annotation and discussion as obsolete, and to only display non-obsolete and new annotations.

However, valuable knowledge may lie in these "obsolete" discussions. Consider for example a workspace recording design rationale for software revisions, or a learning application in which a student posts a document on a web server, an instructor comments on it, and the student then revises the document accordingly. The discussion for such

frequently changed sites needs to include older versions. A better approach is needed. Ideally, the system would enable users to browse the differences between old and new Artifacts, and provide access to the old discussion thread generated by the prior versions.

On the other hand, there are no problems for the contents created in the Workspace because the system can show the part changed in the artifact easily.

6. Summary and Conclusions

The Web has fostered explosive growth in a variety of online communities. Many of these communities would benefit from better tools for online discussions that are focused on the interpretation and/or creation of shared documents and other artifacts. Functional requirements of tools for artifact-centered discourse include the ability to move between discourse and artifacts in both directions: retrieving a discussion associated with an artifact or portion thereof; and bringing up an artifact that is referenced by a contribution to the discussion. This paper described the architecture and interface of Pink, a software system that meets these functional requirements. The system is based on an abstract 3-tier server architecture that is designed to be extensible and exchangeable. The implementation described in this paper uses CGI scripts in the Ruby object oriented language and standard web browser clients.

The significance of this work and similar work by others goes beyond its potential for improving artifact-centered discourse in online communities. It also represents a better approach to the design of web-based tools for collaboration, in which the design is driven by an understanding of the interactions to be supported.

References

[AnnotationEngine] (http://cyber.law.harvard.edu/projects/annotate.html)

[Cadiz] JJ Cadiz, Anoop Gupta, Jonathan Grudin. Using Web Annotations for Asynchronous Collaboration Around Documents. Proc. CSCW 2000, 309-318. 2000.

[Davis] James R Davis and Daniel P. Huttenlocher. Shared Annotation for Cooperative Learning. Third International World-Wide Web Conferenc. 1995

[Gelernter] David Gelernter. Generative Communication in Linda. ACM Transactions on Programming Languages and Systems, Vol. 7, No. 1, pp.80-112. 1985.

[Guzdial] Guzdial, M. (1997, December). Information ecology of collaborations in educational settings: Influence of tool. In Proceedings of the 2nd International Conference on Computer Supported Collaborative Learning (CSCL '97), (pp. 83-90). Toronto: University of Toronto.

[JIME] http://www-jime.open.ac.uk/index.html

[Roscheisen] Martin Roscheisen, Christian Mogensen and Terry Winograd. Beyond Browsing: Shared Comments, Soaps, Trails, and On-line Communities.

[Suthers] Suthers, D. Collaborative Representations: Supporting Face-to-Face and Online Knowledge-building Discourse. Proceedings of the 34th Hawai'i International Conference on the System Sciences (HICSS-34), January 3-6, 2001, Maui, Hawai'i (CD-ROM), Institute of Electrical and Electronics Engineers, Inc. (IEEE). 2001. (http://lilt.ics.hawaii.edu/lilt/papers/2001/Suthers-HICSS-2001.pdf)

[SuthersXu] Daniel Suthers and Jun Xu, "Kukakuka: An Online Environment for Artifact-Centered Discourse, Education Track of the Eleventh World Wide Web Conference (WWW 2002), Honolulu, May 7-11, 2002, pp.472-480

[Tecfa] TECFA newsletter http://tecfa.unrge.ch/

[W3C DOM] The World Wide Web Consortium. Document Object Model (DOM) Level 1 Specification. (http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001/) 1998.

[WikiWikiWeb] http://www.c2.com/cgi/wiki?WikiWikiWeb

[Yee] Ka-Ping Yee. CritLink: Better Hyperlinks for the WWW. Hypertext 98. 1998. (http://www.crit.org/~ping/ht98.html)