From the Writable Web to Global Editability

Angelo Di Iorio
University of Bologna
Mura Anteo Zamboni 7
40100 Bologna (Italy)
+390512094871
diiorio@cs.unibo.it

Fabio Vitali
University of Bologna
Mura Anteo Zamboni 7
40100 Bologna (Italy)
+390512094872
fabio@cs.unibo.it

ABSTRACT
The technical and competence requirements for writing content on the web is still one of the major factors that widens the gap between authors and readers. Although tools that support an easy approach to web writing, such as blogs and wikis, are becoming increasingly important and mainstream, they still lack in terms of layout and typographical sophistication, and, most importantly, only allow local editing (on the pages that are stored by the application itself). In this paper we re-propose an old paradigm for writing content on the net, directly derived from the Xanadu vision by Ted Nelson: *global editability* foresees that all documents on the web can be accessed for editing and modified on line, very much as in a global wiki. Global editability needs to address a number of issues, including correct support for intellectual property and legal issues, before it can be accepted as an idea. We provide some considerations on technical issues of global editability, and describe the architecture and implementation of a system, called IsaWiki, that is being developed at the University of Bologna.

Categories and Subject Descriptors
H.5.3 [Group and Organization Interfaces] Computer-supported cooperative work; Web-based Interaction

H.5.4 [Hypertext/Hypermedia] Architectures; User issues

General Terms
Design

Keywords
Web authoring, global editability, collaboration, customization, data collection.

1. INTRODUCTION
Surfing the web has never been as simple and fast: an Internet connection, a browser and few technical skills allow users to search and read a huge amount of information for their work and personal interests.

HT'05, September 6–9, 2005, Salzburg, Austria.

On the contrary, writing the web is not so straightforward and easy, and it still requires expertise and proper tools and write-permissions. Yet, a lot of different solutions exist for the creation of web pages, with a wide variety of complexity of use: from the advanced editors used by the professionals to the simple editors for unaware users, from the content management systems to the wikis and so on. On the surface, web authoring may seem completely addressed but a lot of issues still need to be solved [11], first of all the strong asymmetry between the roles of the author and the reader.

While a reader has no troubles to read content on the Web, in fact, an author needs to master specific technologies and tools and have write permissions on a server. Thus, the first steps towards a full overlapping of these roles are to address the simplification of the editing process.

Originally the web browsers could also be used to create new pages and links, but after the success of Mosaic, these editing capabilities were increasingly dismissed. A few years ago the W3C introduced Amaya [1], a browser/editor allowing users to directly modify web pages during the navigation, all within the browser. Such approach has some evident benefits for the users, since they do not need additional skills and tools for editing, and they do not have to switch between different applications.

Unfortunately, improvements to our daily web experiences, what we might call "web augmentation" do not simply happen through a simplification of the editing process: the most important innovation we strive for is the possibility of performing personal versions of other authors' pages and customizing all web content regardless of ownerships and write permissions. Although web users today can easily read web pages they still cannot add content to and customize other peoples' materials while surfing.

These concepts have been frequently examined in the field of open hypermedia in the 90s: many systems were proposed that allow readers to add links and annotations to a wide range of existing documents, integrating original content and external interventions [19][2]. Unfortunately the World Wide Web moved towards a different direction and many advanced hypertextual functionalities have been neglected in its design, such as private and public links, personalization, reuse of contents, on-line editing and so on[3]. However several proposals addressed the integration of hypermedia concepts into the World Wide Web, such as DLS [6], Webvise [16], and Arakne [5].

This integration cannot be considered the definitive step towards a sharable and global publishing environment. In that direction was aiming Xanadu [35], a universal environment where everyone accesses, reads, re-uses, modifies and comments any material of other users, tailoring it to his/her own purpose. Xanadu never

came to exist but its principles and features are still topical, and might become fashionable pretty soon also in mainstream Internet applications.

At first glance, a global editing environment could seem a shared workspace where all pages can be modified by all users in a chaotic and uncontrolled way. This is a scenario that can be avoided and that we want to avoid. On the contrary we think that customization and reuse of other peoples' materials can be the right moves towards an augmented web publishing environment, **if and only if** they are performed *in a controlled and safe way*, i.e., with good support for individual merits and authorship.

Global editability, thus, does not consist of *content hacking*, i.e. of modifying and substituting content in web pages regardless of ownerships and write permissions; rather it consists of creating new resources and personalize the existing ones, drawing contents and ideas from a knowledge base shared by all the web users.

This paper focuses on global editability as a new paradigm for the creation of web authoring applications. It also describes a system, called IsaWiki [12], that allows global editability within the browser aiming to profiting from the contribution of the ideas introduced in Xanadu. Global editability in IsaWiki is accompanied by a radical approach in differentiating content and presentation that allow global editability to happen only on the content, leaving presentation intact and untouched. This further helps in naturalness and integration of authorship and readership, as it situates the editing in the same environment of the browsing.

IsaWiki is the result of the conjunction of a number of existing ideas (wikis, blogs, Xanadu, external link bases and external comment bases) to a few ideas introduced in other systems and projects developed at the University of Bologna. In particular, it exploits the idea, already introduced in ISA [43] of shielding the user (both content authors and graphic designers) from details about the web technologies used in the system by letting him/her use the standard desktop tools he/she is already used to. Furthermore, it exploits the idea of the existence of a simpler document structures expressible with a simple markup language and to which all documents can be easily and automatically converted to, allowing us to introduce an intermediate format, called IML (IsaWiki Markup Language) that keeps structure and content but releases unneeded information about the presentation. Finally, it exploits the idea, introduced in the elISA system, that content parts of an HTML document can be automatically extracted from the presentation-oriented page layout through a rule-based system. This helps the browser-based editor to clearly identify the content parts of a web page, and focuses the user to modify only those parts of the document that are worth being modified, i.e., the content.

In section 2 we will discuss the benefits of our definition of global editability, comparing it with related proposals and works. In section 3 some technical and non-technical issues about global editability are discussed. Finally, section 4 describes the implementation details of IsaWiki, the system developed at the University of Bologna that takes these ideas to implementation.

2. GLOBAL EDITABILITY

We like to define "global editability" as the possibility for any web user to modify, customize and re-use any web content in a *controlled and safe* way. By this definition we want to exclude any anarchistic permission to modify and hack irresponsibly and without track or control all web pages. Several different approaches have been developed towards this goal in the current World Wide Web, in particular towards the transformation of all users from *passive readers* into *active writers*.

We can identify four different paradigms for global editability: writable web, external layered contributions, anthologies and collaborative editing.

2.1 Writable Web

With the term "writable web" we indicate the possibility for users to write web content with the same skills and tools used to read them. Weblogs [4] are tools for fast editing (mostly based on web forms) and fast publishing of personal diaries, while wikis [9] are collaborative tools for shared writing and browsing, allowing every reader to access and edit any page of the site through simple web forms and a very intuitive text-based syntax. Many wikis and weblogs are based on HTML forms. Some, on the other hand, provide a limited WYSIWYG editor for the specification of text. These tools allow what we might call *in-place editing*, where both the layout and the content of the page remain in the main browser window while editing, in much the same appearance as the final browsable page. This offers evident benefits to the users: no additional tool and skill to learn and no switching between different applications to disorientate them.

A particularly interesting proposal among the wikis is JotSpot [23]. JotSpot is a wiki providing all the basic wiki functionalities in a simpler way than the average wiki clone (for instance, it provides a true WYSIWYG editor). The pages are not unstructured text but the users can integrate advanced elements such as calendars, forms, tables from submitted data, polls and emails and so on. External web content can be integrated too, such as results from a search-engine or news from a newsgroup. JotSpot partially extends the wiki concept over the whole Web: external materials can be collected and integrated in the pages but the editability still involves only the local resources.

In a sense, thus, plain wikis and weblogs are "local": pages are designed and inserted in a special server-based system before they can be edited. Although all web users can edit them while browsing, the editability is limited to the local resources stored on the site hosting the wiki or the weblog.

Before arriving to true global editability we need to extend JotSpot approach and allow editability on all web pages. Put it in another way, we need to express content modifications as contributions in a tool for external layered contributions.

2.2 External layered contributions

We call "external layered contributions" the external annotations and links that users can add to web documents. Several projects have been developed in order to provide users with the possibility of annotating, criticizing and commenting on external resources, regardless of their locations, access permissions and ownership [5][22]. These systems are based on the same principle: any user can annotate any page, since these interventions are stored externally and, whenever the page is accessed, they are inserted on-the-fly onto the actual content of the page (that remains stored unmodified on the origin server). Arakne [5], iMarkup [22], XLink-based XLinkProxy [8] and W3C's Annotea [24] are examples of external annotation and/or linking systems, allowing users to add external contributions to any web page. The browser/editor Amaya [1] supports the Annotea protocol for the creation and distribution of external annotations.

We call these contributions *layered* since links and annotations are not inserted in the original copy of the document, but simply added on top of the original content: there is no deep intervention on the actual content of the page, no deletion, no change in the overall structure. A new content layer is added over the document, where all links and annotations belong.

Recently a number of tools for modification of third-party web pages have been implemented within the Mozilla Community [31]. For instance, Greasemonkey[15] is a Firefox extension that allows users to dynamically add DHTML scripts to any web page in order to change the page behavior and properties: these scripts can transform all the URLs of a page into clickable links, can add new content automatically retrieved from another page, can fix CSS bugs or remove unsupported HTML features, and so on. Greasemonkey scripts can be manually coded or automatically produced using Platypus [38], an extension of Firefox that allows users to modify pages directly within the browser through a WYSIWYG editor: fragments of content can be deleted, moved to other locations, formatted with a different style or added from scratch to any page. Those changes are saved locally as Greasemonkey scripts, which will repeat the same changes next time that page will be accessed by the same user.

Despite being an easy and immediate extension of its features, GreaseMonkey does not, and no web-based system that we know of does, allow the layered contribution of actual content, although this is the basic idea behind Ted Nelson's Xanadu [35], and it is not such a great additional insight to adding links and annotations.

In fact, a document whose content is layered in contributions from the original author and contributions from a number of other individuals customizing the text for their own purposes can be considered a simple example of an anthology.

2.3 Anthologies

New documents can be obtained by combining existing information collected from different resources and additional contents added by the author. These collections, which we might call "anthologies" (but they have been called also "harvestable data collections") would be composed of text fragments as well as comments, annotations and summaries written by the users. An infrastructure of composition links would glue these fragments into a single and unified documents.

In most cases the process for obtaining such anthologies is based on explicit operations performed manually through individual tools: *bookmarking*, *copy&paste* and *word-processing (BCW)*:

- Bookmarks are used to store and retrieve references to the interesting pages,
- copy&paste is used to select, extract and insert existing fragments into a new document and
- word-processing is used to merge personal comments and adjust pasted and new text in the final document.

The BCW method is cumbersome, technically intensive, and separates the content from the context information of where the content originates from.

In the pre-web hypertext literature several solutions were proposed in order to connect occurrences of the same information in different documents: disappeared but not forgotten have been the proposals for hot links, warm links and transclusions. Warm and hot links were defined (in [29] and [7]) as "live" connections between two end-points through which data can move from one document to the other: whenever one of the two end-points is modified the other is automatically updated. Similarly, transclusions [33] are the core mechanism of the Xanadu system [35]. In Xanadu a document is not stored as a whole memory or disk block but as a list of references to fragments combined into the final document, still connected to the original sources and automatically updated.

More recently, Hunter Gatherer [41] is a browser-based solution that allows users to collate information within web pages and organize them in structured and navigable collections. Hunter Gatherer is based on a special *copy&paste* operation similar to the Xanadu transclusions: the user surfs a page, selects a fragment and copies it to his current collection. The collection appears as a list of clickable links through which the user can move to the original document or display the fragment in a floating window. Hunter Gatherer manages the information collections as structured sets of fragments and links but does not allow users to re-use content in new documents.

MRS [29] is a management reporting system that allows users to harvest content from different sources and compose them into new documents. The application was designed to support report management in a firm where the roles are hierarchically structured: managers should be able use content from the reports produced by their subordinates as basis for their new reports. A server-side application manages the collections and a plug-in for Microsoft Word allows users to create and edit composite documents assembling results from the others' reports.

Anthologies are also a simple way to provide for an environment for asynchronous, emergent collaboration over the content of textual documents.

2.4 Emergent collaboration

Collaborative editing, i.e. the possibility of producing documents in collaboration, has been deeply studied by researchers and professionals: many advanced groupware applications have been proposed (see for instance [25]).

Most of the current groupware applications are based on the assumptions that the group working together on the shared resource is fully formed and defined at the onset of the collaboration. The philosophy of wikis [9], on the contrary, is that effective collaboration starts spontaneously by allowing anyone, including passersby, to add ideas and contributions to a discussion or a text document.

Both Xanadu [35] and RHYTHM [26] discussed about the merits of this approach. RHYTHM was inspired by Xanadu and provided many similar functionalities, including free linking, annotation, inclusion and document modification to all users and a complete scalability relatively to the number of users, documents and links.

Emergent collaborations [27] happen when users having reading rights but no formal connections to the authors of a document are allowed to suggest modifications and content integration over some documents, and these modifications are actually accepted by the authors of a document, thus allowing these proponents into the group of collaborators after the fact rather than before.

Global editability can easily integrate emergent collaboration with its basic functionalities. Documents are given a list of official authors (the group collaborating on the resource). Whenever an official author modifies the document he/she creates a new official version of the document. Any other user can still modify the

document (since we assume global editability) but only a personal variant is created.

In a xanalogical environments where versions and variants are composed of references to original fragments, each personal intervention and each reference to outside material is identifiable and distinguishable, yet clearly traceable back to the original source. No technical distinction exists between official versions and personal variants of some documents, but only in their names, status and access rights. If a personal variant is proposed to the official authors and approved to be included in the list of the official versions of the document, the integration of the changes in the document can be made straightforward.

A positive aspect of emergent collaboration is evident: unexpected and unpredictable interactions can spontaneously result and a document can be enriched with unexpected contributions. The philosophy of emergent collaboration, and in particular regarding the expected improvement in the quality of the documents, can be likened to the *open-source* philosophy [39], according to which several revisers and developers contribute towards the same task and everyone share his/her skills and knowledge with the others. While the process of the production of software requires an engineered and methodical approach, a more dynamic and unpredictable collaboration can be equally useful and suitable in the case of multi-authored text documents.

3. TECHNICAL AND NON-TECHNICAL ISSUES IN GLOBAL EDITABILITY

Complex technical and non-technical issues about the feasibility of global editability exist and need to be deeply investigated. In this section we will just underline some of them as we are foreseeing them, describing the open problems and proposing our solutions. In particular, five issues about global editability need to be clarified: (i) technical feasibility of global editability, (ii) intellectual property management, (iii) assets content and layout management, (iv) versioning and (v) scalability.

3.1 Technical feasibility of global editability

Today the WWW is increasingly becoming a place for supplying services and showing products and initiatives, rather than remaining a hypertextual web of connected content. Web pages are often complicated and full of graphical elements, since their main goal is catching users' attention and winning an audience over the competition. Thus, is it possible to revolutionize the current WWW architecture and twist behaviors and practices of millions of users towards global editability? Global editability cannot involve uprooting the current WWW structure and forcing an inadmissible revolution on a system so stable and widespread.

The only viable solution is to build *over the current web* an infrastructure to provide users all the required functionalities for publishing and collaboration. In particular we cannot:

- Suggest new transport protocols, a new architecture or new standards and languages, to replace the existing ones.
- Destroy the current WWW with all the services, e-commerce tools, graphical elements, lovely sites, collaborative tools, and so on.

On the contrary, we can and should:

- Exploit the existing technologies and protocol towards the creation of a new environment that coexist and augment the WWW without revolutions in what is already available.
- Create an environment addressed only to the interested users, without interfering with the navigation and the publishing processes of the rest of the WWW.

Not all web users are interested in global editability and many of them do not like to be disturbed during their navigation. For this approach to work, it needs to be shaped in the *opt-in* fashion of "good" email marketing, the one that makes sure that only interested users are contacted [36]. Therefore global editability needs to be designed as a service for registered users: subscribers surf on personal variants of the pages and can access all the contributions, while the others surf on the original pages as created by the original author.

Another key issue is providing a stable addressing base for layered contributions to work. Currently, dynamic web sites using obscure URLs for retrieving their content, rapidly evolving home pages, and lack of public archive access can easily disrupt the access and the flow of content in anthologies in a much worse and evident way than with the famed 404 File Not Found error. We need to find strategies for making base resources (those that are modified and edited in the global editing environment) be accessible reliably and continually. URNs [40] and PURL [37] are all proposed solutions for a problem that is clearly and heavily felt even outside of this specific application.

3.2 Intellectual property management

All web users can modify and customize all web content: has intellectual property management been neglected in our proposal for global editability? In a way, we believe that global editability does not ignore, but rather by-passes the problem: instead of preventing malicious and illegal reuse of others' people materials, it provides an infrastructure for monitoring who and when modifies content. Thus, users will be better able to control intellectual property and copyrights.

Currently the problem of copyright are dealt with by Digital Rights Management systems, tools to control and limit the use of digital materials. In [14] DRMSs are divided into two generations: the first-generation relies on security and encryption to solve copyright problems, the second-generation defines languages and standards to describe the information assets and define right holders relationships and protection rules. For instance, DOIs [42] are names used to identify and exchange intellectual properties, and ODRL [21] is a language to express rights information over content.

A particularly interesting approach is *transcopyright* [33] as proposed by Ted Nelson for Xanadu, which is conceptually inseparable from transclusions. A Xanadu document is a virtual list of fragments that are "transcluded" from the original source. The owner of each fragment can decide if the other users have to pay for the transclusion of the materials. Ideally any single byte of information can be priced so that a user customizing or re-using others' materials has to pay for it. Although the transcopyright philosophy in the current WWW is not practically admissible for several relevant reasons, the main principle underlying transcopyright, that is, the total traceability of the interventions, can in fact be adapted to the current architecture of the WWW.

The distinction between original content and personal variants could be considered as a barrier against malicious interventions and at the same time as a basis for the support of systems for innovative copyright control and management. The first required step, intrinsic in the global editability, is the complete traceability of the modifications and the external anchoring; mechanisms to

control, to prevent or to subject to payment the usage of some materials can be easily implemented on top of these data.

3.3 Assets, content and layout management

The global editability model presupposes an advanced and fine-grained management of documents' fragments: each document should be divisible into a set of assets that are storable, manageable and reusable in independent ways.

Xanadu addressed this issue exploiting a specific numbering schema called *tumblers* [34] and many other pre-web projects used their own addressing schemas. The situation has changed with the advent of the HTML and SGML and XML-based markup languages, since mechanisms based on offsets and byte-counting are now unsatisfactory. The two standards, XPath and XPointer, introduced to provide addressing and identification mechanisms for XML documents, would not be adequate to completely solve this problem. Two issues are still relevant: the presence of multiple data formats and the fusion (and indeed *confusion*) of content and layout in the WWW documents.

Most web pages are coded in HTML but a lot of other data formats are readable through a web browser, such as PDF, MS Word, SVG, XML and so on. In order to modify these documents and reuse significant fragments of them it is necessary to convert them from a data format to another and vice versa. The same operation is required if we aim at comparing two different documents in different formats or at including fragments from an external resource. Obviously proper tools can be used to view and modify content in a specific format, but this solution does not allow different sources to be really integrated with each other: each fragment is a black box only readable with a specific tool rather than a really convertible and reusable unit of content.

By implementing a *superior standard model* [13], that is, an intermediate generic data format with features from both source and destination formats, the automatic conversion of each format to and from this one becomes possible. A basic assumption lies beneath this idea: that any document in any data format has a representation in the generic standard model that is capture at least the **real content** of the document, that is, the most relevant part for reuse.

In fact, users customizing web pages are probably interested in the actual content of the pages, rather than their presentational aspects. Probably, when including fragments into a personal anthology or into a new document, they need to end up with a layout and the graphical elements that are appropriate to the new document, and not the source (as it would be in most cases for BCW tasks). On the other hand, the same asset management modules within the system need to handle content units rather than layout information. Thus, the generic data format needs to be designed to capture the actual semantic information within the document, ignoring presentation, navigation, and recurring parts of the page. This conversion must be done with the smallest loss of information and must extract (in case, by storing externally) all the details about the layout and the style.

This content and layout extraction is particularly interesting when applied to HTML pages, whose content and presentational elements are very often mixed together: a tool for the extraction of the layout information from a web page and the identification of the role of each fragment is strictly required in order to implement the global editability paradigm [45].

The advantages of a clear distinction between information and presentation are already known: first of all, the authors can deal only with the real content of a document and not with its decoration and style; secondly, the content production could be fast and easy [43]. This distinction allows users to organize, analyze, correct and modify the pages according to their semantic information too. Finally the same content could be represented with various layouts and displayed by different devices [17].

3.4 Versioning

In the field of hypermedia the advantages of versioning have already been described [44] [46]: historical revisions, security and explorative productions, distributed and asynchronous collaborations, emergent forms of collaborations, workflow support, efficiency and scalability have been used as justification.

In the global editability model, versioning surely is an extra feature providing services and information to the users but, also, it is a *necessary requirement*. An advanced versioning and *diff* engine is necessary to trace the contributions of the different authors of different sources, to outline differences between documents, to restore old changes, to revise modifications, to differentiate all the personal interventions and so on.

The simplest approach to providing these features is to implement a *session-based* mechanism for versioning and external anchoring. After an editing session, a new version of the document is sent to the central server the user registered to: if the user belongs to the list of official authors a new official version of the document is created and becomes immediately available, otherwise a new item is added to the tree of the author's personal variants. This also and most importantly allows the user to customize web pages that are not necessarily stored on the central servers.

Of course, although all the official versions and the personal variants are automatically versioned on the central server, the original web pages (still stored on the origin server) are not. How can these documents be integrated in the whole versioning system?

The problem does not occur in the case of page customization, since whenever a personal variant is created the system saves the entire document and so, even if the original page is modified, all the required information are still available on the central server. Starting from the first one all the subsequent variants are wholly versioned by the system.

For anthologies, where some fragments are imported from ordinary web pages, the problem needs to be addressed. If a user includes a fragment from a web page that will be later modified on the origin-server, all the connections and information about this page will be soon groundless. To solve this problem without losing efficiency and scalability, solutions based on caching mechanisms, data replication and consistency checks need to be addressed.

3.5 Scalability

Another important question concerns scalability. Highly used web pages could be modified and customized by several hundreds or thousands of users so that a huge number of personalized versions could have to be managed: so how can a system for global editability scale?

We think that global editability could be fully realized by a distributed and decentralized environment where many servers live together providing services for many users. Each user initially registers himself to a single server that will support him during the

navigation and the editing. All the operations regarding the documents personalization, editing and storage are managed by the server the user have registered to. Thus, the critical aspect is the number of the registered users (and consequently the amount of personal variants) on a particular server rather than the total number of existing versions or the number of users modifying the same web page. So the system could easily scale by increasing the number of the servers, by replicating resources and by exploiting techniques related to the distributed systems.

In conclusion, in this section we have identified some relevant requirements of the global editability model. To summarize, a publishing environment based on such approach has to:

1. Coexist with the current WWW, without revolutionizing its architecture and protocols
2. Not interfere with the navigation of uninterested users
3. Ignore sites, pages and page areas where content editability is inappropriate, pointless or impossible
4. Monitor all interventions, being able to trace who edited, when and what
5. Be extensible with copyright management tools, protocols and languages
6. Manage content assets with advanced mechanisms based on a clear distinction between content and layout
7. Be independent from the data formats
8. Support advanced versioning features
9. Be completly scalable

4. ISAWIKI

The requirements above are the cornerstones of the design of IsaWiki, a publishing environment which takes global editability to implementation. IsaWiki is a distributed system providing services to registered users. It is a simple client-side architecture based on the HTTP protocol. An IsaWiki server is a PHP application running on a plain web server, while an IsaWiki client is a plug-in for some common web browsers. Thus, the system naturally coexists with the architecture, the protocols, the languages and the tools of the current Web. All the possible interactions among the IsaWiki modules and the web clients and servers are summarized below:

- an IsaWiki server, firstly, is a plain web server delivering contents to any web client; on the other hand, it provides advanced services for customization and collaboration to the subscribed users only.
- an IsaWiki client surfs and downloads web resources; in parallel it communicates with an IsaWiki server and allows user to edit web pages within the browser.

A user interested in IsaWiki would simply install the plug-in and register himself onto an IsaWiki server. From then on, his/her editing and surfing activities would be supported by the system. Both the server and the client use an internal and structured representation of web data, based on a generic language called IML which plays a decisive role within the system.

4.1 IML: IsaWiki Markup Language

Web pages are not to be customized as a whole, but by separating content and presentation, segmenting the page into individually manageable assets of information. However the web pages are far from being so clearly structured: firstly, they are not always well-formed in either XHTML or a reasonably correct form of HTML; rather, all sorts of syntactical problems exist for these documents. Secondly, the actual content of a page is heavily intermixed with graphics and layouts, so that it is very hard to identify information within graphics, logos, footers, headers and so on.

Indeed global editability is the possibility of editing any web document in any data format and layout. IsaWiki converts any web document into a simplified one that expresses only the actual content of the original document and ignores presentational aspects. A limited set of constructs and elements is meant to gather the actual information of **any** page regardless of its storage format and formatting. We call the language composed of these elements IML, IsaWiki Markup Language.

Rather than being an extensive language where any possible element of a text is explicitly coded, IML is a generic language composed by few elements: blocks (e.g. paragraphs) containing text and inline elements (e.g. bold and italics), tables containing individual cells, collections of records (e.g. a drawing composed of many simple graphical elements in a vector-based graphic language), lists, and little else.

IML has an ambitious goal: separating the actual content and the presentational aspects of a document, in order to give users the possibility to edit and capture only the real information. The layout and the storage format are considered as extra layers which can be modified and substituted at any time, without impairing the real information of the document. For instance, a paragraph with style 'important' in MS Word, a fragment `<p class='important'>An important paragraph</p>` in HTML, and the fragment `<important>An important paragraph</important>` in XML, or an emphasized paragraph in PDF are all equivalent. The main principle of IML is just capturing the structural meaning of the elements in order to convert a document back and forth from one format (and layout) to the others, through the translation or better the *normalization* of the same document into IML.

Undoubtedly most documents are not natively conforming to the IsaWiki data format, so the first step performed by the system is just the transformation into this format of the pages. Obviously this transformation can lose some information but it has been just designed to minimize this loss and to keep any relevant content: exploiting the IsaWiki algorithms, any web document can be *normalized* into an IML document where content elements are clearly distinguished from the others. The following picture summarizes the conversion process within the system:

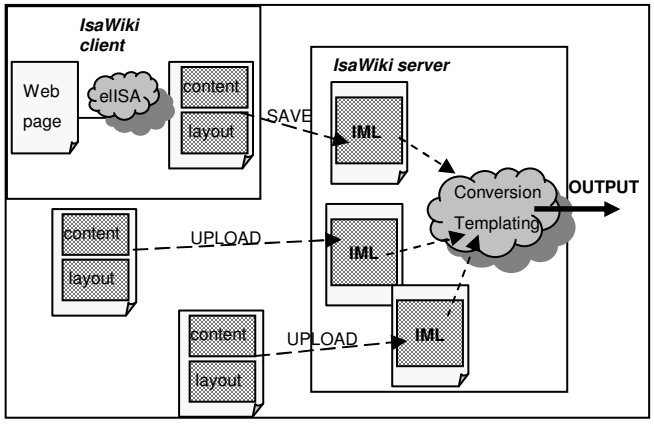

Figure 1. Content conversion through IML

In the IsaWiki client users edit and save the HTML pages within the browser, during the navigation; alternatively, documents can be uploaded via WebDAV, FTP, or direct file system access to the IsaWiki server. Thus, a user can create a MS Word document and manually save it on the server, after which any other entitled user can edit the same document in HTML (though the IsaWiki client) and, in turn, another one (in case, the original author) can re-edit the same page in .DOC or TeX, and so on. All in all, IsaWiki allows users to view and modify documents with their preferred tools, according to their preferences and needs. The independence from the input and output data formats is realized through IML.

Client-side, specific tools make any HTML page conformant to IML. Server-side, a similar task is performed by a module which extracts an IML representation from documents stored in different data formats. Starting from the output of this conversion into IML, the same document can be transformed into and from any supported data format and formatting. It worth to remark that the transformation is not a "literal translation" of the content, layout and graphics (as it happens in case of a printer driver) but a smart re-flow of the mere content of the document into a different format and layout.

4.2 A web browser integrated client

Some interface elements (sidebars, toolbars, menus and buttons) can be added to both major browsers of current generation, Internet Explorer and Mozilla, to provide users with advanced functionalities while browsing. For instance, a sidebar is a module residing within the browser that can monitor the URL loaded in the main window, associate an action to each event, start autonomous HTTP connections and modify the content of the page in the main window.

The IsaWiki client is a sidebar exploiting such functionalities and having a full control on the current page in the browser. It is developed for Internet Explorer 6.0 under Windows and for Mozilla and Firefox for Windows, Linux and Mac OsX. The following figure shows the IsaWiki interface for Internet Explorer during a browsing session:

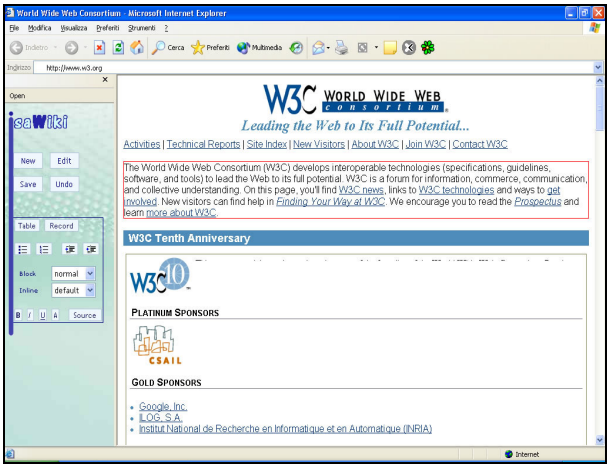

Figure 2. The IsaWiki sidebar

Two user's activities are helped by the sidebar:

- *browsing a modified page*: the sidebar verifies if any accessible user has previously customized this page and, in case, shows the latest available customized variants.
- *editing*: the sidebar includes an editor which allows users to directly modify any page in the main browser window.

Whenever a user accesses a web page, the sidebar catches the corresponding event and verifies (through a HTTP request to the IsaWiki server) if a personal variant of the current page exists. If the response is positive, a customized version of the document is displayed, instead of the original page from the origin server. The communications between the sidebar and the IsaWiki server and between the browser and the origin server are performed in parallel: this parallelism saves both time and user's patience with respect to a proxy architecture, as discussed in [8].

The content of the requested document is then modified according to the data received from the IsaWiki server (javascript and C++ functions are used to read and write the browser internal data in Internet Explorer, while Mozilla and Firefox use XUL [32] and javascript). The end result is then displayed in the browser.

The types of documents handled by the system can be divided into two main categories: (i) *local pages*, that are pages stored on an IsaWiki server, created and updated within the system and entirely versioned by the content manager, (ii) *external web pages*, that are ordinary web pages, not stored on an IsaWiki server and, obviously, not natively versioned. Both types of pages can be edited by IsaWiki registered users: depending on whether or not the user has write permissions on the document, a personal variant or a new official version is created. Thus, whenever a user surfs a web page, the page can be:

1. *an external web page never customized by any user*. The IsaWiki server notifies the sidebar that no personal variant exists, so the original page is displayed in the browser.
2. *an external web page already customized by some user*: the server sends information about all personal variants of the document. The latest one is loaded in the browser window and a list of variants is displayed for the user to access, including the original copy and the latest version on the origin-server.
3. *a local page never customized by this user*: the last official version of the document is displayed while a menu allows user to access any public version of the document.
4. *a local page already customized by this user*: the server sends information about all personal variants of the document and all public versions. The last personal variant is loaded in the browser and a list of versions and variants is displayed.

Thus, any web page can be edited and any intervention is clearly distinguished from the others: in fact, any version/variant is numbered and individually accessible and any customized revision is differentiated from the original external web pages.

Figure 3 shows a zoom on the editor buttons of the IsaWiki client for Internet Explorer:

Figure 3. The IsaWiki Editor

The editor for Mozilla and Firefox has been produced by customizing an extension called Mozile [30] (Mozilla InLine Editor), which is a WYSIWYG editor that allows users to modify the current page, directly within the browser. Mozile gives users full control on the structure and content of the page. Furthermore users can also edit only specific editable sections of the page, identified by a special CSS property called *–moz-user-modifiy*.

A separate step is required to identify the parts of a web page that contain actual content from the parts that provide decoration, navigation, or layout. For this reason, we have integrated in the IsaWiki client a tool for structural analysis, called elISA [45], that can identify the role and meaning of most parts of a web page, by studying structures, patterns and regularities in the HTML code.

Although elISA can identify the role of many page elements, in IsaWiki it is simply used to determine the areas containing the actual content. Thus, the output of elISA is the same page translated into IML, where the content areas are emphasized and labeled with the special properties mentioned above. Only these areas can be modified by the user, while the other areas remain visible in the browser window for contextualization and orientation of the user. Within the editable regions any content and structure can be freely modified, deleted or added by the user and the DOM is consequently and immediately updated according to these modifications. elISA normalizes the document into IML and the commands of the editor reflect the IML model, so that the editor produces a page wholly conforming to the IsaWiki generic data format.

After the editing session, the sidebar sends the modified document to the preset IsaWiki server, adding a few metadata such as the title of the document, the access list for reading and writing, the modification time, the author and so on, information used to manage, version and organize the documents on the server.

4.3 A multi-services server

Server-side, IsaWiki provides all the services to support the functionalities in the client-side interface and, in particular, it handles different versions, layouts and formats of the documents. The server is a PHP application running in an Apache HTTP server, tested under Linux, Windows 2000, and Mac OSX 10.2. The following picture schematizes the server architecture:

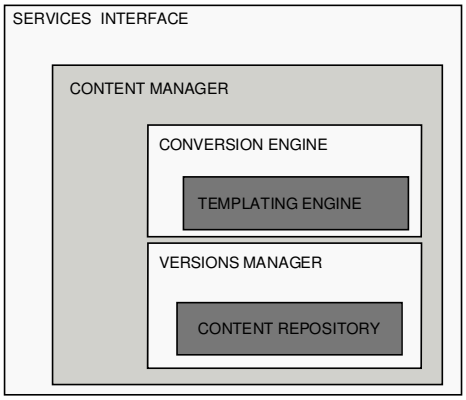

Figure 4. The IsaWiki server architecture.

Two main components compose the architecture of an IsaWiki server: *a service interface*, in charge of parsing and interpreting the clients' requests, and a *content manager*, in charge of versioning, converting and formatting data. The content manager, in turn, is composed of two modules: a *conversion engine*, that can transform documents from and to different data formats, and a *version manager* which handles personal variants and official versions of the documents stored on the server.

4.3.1 Services interface

IsaWiki is a distributed architecture based on the HTTP protocol. The most external layer of the architecture of the server is just a module which parses any HTTP request, extracts the required parameters and passes this data to the conversion and versioning engines. After receiving the output from these components the response is sent to the client. The requests handled by the server can be divided in five main categories:

- *retrieving a local page*: the resource is a public page (or a specific version) locally stored on the server and the request can come from a common WWW client or an IsaWiki client..
- *retrieving a customized variant*: the request comes from an IsaWiki client and carries information about the user, the document URL and the variant number.
- *saving a public version or a customized variant*: the request is an HTTP POST of the content of a page coming from an IsaWiki client (editor). Obviously metadata about the user and the content have to be extracted too.
- *retrieving versioning information*: the request comes from an IsaWiki client to obtain information about the whole versions or variants tree of a page.
- *other wiki-oriented services*: the request of a specific service to handle data on the server. These services include a simple search engine, a recent change document (where information about the most recently modified documents are listed), a configuration panel for the administrator of the server, a simple panel for users management.

Any request is interpreted by the services interface module and passed to the content manager integrated within the server, which wholly manages different versions (or variants) and data formats.

4.3.2 Version manager

IsaWiki implements a *session-based* versioning mechanism and external anchoring system. The content repository is divided in two areas: the local pages area, with a sub-directory for each local page to store the full tree of versions, and a customization area, where each registered user has his/her personal space. In this space, for each personalized page, there exists a subdirectory (in case, within further subdirectories mirroring the original web site) which contains the whole tree of variants. Any web page can be edited through the browser and any customized variant for any user can be added to the file system.

All users can potentially create personal variants of the same page, since each workspace and each version is clearly distinguished from the others. Furthermore, issues depending on concurrent editing are straightforwardly addressed, since the version numbers are directly computed when data are submitted (parallel editing sessions produce different versions at the same level of the tree).

A user can create a new version or variant also by uploading the new file directly on the server, via FTP, WebDAV or direct file access. An uploaded document, in fact, is automatically versioned by the version manager, regardless of its storage format. A daemon scans the file system, checks the consistence of the information, reads metadata of the uploaded file and decides its position in the versions trees.

Each variant or version of a document is entirely stored on the IsaWiki server and a *diff* tool integrated in the versions manager can be used to display differences between two versions. Any pair of consecutive versions of a document can be compared regardless of their storage format: they are first converted into IML by the conversion engine, and then the versions manager normally computes and displays the differences between the two files. Thus, also the identification of the changes between different versions is not performed on the actual documents, but rather on their translations into IML.

XanaWord [10], a previous work from which IsaWiki took inspiration, has already shown the potentiality and the feasibility of such a versioning system. XanaWord is a web publishing environment allowing users to edit web pages during the navigation, by using MS Word. The XanaWord system is based on versioning, external anchoring and a non-transparent HTTP proxy to provide the required service. When a document is requested, all changes introduced by each user (extracted through a forward delta based versioning engine that exploits the versioning functionalities of MS Word) are applied on-the-fly to the original document retrieved from the origin server. Client-side, all versioning features can be requested by the user though an *ad hoc* menu integrated in Internet Explorer. While XanaWord was designed to handle the internal *diff* information of MS Word (and worked fine with this class of documents), the same algorithms and principles about versioning have been adapted to IML and therefore to any data format supported by IsaWiki.

4.3.3 Converter and templating engine

IsaWiki addresses issues about the heterogeneity of data formats, layouts and graphical effects of the WWW through the introduction of IML. The converter is just the server-side module which realizes conversions between IML and the other data formats. As expected, it implements a superior standard model using IML as intermediary format: any document is first *normalized* into IML and then transformed in the destination format. In particular, currently IsaWiki supports bi-directional conversions to and from MS Word, Wiki, HTML, TeX, XML and (partially) PDF files.

A sub-module of the converter is in charge of adding presentational elements to the pure content of an IML document: the templating engine. The following picture summarizes the templating process within IsaWiki:

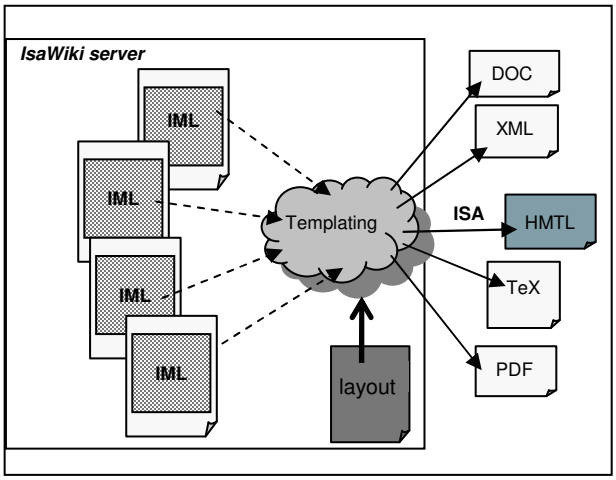

Figure 5. The templating process within IsaWiki

The HMTL templating engine of IsaWiki is directly imported from a previous project called ISA [43]. ISA is a web page production system based on the exploitation of standard desktop tools for the creation of content and layout and their automatic merging into the final web pages. In the scenario of producing a web site with ISA, a graphic designer creates the overall graphical aspect of the page using Fireworks or Photoshop. She/He will then use the slicing tool of the application to specify which area will contain which content. Independently, the content author can proceed to write the content documents through MS Word, using styles as instructed by the layout designer. After the layout has been created and the content document written and saved on the server, ISA is able to merge them to form a complete web page.

In the following pictures, a PDF document is shown in its original form and converted into HTML via the intermediary IML reduction and the re-flowing into a different layout (by ISA). Ideally any content could be re-flowed in any layout and can *move* from a page to another, according to the idea that a document is composed of content and, separately, layout and style.

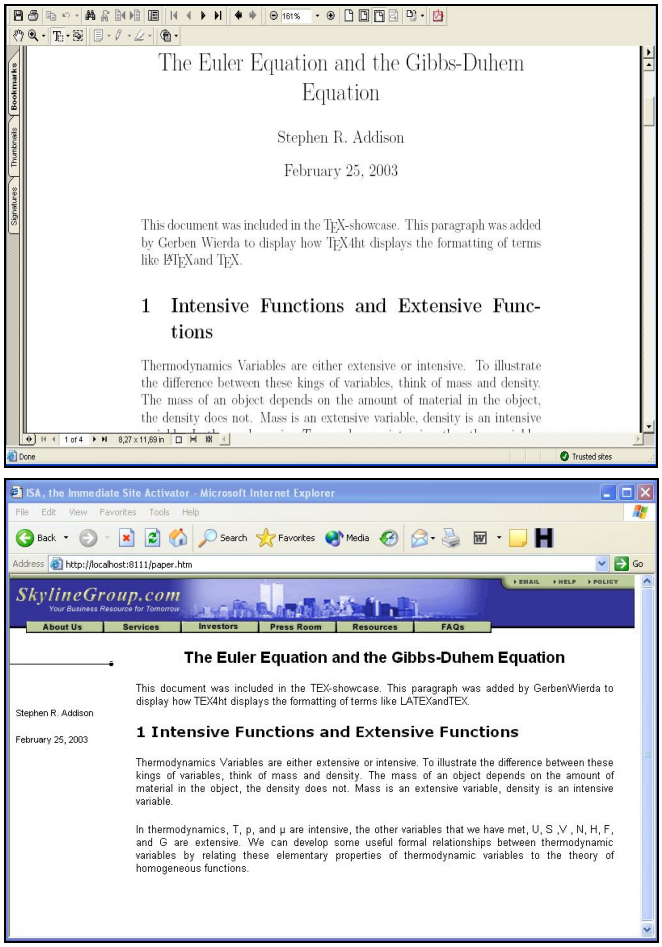

Figure 6. A PDF document re-flowed into a HTML page

The integration of the version manager and the conversion engine provides advanced services for the IsaWiki users: any client can request any version/variant of any document in any data format, through a specific URL. For instance, the version 12.1 of the local document *index.html* in DOC format can be obtained by asking for `http://serverIsaWiki/index.doc_12.1`; similarly `http://serverisaWiki/index.html_12.2` returns the following version of the same document in HTML

format and the `http://serverisaWiki/isawiki.php?doc_req=http://www.google.it/index.xml_3.3` returns the variant 3.3 of the (content of) Google Home Page in XML.

4.3.4 Assets management

Although the whole design of the IsaWiki framework is almost complete, the system does not implement yet an essential feature for the global editability environment described in the introduction: a fine-grained management of assets of content.

In the current implementation the atomic unit is a version or variant: versions are clearly distinguished, can be compared and converted from and to any format, but cannot be further segmented into reusable assets of content. On the contrary, the key aspect of global editability is the possibility of reusing and customizing any fragment, and so the possibility of re-flowing and moving any single fragment from and to any source. Moving these fragments without any constrains is not enough: the system has to keep references to the original source and owner of each fragment, in order to trace any change and keep credits of original authors.

Thus, we are studying the development of IsaWiki into an infrastructure based on IML fragments' management. Primarily we plan to integrate within the system a merging/import engine which allows authors to put together fragments from different sources into the same document, keeping information about each single fragment. This integration will allow users to create multi-sources documents (drawing content and ideas for all the accessed pages) or multi-authored documents (written in collaboration by many "emergent collaborators"). Finally, once this step is completed, a framework to handle intellectual property and copyright could be investigated as well.

5. CONCLUSIONS

In this paper we re-propose an old vision of the World Wide Web as a sharable, universal and democratic environment where all users are entitled to modify, customize and re-use others' content. This model is called "global editability". It has been directly inspired by Xanadu and by the original design of the same WWW, which initially was a wholly writable medium rather than a read-only one.

We have discussed the advantages of the global editability approach and described the IsaWiki system that really implements such a model. Global editability improves and integrates some existing web authoring approaches: the web data collection and reuse of any material, the possibility of annotating and linking external resources, the possibility of editing pages during the navigation and the collaborative editing.

IsaWiki is a distributed system, based on client-side architecture, designed and implemented according to the global editability paradigm. Obviously a lot of technical and non-technical issues had to be solved designing such a system. This paper explains these problems and motivates the global editability solution: the core principle is that any content fragment can be identified and retrieved within the system. Thus, it is always possible to know who edited a page, when it was edited and what was edited, so that any web user can modify any web page in a *safe* and *controlled* way.

Three main features could be integrated towards the implementation of such a global environment: *documents' segmentation* into manageable assets of content, *conversion* between data formats and advanced *versioning and diff*. These functionalities are almost already implemented in IsaWiki and they will be soon integrated and strengthened.

Furthermore, we are defining some criteria and tests for the system evaluation. In particular, we are formalizing and describing each of the functionalities of the system, in order to methodologically test each of the involved modules, both client-side and server-side. By adopting software engineering approaches, methods and tools we are building a testing environment that would support further developments in the system. We plan to use an automated testing framework, such as HttpUnit[20], in order to support, automate and speed up our tests, i.e., to emulate the browsers' behavior, examine returned pages and automatically produce detailed reports.

Then we are adding new functionalities to the system: in particular, we are working towards the creation of an advanced users' manager, the integration of all the functionalities into a unique global environment, the support for new data formats and above all the management of smaller reusable units of content.

6. ACKNOWLEDGEMENT

We wish to thank all the students that have taken part to the development of all the parts of the IsaWiki system that have been created so far: Gabriele Fantini, Pietro Nanni, Simona Orselli, Roberta Zeppilli, Elisa Ventura Campori, Michele Schirinzi, Stefano Monducci, Lauro Cottafavi, Stefano Rizzoli, Matteo Bagnasco, Nicola Bagnasco, Mariano Diasio, Luca Ventura and Vito Sanitate. In particular we wish to thank Antonio Feliziani for his precious and competent contribution to this work.

7. REFERENCES

[1] Amaya, W3C, http://www.w3.org/Amaya/.

[2] Anderson K. M., Taylor R. N., and Whitehead E. J., "Chimera: Hypermedia for heterogeneous software development environments". *ACM Transactions on Information Systems*, 18(3), July 2000.

[3] Bieber M., Vitali F., Ashman H., Balasubramanian V., Oinas-Kukkonen H. (1997) "Fourth Generation Hypertext: Some Missing Links for the World Wide Web", *International Journal of Human-Computer Studies*, 47, 1997, 31-65.

[4] Blood , R. Weblogs: a history and perspective, http://www.rebeccablood.net/essays/weblog_history.html.

[5] Bouvin N. O. Unifying strategies for Web augmentation. *Proceedings of ACM Hypertext'99*, p 91-100, 1999.

[6] Carr L. A., DeRoure D., Hall W., and Hill G. "The distributed link service: A tool for publishers, authors and readers". *Proceedings of the 4th International World Wide Web Conference*, 1995.

[7] Catlin T., Bush P. and Yankelovich N., "InterNote: Extending a hypermedia framework to support annotative collaboration". In: *Proceedings of Hypertext'89*, (ACM Press, New York, 1989,) pp. 365–378.

[8] Ciancarini P., Folli F., Rossi D. and Vitali F. "XLinkProxy: external linkbases with XLink". In *Proceedings of the 2002 ACM symposium on Document Engineering* edited by ACM Press, 2002, p. 57-65.

[9] Cunningham, W. & Leuf B. *The Wiki way*. New York, Addison-Wesley, 2001.

[10] Di Iorio A., Vitali F. "A Xanalogical collaborative editing environment". In *Proceedings of the second international workshop on Web document Analysis*, University of Liverpool, (2003).

[11] Di Iorio A., Vitali F. "Web authoring: a closed case?". Accepted at *Hawaii International Conference on System Sciences*, January 3-6, 2005, Big Island (Hawaii). To be printed.

[12] Di Iorio A., Vitali F. "Writing the Web" in *Journal of Digital Information,* Volume 5, Issue 1, May 2004.

[13] Diaz L.M., Wustner E., Buxmann P. "Inter-organizational Document Exchange - Facing the Conversion Problem with XML", *Proceedings of the ACM Symposium on Apllied Computing (SAC 2002)*, Madrid 2002

[14] Fetscherin, M. (2002): "Present State and Emerging Scenarios of Digital Rights Management Systems", in *:The International Journal on Media Management*, Volume 4, No. 3, 2002

[15] Greasemonkey, Mozdev.org, http://greasemonkey.mozdev.org/

[16] Grønbæk K., Sloth L., and Ørbæk P. "Webvise: browser and proxy support for open hypermedia structuring mechanisms of the World Wide Web". *Proceedings of the 8th World Wide Web Conference*, p 253–267, 1999.

[17] Gupta S., Kaiser G., Neistadt D., Grimm P., "DOM-based Content Extraction of HTML Documents", *Proceedings. of WWW2003*, May 20-24, 2003, Budapest, Hungary.

[18] Haake A. and Hicks D. "VerSE: Towards Hypertext Versioning Styles" in *Proceedings of ACM Hypertext '96*, Washington DC, 224-234, March 1996.

[19] Hall W., Davis H. C., and Hutchings G. *Rethinking Hypermedia: The MicroCosm Approach*. Kluwer Academic, Norwell, USA, 1996.

[20] HTTPUnit, Sorceforge.net, http://httpunit.sourceforge.net/

[21] Iannella R., "Open Digital Rights Language", W3C, http://www.w3.org/TR/odrl/, last visited 8th November 2004.

[22] iMarkup: Annotate, organize and collaborate on the Web, http://www.imarkup.com/products/annotate_page.asp

[23] JotSpot Inc., "JotSpot Beta: the application wiki", http://www.jotspot.com/

[24] Kahan J., Koivunen M., Prud'Hommeaux E., and Swick R. "Annotea: An open RDF infrastructure for shared Web annotations". *Proceedings of the WWW10, International Conference*. Hong Kong, 2001.

[25] Laurillau Y. and, Nigay L., "Clover architecture for groupware" in the *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, 2002, ACM Press, pp. 236-245.

[26] Maioli C., Sola S., and F.. "Wide-Area Distribution Issues In Hypertext Systems" in *Proceedings of ACM SIGDOC '93*, Kitchener, Canada, 185-197, 1993.

[27] Maioli C., Sola S., Vitali F., "The Support for Emergence of Collaboration in a Hypertext Document System" in: *ACM CSCW'94 Workshop on Collaborative Hypermedia Systems*, Chapel Hill (NC), GMD Studien n. 239, ACM, 1994.

[28] Meyrowitz N., "`*Hypertext---does it reduce cholesterol, too?*"", IRIS Technical Report 89-9, Brown University, Institute for Research in Information and Scholarship, (November 1989). Transcript of Hypertext '89 keynote address.

[29] Miles-Board, T., Carr, L., Kampa, S. and Hall, W. (2003) "Supporting Management Reporting: A Writable Web Case Study". In *Proceedings of The Twelfth International World Wide Web Conference (WWW2003)*, pages pp. 234-243, Budapest, Hungary.

[30] Mozile, Mozdev.org, http://mozile.mozdev.org/

[31] Mozilla Community, Mozdev.org, http://mozdev.org/

[32] Mozilla Organization, "XML User Interface Language (XUL)", 2003, http://www.mozilla.org/projects/xul/

[33] Nelson T.H., "Transcopyright: Dealing with the Dilemma of Digital Copyright." *Educom Review*, vol. 30, Jan/Feb 1997, p. 32.

[34] Nelson T.H., "Managing immense storage". *BYTE*, Volume 13, Issue 1 , January 1988, pp. 225-238.

[35] Nelson T.H., *Literary Machines*. Sausalito (CA), USA, Mindful Press, 1987.

[36] Opt-in email marketing Resource Center, http://www.opt-in-email-marketing.org/, last visited 8th November 2004.

[37] Permanent Uniform Resource Locator, OCLC Online Computer Library Center, http://purl.oclc.org/.

[38] Platypus, Mozdev.org, http://platypus.mozdev.org/

[39] Raymond E.S, "*The Cathedral & the Bazaar (Hardback) Musings on Linux and Open Source by an Accidental Revolutionary*", O'Reilly Editions, first edition October 1999.

[40] RFC 2141, URN Syntax, R. Moats, May 1997.

[41] Schraefel, M. C., Zhu, Y., Modjeska, D., Wigdor, D. and Zhao, S. (2002) "Hunter Gatherer: Interaction support for the creation and management of within-web-page collections". In *Proceedings of International World Wide Web Conference*, pages pp. 172-181, Honolulu, Hawaii, USA.

[42] The International DOI Foundation, "The Digital Object Identifier", http://www.doi.org/, last visited 8th November 2004.

[43] Vitali F., "Creating sophisticated web sites using well-known interfaces" in: *HCI International 2003 Conference*, Crete (Greece), June 2003.

[44] Vitali F., "Versioning Hypermedia", *ACM Computing Surveys*, 31(4), 1999, article n. 24, pp. 7.

[45] Vitali F., Di Iorio A., Ventura Campori E., "Rule-based Structural Analysis of Web Pages". In Simone Marinai and Andreas Dengel Eds., *Document Analysis System VI*, Volume 3163 of *Lecture Notes in Computer Science*, pp. 425-437, Springer Verlag, Berlin 2004.

[46] Vitali F., Durand D., "Using Versioning to Provide Collaboration on the WWW", *The World Wide Web Journal*, 1(1), 1995, 37-50.