Web authoring: a closed case?

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

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

ABSTRACT

Writing for the web is still a complex and technically sophisticated activity. Too many tools, languages, protocols, expectations and requirements have to be considered together for the creation of web pages and sites. The complete overlapping of readers' and authors' roles are important evolution steps towards a fully writable web, as is the ability of deriving personal versions of other authors' pages. Much like Xanadu was dreamt of providing, we discuss the features of IsaWiki, a browser-based editing environment sporting a number of interesting functionalities, including our own idea of xanalogical storage. Through the use of IsaWiki every web page, local and remote, can be edited and customized during browsing, where links can be created and collaboration can be set up with a minimum of complexity.

Categories and Subject Descriptors

H.5.1 [Information Interfaces and Presentation] Hypertext/Hypermedia

H.5.3 [Information Interfaces and Presentation] Web-based Interaction

General Terms

Design

Keywords

Web authoring, xanalogical storage, collaboration, customization.

1. INTRODUCTION

The technical prowess necessary for reading the web is very low: mostly everyone can read the Web, regardless of location, age, education, and (when WAI guidelines are used) physical ability. But this ease does not extend to authoring web pages. Too many tools, languages, protocols, expectations and requirements have to be considered together for the creation of web pages and sites. Yet, expert authors can routinely create and publish graphically advanced web pages that provide a lot of visually and intellectually compelling content, so it may seem that issues about documents creation, management and publishing are already more or less solved.

Unfortunately, this is far from true. The publishing model of the World Wide Web is asymmetric: a clear distinction exists today between readers and authors, whereby readers can only choose reading paths explicitly provided by the authors and cannot comment, create new content, create new links or create personal variants of web pages during browsing. What is needed is not to invent more advanced editors, however powerful and easy to use; what is needed, we think, is a stronger integration mechanism that connects reading and authoring, so that they can be carried out with the same technical knowledge, the same tools and even at the same time. In other words, we believe that an important evolution is going from a universally readable Web to a universally writable Web.

This evolution does not simply happen with a simplification of the editing process, though: an important innovation is the ability of performing personal versions of other authors' pages. In other words, web users should be able not only to easily create and update their documents, but also to personalize every page available on the web, regardless of its write permissions, creating personal variants where every contribution is clearly identified and distinguished from the original document. The web should therefore become a universal knowledge base from which all users can draw information and ideas, profiting from the contributions of the others.

Nevertheless, being the web an inexhaustible source of content, sharing ideas and thoughts trough web pages is not easy and natural yet. In the most cases documents and reflections remain "hidden" within the users' computers (or minds) but unavailable for the others and even if the materials are published they cannot be collected, edited and re-used by everyone. On the contrary, a larger and more profitable collaboration among users can be achieved transforming the web into a wholly writable environment. Benefits involve both the personal and the collaborative reading/editing processes.

The personal process is improved by allowing the users to re-organize and comment any web content according to each one's needs and preferences, creating new documents where contributions from different sources meet and mix. This scenario leads to two positive results:

- Other authors' ideas and documents can be used to develop new writings and comments. Easy modifications of other people's documents supports better content and reflections in the generation of new documents.
- Users can create new personalized documents by collecting and composing content fragments from different sources, which automatically become references to the original resources. Thus everyone can create one's own navigation path, real anthologies of content. A use that has been foreseen and described by Vannevar Bush [34], Ted Nelson [28] and, more recently, HunterGatherer [33].

The collaborative process is improved by the new simplicity of the collaboration itself: collaboration is not limited to a restricted set of people having the same tools, the same access rights to a shared storage, and the same objectives, but to anyone contributing content that ends up being useful. This creates new forms of emergent collaboration [35], by which the web can become a single, huge pool of human resources ready to collaborate, and not just of documents.

This ideas were first suggested for Xanadu [28], an ambitious project by Ted Nelson which unfortunately never came to exist, that proposed a distributed macro-system where everyone could access, read, reuse and comment on any material of other users, stored in a universal information pool called the *docuverse*. The original World Wide Web was widely inspired by Xanadu, but many of Xanadu's most interesting functionalities were either never introduced or abandoned very early, due to the complexity intrinsic to their development.

Recently this trend has been inverted and various efforts towards a writable web have been started by several researchers and projects. Different levels of overlap between the authors' and readers' roles can be found in today's Web: from the complete separation of roles implied by most commercial web sites to partial overlapping given by annotation systems [30], or the total unification of roles allowed by wikis [8].

An additional issue is requiring that all authoring and customization tasks are done while browsing, in a single shared collaborative environment where every page can be accessed, browsed, and edited, where every link can be traversed as well as created, where any kind of data fragment can be included in other documents: where every reader can be an author, too.

This scenario can be considered too extreme. Not everyone would be happy if the web becomes a universally writable environment, and not all circumstances and uses allow or require this. For instance, in Enterprise CMS it is rather unlikely that authors and managers approve of their content being made available to anyone on the web for customization.

Yet, there lies a big difference between a complete rewriting of the web philosophy into a new, anarchistic, "writable web" model without rules or control, and an evolution of the web that allows both traditional and innovative uses of the content that can be found on the web, according to two possible publishing philosophies:

- Restricted publishing, where only authorized authors use the innovative tools to create content that is subsequently officially published, and where particularly important are process management features and easier life for authors.
- Open publishing, where every reader is authorized to use the innovative tools and contribute to existing content.

Yet, every open publishing system can be used as a restricted publishing environment by setting the appropriate access rights. It is not by rejecting features, but by developing appropriate access control mechanisms that one controls the development and evolution of one's content.

In this paper, starting from these assumptions, we propose a classification of some web authoring scenarios and we briefly outline the benefits and drawbacks of each one. We draw up a list of requirements that are relevant to our vision of a complete publishing system. We stress the importance of implementing these features so as to maintain the independence of the global system from the choice of specific editing tools (everyone should be allowed to choose his/her favorite application), from technical skills in the authors (no particular technical knowledge should be required) and from the identification of a particular input or output format (documents should be edited and saved in as large a number of data formats as the system supports).

To summarize, our envision an universal collaborative environment over the Web where personalization, collaboration and on-the-fly browsing/editing are possible. We believe that this is possible without revolutionizing the basic architecture of the World Wide Web, since the current technologies are, surprisingly, already sufficiently sophisticated.

We introduce and describe IsaWiki, a tool developed at the University of Bologna that adheres to the vision of the writable web. IsaWiki is a complete editing and publishing environment, implemented for the most widespread browsers, whereby every web page, both local and remote, can be edited and customized during browsing, whereby links can be created and collaboration can be set up with a minimum of complexity. No particular technical knowledge, no particular tool or configuration are required; yet, through IsaWiki, we believe that the Web can turn into a complete publishing system rather than a simple navigation aid as today.

2. REQUIREMENTS

Starting from Content Management Systems [4][5] and some known limits of the Web architecture [2], we can identify some relevant issues about the requirements for a publishing system and the most important issues in this field, apart from the class of the publishing system (open or closed) we are dealing with.:

Readers' skill: Obviously the first requirement of a publishing system is the ease of reading the information: no particular skill, no specific software should be required.

Authors' skill: On the other hand, an environment where the documents are easy to read but very hard to write is, similarly, rather incomplete: required skills for authoring should also be minimized.

Layout/content separation: the separation between the information and the graphical aspects strongly influences the quality of the documents and the possible operations on them: first of all, the authors can deal only with the real contents of a document and not with its decoration and style; secondly, the content production is fast and easy [28]. 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 [24].

Templating and output flexibility: the development of web reading tools so different from the classical web browsers (cellular, PDA, smart-phones) and the need of a clear distinction between content and layout in the production of documents have increased the importance of the output flexibility, so one of the major issues in the recent (and future) web should be the transition from hand-crafted web pages to template-driven automatic page generation [26]. A different approach to obtain flexibility in the output is the conversion from (and to) data formats, based for instance on tags substitution, structural conversion [20] or different transformation models [11].

Content determination: many current web pages are really an undistinguished mix of actual content and decoration, so that mechanisms to extract the relevant content from them are necessary [24]. Some recent literature discusses this issue, proposing various heuristic approaches [13][17][32]. Determining the roles of each part of an HTML document is most often an essential step in customization: an author, in fact, is surely interested in modifying the actual content of a page while remaining uninterested towards the layout that has been used.

Workflow Management: the benefits of an advanced process management are clear: improvement of documents quality, archiving, revision and versioning, full control on users and permissions, easy management of the site, simplification of each production step, easy updating and so on.

Versions management: versioning carries important advantages [30][31]: historical revisions, parallel asynchronous collaboration, exploratory authoring, control over workflow management, emergent form of collaboration [19], efficiency.

Customization. Re-use of other people materials and xanalogical storage: the relationship between the roles of authors and readers is a litmus paper to evaluate a publishing environment. A system where users can read the pages but cannot customize them and reuse other peoples' materials is not complete. On the contrary, the readers should be able to modify, comment on or reuse other people contents during browsing. The objective can be considered Xanadu [23], the system proposed (but not completed) by Ted Nelson: Xanadu was to be built on the concept of xanalogical storage, a system to store documents not as whole blocks on a file system, but as a list of references to fragments combined into the final document on-demand. Each fragment represents an individual change to a document, stored separately and individually, in order to allow the reconstruction of the original, final and any intermediate state of the document throughout the history of its editing. This mechanism has a lot of advantages, mainly transclusions [22], or the reuse in whole or in part of content from other documents. This is different from pure copy&paste, in that the document stores only a reference to the external material. The software is expected to fetch the current content and place it inline with the main material, so that the included content is always current and updated with respect to its source document.

Metadata: Many operations need to be automated in a publishing system. That is impossible without meta-information, so it is necessary to investigate about their accuracy, languages, internal storage mechanism, external communication, updating and so on.

Link management: evaluating a content management system without exploring the link management issue is obviously going to be incomplete: the documents are connected, the human cognition is a collection of linked ideas and thought, so the system has to reflect this organization. This can be carried out only by links, or better yet by external and bidirectional links [2].

Spatial structure and site organization: an additional functionality that an information system has to provide is the possibility to organize, control and easily update the spatial organization of the pages. The authors can be helped, for instance, by a global vision of the site in a graphical environment.

Users management: many documents are usually produced by different authors who are collaborating, reviewing and correcting them in successive stages. So, within a workflow management system, a relevant module has to set and control users permissions, accountability and authentication. In an open context where everyone can read and modify pages, indeed, this issue stands for the control (and log) over the authorship of each modification.

Storage: authored content has to be placed into a repository. Storing does not means only copying resources as they were edited by the authors, but also versioning and breaking down into structures and meaningful components, ready for a fast and powerful retrieval.

Integration with other applications: the possibility to import and export contents from and to other applications and interaction with external applications (authentication, communications and so on).

Services: finally, every system can be evaluated by the additional services it provides. Chat, newsgroups, searching and indexing mechanism, email support, installation tools, activities diaries and all the extra applications to plan and simplify own activities.

3. A WEB AUTHORING TAXONOMY

Creating web contents is far from being as easy as reading them but a lot of different scenarios, tools, possibilities and requirements coexist in the current Web. We propose a classification that divides authoring scenarios in four category, according to the relationship between authors and readers roles as shown in tab. 3.1.

Table 3.1 Web authoring scenarios

Total separation	HTML Editors HTML Editors and converters Dynamic pages by server-side scripts Professional web tools
Separation with facilities for the authors	Drag'n'drop Stand-alone content pages Content Management System
Separation with external collaboration	Annotations External linking
Overlap of roles	Weblogs Wikis Browsers Editors

3.1 Total separation

The first category reflects a clear distinction between the author and reader roles: different required skill, different tools, different moments to come into play. This scenario is extremely limited: no customization and reuse of materials is possible, so the Web can be viewed as display rather than as a publishing environment.

3.1.1 HTML Editors

The most basic mechanism to publish web contents is certainly the editing of static web pages through HTML editors, either generic textual editors or dedicated WYSISYG applications.

3.1.2 HTML Editors and converters

Going up the level, we find HTML editors and conversion filters [15], that transform a number of data formats into HTML.

3.1.3 Dynamic pages by server-side scripts

Expert authors may be coding server-side scripts, that create web pages by programming the interaction with other server-side applications.

3.1.4 Professionals web tools

Professional tools such as Dreamweaver [18] automate and simplify, through wizards and (quite) clear interfaces, the management of the web pages. Professional users can easily get satisfactory results but these software require to be used with sufficient expertise for really good results.

3.2 Separation with facilities for the authors

The second category of web authoring scenarios involves all the situations where the two roles still have no overlap, but many functionalities are provided to simplify the authoring. No personalization or xanalogical reuse is possible, but at least becoming a proficient author is quite simple.

3.2.1 Drag'n'Drop

The documents to edit are ready-made pages divided in different zones: some blocks that contain exclusively graphical elements, and others that the authors can select and directly fill with the content [14].

3.2.2 Stand-alone content pages

In the previous scenario content and layout are in the same document and the author looks at the final effect of the page while he is creating it. In [28] we proposed a different solution: the author writes only the content independently from its layout and saves it in a specific directory on a web server. Server-side scripts merge content and layout into the final result whenever the page is requested.

3.2.3 Content Management System

The term CMS involves all the applications that manage creation, authoring and publishing of web documents. A CMS usually provides tools to control and organize the whole process of production of documents, above all separating the actual authoring phases (from the creation to the revision and approval) from the final delivery. Easy interfaces are provided to edit, correct and update documents and external materials can be easily imported in a CMS, so few skills are required to the users. Therefore through simple interfaces it is possible to manage metadata, spatial organization, versioning, archiving and so on. Many commercial applications can be considered CMS : from e-commerce to portal, from magazine publishing to electronic news and so on [5].

3.3 Separation with external collaboration

Some systems allow personal intervention on external materials providing a more sophisticated approach: the readers can improve and comment on other people's material, but the customization is only partial. Whenever a user adds a new annotation, in fact, the system does not create a variant of the document but all the annotations are merged in the same one version of the document. Furthermore the readers cannot create personal view of the same materials.

3.3.1 Annotations

Annotations are comments that can be added to the pages (not only by the official authors) stored in separate link-bases and merged subsequently into the same document while browsing. One example of the many existing ones could be W3C's own Annotea [16].

3.3.2 External linking

Similar observations can be extended to the management of links. The W3C standard XLink [9] introduced, among other improvements, the possibility of expressing external links. Linkbases and non-transparent HTTP proxies can be exploited to build applications that allow every user to add links from and to every Web page, regardless of ownership and access rights [6].

3.4 Roles Overlap

Finally in some scenarios all users can be at the same time readers, authors and reviewers of documents.

3.4.1 WebLogs

Weblogs[3] are tools for fast editing and publishing of personal diaries, targeted towards individuals and small communities. The editing is mostly based on web forms, and in some cases on WYSIWYG editors. The most relevant aspect is certainly the possibility of publishing content on-the-fly directly within the browser, through a simple HTTP POST. On the other side, weblogs have a fairly limited variety of document types (usually date-sorted notes). In fact, weblogs are little more than personal diaries, that only become a collaborative environment when the same diary is shared by multiple authors.

3.4.2 Wikis

Wikis [8] are collaborative tools for shared writing and browsing on the web, allowing every reader to access and edit any page of the site, through simple web forms and a very intuitive text-based syntax for typographical effects. Characterized by simple interfaces, an open editing philosophy, internal revision tracking and differencing mechanisms, wikis are graphically limited but are excellent tools for collaboration and easy publishing.

3.4.3 Browsers Editors

The World Wide Web first client was a browser/editor that allowed creating and updating documents directly on the Web during browsing. The W3C has recently activated a project called Amaya [27] that is a direct descendant of this kind of browser. Unfortunately the interface is complex and, unless the user has write permission on the resource, the application needs a service-server to store users' variants.

4. Towards an open publishing system

In the previous sections we have investigated some important issues about information systems, outlining a list of necessary steps to realize a really sharable and customizable content management system. In this section we present IsaWiki, a research prototype being developed at the University of Bologna aimed to merge all these benefits in a whole system and create an universal publishing environment on the WWW. IsaWiki draws its inspiration directly from ISA [28] and XanaWord [10] two previous research prototypes that we have used as starting points for this endeavor.

4.1 ISA: creating sophisticated pages with no technical knowledge

ISA is a parasite web page production system, providing an intermediate solution between the advanced effects of professional tools, powerful but difficult to learn, and the simple but ugly results obtained when exporting HTML from a word processor. The main idea of ISA is to exploit standard desktop tools for the creation of content and layout, and to employ a server-side application for the delivery of 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 a desktop tool such as Fireworks or Photoshop. Independently, the content producer (having no knowledge of HTML or any other markup language) can proceed to write the content documents. He will use either an HTML editor, or, more frequently, a word processor. The content producer will thus create any number of Word files, using styles as instructed by the layout designer, and saving them on the site as Web documents. After the layout has been created and the content document written and saved on the server, ISA is able to merge the layout and the content document to form a complete web page. Not that the author has only to deal with the contents, while the layout application is subsequent and automatic.

4.2 XanaWord: a xanalogical editing environment

While ISA still relies on web pages to be accessible through a web site under the control of the authors, XanaWord [10] is meant to provide web customizability for all web pages, regardless of their authors and access parameters. In the basic scenario, a user normally browsing WWW pages through a browser requests the browser to edit the currently displayed page. An instance of MS Word is launched and loaded with the requested page; the user edits the content and saves it in a specialized server. The system will then extract the changes introduced in the editing session, and will create a personal variant of the document (or a new version in case the user is also the original owner of the document). Anytime the user requests the same document the system will then add the changes again and will provide the user with the modified resource, although the original copy might still be unchanged. The XanaWord system is based on versioning, through the imposition of a structure on versions, and the ability of browsing and accessing every intermediate state of document. Since it is impossible to modify the main copy of the documents directly on their origin server, we rely, as XLinkProxy [6] do, on 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, and stored in a separate database in XML format) are applied on-the-fly to the original document retrieved from the origin server. Note that this algorithm is not a generic differencing mechanism on XML data (see instead [7]) but it has been designed and implemented to handle and exploit specifically MS Word documents, which already include the change tracking information. On the client side, through an *ad hoc* browser menu, users can request all versioning features (client/server communication is based on WebDAV).

4.3 IsaWiki

The experiences of ISA and XanaWord have converged into IsaWiki, a collaborative editing tool that integrates the authors' and readers' roles directly within the browsing experience. IsaWiki has a client-server architecture in which each service is clearly residing either on the client or on the server: the client is a module easily installable on common web browsers, the server-side application can run over each web-server and provides advanced services for registered users, so no new application, transmission protocol or markup language is needed [29].

The basic scenario we propose shows a user normally browsing web pages with a browser. By subscribing to the IsaWiki service and activating it, an interface would appear during the navigation, so the user can normally access the page or edit its content, access edited pages or request services on the current document.

By selecting the edit command, a content editor would appear and allow the user to add, delete and modify to the text content of the page. Ideally the system would be smart enough to allow in-place editing and to differentiate between the actual content and the presentational parts of the page. Every modification to the page is recorded, with author and time information. Upon saving the changes, the modifications would be sent to the IsaWiki server and made available to all the subscribers. The navigation behaves normally.

Every time the user surfs to a page, the browser would interrogate with the IsaWiki server. If a variant is present, the server would send it and it would be displayed in the browser in stead of the original page. An analogous service would be made available for user defined links and comments on web pages: rather than storing text and text modifications, it would store link information and pointers to places in the document, but the overall mechanism would be otherwise identical.

Our architecture is based on a simple read-only mechanism: official content remains unmodified on the origin server, while all personal variants produced by the user are saved on an external IsaWiki server. The final result is meant to realize an "open" publishing environment allowing users to enhance collaboration and personalization. By using the IsaWiki system we expect to convert the web editing process into a "natural" and "universal" one. By "natural" we intend the possibility to enact immediate in-place editing during the navigation through the use of a WYSIWYG editor. By "universal" we mean that all pages can be modified by every user independently on their write permissions and ownerships.

IsaWiki enforces total overlapping of author and reader roles, which weblogs, wikis and browsers/editors only partially do. Thus, while a wiki only edits local resources, IsaWiki does so to all resources on the whole web. A wiki is a web site whose pages are automatically generated by a server-site script and are designed to be modified within the browser. Editability remains limited to the wiki local resources. IsaWiki entitles users to modify pages which are not intended for this. Moreover, wikis imply knowledge of a specific syntax, however simple it may be. On the contrary, IsaWiki exploits a WYSIWIG editor, where no technical knowledge is required and even unaware users can modify complex pages.

As far as browser/editors are concerned, they do indeed share with IsaWiki the idea of in-place editing while browsing, but they lack completeness in the publishing environment. Browser/editors are wholly client-side. They require the implementation of a server-side application giving support to personalization, collaboration, versioning and users management. IsaWiki is an integrated environment where such functionalities are already available. It would then secure a shifting from a "readable" web to a "writable" web.

Two practical questions arise. The first relates to the copyright:. how would IsaWiki prevent a user from maliciously manipulating other authors' web pages? Fundamentally IsaWiki bypasses the problem: it simply records all personal variants on local servers while the original documents remain unmodified on the origin servers. So, it is always possible to identify who edited, when he/she edited, and what has been edited. Moreover the personal variants are not as accessible by everyone as the original pages and their existence does not interfere with the browsing of uninterested user. This approach is well within the *opt-in* philosophy of "good" email marketing, that makes sure that only interested users are contacted: subscribers surf on personal variants of the pages while the others surf on the original pages as created by the original author. Clearly, issues about copyright management and protection could raise further ethical and legal debates, but we believe this is out of the scope in the present paper.

The second 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: would IsaWiki scale in this scenario? This problem is addressed by the architecture of the system: IsaWiki is a distributed and decentralized environment where many servers live together providing services for many users. Each user initially registers himself to a single IsaWiki server that will support him during the navigation and the editing. The critical aspect is the number of the registered users (and consequently the amount of personal variants) on a particular IsaWiki server rather than the total number of existing versions or the number of users modifying the same web page. So the IsaWiki system could easily scale by increasing the number of the servers, by replicating resources and by exploiting techniques related to the distributed systems.

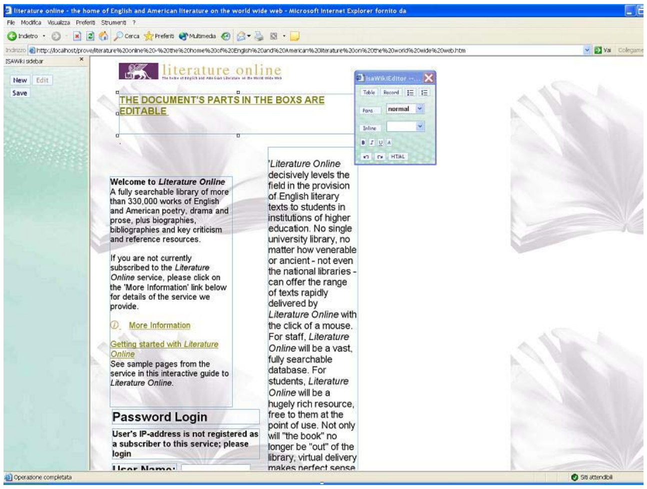
Two modules cooperate to realize the IsaWiki environment: a web browser integrated client and a multi-service web server.

4.3.1 A web browser integrated client

The IsaWiki client is composed of a sidebar installed on the user's browser (the current version only works on Internet Explorer under Windows, but the Mozilla version under Windows, Linux and MacOsX in in an advanced development stage) and communicating with a preset server.

The main tasks of the sidebar are navigation monitoring and page customization: whenever the URL of the main browser window changes, as the browser is loading the actual document from the origin server, the sidebar activates and verifies with the IsaWiki server if data about previous editing sessions exist. If appropriate data is found, the sidebar downloads it. When the data arrives the result is displayed in browser window.

When the user requests to edit a page, a WYSIWYG editor appears. The fig. 4.1 shows this scenario.


literature online - the home of English and American literature on the world wide web - Microsoft Internet Explorer fornito da
File Modifica Visualizza Preferiti Strumenti ?
Indietro Cerca Preferiti Multimedia
Indirizzo http://localhost/prove/literature%20online%20-%20the%20home%20of%20English%20and%20American%20literature%20on%20the%20world%20wide%20web.htm
Vai Collegamenti
ISAWiki sidebar
New Edit
Save
literature online
IsaWikiEditor --...
Table Record
Para normal
Inline
B I U A
HTML
Operazione completata
Siti attendibili

THE DOCUMENT'S PARTS IN THE BOXS ARE EDITABLE

Welcome to *Literature Online*
A fully searchable library of more than 330,000 works of English and American poetry, drama and prose, plus biographies, bibliographies and key criticism and reference resources.

If you are not currently subscribed to the *Literature Online* service, please click on the 'More Information' link below for details of the service we provide.

More Information

Getting started with *Literature Online*
See sample pages from the service in this interactive guide to *Literature Online*.

Password Login

User's IP-address is not registered as a subscriber to this service; please login

User Name:

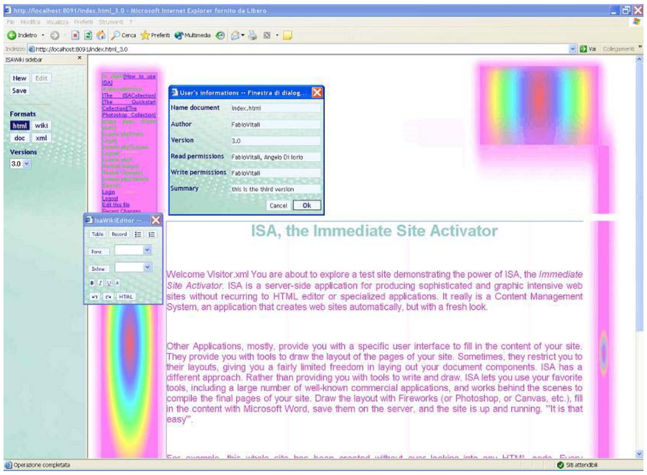
'*Literature Online* decisively levels the field in the provision of English literary texts to students in institutions of higher education. No single university library, no matter how venerable or ancient - not even the national libraries - can offer the range of texts rapidly delivered by *Literature Online* with the click of a mouse. For staff, *Literature Online* will be a vast, fully searchable database. For students, *Literature Online* will be a hugely rich resource, free to them at the point of use. Not only will "the book" no longer be "out" of the library, virtual delivery makes perfect sense

Figure 4.1 A web page editing with IsaWiki

This editor provides inline editing of the content within the actual page, keeping the same styles and layout.

An important assumption of our work is that users editing and customizing web pages are mainly interested in their content, and would prefer to be spared the details of the presentational parts of the page. On the other hand, there are many web pages where the presentation part is heavy in comparison to the content, and whose content is practically indistinguishable from the presentation. Examining the HTML code does not help in finding an easy way to separate content and presentation, so we have to resort to empirical mechanisms to identify the parts of the page containing real content, and those only providing decoration and layout. This page analysis algorithm identifies content blocks and makes them editable, so the IsaWiki editor allow user to access only to the relevant part of the pages, as you can see in fig. 4.1.

After the editing session, the user has simply to select the save command to send the modified document to the IsaWiki server, which saves it in the appropriate directory and updates the version data for the document. Note that the sidebar asks for a few metadata (the title of the document, the access list for reading and writing, etc.) before posting the data. These information are useful to manage, version and organize the documents, but we plan in the near future to extend them towards a more complete metadata management. The figure 4.2 shows an example of editing and save a document and its metadata with the IsaWiki client.


http://localhost:8091/index.html_3.0 - Microsoft Internet Explorer fornito da Libero
File Modifica Visualizza Preferiti Strumenti ?
Indietro Cerca Preferiti Multimedia
Indirizzo http://localhost:8091/index.html_3.0
Vai Collegamenti
ISAWiki sidebar
New Edit
Save
Formats
html wiki
doc xml
Versions
3.0
[How to use ISA]
[The ISACollection]
[The Quickstart Collection][The Photoshop Collection]
Login
Logout
Edit this file
Recent Changes
User's informations -- Finestra di dialog...
Cancel Ok
IsaWikiEditor --...
Table Record
Para
Inline
B I U A
HTML
Operazione completata
Siti attendibili

Name document	index.html
Author	FabioVitali
Version	3.0
Read permissions	FabioVitali, Angelo Di Iorio
Write permissions	FabioVitali
Summary	this is the third version

ISA, the Immediate Site Activator

Welcome Visitor.xml You are about to explore a test site demonstrating the power of ISA, the *Immediate Site Activator*. ISA is a server-side application for producing sophisticated and graphic intensive web sites without recurring to HTML editor or specialized applications. It really is a Content Management System, an application that creates web sites automatically, but with a fresh look.

Other Applications, mostly, provide you with a specific user interface to fill in the content of your site. They provide you with tools to draw the layout of the pages of your site. Sometimes, they restrict you to their layouts, giving you a fairly limited freedom in laying out your document components. ISA has a different approach. Rather than providing you with tools to write and draw, ISA lets you use your favorite tools, including a large number of well-known commercial applications, and works behind the scenes to compile the final pages of your site. Draw the layout with Fireworks (or Photoshop, or Canvas, etc.), fill in the content with Microsoft Word, save them on the server, and the site is up and running. '"It is that easy"'.

For example, this whole site has been created without ever looking into any HTML code. Every

Figure 4.2 A web page editing and save in IsaWiki

Furthermore the sidebar allows version access: a version list of local documents is always displayed to let the user access every individual version of the document. The original copy is always selectable, to let the user access the document in its real origin server. If the document on the origin server has changed since the first modification of the document on the local server, the current version of the document is also shown in the list as an autonomous version. Finally the sidebar contains a menu to request data format conversion,searching, monitoring and listing functionalities.

Other browsers cannot offer the sidebar, so it is necessary to foresee alternative input mechanisms for those applications. In this case, a server-side mechanism has to extract the content of the page and place it within a form element in a dynamic HTML page, which is then displayed to the user. Furthermore a set of predefined URLs allow to manually send the IsaWiki server the same requests available through the sidebar.

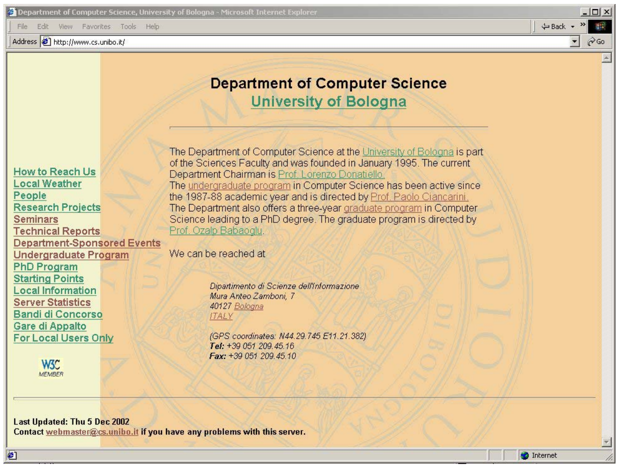
4.3.2 A multi-service server

The server is a PHP application running in an Apache HTTP server and tested under Linux, Windows 2000, and Mac OSX, obviously able to satisfy all the client requests. Two kinds of documents can be hosted on this server: local documents (new document created directly on the server), or remote ones (the local versions of a document residing on a different server that a registered user has previously customized. The IsaWiki server allows registered users to personalize web pages, send new versions, create or navigate official versions and personal variants trees and so on. Everything is stored on the server, everything is easy to access and update, everything is accurately structured.

Each version of each document is stored in a specific data format, exactly as saved by the editor. Allowing users to edit document only in the original format is an unsatisfactory solution: users cannot use their preferred editor to modify successive versions, pages always have to be displayed in the same format and differencing mechanism have to be implemented for each data format. IsaWiki instead proposes a different solution: the document is stored exactly as saved by the editor but a conversion module can transform any document from any format to any other . So, regardless of the format the document is stored on the server, a user can request the same document in any other format to be displayed or edited. Currently we support HTML, XML, MS Word, plain text (with loss of information) and the wiki data format. Different versions can have been edited with different editors (so a document started with MS Word was subsequently modified with the WYSIWYG editor, and then again with Word, etc.) and everyone can choose a preferred editor. The conversion module exploits an intermediate conversion to the generic data format according to the "superior standard" model ([11], as well as [1][21]). The IsaWiki generic data format is a subset of XHTML which captures the fundamentals of a data document, and ignores the irrelevancies, such as the typographical properties. A core set of significant fragments (paragraphs, tables, in-lines, records, etc.) allows to exhaustively describe the actual semantic of a document. The advantage of this approach is that it is easy to extract the relevant parts of a document, and it is easy to rebuild the original format afterwards, by adding (even randomly) the typographical properties that were previously extracted. On the whole, the access and the editing is completely independent of any input or output format.

Also the differencing and versioning system have to be adapted to this vision. If different versions can be edited with different editors, how can the IsaWiki server determine the delta between two versions that can be in different format? The two versions are first converted to the generic intermediate format, and then the diff is taken. The delta is usually taken from the result of an external diff engine but if the editor adds change-tracking information (as is the case with MS Word or our WYSIWYG editor), the delta is automatically extracted and saved whenever a new version of the document is saved. The delta is then used to create display of the differences between two or more subsequent versions of a document and compact saves of long version trees.

The relevant part of the document is the actual content that is expressed by the generic data format. Whenever the document is requested as HTML, the presentation module, starting from this generic representation can apply one of the many layouts created with a graphic application. The actual layout can be specified as a global option, within the document, or within the request URL. This module is taken directly from ISA: note that it allows to re-flow a modified content within a presentation and layout different from the original one, with a good graphical effect.



Department of Computer Science
University of Bologna

The Department of Computer Science at the University of Bologna is part of the Sciences Faculty and was founded in January 1995. The current Department Chairman is Prof. Lorenzo Donatiello.
The undergraduate program in Computer Science has been active since the 1987-88 academic year and is directed by Prof. Paolo Ciancarini.
The Department also offers a three-year graduate program in Computer Science leading to a PhD degree. The graduate program is directed by Prof. Ozalp Babaoglu.

We can be reached at

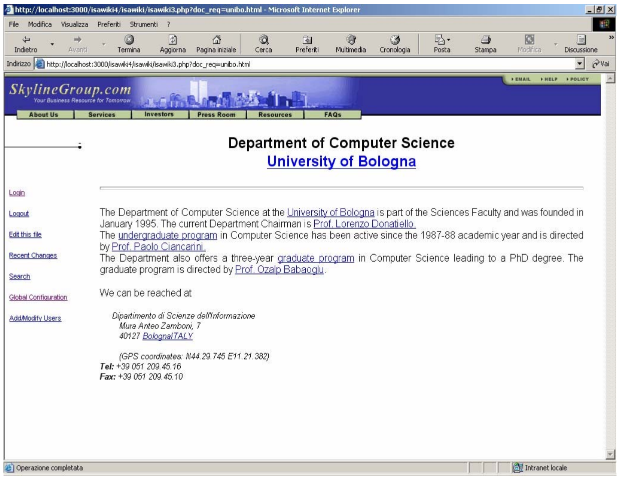
Dipartimento di Scienze dell'Informazione
Mura Anteo Zamboni, 7
40127 Bologna
ITALY

(GPS coordinates: N44.29.745 E11.21.382)
***Tel:** +39 051 209.45.16*
***Fax:** +39 051 209.45.10*

Last Updated: Thu 5 Dec 2002
Contact webmaster@cs.unibo.it if you have any problems with this server.

Figure 4.3 The original page page http://www.cs.unibo.it

For instance, the fig. 4.3 and 4.4 show the original home page of the department of the Computer Science of the Univesity of Bologna and a variant on a IsaWiki server with a different layout and few modifications.



Department of Computer Science
University of Bologna

The Department of Computer Science at the University of Bologna is part of the Sciences Faculty and was founded in January 1995. The current Department Chairman is Prof. Lorenzo Donatiello.
The undergraduate program in Computer Science has been active since the 1987-88 academic year and is directed by Prof. Paolo Ciancarini.
The Department also offers a three-year graduate program in Computer Science leading to a PhD degree. The graduate program is directed by Prof. Ozalp Babaoglu.

We can be reached at

Dipartimento di Scienze dell'Informazione
Mura Anteo Zamboni, 7
40127 BolognaITALY

(GPS coordinates: N44.29.745 E11.21.382)
***Tel:** +39 051 209.45.16*
***Fax:** +39 051 209.45.10*

Figure 4.4 A variant of the page http://www.cs.unibo.it created with IsaWiki

Just as most wikis do, IsaWiki provides a number of services for the reader of local data. These include a simple search engine and a "recent changes" document where the title of the most recently modified documents are listed, as well as their modification date. The system is completely configurable through a web interface and provides a complete user management module. IsaWiki is a service-based environment for registered users, so at every access the system controls who has sent the request and the read and write permissions to the requested resource. This control is particularly important when a page is edited, since a list of official authors is associated to every document and, if the current user belongs to this list, a new official version is created, otherwise a variant visible only to registered users or to this author.

The IsaWiki server does not require a specific client to work. Every request can be expressed though a specific URL: certainly the supported sidebars provide an easy and fast access to all the functionalities but, with some efforts, from every browser, every user can interrogate this server. Also editing for old browsers is supported: in this case the server will extract the content of the page, convert it into HTML or wiki, and place it within an HTML form to be displayed to the user.

5. CONCLUSIONS

Different scenarios, tools, possibilities and requirements live together in today's web authoring; they prove that the Web today is not a complete publishing system yet and has still some important limits, first of all the still existing separation between the readers' and authors' roles.

In this paper, we have outlined many of the requirements that our idea of complete publishing environment has to satisfy and, starting from a classification of possible scenarios in web authoring, we have shown that there not exists a solution yet that satisfies all these requirements.

Hovewer we believe that is possible to transform the World Wide Web into a real collaborative editing system, without necessarily revolutionizing its basic architecture, but rather exploiting existing technologies and protocols.

We believe that our proposal, IsaWiki, can be considered an innovative step in the direction of this goal, providing a client-server environment where a user, while browsing, can create, modify and customize any accessed page. As required for a complete content management, the application is completely independent form any data format, editing tool and user skills.

6. ACKNOWLEDGMENTS

We would like to thank all the master and undergraduate students at the University of Bologna who took and are taking part to this project for their master thesis and final projects, respectively. They are: Clara Bianconcini, Gabriele Fantini, Erica Gazzani, Pietro Nanni, Simona Orselli, Elisa Ventura Campori, Roberta Zeppilli. We would also like to thank Microsoft Research for their partial support to this endeavor.

7. REFERENCES

[1] Abiteboul S., Clouet S., Milo T. " Correspondance and Translation for Heterogeneous Data", Proceedings of ICDT'97, 1997

[2] 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.

[3] Blood R. "Weblogs: a history and perspective", http://www.rebeccablood.net/essays/weblog_history.html, last visited 14th November 2003.

[4] Boiko B., "Content management Bible", John Wiley & Sons Inc.,2001.

[5] Browning P. and Lowndes M., "Fourth Institutional Web Management Workshop Report", University of Bath, 2000

[6] 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.

[7] Cobena G., Abiteboul S., Marian A., "Detecting changes in XML Documents" in: Proceedings of ICDE 2002, San Jose, California, USA, IEEE, 2002, pp. 12.

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

[9] De Rose S.J., Maler E., Orchard D., "XML Linking Language (XLink) Version 1.0", World Wide Web Consortium, 2001, http://www.w3.org/TR/2001/REC-xlink-20010627.

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

[12] Doyle L.,"Content Management Systems Workshop Report," http://www.bris.ac.uk/ISC/cms/summary.htm, University of Bath, 2000

[13] Gupta Suhit, Kaiser Gail, Neistadt David, Grimm Peter "DOM-based Content Extraction of HTML Documents", Proc. of WWW2003, May 20-24, 2003, Budapest, Hungary.

[14] Homestead home page, Homestead technologies, http://www.homestead.com/

[15] HTML Converters, World Wide Web Consortium, http://www.w3.org/Tools/Filters.html

[16] Koivunen Marja-Riitta, "The Annotea Project", World Wide Web Consortium, 2001, "http://www.w3.org/2001/Annotea/

[17] Kunze M., Rosner D. (2001) "An XML-based Approach for the Presentation and Exploitation of Extracted Information".In Proceedings of the first international workshop on Web document Analysis, University of Liverpool, http://www.csc.liv.ac.uk/~wda2001/Papers/16_kunze_wda2001.pdf

[18] Macromedia corporation, "Macromedia Dreamweaver Home Page", retrieved November 14, 2003 from http://www.macromedia.com/software/dreamweaver/

[19] 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.

[20] Mamrak, S.A., Barnes J., C. O'Connell. Benefits of automating data translation, IEEE Software, July 1993, 82-88

[21] Milo T., Zohar S.. Using Schema Matching to Simplify Heterogeneous Data Translation", Proceedings of the 24th VLDB Conference, New York 1998, 122-133

[22] Nelson T.H., "Transcopyright: Dealing with the Dilemma of Digital Copyright.", *Educom Review*, 32(1), 1997, 32-35.

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

[24] Rahman A. F. R., Alam H. and Hartono R. (2001) "Content Extraction from HTML Documents". In Proceedings of the first international workshop on Web document Analysis, University of Liverpool, http://www.csc.liv.ac.uk/~wda2001/Papers/11_rahman_wda2001.pdf

[25] Soderland S. (1997) "Learning to extract text-based information from the World Wide Web". In Proceedings of

Third International Conference on Knowledge Discovery and DataMining (KDD-97), pp. 251-254, 1997.

[26] Template Attribute language, Zope Coprporation, http://zope.org/Wikis/DevSite/Projects/ZPT/TAL, last visited on 14th November 2003.

[27] Vatton I., “Amaya”, World Wide Web Consortium, 2003, http://www.w3.org/Amaya/.

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

[29] Vitali F., “Functionalities are in systems, features in languages. What is the WWW?”, IV Hypertext Functionalities Workshop, Seventh International World Wide Web Conference, Brisbane Australia, 14th April 1998, http://www.cs.nott.ac.uk/~hla/HTF/HTFIV/fabio.html.

[30] Vitali F., “Versioning Hypermedia”, ACM Computing Surveys, 31(4), 1999, article n. 24, pp. 7.

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

[32] Yu Chen, Wei-Ying Ma, Hong-Jiang Zhang "Detecting Web Page Structure for Adaptive Viewing on Small Form Factor Devices" Proc. of WWW2003, May 20-24, 2003, Budapest, Hungary

[33] schraefel, m.c., Modjeska, D., Wigdor, D. and Zhu, Y. (2002) "Hunter Gatherer: Interaction Support for Within-Web-page Collection Making". Proceedings of WWW2002: the 11th International World Wide Web Conference, Honolulu, Hawaii, 7-11 May http://www2002.org/CDROM/refereed/130

[34] Bush, Vannevar, "As We May Think." Atlantic Monthly, July 1945.

[35] Maioli C., Sola S., Vitali F. (1994), "Versioning for Distributed Hypertext Systems". In Proceedings of the Hypermedia '94 Conference, Pretoria, South Africa.

[36] Tim Berners-Lee with Mark Fischetti, Weaving the Web, Harper San Francisco, 1999