CHEOPS: Adaptive Hypermedia on World Wide Web

SALVATORE FERRANDINO

XCom Wide Communication

Via E.De Filippis 107/A –P.co Luciano

84013 Cava De' Tirreni (SA) – Italy

ALBERTO NEGRO VITTORIO SCARANO

Dipartimento di Informatica ed Applicazioni "R.M. Capocelli"

Università di Salerno

84081 Baronissi (SA) – Italy

Abstract

Because of stateless characteristics of HTTP, it is not possible for HTTP servers to adapt responses on the basis of previous interactions with the same user. Although efficient, this is a limiting aspect of the protocol, particularly for Web documents that are directed toward a broad audience, including experts and novices and an adaptive response would be helpful in providing information with the right *style*.

We present here the key concepts of CHEOPS, a system that is being developed as a tool that can be used by hyperdocument designers to provide their document with adaptivity and navigational aids. Version 0.9 of CHEOPS, actually available, is implemented by several CGI-BIN scripts that enables an HTTP server to interact with a user and follow her on the "Path to Knowledge". Solution is transparent to the user and to the client, and has a moderate impact on server performances.

Further directions of the work toward version 1.0 is also presented.

1 Introduction

The World Wide Web (WWW) is an information retrieval system on the Internet that can be considered the first real global hypermedia network. It was conceived as an environment

where information from any source can be accessed in a consistent and simple way from any place in the network.

The World Wide Web was developed in 1989 at CERN as a means of sharing information through the organization. Since then, WWW's growth has been exponentially fast, by reaching tens of millions of users through the whole Internet.

Although originally developed to facilitate information sharing (especially in academic environments), the World Wide Web has large potentials as an educational support, especially for distance learning. In fact, several educational systems based on WWW have been developed in the recent past [4, 11].

It is our opinion that in order to fully exploit WWW potentiality in the educational field, server response must be aware of user's behaviour so that it can take into account the level of knowledge and, as a consequence, provide the user with documents that she can read, given her background/capacity. This ability is well beyond the usual interaction with students through exercises and automated response for exercises or tests [4]. In fact, what we propose and implement here is to have an "opaque" mechanism, that, based on the history of user's requests, can provide a different document that is likely to be understood and lightly challenging while not being intimidating.

Such a mechanism can be also helpful to avoid the risk that users can "*have trouble in finding the information they need*" [16] because of the large amount of information available. As an example, many WWW servers of official institutions (like Universities, Colleges, Research Labs and so on) include a "Presentation" of the institution and its size can be as small as few Kilobytes of text or as large as hundreds of Kilobytes including text, images, audio, history and so on. On the increasingly overcrowded network, downloading the "Presentation" can take up to few minutes, when to make a "hit" few lines would be usefully delivered to a "passer-by" and could induce her to continue the visit of the site. An adaptive response from server could slowly increase the amount of information given to the user and,

therefore, avoid the risk of an "information overloading" that could "scare her off" the site.

1.1 HTTP and Adaptivity.

HyperText Transfer Protocol (HTTP) is a very simple Internet protocol, similar to the File Transfer Protocol (FTP, RFC 959) and Network News Transfer Protocol (NNTP, RFC 977).

HTTP is a fast and efficient protocol for searching and retrieving information from a server: the client makes a TCP-IP connection to the host by using domain name (or IP number) and the port number (default port is 80). Then the client sends a request document, consisting of lines (CR-LF terminated) of ASCII characters. The server sends back to the client its answer: the document required or (if that is the case) an error message.

Because of efficiency requirements, HTTP 1.0 was designed as a stateless protocol, that is the server does not keep any state on behalf of the client. In this way, HTTP servers can handle a large number of requests at the same time.

Although very efficient, this design requirement is limiting many possible activities that the server can perform on request. Examples are the ability of analyze the efficacy of cross references within the same Web site, build statistical profiles of users and to adapt its behaviour on the basis of previous interactions.

Solutions to this limitations fall in two categories: those proposing an extension to the HTTP 1.0 protocol and those devoted to desing a new protocol. In the first category, we see, for example, an extension to the HTTP 1.0 protocol [12] which is a refinement of the "Cookie" solution proposed by Netscape [13]. On a different direction lies the effort to overcome this limitation in the next future by designing HTTP 1.1 [8], where it is allowed to use the same TCP/IP connection to perform multiple operations.

Both kind of solutions offer advantages but also suffer from some limitations. New HTTP 1.1 is still being designed and so are the browsers that can take advantage of the multiple operations capability. Multiple operations should be performed in HTTP 1.1 by keeping the

connection open between client and server (*Keep-Alive* connections). The mechanism looks ill-suited for mantaining a session in an educational hyperdocument. In fact, typical usage would be to browse through a hyperdocument for a non-negligible amount of time, which can be burdensome on the server-side, being a child process of the HTTP server devoted exclusively to the goal of serving the user requests.

On the other side, the "cookie" extensions proposed suffer from the drawback that they can be used on specific (also if widespread) browsers like Netscape 2.0 and following versions [13]. Also, widespread complaints over similar mechanisms that can be used to mantain surveillance of users accessing a Web server are reported over the Internet and taken into account by several authors [7].

1.2 CHEOPS System.

CHEOPS is more oriented toward a "server-side", application-specific solution to the adaptivity requirement, with particular care in avoiding overloading the server with many tasks. Its architecture is designed to limit concerns on users' privacy: the session that CHEOPS keeps track of is strictly limited to the hyperdocument being designed and, therefore, is not extended on user's behaviour on the whole Web site. Moreover, CHEOPS, although specific to educational application can be also generalized to other similar situations when presenting information at the right rate (for the user) is crucial.

Potentialities of WWW as an educational tool were immediately recognized and many results are available on the educational use of the Web for distance learning [4, 9, 10, 11]. In particular, in [10] the need for a "WWW work session" (using our terminology) in an educational setting was recognized and an application based on CGI-BIN programs was developed that adds a similar capability to the Web server.

Our solution differs from [10] in two major aspects: first, CHEOPS is a design system that should make easy for a designer to add adaptivity to a hyperdocument in a modular way.

Especially with the extensions planned (and currently under development) for version 1.0, this goal looks quite reachable. The second one is an architectural difference: in [10], each time a session was started there was a sleeping process on the server which each successive request would interact with. That would pose two problems: the first is that one needs (as also pointed out by the author in [10]) an additional mechanism to eliminate "hanging" unwanted sleeping process ("zombie" processes that do not correspond to an active session anymore). Then one should consider carefully the impact on server performances: if many sessions are to be taken care of concurrently, there would be a proliferation of spawned processes with a consequent, possible, performances degradation on heavily loaded servers.

It should be noted that, from a design point of view, one does not need a permanent "sleeping process" to interact with during the whole session, but only a process that loads the server only when needed, that is, *at the moment of the request* and, if correctly instructed, can recognize the request as belonging to a specific session.

Our solution, therefore, adopts a different strategy: each time a session is started, the server creates a user's profile, "distributing" (virtually) a "session card" *embedded within the hypertext* file that is sent back to the browser. Each successive requests to the server will be done with this session card and, therefore, recognized as belonging to a specific session.

CHEOPS System is non-obtrusive, widely usable, (one does not need specific browsers to use our system), has moderate impact on server load (putting all the "memory" of the session where it belongs, i.e. on the disk) and is modular enough to be easily installed on a Web server for a specific hyperdocument.

CHEOPS is implemented through several CGI-BIN scripts, used to build "on-the-fly" documents that depend on user's previous inquiries at the same server *within the same session.* CGI-BIN scripts are program run on the server and triggered by input from a browser.

1.3 Structure of the paper

In the next section, the system design and architecture of CHEOPS is presented with a brief description of requirements, the Session Protocol, the Knowledge Model we propose and few implementation details. Then, in section 3, we conclude by showing a case study that led to the development of CHEOPS and giving the guidelines of future work that is currently planned (or under development) within the CHEOPS project toward a version 1.0 of the package.

2 System Architecture

CHEOPS, actually being developed at the Dipartimento di Informatica ed Applicazioni of the University of Salerno in Italy is a "server-side" implementation of a session-based interaction model based on several specifications required by an educational, WWW-based, software.

CHEOPS version 0.9 consists of several Unix Shell scripts that provide the designer of a hyperdocument with an automatic mechanism that satisfies user's requests according to previous inquiries.

We begin the section by defining several requirements, identified during a previous project briefly described in section 3, that led us to the design of CHEOPS. Then, we show our Session Protocol and illustrate the Knowledge Model which CHEOPS is based on. Finally we provide a description of the implementation, with few key concepts of CHEOPS design.

2.1 Analysis of requirements

Being focussed on developing a design tool for hyperdocument to be employed in the educational field, several requirements were identified during a preliminary project:

Lecture-type Interaction: The user should be able to interact with a WWW hyperdocument that makes her feel like having a teacher/instructor/field-expert "*on the other*

side of the screen". To such a goal, it is necessary to have tools able to distinguish between requests of different users. It is also recommended the usage of context-sensitive help, where by context it is not only meant the part of hyperdocument currently visited but also the user knowledge and previous interactions with the help system.

Categories: It is often the case that there is a "natural" subdivision of the hyperdocument in categories that are well-suited to the argument. It is important that user is able to check her "knowledge level" for each category and also able to choose to further explore a category that, willingly or not, was ignored at a previous step.

History Mechanism: User should be able to check the path followed during her interaction with the hyperdocument: being able to trace back previous interactions (within the same session) is helpful in developing and reviewing the actions taken.

Avoid the "*Too much*" syndrome: Overloading the user with too much information, too early is a bad policy well-known to lecturers.

Tours: User should be able to learn according to her own character: some users would like to "explore" on their own the hyperdocument while others could be more at their ease by being provided a guided tour through a part of the document.

It was clear, during the design of CHEOPS, that a WWW hyperdocument that develops the full potentialities of above mentioned requirements needed adaptivity in the server response. Furthermore, the project was to be realized by using "off the shelf", state-of-art tools that were not to make heavier server load.

Then, we, first, defined a model of the interaction between user and server.

2.2 The Session Protocol

The session protocol that has been implemented is based on the following sequence of actions:

1. **Introduction.** A new user is introducing herself to the server through the choice of an "introductory" link.

2. **Beginning of the Session.** The server responds by sending the user a *Session Card*, that is a unique, opaque, identifier that will be used in the following interactions by the client thus identifying itself as "known" for the system.

3. **Session.** An (unlimited) number of interactions user-server where each action is accounted for and used by the server in order to adapt its future responses to the same user.

4. **End Session.** The Session Card is destroyed and no longer valid.

2.3 Knowledge Model

When a hyperdocument is designed for a broad audience, it is customary to face several problems in trying to make its *style* acceptable for novices, amateurs and experts at the same time. They all have different, some time opposite needs: a novice may need information that can be boring for an expert since it makes her loose time in finding the more advanced part of the hyperdocument. On the other side, a novice can be scared and confused by an excessive usage of technicality and leave the site before she could have the chance to access information that are suitable for her experience in the field. Then, great care must be taken to fulfill this important, we dare say paramount, requirement if one is asked to use at its best such an efficient communication media like the World Wide Web.

Another orthogonal[1] need is to help the user (whichever level and experience they have) in navigating through the (sometimes huge) amount of information. This is often achieved by subdividing data in a limited number of *categories* that can be seen as providing an hypertext of a small degree of traditional linearity within the hypertext itself.

[1]Use of the word is not casual, as the patient reader will soon discover.

Putting all together, the view we propose as a model for dealing with the "shapeless" amount of information is a geometric one: let us build a *Knowledge Pyramid*, having as bases two regular polygons of the same shape but of different size, the smallest on the top. Each vertex of the polygon represents one of the categories which the designers subdivided the whole amount in; the edges of the pyramid represents the experience level in each category.

The experience acquired by a user is defined as the surface obtained interpolating the intersection of the current experience levels on each category. The learning process can be seen as trying to get the surface as low as possible.

A possible variation of the model can be to have fewer category on the top than on the bottom. This corresponds to the natural mechanism to avoid mentioning some categories if the user may not be ready and actually may be disturbed by that.

THE KNOWLEDGE PYRAMID

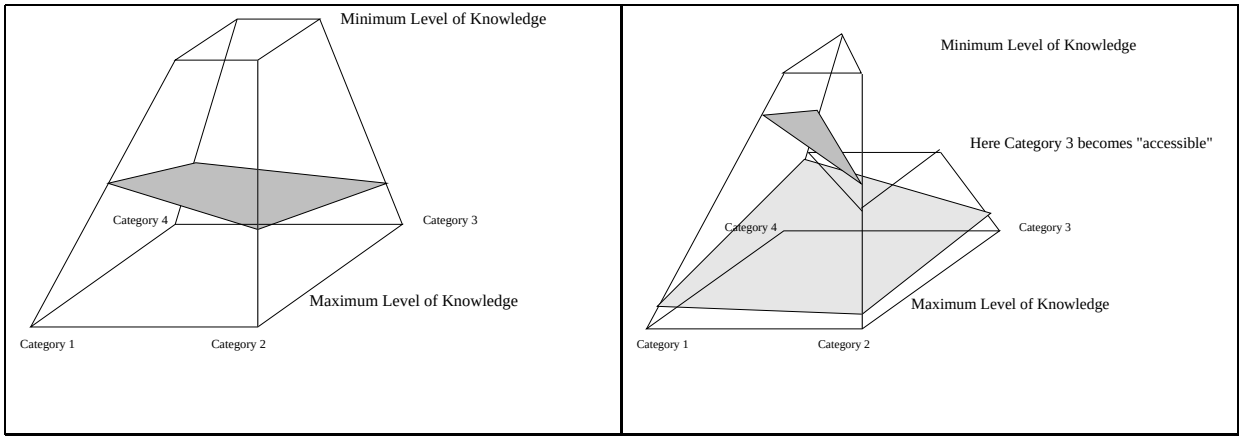

Figure 1: **Left.** The Knowledge Pyramid: four corners represent categories and the surface is the "current level" of knowledge of the subject. **Right.** The Model when Category 3 is accessible only given a certain knowledge of the subject.

CHEOPS uses the Knowledge Pyramid as an important navigational aid to the user: at each step, the user knows what her confidence for each category is and act as a consequence. She may choose to explore a part of the hyperdocument she left behind, or she may want to go deeper in a category or she may go on with her hypertextual visit of the document.

2.4 The Implementation

CHEOPS 0.9 provides the hyperdocument with a mechanism base on the Knowledge Model that makes explicit for the user the access to different categories further differentiated by knowledge level. The mechanism is non-obtrusive both for designers and users. In the first case, in fact, hyperdocument is built as usual, the only difference being the way "internal" anchors are defined (that is as links to CGI-BIN scripts with arguments) and the fact that for each HTML document some information (like category and level) has to be inserted in a configuration file. For the user, nothing is lost of the well-known hypertextual browsing of the document, only few additional capabilities are added and placed on two toolbars, non-obtrusively placed at top and bottom of the page, available at user's will.

The main mechanism is implemented through a CGI-BIN script that takes as input the document required by the user and her session card and builds "on-the-fly" a document that has all the links changed according to the current session card and introduces, if required, two toolbars as header and footer to the page.

We describe, now, the files used by CHEOPS and the CGI-BIN scripts that make part of version 0.9. Then we present a sort of "designer guide" to the usage of CHEOPS.

2.4.1 CHEOPS Support files

Three are the files involved in the system behaviour of CHEOPS 0.9.
The first is a *configuration file* located in the starting directory of the hyperdocument and called `general.dat`. In this file, the designer is asked to introduce (in this version through a text editor) information about the hyperdocument in general and about each HTML file.

The general information about the hyperdocument includes the name and location of the scripts, a prefix used to identify the *user profile* and *history* files (more details follow immediately), location of a temporary directory (inder the Web root) where temporary images can be stored, location of a directory where history files are archived, the starting

page of the hyperdocument, the home page which the user can return to at the end, and the names of the Categories.

For each HTML files, the designer is asked to include information about (*a*) the category and the level the file belongs to, (*b*) if one wants the header and bottom toolbar to appear (more details follow) and (*c*) the context-sensitive help file.

The *user profile* file is generated by one of the scripts at browsing time and contains information about the current levels of expertise the user has reached during her inquiries in each category. User profile is located in the `/tmp` directory and is prefixed as indicated by the designer in the configuration file and suffixed by the session card of the session they refer to. Profile files are not supposed to be altered by hand and are deleted at the end of the session. A recurrent procedure that cleans up profile files should be executed at a fixed amount of time (each day or week) in order to delete information of users who choose to exit the browsing without the explicit "End Session" link.

The *history* file is also script–generated, and holds information, in HTML format, on user requests within the same session. The history file is also kept in the `/tmp` directory and, as well as the user profile file, is prefixed as indicated in the configuration file and suffixed by the session card of the session and is not supposed to be altered by hand. The designer can choose whether history files are deleted or not at the end of the session by setting in the configuration file an empty target directory. We think that history files provide a useful mean to the designer to check efficacy of links and the possibility of elaborate statistics on accesses to the hyperdocument and, therefore, suggest that, at least during a test phase, history files should be saved and checked.

2.4.2 CHEOPS CGI-BIN Scripts

CHEOPS 0.9 consists of several Unix shell scripts (CSH); the main component is the script named `builder`. This script takes two input parameters: the first one is what we call the

session card, while the second is the HTML file actually required by the user. Its output consists of the required HTML file, where (if required by the configuration file) two toolbars are introduced. Three are the main tasks that `builder` is to accomplish:

1. Insert a *header toolbar*.
2. Change the reference to the hyperlinks within the same hyperdocument in such a way to include the current session card.
3. Insert a *bottom toolbar*.

The *header* toolbar offers hyperlinks to the four Category Summaries with the actual confidence level (according to the session card that is a unique identifier in the system) plus hyperlink for the context-sensitive help and a hyperlink to another script that gives the user the chance of explicitely changing her confidence levels (more details follow later in this section). Moreover, the name of the "current category" (that is the one which the document required belongs to) is shown in capital letters and a GIF image is placed on the right part, which represents the current view that user has of the Knowledge Pyramid.

Category Summaries are HTML files written by the designer to help the user to get information on what is available given her degree of expertise as obtained by her previous interactions (called *confidence level* for the category). Depending on the confidence levels, they can be as concise as a single file with little relevant information on the subject or as extended as a thorough review of the material available, with many hyperlinks to other documents.

Access to Category Summaries, further differentiated by current confidence levels, are an important navigational aid for the user: at each step of her browsing through the hyperdocument she can retrieve the starting point in each category, and, automatically among them, the one that is more suitable to her actual knowledge. In fact, the above mentioned non-

obtrusiveness is obtained by not disturbing any of the usual ways of accessing an hypertext but only proposing the user a complementary way of navigating through the hyperdocument.

The advantages of having this information displayed on the header toolbar is twofold: on one hand, they provide (through the above mentioned links to Category Summaries) an easy way to visit some places of the hyperdocument that were skimmed over the first time; on the other hand they are also a useful way of helping the user to keep track both of the current category (and, therefore, of the "section" of the hyperdocument currently shown) and of the level of knowledge in each category that she supposedly acquired by navigating through the hyperdocument.

Image of Knowledge Pyramid is built "on-the-fly" through a C program called `gifcreate` that uses GD 1.2, a graphic library for GIF creation [6]. The image is prefixed and suffixed as profile and history files and is located in the directory indicated in the configuration file.

At this point, after inserting (if so required by the configuration file) the header toolbar, the script provides as output the HTML file required, taking care of instantiating each occurence of the pattern `SESSION_CARD` with the value of the session card passed as first parameter. The designer takes care, in fact, to insert as first parameter of the internal links (i.e. links to the `builder` script) the pattern `SESSION_CARD`. That allows CHEOPS recognizing further choices made within the same session.

The *bottom toolbar* gives hyperlinks to the beginning of the whole document, to another script that let the user see the history of her browsing of the hyperdocument, to the "End Session" part of the session protocol previously described and to a general help file.

The second script involved in CHEOPS is the `level-modification` script. It is important to give the user the chance to modify current confidence levels in such a way to access, if desired, more complete information on a subject or if the user realizes that "she has gone too far" and is confused by the technicalities presented at that level.

This script proposes the user with a form where she can modify confidence levels, where

the current value is shown as a default choice. The `action` associated with the form is the `builder` script itself. This gave an additional degree of difficulty to the `builder` script that has to recognize whether it has been called from an "ordinary" link (that is with two parameters) or it has been called from the `level-modification`. A *Back* hyperlink is also provided to give the user the possibility to go back to the previous document without changing levels.

The third script is the `view-history` script that allows the user to see the history file that is step-by-step updated by `builder` each time the user takes an action. Each entry is available as an hyperlink so that the user can access files that she previously visited. Each manual levels modification (i.e. through the `level-modification` script) is also shown.

2.4.3 The Session Protocol

The Session Protocol described at the beginning of this section is implemented through different ways of setting the first parameter (i.e. the session card) of `builder` script. At the beginning, the designer provides an entry point to the hyperdocument by calling the `builder` script with first parameter set to `START`. In this case `builder` will have (transparently to the user) a different behaviour, taking care of getting a new session card and initializing the history file before doing the ordinary actions previously described.

The "Close Session" hyperlink provided in the bottom toolbar is a link to `builder` with first parameter set to `END` and, in this case, the script will take care to provide the user with a conclusive page that has the home page link as indicated in the configuration file. The script will also delete the user profile from the `/tmp` directory and move the history file to some other directory so that there is no possible interference with successive sessions.

2.5 How a designer can use CHEOPS

CHEOPS 0.9 is designed taking into account the needs of designers: not much additional

work is required in order to make a hyperdocument adaptive to user's responses.

Here is a brief check-list for the designer:

- Design the WWW hyperdocument as usual: HTML files with hyperlinks pointing to other files within the whole hyperdocument.
- Categorize the files in 4 categories with 3 knowledge levels. Version 1.0, actually under development, will remove any constraint on the number of categories and confidence levels.
- Edit the default configuration file provided with the distribution. Insert all the information relevant to the hyperdocument being designed as its directory, home page, the first file to be shown, names of the categories and their *short names* and so on. Short names of categories are strings with no spaces, used by `builder` to get the name of the files containing Category Summaries, indexed by the confidence level.
- Build the Category summaries; often as the experience suggests, summaries are already been developed during the usual design of the hyperdocument.
- Change the internal links so that they call the `builder` script with first parameter `SESSION_CARD` and second parameter the name of the file which the link points to. This is a step that will be removed by version 1.0 that will change automatically internal links through the `builder` script.
- Provide an entry point to the document as a call to `builder` with first parameter START.

3 Conclusions

The design of CHEOPS has been highly influenced by a project in collaboration with Echos Studio Media (a Musicology association) of an hypertextual document about Giovan Battista Pergolesi, a Neapolitan composer of the XVII century. In such a framework, we realized that our design choices to obtain adaptivity were general and not specific to that document.

Hyperdocuments on Theory of Music represent a useful paradygm since the argument can be accessed both by a novice (interested in classical music) and by an expert that can be interested in accessing (for example) Pergolesi's handwriting manuscript in order to check and ascribe (or definitely do not ascribe) to him some newly discovered manuscript.

Case study was provided with all the characteristics now in CHEOPS plus some additional capabilities like an automatic mechanism to perform guided Tours (like to Bush's trails[5], Trigg's guided tours[17] and "Footstep" mechanism[15]).

3.1 Future Developments

CHEOPS 0.9 is actually undergoing further modifications and improvements toward version 1.0 that should be available soon. Information on the updated version of CHEOPS can be retrieved at

[URL: `http://stromboli.dia.unisa.it/CHEOPS/`]

where a demo of its capabilities is also available. The hyperdocument developed in collaboration with *Echos Studio Media* on *Giovan Battista Pergolesi* is fully operational and it is available (in italian) at

[URL: `http://www.xcom.it/pergolesi`]

Modifications and improvements toward version 1.0 are the following:

- Chance for the designer to provide in the configuration file the number of categories (i.e. not limited to 4 as it is in version 0.9) and the the number of available confidence

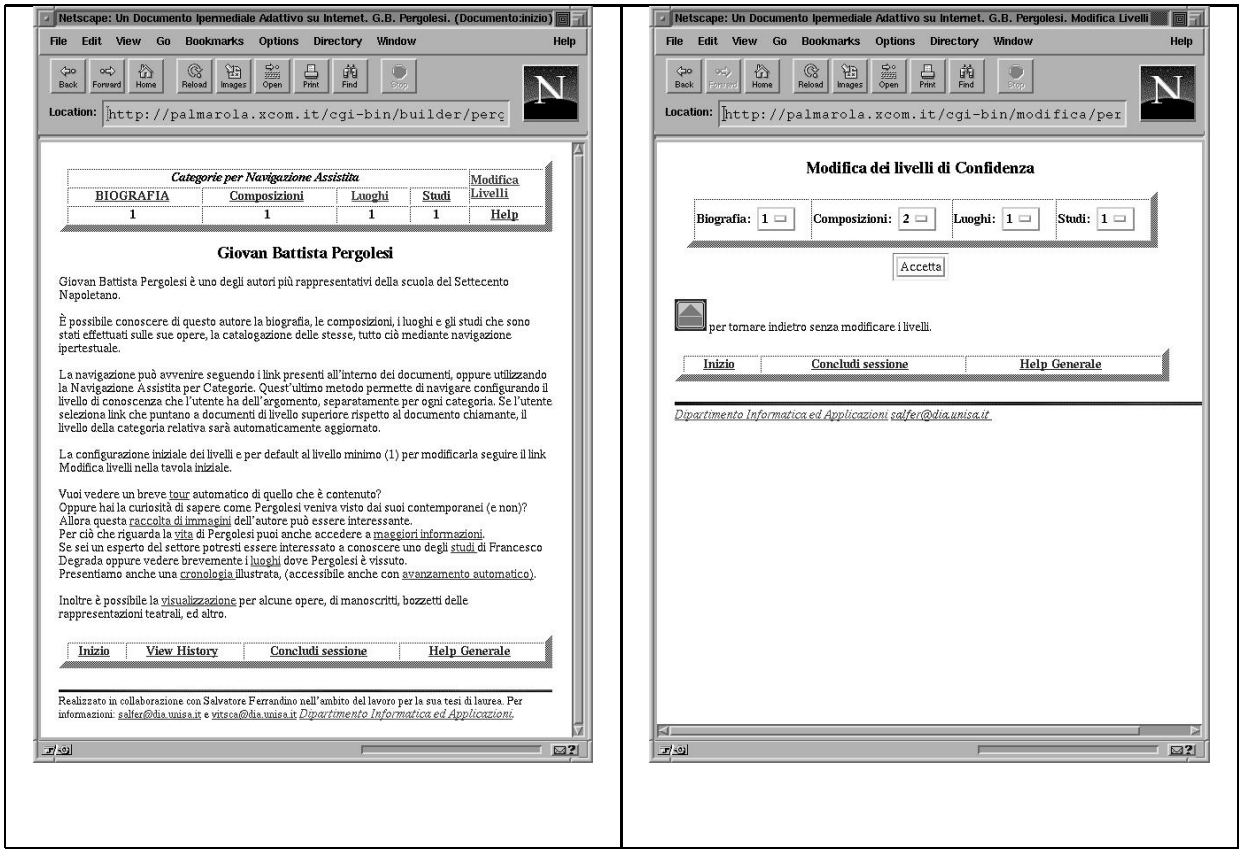

Figure 2: **Left.** The starting point of the hyperdocument on Pergolesi. Notice that *Biografia* is in capital letters (being the current category) and that levels are all at novice level (i.e. 1). **Right.** The form to change confidence levels (*Modifica livelli di confidenza* in italian).

levels (currently 3 levels are allowed).

- Include an automatic mechanism to present "long" HTML files in several pieces linked together through `Next` and `Previous Page` links.

- Change not only the pattern `SESSION_CARD` but the whole URL, so that the designer can build her hyperdocument as usual, test links correctness and then let CHEOPS add the adaptivity to it.

- Design and realization of a mechanism that supports the construction of tours.

Future versions of CHEOPS will also include the following modifications (in increasing order of significance):

- Add a user fiendly interface for editing the configuration file.
- Add the possibility to define a multi-faceted knowledge pyramid (fewer categories on the top and more on the bottom).
- Design and realize a mechanism that supports the construction of adaptive tours, that is based on the amount of the hyperdocument yet to be visited, and/or tours by "degree of expertize" acquired by the user.
- Give the possibility to the designer to insert a more exotic mechanisms for the adaptivity. At this moment, the adaptivity is based on our knowledge model: a user is "given" only the "cross-section" of the knowledge pyramid according to her profile. For example, a different mechanism could be an average of the experience level of the past documents read. The modification should give the designer, at least, a choice among several adaptivity response algorithms.
- Design and realization of a mechanism to allow the "enrollment" of users so that they can re-use hyperdocuments starting where they left. The idea is to enclose the session-based protocol within a layer that supports user-based access like autenthication, assigning access password and user management.
- Future versions of CHEOPS should be multiplatform and so written in a portable language like C, PERL or TCL.

Acknowledgments

It is a sincere pleasure to acknowledge an intense collaboration in the preparation of the WWW hyperdocument on Pergolesi with *Echos Studio Media* and, in particular, with *Maestro* Eugenio Ottieri. In the same project, we also thank Walter Balzano for several interesting discussions.

References

[1] T. Berners-Lee. "*World Wide Web Initiative*". WWW Home Page.
[URL: `http://info.cern.ch/hypertext/WWW/TheProject.html`]

[2] T. Berners-Lee, R. Cailliau, J.F.Groff. "*The World Wide Web*". Computer Networks and ISDN Systems, Nov. 1992, vol.25 (no.4-5), 454-9.

[3] T. Berners-Lee. "*Hypertext Transfer Protocol*". Internet Draft.

[4] D. Dwyer, K. Barbieri, H.M. Doerr. "*Creating a Virtual Classroom for Interactive Education on the Web*". Proc. of WWW 95, Third International Conference on World Wide Web.

[5] V. Bush. "*Memex Revisited*". In *Science is not enough*, W.Morrow and Co. Reprinted in Nyce, J.M. and Kahn, P. (Eds.) (1991), From to Hypertext: Vannevar Bush and the Mind's Machine. Academic Press.

[6] T.Boutell, "GD 1.2, *A graphics Library for fast GIF creation*". Quest Protein Database Center, Cold Spring Harbor Labs (US).

[7] P.M. Hallam-Baker, D.Connolly. "*Session Identification URI*". W3C Working Draft WD-session-id-960221.
[URL: `http://www.w3.org/pub/WWW/TR/WD-session-id.html`]

[8] R.Fielding, H. Frystyck, T. Berners-Lee. "*Hypertext Transfer Protocol,* HTTP *1.1*". HTTP Working Group Internet Draft.

[9] N. Hammond, L. Allison. "*Extending Hypertext for learning: An investigation of access and guidance tools*". In Sutcliffe,A. and Macaulay, L. (Eds.): *People and Computers*, Cambridge University Press, 1989.

[10] B. Ibrahim, "*World-Wide Algorithm Animation*". *Computer networks and ISDN Systems*, Vol. 27 No.2, Nov. 1994, Special Issue of the First World Wide Web Conference, pp.255-265.

[11] B. Ibrahim, S.D. Franklin. "*Advanced Educational Uses of the World Wide Web*". Proc. of WWW95, 3rd International Conference on World Wide Web.

[12] D.K. Kristol. "*Proposed* HTTP *State Management Mechanism*". HTTP Working Group Internet Draft (2/22/96).

[13] Netscape Communications Corporation. "*Persistent Client State* HTTP *Cookies*". 1995.

[14] NCSA Mosaic Project (1994). "*NCSA Mosaic Home Page*".
[URL: `http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/NCSAMosaicHome.html`]

[15] D. Nicol, C. Smeaton, A. Falconer Slater. "*Footsteps: Trail-blazing the Web*". Proc. of WWW 95, Third Internazional Conference on World Wide Web.
[URL: `http://www.igd.fhg.de/www/www95/proceedings/papers/60/footsteps.html`]

[16] J. Nielsen. "*Hypertext and Hypermedia*". Academic Press Ltd, 1990.

[17] R.H. Trigg. "*Guided Tours and tabletops: Tools for communicating in a hypertext environment*". ACM Transactions on Office Information Systems 6,4 (October 1988).