Workgroup Hypermedia Editor: a tool to support a strategy for co-operative hypermedia production

M. Cossentino, U. Lo Faso
Dipartimento di Ingegneria Automatica ed Informatica
University of Palermo - Viale delle Scienze – 90128, Palermo - Italy
maxco@unipa.it, lofaso@unipa.it

Abstract: A software tool for building hypermedia in a co-operative distributed environment is presented which supports both authoring and assembling processes and publishes automatically the work to the Internet. The design methodology and the hypermedia structure are put into evidence.

Keywords: Hypermedia, Internet, Computer Supported Co-operative Work, Tools for CSCL.

1 Introduction

Human processes are often concerned with the co-ordination of people and organisations toward a common aim. In the last years, organisations have been involved in identifying and optimising their information processes in order to induce new collaborative working styles based on the use of IT. The dimension of this problem has grown-up due to the transition from a local to a geographical dimension of working scenarios. In this new situation, suitable software tools have a strategic value to let managers co-ordinate the activity of many persons and to let workers interact and co-operate at a distance.

2 The problem

In our scenario many authors, geographically dispersed, co-operate to write a hypermedia: our aim is building a software tool suitable to support and manage this type of work.
Each author in his production activity has to know the structure of the hypertext because, probably, he needs to insert links from his own pages to other ones. At the same time other authors are working on the pages that the first author is referring to.
Obviously it is desirable that contributions from different authors share the same structure and style.
More precisely, in the special case of a multimedia distributed authoring environment, a workware application should have these specific functions:

- to support the definition of the structure of the whole hypermedia;
- to prescribe format and styles;
- to let an editorial office:
  - modularise the work and assign modules to authors,
  - collect authored modules coming from various places at different moments,
  - assemble the modules;
  - guide authors to observe the prescriptions about structure and style.

3 The theoretical background

The scientific literature of the latest years counts many works on the design of hypertext and on co-operative work. In the design of our application, with reference to the hypertext design, we have taken into account the following conceptual contributes:

1) The Dexter model [1],[2],[3] that offers a standard terminology and a widely used hypertext formal model. The Dexter Model consists of 3 layers (run-time, storage, within-component). The main focus of the model is in the storage layer which describes the network of nodes and links that are the essence of hypertext. In the model, nodes and links are called components which can be of three types: atomic, composite components (made of other components) and link-components.
2) The RMM (Relationship Management Methodology) [4] for the data structure and navigation design. The authors of RMM say: "The class of applications for which RMM is most suited exhibits a regular structure for the domain of interest, i.e., there are class of objects, definable relationships between these classes and multiple instances of objects within each class". Our problem satisfies this requirement. Moreover, the RMM is centred upon entities (like in HDM [7], [8]) that 'represent abstract or physical objects', and upon their attributes and relationships. In our design methodology Entity Design, Entity-Relationship Design, and Navigation Design have a particularly important role. Navigating in the hypertext is supported by six primitives: unidirectional link, bi-directional link, grouping, conditional index, conditional guided tour and conditional indexed guided tour.

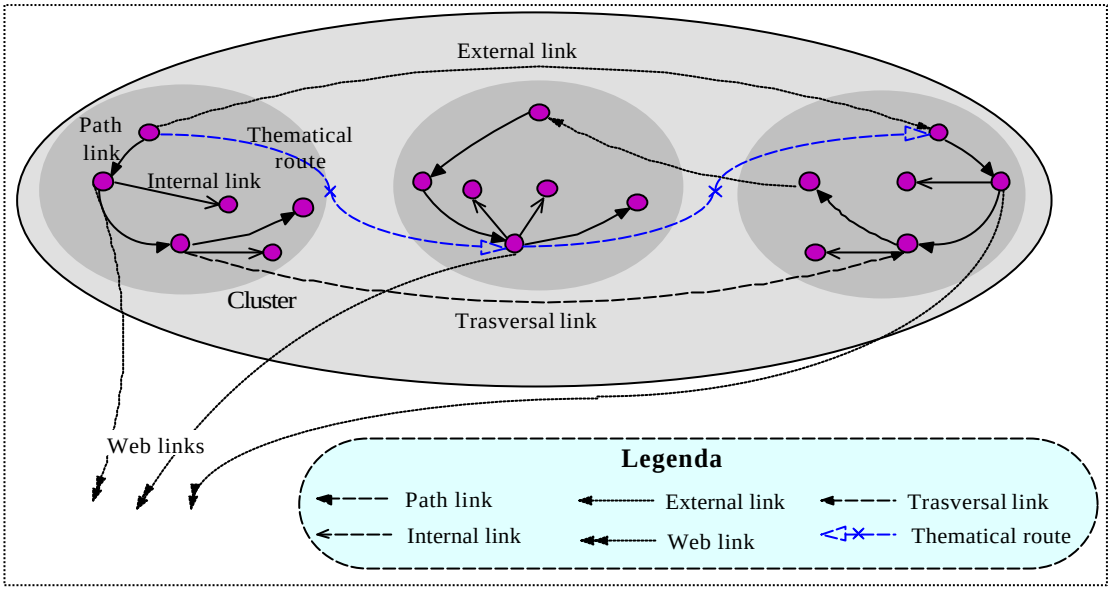

Fig. 1 - The main structure of the hypermedia is a graph whose nodes (grapes) are grouped by argument (cluster). Branches in the graph represent the various types of hypermedia links.

3) In OOHDM (Object Oriented Hypertext Design Methodology, [9]) the design process begins with conceptual design in which firstly the domain semantic is captured, then the navigational design is performed interpreting the navigational model as a view of the conceptual model. Navigational design is made using the concept of navigational context (a set of nodes, links, context classes and other navigational contexts). Authors of OOHDM used ADV [5] for the interface design.
4) About co-operative aspects, Gronbaeck et al. [2,3] underline that the authoring process in large projects leads to a co-operative work between several authors and they list some different kinds of co-operation between them. They classify various different modalities of co-operation depending on the different possibilities of modifying and reading the same hypertext part at the same time.
5) Butler et al. [16] connect software design with the structure of the workflow in the context where the software will be used. They propose a precise lifecycle in which the design of software has to be done keeping in mind the workflow and the influence on it due to the presence of software.
6) According to Nanard and Nanard [15] the hypermedia production process can be represented in a two dimension space. The first dimension is connected to the design techniques applied to the hypertext structure, the second dimension is relative to designer's mental processes moving through different abstraction levels of his plan.

4 The proposed solution

Our software application supports a distributed, asynchronous multimedia authoring process. For the sake of clarity we describe how things work with reference to a real context: suppose, for instance, that a group of universities decide to write together an encyclopaedic hypermedia on some discipline.
First of all an editorial staff, having co-ordination responsibility, has to be established in one of the universities chosen to have a leading role within the group. The authoring group is supposed to be composed of professors and researchers in the various universities, familiar with personal computers simply as users, not as technicians. The whole work has to be controlled and managed through the software. Both the editorial central staff and the authors are supposed to be connected to the Internet.
The editorial central staff writes a list of arguments (called clusters) to be dealt with and assigns each of them to an author. Each author can develop his work as he likes: he can use as many grapes (pages of the hypertext) in his clusters as he needs and can also connect them by internal links or order them creating a path. For the external links (i.e. the links going outside a cluster), any author receives from the editorial central

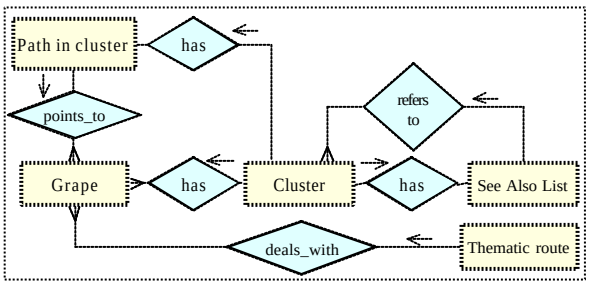

Fig. 2 - The Entity-Relationship diagram of the hypermedia

staff, the whole list of all the clusters and so he can establish links to any argument.
At end, when his work will be finished, each author will send his file to the editorial central staff. The software will assemble automatically the various contributions and at this point, the co-ordinators can add other transversal links (from a cluster to a grape of another cluster, between grapes of different clusters).

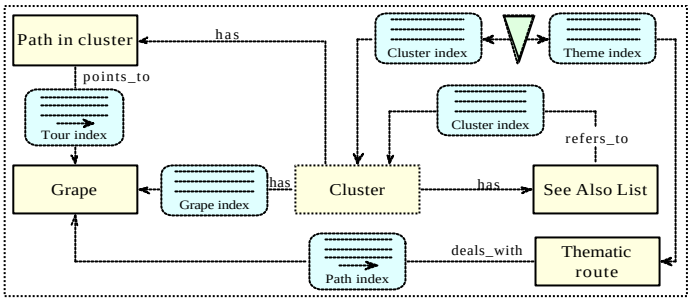

Fig. 3 - This diagram illustrates the navigational connections between entities of the E-R diagram

Finally end-users will be allowed to navigate through the hypermedia by an Internet browser.
Obviously, the software can support other different scenarios, from educational situations where students of many schools co-operate to build their own hypertext to business situations where many department of a firm co-operate to build a catalogue.
As we have already said, the nodes of the hypermedia (grapes) are grouped into clusters (see fig. 1). A cluster represents an argument while a grape (containing text, image, sound, …) represents one of the elements used by an author to develop the argument.
In the following sections will be described:
1. the structure of hypermedia that is possible to build with the software;
2. the software design.

5 The structure of the hypermedia

We could express the elements represented in fig. 1 using the Dexter notation. We can consider grapes as atomic components, clusters as composite components and links as link-components.
A natural way to implement this structure could be a database. The web pages of the hypertext would result from queries of this database.

5.1 The design with RMM

We use the RMM in order to design the structure of hypermedia that our software will implement.
The choice of this methodology instead of other ones (HDM [7] e [8], OOHDM[9], MBPAM [10]) is influenced by the fact that we consider this approach more useful for an application which is not an hypertext but is a tool for building a class of hypertexts. According to RMM, in the following three sections, we deal with E-R Design, Entity Design, Navigation Design.

5.1.1 E-R Design

The diagram of fig. 2 illustrates the system structure from an entity-relationship (E-R) point of view which is the first step of the RMM design methodology. In the diagram we can see that a “cluster” (e.g.: the cat) contains many “grapes” (e.g.: behaviour, diet, morphology, breeds, …). It also has a “See Also List” which refers to other clusters dealing with other related arguments (e.g.: other felines).
Clusters can be visited through the “Path in cluster” entity which lists a subset of the cluster’s grapes (e.g.: morphology, behavior) that constitute a basic description of the argument treated in the cluster.
A “thematical route” crosses grapes illustrating some kind of transversal argument (e.g.: digestion in mammals); grapes in this case, obviously, will not belong to the same cluster.

5.1.2 Entity design

Information relative to each entity should be now divided into units to be presented as separated but interrelated wholes. Each unit, called slice, groups one or more attributes of the entity. All these aspects are strictly dependent on the own author’s choices: in order to advice authors and to assure a style uniformity of contributions, we introduce a set of layout (see section 5.2.1) which can be used to format each grape. The central editorial staff will be able to introduce as many layouts as needed.

5.1.3 Navigation design

From the E-R diagram we can obtain the diagram of fig. 3 that illustrates several kinds of navigational connections between entities. In this figure:
a) rectangles represent entities (as in the E-R diagram),

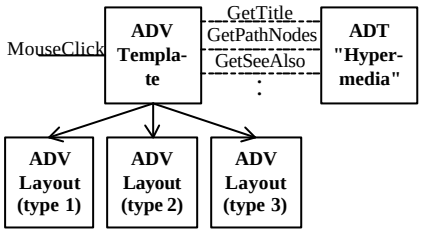

Fig. 4 - This Configuration Diagram shows the inheritance relationship between the ADV of pages (ADV Layout (type x)) and ADV Template from which some elements are derived

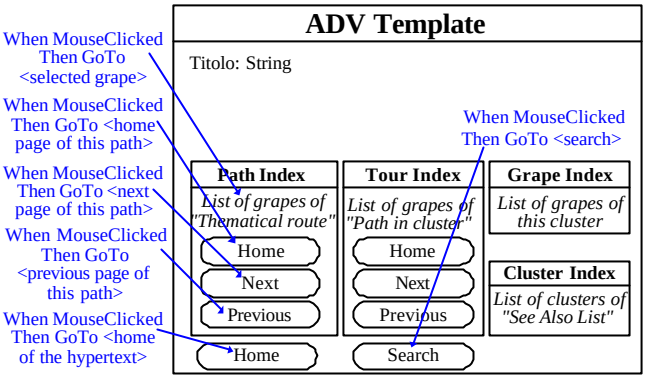

Fig. 5 – Detail of the configuration diagram for ADV Template

b) rectangles with rounded edges represent navigational indexes, the presence of the arrow in some of these indexes indicates that their elements will be visited, by default, in sequence (guided tours);
c) continuous lines represent navigational links
d) dashed lines represent the relationships already present in the E-R diagram.

We can explore this diagram starting from the triangle which represents a grouping primitive putting together the "Cluster index" and the "Theme index". This new composed group is implemented in the home page of the whole hypertext from which the user can navigate to clusters and thematic routes.
The "Cluster index" lists all the clusters in the hypertext, the "Theme index" lists the thematic routes created by the editorial central Staff. Each thematic route is a guided tour ("Path index") that illustrates a certain argument crossing transversally different grapes.
The cluster can be visited following a path created by the author with *path links* (see fig.1). The steps of this path are listed in the "Tour Index". Another way of visiting a cluster is starting from each element of the list of its grapes ("Grape index").
From a cluster the user can make an associative jump using the "See Also List" which contains a list of clusters dealing with related arguments.

5.2 Interface design

In order to compose pages dealing with different arguments, we suppose that authors need at least of:

- pages with text and one only image,
- pages with images with or without text,
- pages with movies or sounds and associated buttons.

For this reason and in order to guarantee some style uniformity to the whole work, we have decided to introduce a set of layout pages. Depending on his needs, the author should choose between pages containing a different number of images with or without captions, with shorter or longer text fields, movies, sounds,…
In order to specify the structure of these pages we use ADVcharts [5].

5.2.1 The structure of pages produced by authors

Authors can choose between several layouts having some elements in common: title, navigational buttons (Home, Next, Previous, …); obviously, pages using these layouts will differ for their content.
This kind of structure suggests an implementation based on a simple inheritance relation between a page-template and various page-layouts (many layouts inherit the same template): the configuration diagram in Fig.4 shows this relation.
The ADV (Abstract Data View) Template (see fig.5) contains any element which is useful for navigating the hypertext with the only exception of any link created by the author internally to his texts. We can find:

- Buttons for navigation in the whole hypertext (home, search);
- The "Tour Index": a list of the grapes of the "Path in Cluster"
- Buttons for navigating the "Path in Cluster" when the current node belongs to the "Path in cluster" of the current cluster (home, next, previous);

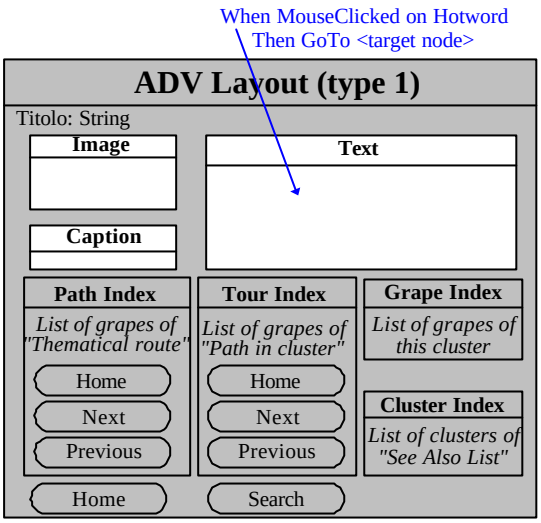

Fig. 6 – Detail of the configuration diagram for ADV Layout (type 1)

Use Case	Software Module	User
Author archive creation	CreateNodes	Editorial Central Staff
Assignement of clusters to authors		
Cluster Creation		
Reading of assigned Clusters	WriteNodes	Author
Edit Nodes		
Export Nodes		
Nodes Assembly	Assemble Nodes	Editorial Central Staff
Edit Nodes for Central Staff		
Publication		
Web server DB	Hosting web server	
Navigation with Browser	Browser	End User

Table 1 – Use case and software mo dules

- The “Path Index”: a list of steps of “Thematic route” when the current node belongs to a thematic route;
- Buttons for navigating the “Thematic route” (home, next, previous);
- The “Grape Index”: a list of the grapes of the current cluster;
- The “See-Also Index”: a cluster list dealing with arguments connected to the current one.

The ADV Template also contains the title of the page.

To provide these functions (see dashed lines in fig. 4) some services are required from the ADV to the ADT (Abstract Data Type). In the list of services we can find:

- GetTitle: exports the title of the page;
- GetSeeAlso: exports the list of the arguments which are connected to the current page.
- GetThematicNodes: exports the list of nodes of the thematic route containing the current page.
- … and so on.

As already said, the ADV Layout inherits the ADV Template. The detail of ADV Layout structure is represented in fig. 6, where we find a possible configuration: an image field, an image caption text field and, finally, a main text field.

As already done, we should now draw a configuration diagram also for the ADV Layout. In this diagram we could see various services (GetImage(i), GetCaption(i), GetText(i)), the display function and the mouse management function (as prescribed in fig.4).

The services Getimage, GetCaption, GetText are parameterised in order to be reused in different configurations.

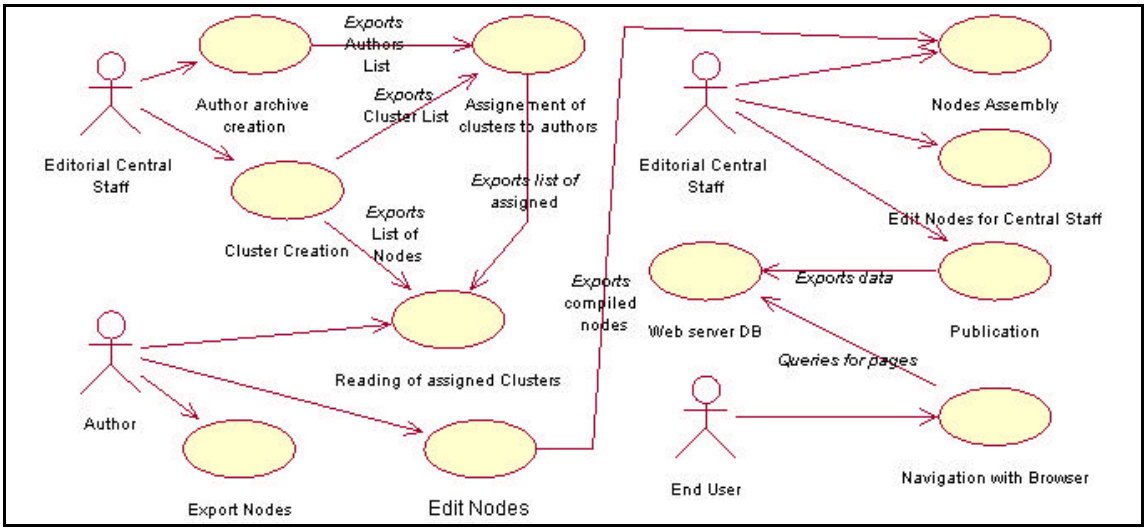

Fig. 7 – Use case diagram of the system

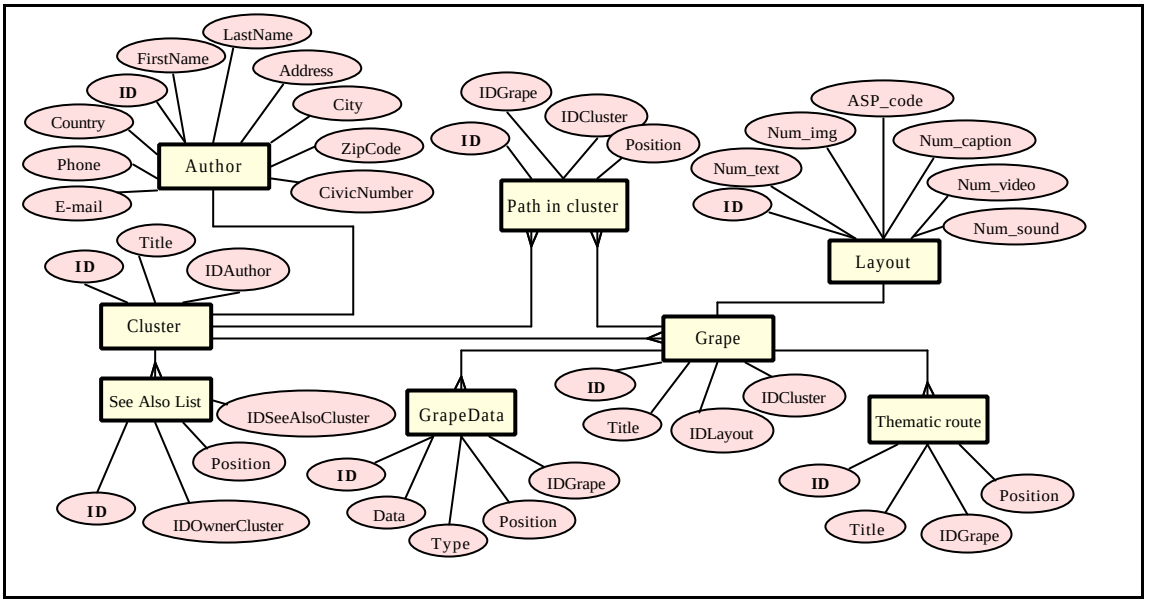

Fig. 8 – The structure of the DB can be deduced from this ERD

6 The software design and implementation

The software has been designed using UML (Unified Modelling Language) [11], [12].

The **use case diagram** in fig. 7 illustrates the different phases of the hypermedia production. In the first phase the central editorial staff (upper-left side of the diagram) is the actor. The use case "Author Archive Creation" deals with building the archive containing authors' data; this data will be useful to editorial central staff for communication and data exchange purposes. The editorial central staff also creates the list of clusters that compose the hypermedia (use case "Cluster Creation"). From these two use cases, the list of authors and the list of clusters are exported to the "Assignment of Clusters to Authors" use case; the last operation of this phase is the delivery of the hypermedia skeleton data to authors (use case "Sending HM skeleton data").

In the second phase the actor is the author. The "Reading of assigned clusters" use case imports data that author will use for his work. These data contain: the list of all clusters of the hypermedia, the list of the clusters assigned to the author, the list of the available layouts. The author, first, uses this data to construct his own part of the hypermedia (use case "Edit Nodes") and then will send back the result to the editorial central staff.

In the third phase (upper-right side of the diagram) the actor is again the editorial central staff who will receive and collect the contributions from all the authors. With these data the skeleton of the hypermedia will be stuffed and the DB structure will be completed. Nodes can be modified and new links can be added (trasversal links and thematic route links). At last the hypermedia will be published trasferring all data to the web server.

In the last phase actor is the end user; he can navigate the hypermedia using his own browser.

From the **implementation** point of view several application programs have been produced, one for each of the four phases described in the use case diagram. Table 1 describes the connections between modules and use cases.

A prototype has been developed in which all the modules are coded in Visual Basic, ver. 6.0. This language has been chosen for its rapid development feature, for its good support of ASP technology and for its DB interfacing capability.

6.1 Data structure

Taking into account the structure shown in fig. 2, we can define the database that we use to contain the whole hypermedia (see the E-R diagram in fig. 8). In the E-R diagram we see the following entities:

- Grape: used to collect (for each grape) the assigned layout and the cluster which contains the grape.
- GrapeData: in which we can find the data stuffing the layout of the grape. The value of the field "Type" is used to declare the type of data contained in the field "Data": a text, a caption, the name of a graphic file, the name of a sound file, …
- Cluster: containing the title of the cluster, the author's information.

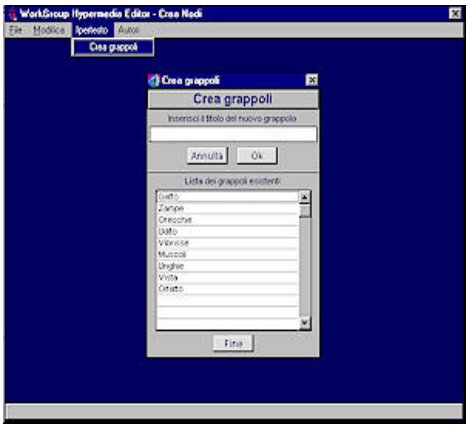

Fig. 9 – The creation of the list of the clusters

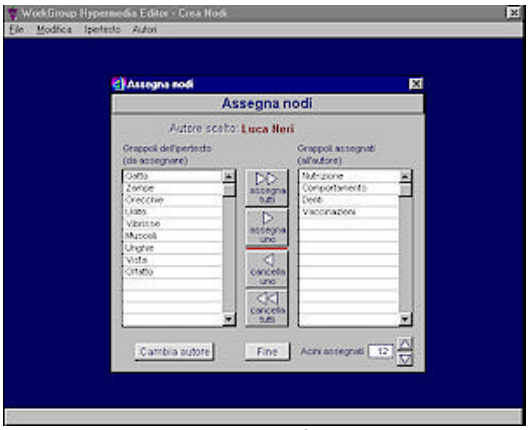

Fig. 10 – The assignment of clusters to authors

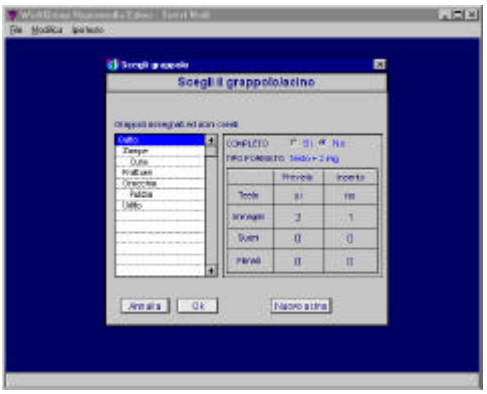

Fig. 11 – The choice of the cluster/grape to edit

Fig. 12 – Editing a grape

- SeeAlsoList: in which we can find a list of clusters which can be associated to the argument treated in the current cluster.
- PathIncluster and ThematicRoute: in which we can find lists of grapes or clusters forming navigation units.
- Layout: in which we can find the layout which can be used to format a grape. In the field "ASP_code" is stored the ASP code[1] necessary for creating the dynamic page; other fields in the table describe the page structure (number of text field, images, sounds, …).
- Author: in this table we can find any information relative to authors.

7 The prototype

The prototype we have already developed is composed of the first three modules of table 1.
Now we will describe the functionalities of the various parts of the software through interfaces.

7.1 The "CreateNodes" Module

The editorial central staff creates all the clusters and give them a name as shown in fig. 9.
Then the editorial central staff creates an archive of the authors that will be involved in the hypertext production and divide the grapes among each them (fig. 10).
At this point all the information have been provided and the program prepares the authors' data file. In these file each author can find the structure of the whole hypertext and the list of the clusters that he has to write. The files are sent to the authors together with the WriteNodes program via e-mail.

7.2 The "WriteNodes" module

Each author receives his own data file, opens it in his copy of WriteNodes, chooses a grape of a cluster (fig. 11) and begins its composition.

[1] The hypertext will be presented to the user using ASP pages. When an user request is sent to the server, data will be extracted from the DB as described in the code of the ASP page, a new page will be built on the fly and then sent to the user who can read it with almost any modern browser.

During the editing phase (fig. 12), the author can introduce a text clicking on the text field that (in this layout) is in the upper right side of the page. It is also possible to create links to other grapes of this cluster and to the first grape of other clusters.
When an author has completed his work he can assemble and send all the files (text, images, …) to the central editorial staff.

7.3 The "AssembleNodes" module

The central editorial staff, collects the contributions of all the authors, imports them into the database of the whole hypertext and then publishes it over the Internet.
At this point the work is finished and the final user can navigate the hypertext

8 Conclusion

In this work a co-operative software tool for building hypermedia in a geographically distributed authoring environment has been presented. The package supports both authoring and assembling processes and publishes automatically the whole work to the Internet.
At present the software requires that authors use the same language (e.g. English, Italian, …): this appears to be a rather serious limitations if an international co-operative context is considered. A further release should be developed implementing some facilities suitable to guide and manage the multiple translation process needed when a multilingual work has to be produced.

References

1. F. Halasz, M. Schwartz – "The Dexter hypertext reference model" - Comm. of ACM Feb 94, vol.37, No.2, p. 30
2. Kaj Grønbaek, R. H. Trigg - "Design issues for a Dexter based hypermedia system" - Comm. ACM, 37,2, Feb 1994
3. K. Grønbaek, J.A. Hem, O. L. Madsen, L. Sloth - "Cooperative Hypermedia Systems: A Dexter Based Architecture" - Comm. ACM, 37, 2, Feb. 1994
4. T. Isakowitz, E. Stohhr, P. Balasubramanian - "RMM: A Methodology for Structured Hypermedia Design", Comm. of ACM Aug 95, vol.38, No.8, p. 34
5. L.M.F. Carneiro, D.D. Cowan, C.J.P. Lucena - "ADVcharts: A Graphical Specification for Abstract Data Views" - Proc. CASCON '93 - Toronto, Canada.
6. D.D. Cowan - "Abstract Data Views: An Interface Specification Concept to Enhance Design for Reuse" - IEEE Trans. Soft. Eng., 21, 3, Mar 1995
7. F. Garzotto, P. Paolini, D. Schwabe - "HDM - A model based approach to hypertext application design". ACM Trans. on Inf. Syst. 11, 1 - Jan 1993.
8. F. Garzotto, L. Mainetti, P. Paolini - "Hypermedia design, analysis and evaluation issues". Comm. ACM 38, 8 - Aug. 1995.
9. D. Schwabe, G. Rossi, S. D. J. Barbosa - "Systematic Hypermedia Application Design with OOHDM" - HyperText '96 proc.
10. D.B. Lowe, A.J. Bucknell, R.G. Webby - "Improving Hypermedia Development: A reference Model-Based Process Assesment Method" – Proc. ACM Conf. Hypertext '99 – Feb. 1999 - ACM Press
11. J. Rumbaugh, I. Jacobson, G. Booch - "The Unified Modelling Language Reference Manual" - Addison Wesley ed.
12. I. Jacobson, G. Booch, J. Rumbaugh – "The Unified Process" – IEEE Soft. May/June 1999
13. M. Accascina, M. Cossentino, U. Lo Faso – "Un progetto per la redazione di ipertesti cooperativi a distanza" – Proc. of NIR-IT '99 [on-line at: http://www.cilea.it/nir-it/1999/RA/agendafinale-v.html]
14. M. Cossentino, U. Lo Faso, M. Spagnolo – "Produzione di ipertesti multimediali cooperativi" – Didamatica 2000 (AICA) – May 2000 - accepted paper
15. J. Nanard, M. Nanard – "Hypertext Design Environments and the Hypertext Design Process" – Comm. ACM, 38,8, Aug. 1995
16. K. A. Butler, C. Esposito, and R. Hebron - "Connecting the design of software to the design of work". Comm. ACM, 42, 1 - Jan 1999