Published in the proceedings of IWWOST 2001,
http://www.dsic.upv.es/~west2001/iwwost01/, Valencia, Spain (2001)

The Conference Review System with WSDM

Olga De Troyer, Sven Casteleyn
Vrije Universiteit Brussel
WISE Research group
Pleinlaan 2, B-1050 Brussel, Belgium
Olga.DeTroyer@vub.ac.be, svcastel@vub.ac.be

1 Introduction

In this paper we elaborate the Conference Review System [1] using the WSDM method [2,3,4]. Because the problem statement for the Conference Review System is on some aspects (deliberately) vague, we have taken the following assumptions:

- The system is dealing with only one conference
- At installation, the PC Chair is register as such and receives a username and password. This is not covered by the web-application.

The paper is structured as follows. In section 2 we give a short overview of the different phases of WSDM. In the next sections we elaborate the Conference Review case study. Section 3 gives the mission statement of this web application, section 4 deals with the audience modeling and in section 5 we make the conceptual design of the web site. In section 6, we deal with some aspects of the Implementation design. The case study is too large to discuss all aspects in detail. Therefore only parts are elaborated in the different sections. A more elaborated design is given in the Appendix.

2 WSDM Overview

The main characteristic of WSDM is the audience-driven approach. This means that instead of letting the structure of the available data set drive the design of the web site, as in most methods, we create a web site based on the requirements of the intended audience(s). In this way, WSDM gives consideration to the fact that web sites usually have different types of visitors that may have different needs.

A second important characteristic of WSDM is the distinction between the conceptual design (which is free from any implementation detail) and the design of the actual presentation: the grouping in pages, the use of menus, static and dynamic links, etc. This distinction is similar to the distinction made in database design between the conceptual schema (e.g. an E-R schema) and the logical schema (e.g. a relational schema). This distinction has proven its usefulness for more than 15 years. It allows to make web site

designs that are not biased by the diversity and rapid growing obsolescence of the web technology as well as by current implementation limitations.

In figure 1 an overview of the WSDM method is given. The first step is to define the *Mission Statement*. The Mission Statement should express the purpose and the subject of the web site and declare the target audience. Based on this Mission Statement a two-step *Audience Modeling* phase is performed. In the first step, *Audience Classification*, the different kinds of users are identified and classified (i.e. their Audience Class is formally identified). Members of the same Audience Class have the same information and functional requirements. In the next step, called *Audience Class Characterization*, the characteristics of the different Audience Classes are given. The result of the Audience Modeling is a hierarchy of Audience Classes together with an informal description of their requirements: information- and functional - as well as navigational- and the usability requirements, and their characteristics. If within one Audience Class, members have different characteristics *Variants* of Audience Classes are introduced that group members of the same class with the same characteristics.

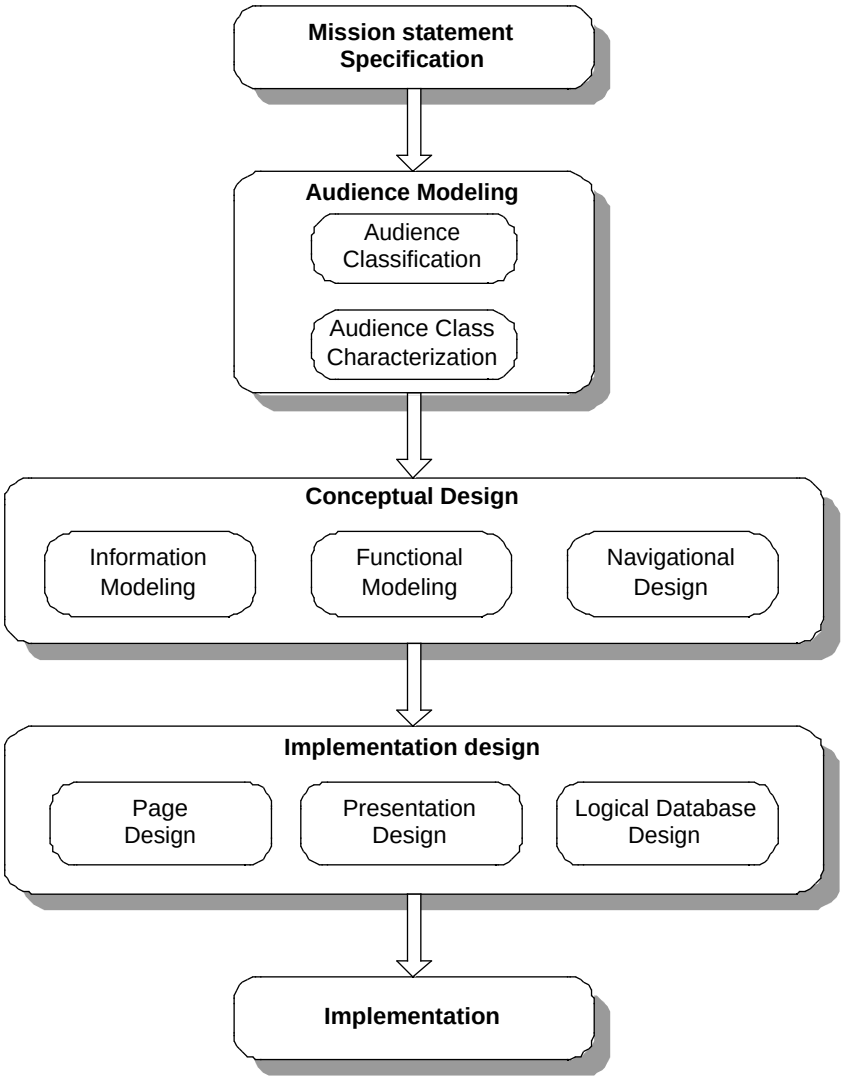

Figure 1: Overview of WSDM

Next we perform a *Conceptual Design* of the site. The conceptual design phase is divided in three steps: *Information Modeling, Functional Modeling* and *Navigation Design.*

During Information Modeling, *Information Chunks* are created. These chunks model the information requirements of the different audience classes. The different Information Chunks are linked together by a single information model, called the *Business Information Model.* The Business Information Model describes the information available in the organization, independently of any particular use. All Information Chunks are defined as views on this model. In this way, possible redundancy is described and therefore can be controlled.

During Functional Modeling, the functionality needed for the different audience classes is described. This is done using *Functional Chunks*.

During Navigation Design we describe the (conceptual) structure of the web site and model how the members from the different audience classes will be able to navigate through the site. For each audience class (variant) a *Navigation Track* is created. Navigational requirements are taken into consideration in this step. All Navigation Tracks together form the *Navigation Model* of the site.

The integration of the Information Chunks and Functional Chunks in the Navigation Model is called the *Conceptual Model* of the web site.

During *Implementation Design* we essentially design the (page) structure as well as the 'look and feel' of the web site. The aim is to create a consistent, pleasing and efficient look and feel for the conceptual design made in the previous phase by taking into consideration the usability requirements and characteristics of the Audience Classes. The page structure is derived from the Navigational Model. The amount of information on a page should not overwhelm the user. In addition it should also not cause a long download time. On the other hand, too little information on a page may force the user to "click" too much. If the information provided by the web site (or parts of it) will be maintained by means of a database then the Implementation Design phase will also include the Logical Design of this database. This logical database schema can be derived from the Business Information Model.

The last phase, *Implementation*, is the actual realization of the web site using the chosen implementation environment. E.g. for an HTML implementation this means that the implementation model must be converted into a set of files containing HTML source code. Depending on the complexity of the web site, parts of it can be automated using available tools and environments for assisting in HTML implementations.

3 Mission Statement for the Conference Review Site

The first step in WSDM is to define the Mission Statement of the web site. The Mission Statement must answer the following questions: *what is the **purpose** of the web site*, *what is the **subject*** and *who are the **target audience(s)***.

The Mission Statement for the Conference Review Site can be formulated as follows:
"*To support the overall selection process (submission by authors, evaluation and selection by the Program Committee) of papers for a conference*".

So, the purpose of the site is to support the paper selection process. The target audiences are authors and PC-program Committee and the subject of the web site are papers for a conference.

4 Audience Modeling for the Conference Review Site

4.1 Audience Classification

In general, the mission statement only gives a general indication of the target audience of the site. To identify the different audience classes the method prescribes to look at the activities of the organization that are related to the purpose and subject of the web site. Each activity involves people. These people are potential users if they belong to the target audience given in the mission statement. These activities are decomposed in order to refine in each decomposition step the target audience given in the mission statement.

For the Conference Review Systems, the activities are Paper Submission, Assignment of Papers to PC Members, Assignment of Papers to Reviewers, Entering a review, Selecting Papers, Notify Authors (see figure 2).

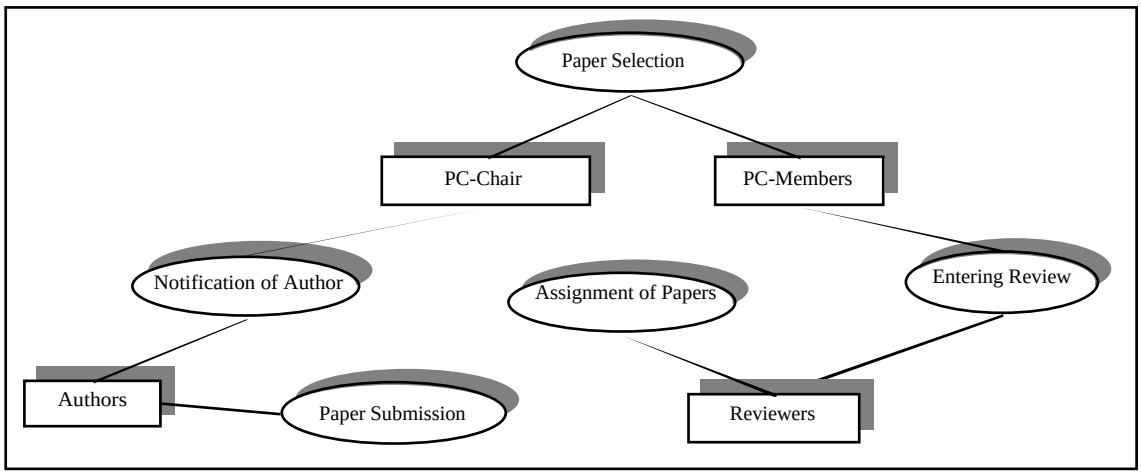

Figure 2: Activity diagram for the Conference Review System

The people involved in these activities are Authors, PC Chair, PC Members, and Reviewers. To decide whether these can be in one audience class or we need several audience classes we look to their requirements. By definition all members of an audience class have the same information and functional requirements. We can formulate the following (high level) requirements:

Authors
Functional requirements:
- Submit paper info and upload file
- Change submission until submission deadline
- Pre-register Co-author

Information requirements:
- Information about the author's submissions

PC Chair
Functional requirements:
- Create conference
- Update Conference information
- Pre-register PC Members
- Pre-register reviewers
- Indicate conflict of interest for papers
- Open list of submitted papers for PC Members when submission deadline has been reached
- Assign papers to PC Members (based on preferred papers and preferred tracks and subjects)
- Inform PC-Members about papers assigned to them
- Change track or topic of a submitted paper
- Open virtual PC meeting
- Mark paper as accepted, not accepted or undecided until all papers are either accepted or not accepted
- Close virtual PC meeting

Information requirements:
- All available information on the conference
- All available information on papers
- All available information on authors
- All available information on PC-members
- All available information on reviews
- During PC meeting: statistics about the reviews

Navigation requirements:
- Easy navigation from an author to his papers; from a paper to its reviews; from a review to the PC-member and reviewer; from a paper to its authors; from a review to the paper

PC Members
Functional requirements:

- Indicate preferences for tracks and subjects
- Pre-register reviewer
- Indicate interest in and conflict with proposed papers when list of papers is available
- Re-assign paper to reviewer
- Download papers assigned to him
- Submit review
- Change review if not final and until review deadline
- Advice PC chair during PC meeting

Information requirements:

- List of papers
- Own reviews
- State of reviews of papers PC Member is reviewing himself, after he submitted his own
- During PC meeting: all reviews and statistics

Reviewers
Functional requirements:

- Download papers assigned to him
- Submit review
- Change review if not final and until review deadline

Except for PC-Members and Reviewers, the requirements of the different groups are sufficiently different to put them in different Audience Classes. The requirements of Reviewers is a subset of the requirements of the PC-Members, therefore PC-Member is defined as a sub-class of Reviewer. This results in the initial Audience Classes: **Author, PC Chair, PC Member, Reviewer**. Please note that sub-classes are not defined on the basis of a "is-a" relationship but on the basis of extra needs.

In order to check whether we have to refine these Audience Classes (and possibly introduce new sub-classes), we check if we can decompose the activities into sub-activities. However in this case there are no meaningful decompositions and therefore no further sub audience classes.

In WSDM, the Audience Classes form a hierarchy. All identified classes are sub-classes of the top Audience Class *Visitor* (see figure 3). The requirements of the Audience Class Visitor are those that are common to all Audience Classes. For this Conference Review System, all visitors are interested in the general information about the conference, like date and place of the conference, tracks and subjects, deadlines, etc.

Visitors
Information requirements

- General information about the conference

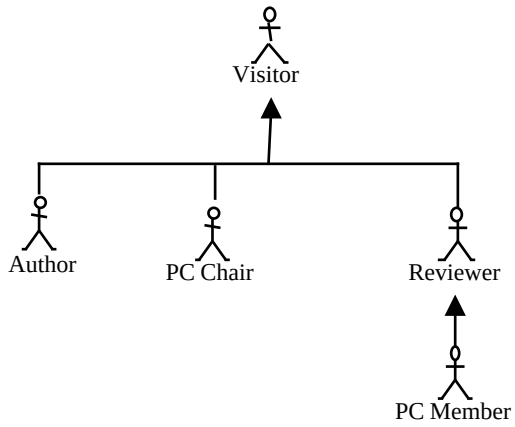

Figure 3: Audience Class Hierarchy for the Conference Review System

Because the site needs to be protected against unauthorized use, we also define a second Class hierarchy: *the Security Class Hierarchy.* In this class hierarchy the visitors are classified according to the security aspects needed by the system.

Registered Users
Security requirements:

- Logging in

Pre-Registered Users
Security requirements:

- Logging in
- Confirm their registration (changing password, entering affiliation and contact information)

Not-Registered Users
Security requirements:

- Register

Based on these requirements, the Security Class Hierarchy can be defined (see figure 4). We define the class Pre-Registered Users as a sub-class of the Registered Users because pre-registered users need some extra functionality (Confirming their registration).

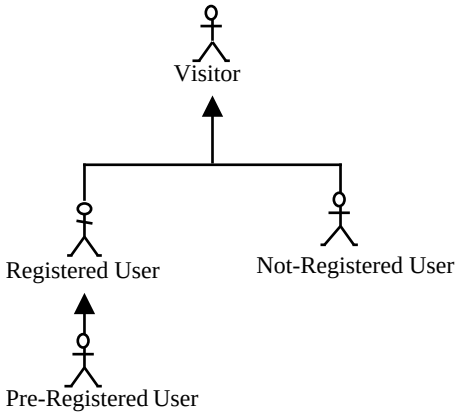

Figure 4: Security Class Hierarchy

The relationship between the classes from the Audience Class Hierarchy and from the Security Class Hierarchy is given in figure 5.

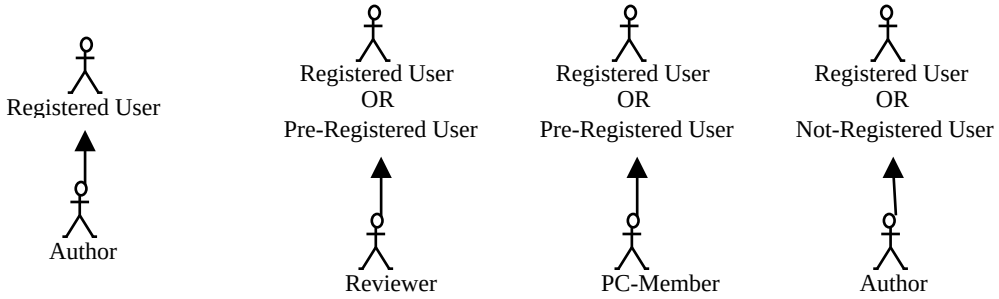

Figure 5: Relationship between Audience Class Hierarchy and Security Class Hierarchy

We also document the possible class transitions. Here, only for the security classes there are some possible transitions. A Pre-Registered User becomes a Registered User after confirming the registration, a Not-Registered User becomes a Registered User after registration. This is graphically represented in figure 6.

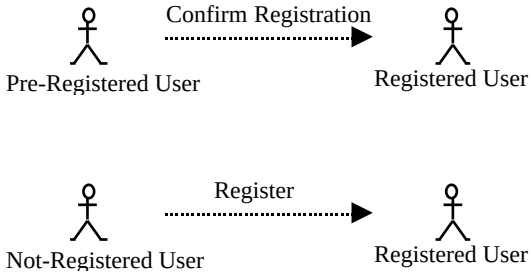

Figure 6: Class Transition Diagrams

4.2 Audience Characterization

As already explained, all members of an audience class potentially have the same information and functional requirements. However the members of one audience class may diverge on how the information should be presented to them, i.e. they may have different *usability requirements*. E.g. younger people are more visually oriented than older people are. Therefore we also analyze the characteristics of the audience classes.

Some examples of user's characteristics are: level of experience with web sites in general, frequency of use, language issues, education/intellectual abilities, age, income, lifestyle… Some of the characteristics may be translated into usability requirements while others may be used later on in the implementation phase to guide the design of the "look and feel" of the web site, e.g. choice of colors, fonts, graphics, etc.

If within one audience class we can distinguish groups of members with different characteristics, we introduce *Audience Class Variants*. However in this case all visitors have the same characteristics and therefore there are no Audience Class Variants.

Characteristics of all Audience Classes:

- are able to communicate in English
- have reasonable experience with web sites
- age: older than 20
- professionals

5 Conceptual Design of the Conference Review Site

During the first phase, audience modeling, the information-, functional requirements, and navigational requirements and the characteristics of the potential visitors are identified and different Audience Classes and Variants are recognized. The goal of the Conceptual Design is to turn these requirements into high level, formal descriptions which can be used later on to generate (automatically or semi –automatically) the web site.

During conceptual design, we concentrate on the **conceptual** "what and how" rather than on the visual "what and how". The conceptual "what" is mainly covered by the information and functional modeling step, the conceptual "how" by the navigation design.

The Conceptual Design consists of three steps, the Information Modeling, the Functional Modeling and the Navigation Design. The steps can be performed in any order and it is not needed to complete a step before starting another. To illustrate the process, we will elaborate the steps for one part of the system: the author part. The complete Conceptual Design is given in the Appendix.

5.1 Information Modeling

The Information Modeling step is intended for the so-called data intensive web sites. These are web sites dealing with structured data. The purpose of the Information Modeling step is to model this structured data. This is done by modeling the information requirements of the different Audience Classes. Each information requirement is translated into so-called *Information Chunks*. An Information Chunk is a tiny conceptual schema that describes the information needed to satisfy a single (usually elementary) information requirement. We call these models *chunks* because they can be seen as parts of a larger model (i.e. the Business Information Model). We have used ORM [5] to do the information modeling. However, in principle any other information modeling technique can be used (e.g. ER or Class diagrams).

ORM views the world as objects playing roles. The basic building blocks of ORM are Object Types (OT) (graphically represented as circles) and (usually binary) relationships composed of roles (graphically represented as a rectangle composed into boxes, each box connects with a line to the corresponding OT). Identifiers (represented by lines on roles) are used to indicate one-to-one, one-to-many or many-to-many relationships. A mandatory role is indicated by a dot. Object Types are divided into Lexical Object Types (graphically represented by a dashed circle) and Non Lexical Object Types (graphically represented by a solid circle). Instances of Lexical Object Types are utter-able, instances of Non Lexical Object Types not. E.g. "Conference" is a Non Lexical Object Type; "Conference Name" is a Lexical Object Type. If no explicit distinction between Lexical and Non Lexical is needed, the Object Type is represented as a Non Lexical Object Types (by a solid circle) and the lexical representation is put between brackets under the name of the Object Type. See e.g. the Object Type "Date" in figure 8.

The information modeling goes as follows:

- Elaborate the informal information requirements (usually in the form of structured natural language text);
- Divide the information requirements into so-called *elementary information requirements*;
- For each elementary information requirement: make an Information Chunk.

We define an *elementary information requirement* as an information requirement that only deals with one object type.

The complete elaboration of the requirements is given in the Appendix. As an example, we will show here the elaboration of the information requirement of the Author Audience Class. The informal description was:

"*Information about the author's submissions*"

This can be elaborated into the following information requirement:

"*For each paper submitted by the author either as main author or as co-author allow to give paper-id, title, abstract, main author (name) and co-authors (name), track and if available the subjects, and file (url) containing the paper*".

We split this requirement into two elementary requirements:

1. "Give *Paper-id and title of all papers submitted by the author as main author or as co-author and allow user to select a paper"*
2. *"For a paper give: paper-id, title, abstract, main author (name) and co-authors (name), track and if available the subjects, and file (url) containing the paper"*

Now, for each elementary information requirement a (specific) information model describing the information needed to satisfy this requirement is built. These are the so-called Information Chunks. Figure 7 shows the Information chunk for the first requirement and Figure 8 gives the Information chunk for the second requirement.

Because we need to be able to refer to instances of Object Types, we have extended the ORM notation in the following way. To refer to object instances we use *referents.* A referent is placed inside the circle depicting its Object Type. For Lexical Object Types, individual referents are values, e.g. 'S. Casteleyn' is an instance of the Lexical Object Type *PersonName* and 1 is an instance of the Lexical Object Type *Rate*. A *generic marker* is used to represent an instance of a Non-Lexical Object Type, e.g. **a* in figure 7 represents an instance of type *Author.* Sets are represented by the traditional set brackets '{' and '}', e.g. {*p} placed in the Object Type*Paper* represents a set of Paper instances. The traditional set operators can be used on sets. The notation {…}@ indicates the number of elements in the set. To represent a relationship between instances, we can place the referents either in the Object Types (if no confusion is possible) or in the boxed of the roles. In figure 7, there is a relationship between *a and {*p} through the *Author main of Paper* relation and a relationship between *a and {*p'} through the *Author co-author of Paper* relation. To indicate the selection of an instance by the user the symbol '!' is used, e.g. !{*p} allows to specify that the user can select one instance from the set {*p}. The symbol '!!' indicates that more than one instance can be selected. Chunks can take input parameters and may return output parameters, E.g. the Information Chunk AuthorSubmission has *a of type Author as input parameter and an instance of type Paper as output parameter (specified as: !({*p} union {*p'}))

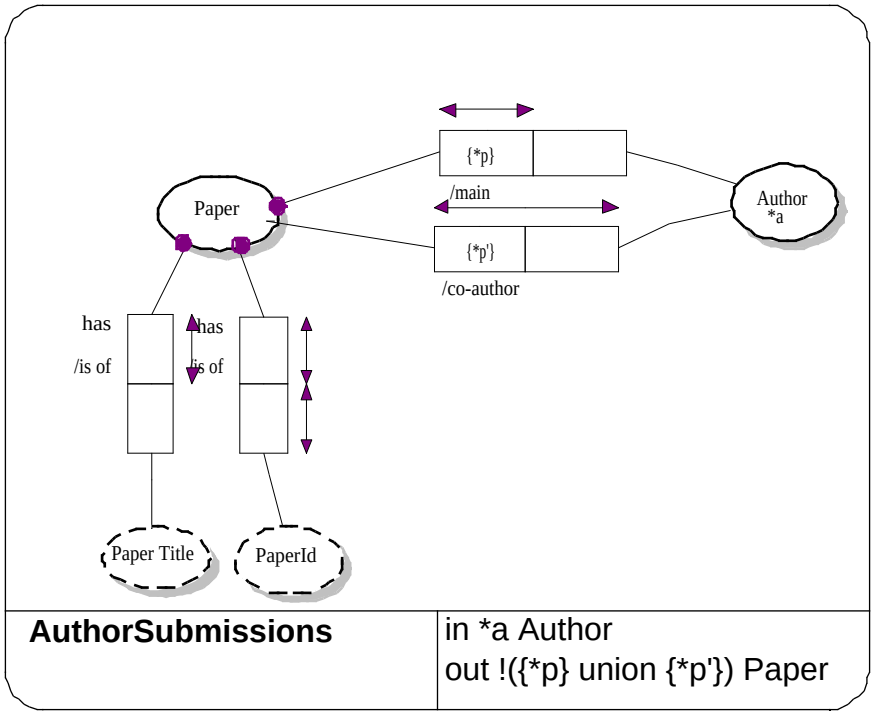

Figure 7: Information Chunk: AuthorSubmissions

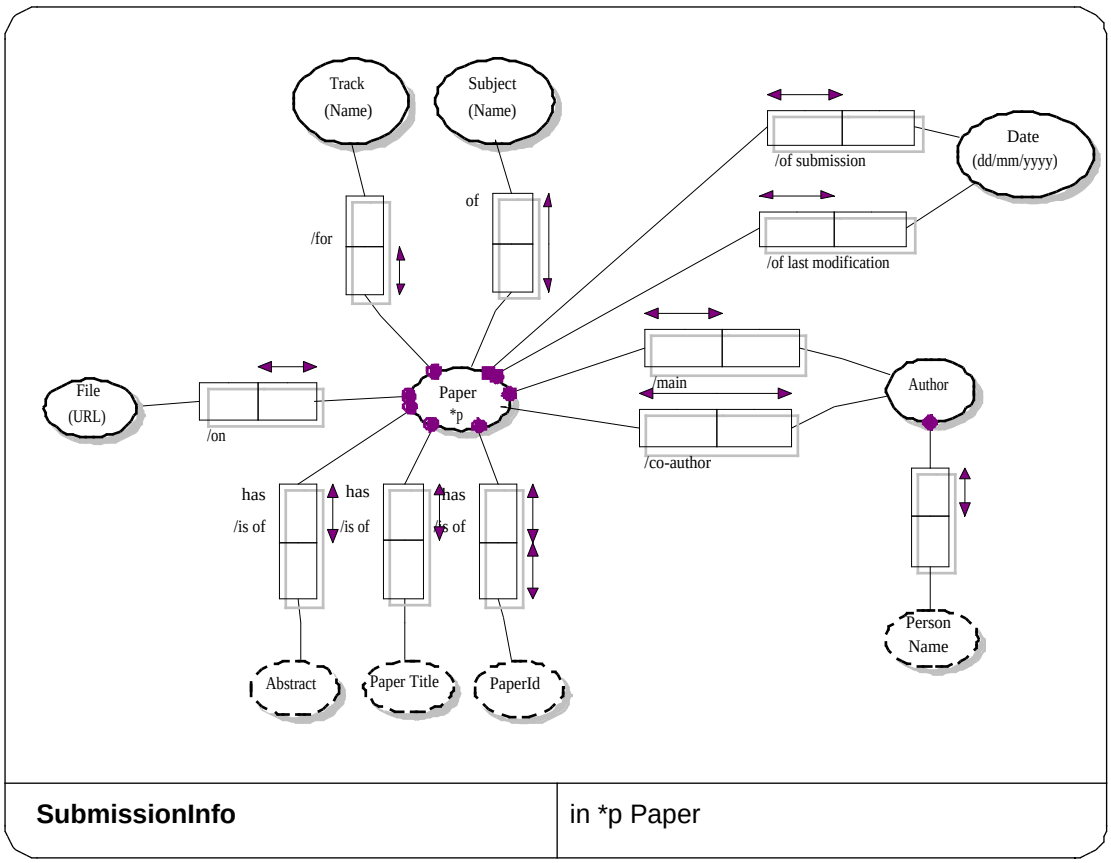

Figure 8: Information Chunk: Submission Info

Other Information Chunks are given in the Appendix.

5.2 Functional Modeling

The Functional Modeling step is similar to the Information Modeling step, except that we now concentrate on modeling the functional requirements. Functional requirements are elaborated into more elementary requirements and are modeled using *Functional Chunks* (and also Information Chunks). This may involve some reformulation of the requirements, e.g. for the pre-registration of co-authors we decide that co-authors should directly be attached to a paper; this to avoid that any existing author can be attached as co-author to a paper. Therefore pre-registration of co-authors becomes a part of submitting a paper. The elaboration of the functional requirements of Author results in the following decomposition of requirements. The detail descriptions of the elementary requirements are also given.

Submit New Paper
- Register New Paper
- Submit Paper Info and File
- Add Co-author

Change Submission
- Select submission
- Show Submission Info
- Update Paper Info and File
- Delete Co-authors
- Add Co-author

Register New Paper
The user enters the title of the paper. The system will generate and return a new paper id for this paper and assign the author as main author for this paper.
Submit Paper Info and File
The user enters the abstract, track, subjects and indicate the file that need to be uploaded. The system will automatically record the date of the submission and the date of the last modification.
Add Co-author
The user enters the name of a co-author of a given paper. The system will pre-register this co-author by generating and returning a new username and password for this author. The author will be added as co-author for the given paper.
Select Submission
The user selects one of his submissions from a list showing title and paper id of all his papers.
Show Submission Info
Gives paper-id, title, abstract, main author (name) and co-authors (name), track and if available the subjects, and file containing the paper for a submission
Update Paper Info and File

The user can change title, abstract, track, and subjects of the paper and upload a new file that replaces the old one. The system will automatically update the date of last modification for this paper.

Delete Co-authors

The user identifies the co-authors to be deleted from the list of co-authors of the paper. The system will remove this author as co-author.

Now, similar as during Information Modeling, for each elementary functional requirement a (specific) functional model describing the functionality needed by this requirement is built. These are the so-called Functional Chunks. The same basic notation as for Information Chunks is used but extra symbols are needed to express e.g. adding information, changing and deleting information, entering input, etc.:

- ? is used for expressing interactive input of a value; ?? is used for expressing interactive input of more then one value;
- = is used for assigning values to referents, e.g. **t =?* placed in the Object Type *PaperTitle* allows to specify that the user can enter a title, which is assigned to the referent *t;
- → is used to indicate that a value of a referent can be changed, e.g. *t →? placed in the Object Type *PaperTitle* allows the user to change the given paper title by entering a new title, which is then assigned to the referent *t;
- → Ø is used to remove information, e.g. {*a} → Ø placed in the role *co-author* will remove the relationships in which the given set of instances participate as co-author;
- == allows to test equality of values;
- **is** alows to test membership of an Object Type;
- **exist** allows to test the existence of an instance;
- **if** and **for each** are used in the general sense;
- NEW indicates the generation of a new Object Type instance;
- NEXT is for generating system identifiers;
- REMOVE is used to remove instances from an Object Type;
- TODAY represents the current date;
- UPLOAD and EMAIL are supposed to be built-in functions.

Figure 9 and 10 shows two examples. The others can be found in the appendix.

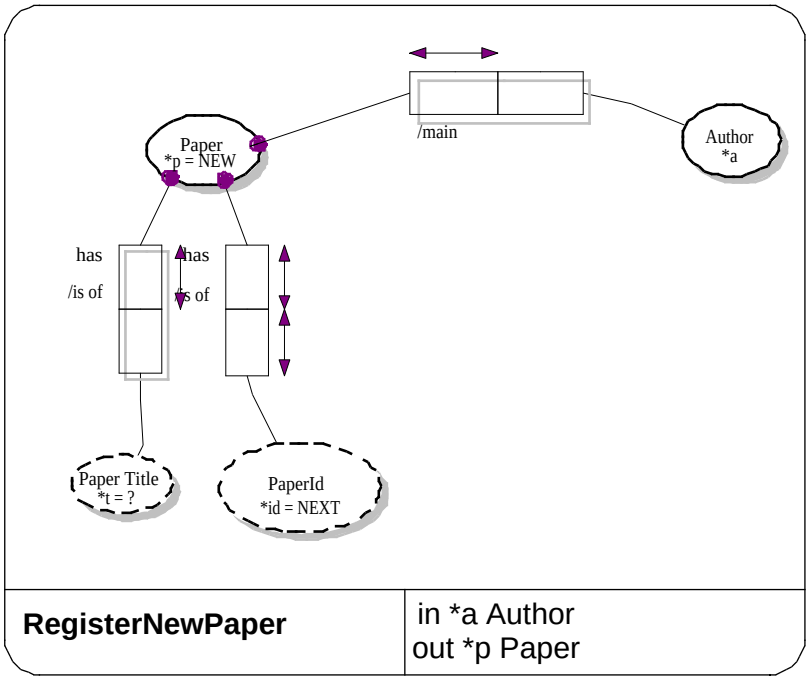

Figure 9: Functional Chunk for Register New Paper

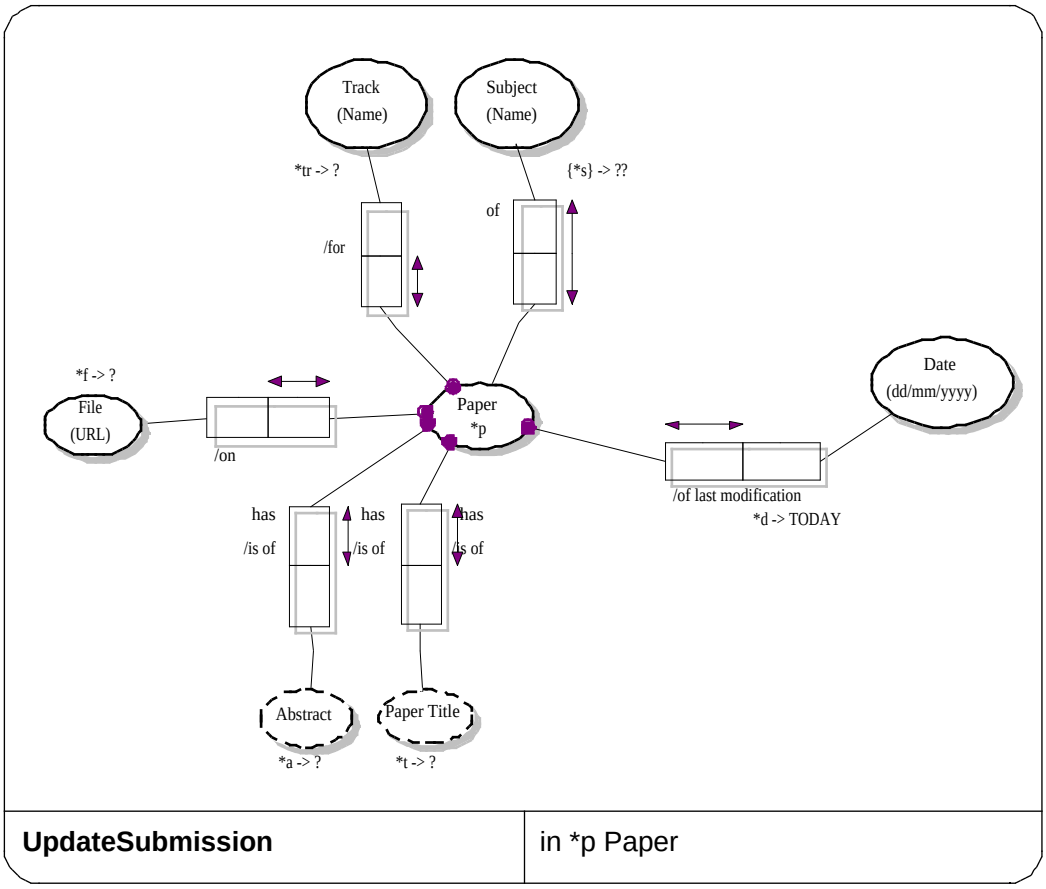

Figure 10: Functional Chunk forUpdate PaperInfo and File

5.3 Navigation Design

In the navigation design we describe the (conceptual) structure of the web site and model how the members from the different Audience Classes will be able to navigate through the site. For each Audience Class (variant) a different *Navigation Track* is created. All Navigation Tracks together form the *Navigation Model*.

A Navigation Model is described in terms of *tracks*, *components* and *links*. Components represent units of information or functionality. Components are connected by means of links. Links are used to model the structure of the web site as well as to indicate the need for navigation. We can put *conditions* on links to indicate that the availability of the link is dependent on the truth-value of the condition.

In figure 11 the graphical notation used for tracks, components and links is given. A multiple link is used to indicate that one component is linked to several instances of the other component.

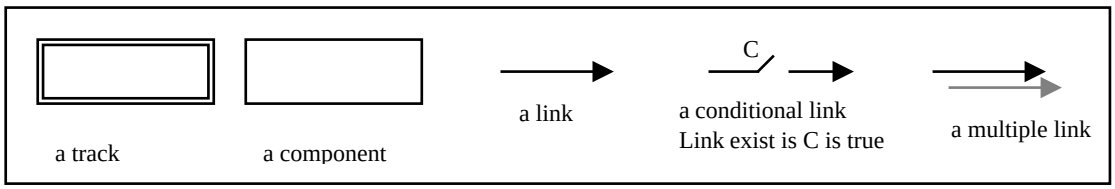

Figure 11: Graphical notation of components and links.

The main structure of the Navigation Model is easily derived from the Class Hierarchies. As already indicated, with each Audience Class corresponds a Navigation Track. The different tracks are linked together following the Class Hierarchies and the relationships between the hierarchies. The resulting track structure for the Conference Review System is given in Figure 12.

The next step is to elaborate each Navigation Track into more detail. To illustrate this process, we will elaborate the Navigation Track for the Author Audience Class.

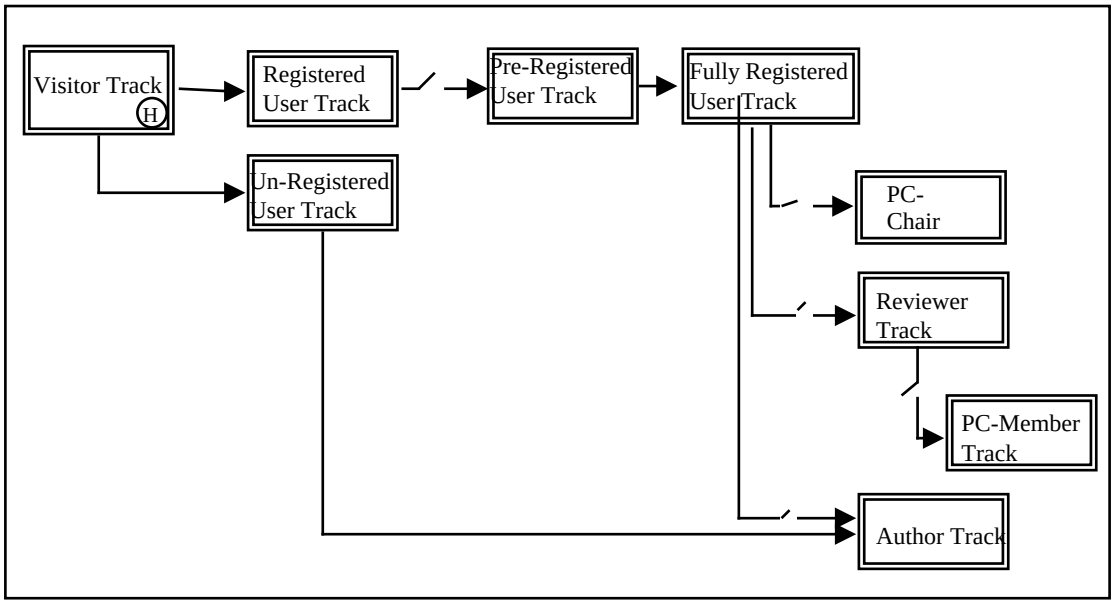

Figure 12: Main Track Structure of the Navigation Model

5.3.1 Navigation Track for the Author Audience Class

The starting point for designing the Navigation Track for an Audience Class are the requirements formulated for this Audience Class in the Audience Modeling phase and elaborated further on in the Information and Functional Modeling steps. For the Audience Class Author these are:

Author.R1: Information about author's submissions

- **Author.R-.1: Submissions of author**
  List of (paper-id and paper-title of) papers submitted by the author as main author or as co-author
- **Author.R-.2: Submission info**
  Paper-id, paper-title, abstract, main author (name) and co-authors (name), track and possible the subjects, and file (url) containing the paper

Author.R2: Submit new paper

- **Author.R-.3: Register new paper**
  The user enters the title of the paper. The system will generate and return a new paper id for this paper and assign the author as main author to this paper.
- **Author.R-.4: Add co-author**
  The user enters the name of a co-author of a given paper. The system will pre-register this co-author by generating and returning a new username and password for this author. The author will be added as co-author of the given paper.
- **Author.R-.5: Submit paper info and file**
  The user enters the abstract, track, subjects and indicate the file that need to be uploaded. The system will automatically record the date of the submission and the date of the last modification.

Author.R3: Change submission

- **Author.R-.6: Select Submission**
  The user selects one of his submissions from a list showing title and paper id of all his papers.
- **Author.R-.7: Show Submission info**
  Same as Author.R-.2: Submission info
- **Author.R-.8: Update paper info and file**
  The user can change title, abstract, track, subjects of the paper and upload a new file which replaces the old one. The system will automatically update the date of last modification for this paper.
- **Author.R-.9**: **Delete co-authors**
  The user identifies the co-authors to be deleted from the list of co-authors of the paper.
- **Author.R.-10: Add co-author**
  Same as Author.R-.4

The navigation track for Author can now be composed easily by defining a component for each requirement following the decomposition structure of the requirements. Because the decomposition of Author.R1 (Information about the author's submissions) is part of the decomposition of Author.R3 (Change Submission), we can share those components. Figure 13 shows the first version of the Navigation Track for the Author Audience Class. Also note that the link to "Register New Paper" and to "Update Paper Info and File" are conditional links. These links only exist if the submission deadline is not yet passed. The exact formulation of the condition will be given later.

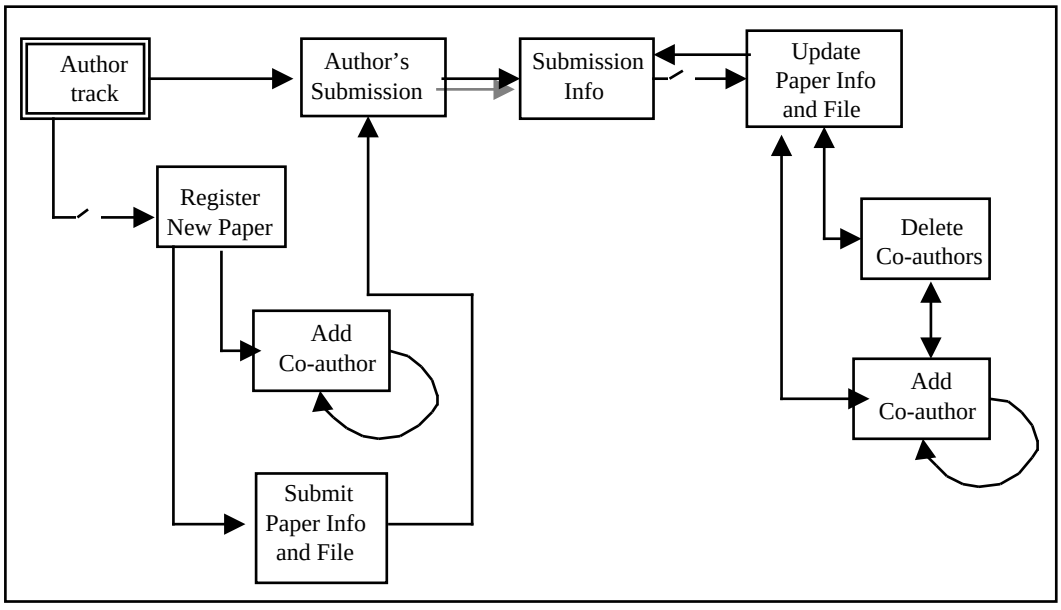

Figure 13: Navigation Track structure for Author

To complete the specification of the Navigation Track, we have to specify the content of the different components. A component corresponds with a requirement. An *information*

component corresponds with an information requirement, a *functional component* with a functional requirement. During Information Modeling, Information Chunks are defined to specify how to satisfy the information requirements. Therefore, the information needed by an information component is described by the corresponding Information Chunk. Similar, the functionality needed by a functional component is described the corresponding Functional Chunk. Therefore, to indicate which information or functionality is represented or needed by a component, the component is associated with the corresponding Chunk. Graphically this is represented by a dotted line between the component and the chunk (see figure 14). If no chunk is associated with a component, the component is a *navigation component,* i.e. it only allows further navigation. Components may be mixed; i.e. they may contain information as well as functionality as well as navigation links.

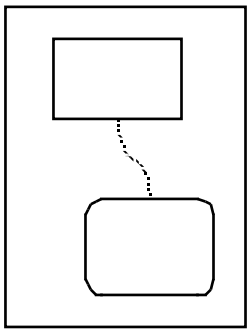

Figure 14: Association between component and chunk

To make the specification complete, we also have to indicate the in- and out-parameters for the chunks and which variables should be passed between components. The complete specified Author Track is given in Figure 15.

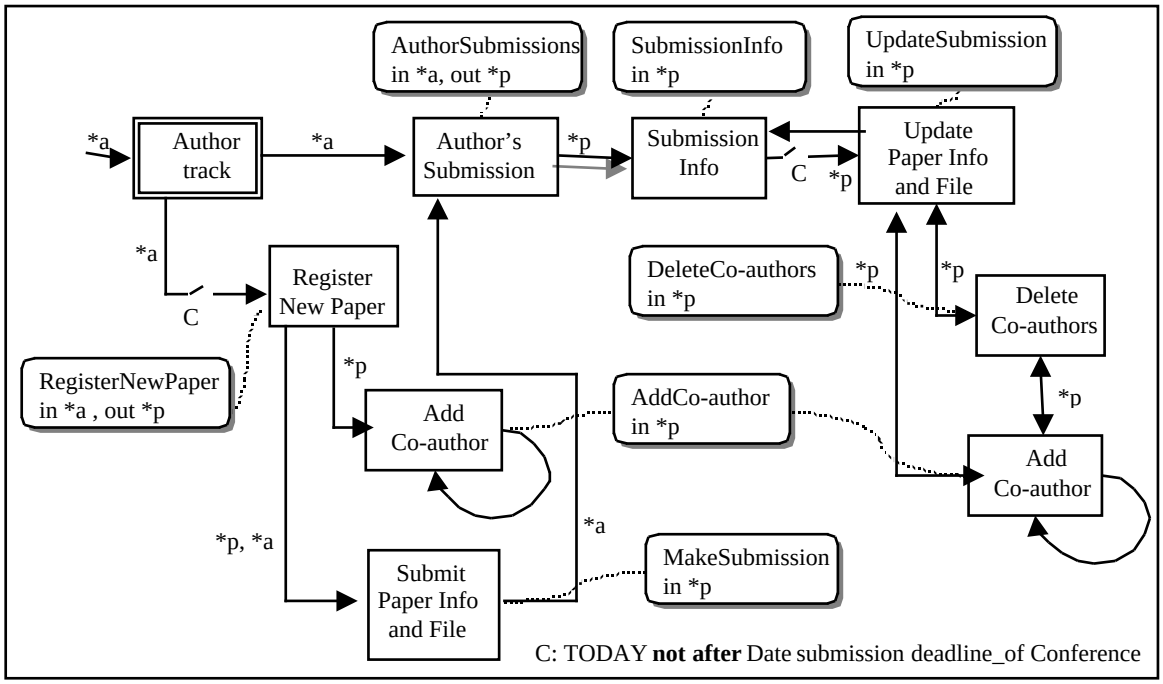

Figure 15: Completely specified Author Track

5.4 The Business Information Model

Different audience classes may need the same information. For example, Authors as well as PC-Members see information about papers. This means that different Information Chunks may model the same information. This redundancy is deliberately (the same information may need different layouts or different navigation paths for different audience classes) and will not cause any problems as long as we recognize that it is indeed the same information and that it can be maintained in a single place. Therefore, we have to relate the information in the different Information and Functional Chunks to each other. To do this we use an overall information model, called the *Business Information Model*. This model describes the information needed by the different Audience Classes is an independent way. In fact it is a conceptual description of the information available in the organization. This may already be available in the organization. If not or it is not in a shape usable for our purpose it must be (re-) developed. The Business Information Model for the Conference Review System is given in the Appendix.

In principle each Information Chunk and Functional Chunk must be expressed as a view on this Business Information Model. We have not done this explicitly because in most cases the relationship between the chunk and the Business Information Model is clear.

6 Implementation Design of the Conference Review Site

We will only discuss briefly the implementation design phase. Currently, the method identifies the different steps to be performed during this phase but is not yet completely elaborated on this point.

In the implementation design we essentially design:

- the page structure of the web site (Page Design step);
- the "look and feel" of the web site and the layout of the individual pages (Presentation Design);
- optionally a logical data schema for the database that will be used to maintain (parts of) the information (Logical Database Design step).

The page structure is derived from the navigational model. For the components and links in the navigational model we define pages and hyperlinks. By default we can assign each component to a page and each link to a hyperlink. However, the amount of information on a page should not overwhelm the user. In addition it should also not cause a long download time. On the other hand, too little information on a page may force the user to "click" too much. Therefore, the web developer can decide to cluster components connected by links and to represent the information on a single page, or he can decide to distribute the information of a single component onto different pages. Figure 16 shows the page structure for the Visitor Track.

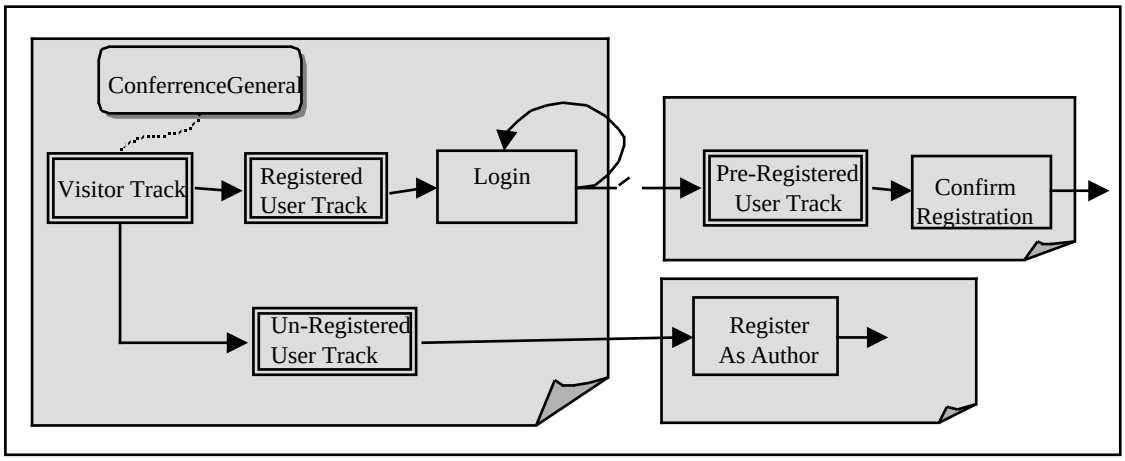

Figure 16: Page structure for the Visitor Track

During Presentation Design, the characteristics of the different audience classes should be taking into consideration. This may result in different styles for different navigation tracks. The purpose is to create a consistent, pleasing and efficient look and feel for the web site. This step of the method is not yet elaborated; currently we refer to the literature

for this. There are many books only dealing with exactly this aspect of web design. We have not elaborated this step for the Conference Review System.

To improve maintainability, more and more web sites have their underlying information (or parts of it) stored in a database. Pages are generated (usually at runtime, but not necessarily) by extracting the necessary information from the database. The information may be extracted from existing databases or a new database may be implemented for this purpose. The basis for such a database design is found in the Business Information Model. The Business Information Model is actually the conceptual schema of the database: it describes the available objects at a conceptual level. From this conceptual schema a logical database schema can be generated (using appropriate database development tools, e.g. InfoModeler [6] for an ORM schema) or manually built. The queries needed to extract the information for the pages can be derived from the Chunks, which can be expressed as views on the Business Information Model. We also have omitted this step.

It should be noted that the implementation design might depend on the chosen implementation environment. Limitations of the implementation language (such as HTML) may interfere with this process.

7 Implementation

The last phase is the actual construction of the web site using the chosen implementation environment. E.g. for an HTML implementation this means that the implementation model must be converted into a set of files containing HTML source code possibly combined with some scripts.

Because of limited resource, we have not implemented our design of the Conference Review System. Only for the purpose of illustration we have made the home page of the Conference Review System (figure 13) and the home page of the Author track (see figure18).

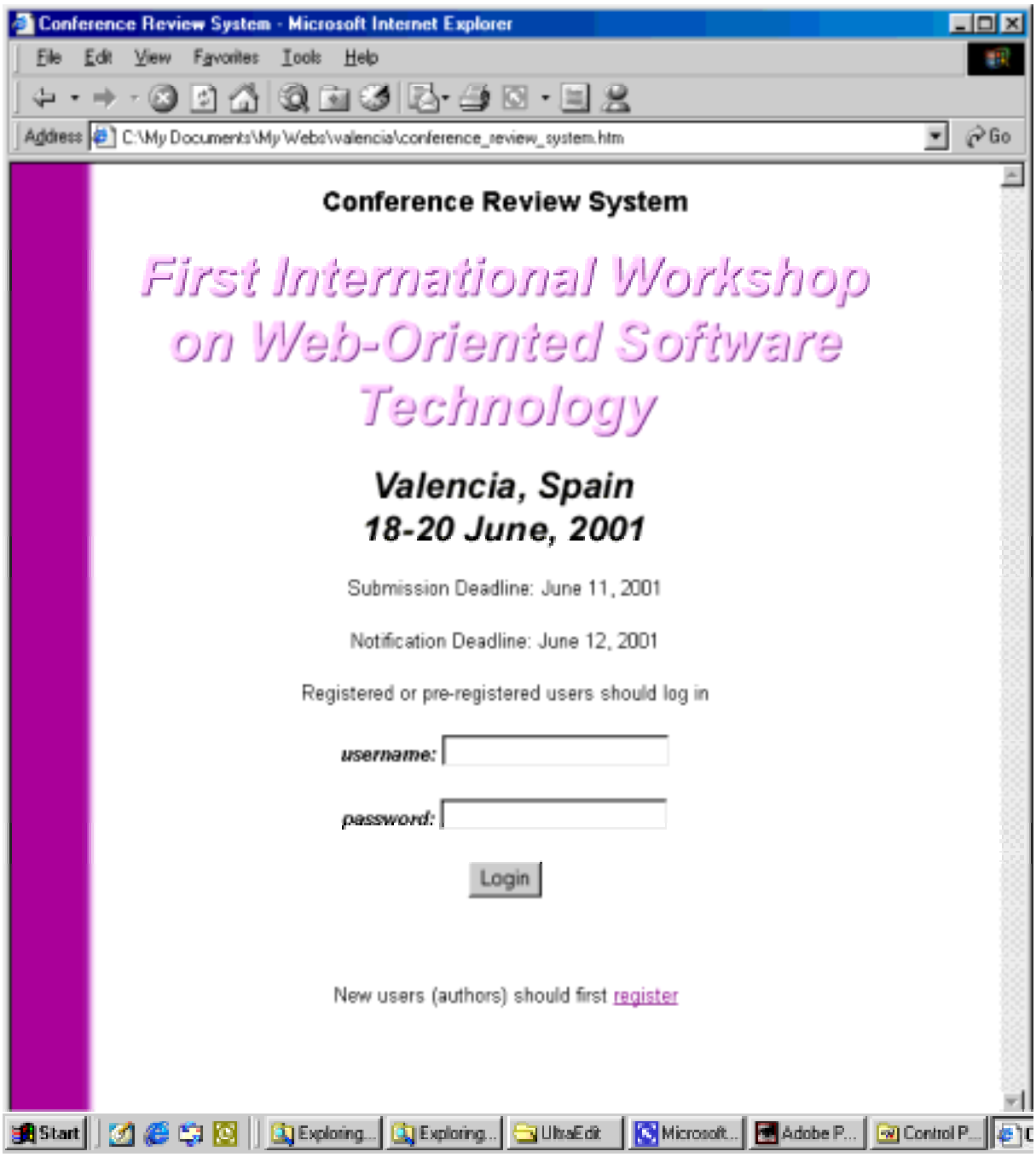

Figure 17: Home page of Conference Review System

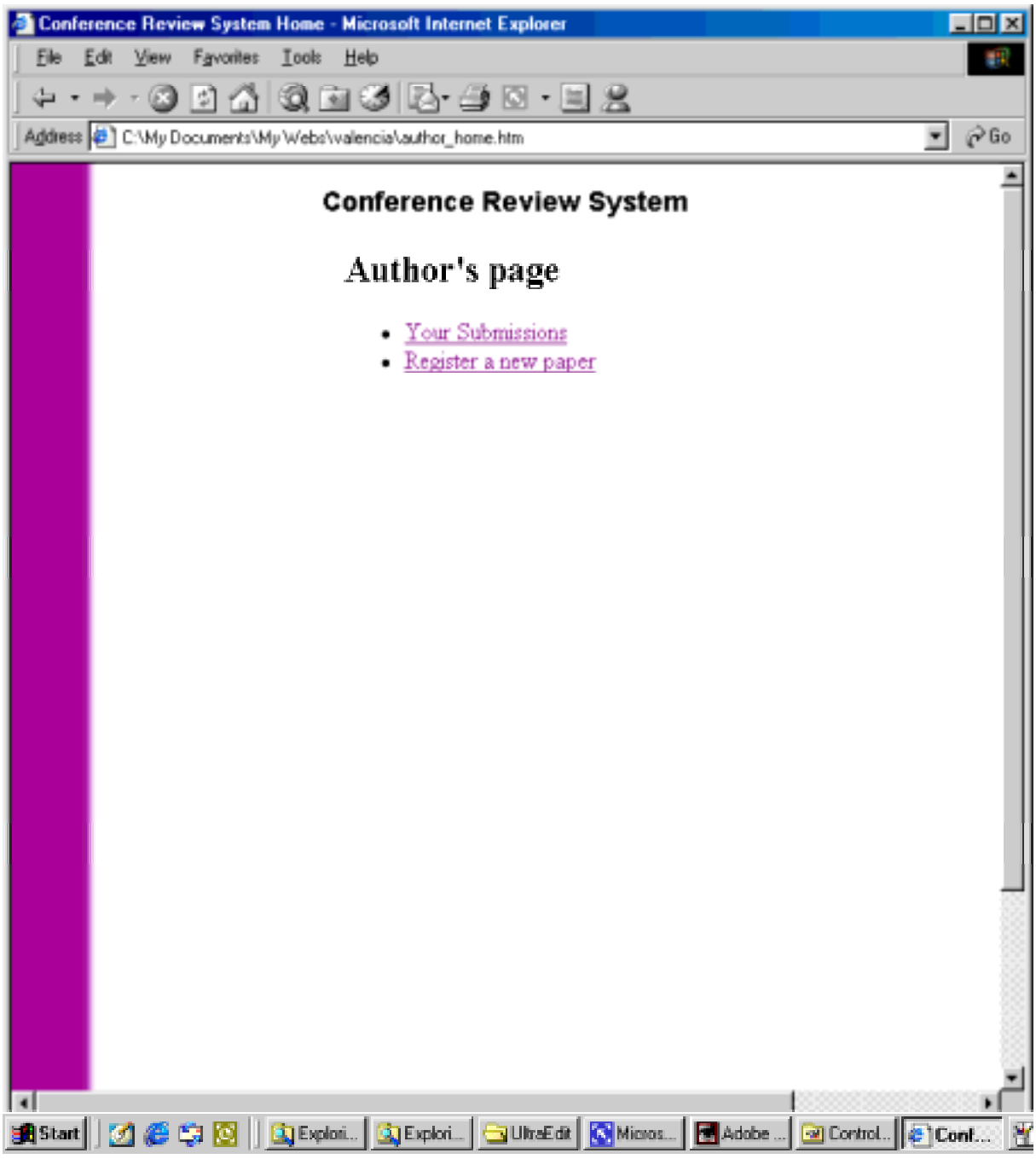

Figure 18: Home Page for the Author Track

References

[1] http://www.dsic.upv.es/~west2001/

[2] De Troyer O., Audience-driven web design. In: Information modelling in the new millennium, Eds. Matt Rossi & Keng Siau, IDEA GroupPublishing, ISBN 1-878289-77-2, 2001.

[3] De Troyer, O. and C. Leune (1998), "WSDM: A User-Centered Design Method for Web Sites," In Computer Networks and ISDN systems, *Proceedings of the 7th International World Wide Web Conference*, Elsevier, pp. 85 - 94.

[4] De Troyer, O.M.F. (1998), "Designing Well-Structured Web Site: Lessons to be Learned from Database Schema Methodology," In *Proceedings of the ER'98 Conference*, Lecture Notes in Computer Science (LNCS), Springer-Verlag.

[5] Halpin, T. (1995), *Conceptual Schema and Relational Database Design*, Second Edition Prentice Hall Australia.

[6] Visio Corporation (1997), Visio Technology, http://www.visio.com/