

Conceptual modeling of data-intensive Web applications

Stefano Ceri, Piero Fraternali, Maristella Matera

Dipartimento di Elettronica e Informazione – Politecnico di Milano

Via Ponzio, 34/5 – 20133 – Milano – Italy

{ceri, fraterna, matera}@elet.polimi.it

Keywords: Data-intensive Web applications, Web Conceptual Modeling, Web Modeling Languages, Web design patterns, Web application taxonomy.

1 INTRODUCTION

Many of the Web applications around us are data-intensive; this term indicates applications whose main purpose is presenting large amount of data to their users. Most of the sites for on-line trading or e-commerce are data-intensive, as well as most “institutional” sites (describing private and public organizations) or digital libraries. Several commercial Web development systems enable the rapid development of data-intensive applications by supporting the semi-automatic publishing of data resources [1]. Such an automatic publishing is typically constrained by database schemas, and offers too limited choices to the designers. Therefore, often Web application development requires adaptation through programming, and programs end-up mixing data, navigation, and presentation semantics in an intricate way; in particular, presentational aspects often hide structural, compositional and navigational aspects.

Despite this trend towards an unstructured development process, it is possible to recognize typical patterns and rules to which data-intensive applications, based on large data sets organized within a repository or database, generally obey. The purpose of this paper is to explain such patterns and rules, by using WebML [2] as a conceptual tool for making such notions explicit.

WebML is a conceptual Web modeling language, which adopts the Entity-Relationship model for data structure description, and an original, high-level notation for representing Web content composition and navigation in form of a hypertext. WebML has been proposed in the context of the research on conceptual models and tools for the Web, which also includes the works reported in [3-9]. An overview and a comparison of such works can be found in [1]. All approaches insist on the relevance of conceptual modeling of Web applications, thereby raising the level of abstraction of specifications. In particular, WebML features an extensive use of diagrams, supported by a powerful GUI of the WebML tool suite.

This paper will show that data-intensive Web sites can be abstracted as complex arrangements of elementary structures, called *skeletons*, which are pairs of data diagrams and hypertext diagrams, expressed according to the WebML notation. Skeletons can be used for a better understanding of Web applications, as well as for driving the design of new sites through the reuse of Web applications solutions [10]. They also facilitate site wrapping for the integration of services and information extracted from autonomously developed sites [11].

This paper will introduce the recognized skeletons, and illustrate them through examples taken from two real Web sites - AcerEuro (<http://www.acer-euro.com>), and Cisco Web site (<http://www.cisco.com>). WebML has been used in the context of grants offered by Acer and Cisco, for developing the commercial version of the former and an experimental version of the latter.

The paper is organized as follows. Section 2 presents the essential features of WebML, modeling a simplified version of the popular Amazon Web site. Section 3 presents a classification of the concepts forming an Entity/Relationship schema, denoted as core, access, and connection concepts. Then, Sections 4 through 6 show the core, access, and connection skeletons, used to support the navigation through the concepts defined in Section 3. Section 7 introduces a content management skeleton, used to input data by using a Web interface. Section 8 discusses the use of skeletons for designing, re-engineering, and classifying sites. Finally, Section 9 draws our conclusions.

2 OVERVIEW OF WEBML

WebML is a modeling language for the Web, which provides some modeling abstractions that a CASE environment is able to translate into concrete page templates. According to the WebML approach, Web applications have two orthogonal conceptual dimensions:

- The **data model**, which enables describing the schema of data resources according to the Entity-Relationship Model.
- The **hypertext model**, which enables describing how data resources are assembled into information units and pages, and how such units and pages are interconnected to constitute a hypertext.

This section introduces the WebML notation, and also shows how pages taken from a real Web site can be expressed through the composition of WebML elements. A more complete and formal definition of WebML can be found at <http://webml.org> and in [2].

2.1 Data Design

The design of a Web application starts with *data design*, which enables the specification of data organization in terms of the relevant entities and relationships. WebML supports the classical Entity/Relationship (E/R) model with generalization hierarchies, typed attributes, and cardinality constraints. Figure 1 shows a graphic representation of a data schema, describing data underlying a site for supporting the sale of books, inspired to the popular Amazon Web site (see <http://www.amazon.com>). Books are grouped into categories and can be available in different editions. Each book is also connected to other books that were also bought by users purchasing that book. This relationship serves the purpose of providing recommendations to users buying a book. Users can order a book edition by filling an order line; trolley groups all the order lines of a given purchase, and is associated to a given user.

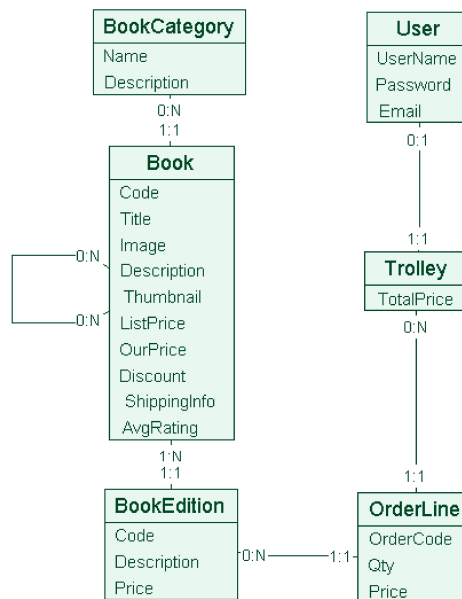


Figure 1. Example of data schema.

2.2 Hypertext Design

After data design, the design process proceeds with *hypertext design*, aimed at defining the hypertext topology of the Web site, i.e., the organization of content by basic units and links between them, and unit composition within pages. Hypertext design is based on five types of *content units*, shown in Table 1. The first four units in the table are devoted to display one or more instances of the entities of the data schema, typically selected by means of queries over entity attributes or over relationships. Moreover, a data entry unit can be used to collect input values through a form.






UNIT NAME	VISUAL NOTATION	DESCRIPTION
Data		Shows data about a single entity instance.
Multi-data		Shows data about all the entity instances.
Index		Shows a list of properties (also called descriptive keys) of a given set of entity instances. A user click on an index entry causes one instance to be selected.
Scroller		Provides commands for scrolling through objects in a set, for example through all the instances in an entity. Scrolling commands allow moving to the first, the last, the previous, and the next element in the set, and cause one instance of the set to be selected and displayed.
Entry		Shows a form with several fields for collecting input by users. This input may consist of conditions used for performing searches over instances of an entity, or of parameters to be supplied to operations, like content updates, login, or generic external operations.

Table 1.Content units of the WebML composition model.

Links relate units and express navigations on the Web site, as well as the transfer of information from a unit to another one. Some links also trigger computations, as it happens for example in case of content updating operations, which are activated by a “submit” link, followed by users after entering new data. Content units can be composed into *pages*, which represent the abstraction of a self-contained region of the screen treated as an independent interface block (e.g., delivered to the user as an HTML page).

Figure 2 shows a hypertext fragment inspired to the Amazon Web site, which illustrates the expressive power of WebML as a model for abstracting and conceptualizing the access mechanisms of a Web site. The Home Page supports the access to books by means of two alternative mechanisms: an index of the available book categories (Categories Index) and a form (BookSearch Entry) for starting a key-based search over books.

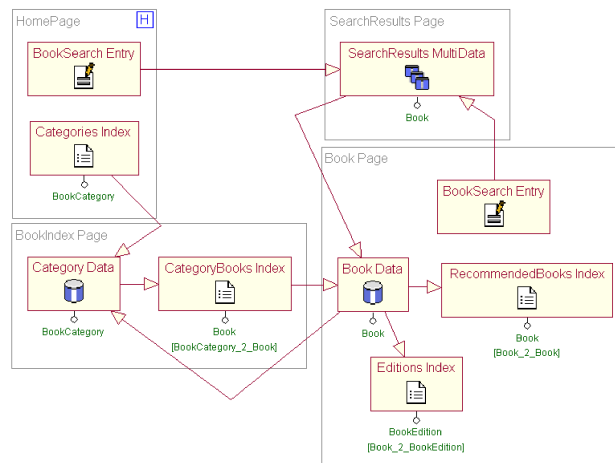


Figure 2. A simplified hypertext, which specifies browsing through the book catalogue.

If the user chooses to follow the index, the BookIndex page is next presented, which includes data about the selected category (Category Data) and the index of its books. If instead the user chooses to perform a search, then information about books matching the search criteria entered through the entry unit is displayed in the SearchResults page, by means of a multidata unit. Both search methods eventually yield to the Book page of a specific book, which shows information about one selected book (Book Data), together with the index of

all the editions of that book (Editions Index), and the index of recommended books based on purchase history (RecommendedBooks Index). An entry unit (BookSearch Entry) allows users to specify a search predicate for searching books; the entry unit is the same as the one included in the Home Page.

Figure 3 presents the book page of the Amazon Web site, which shows the actual rendition on the Web of some conceptual elements included in the Book page of the diagram of Figure 2. Among the different elements that compose the page, it is possible to recognize the data unit about a single book, the index of the various editions of the book, and the index of recommended books (that were bought by customers with similar interests). The page also includes an entry unit for searching books, and several other units, which are here omitted for simplicity.

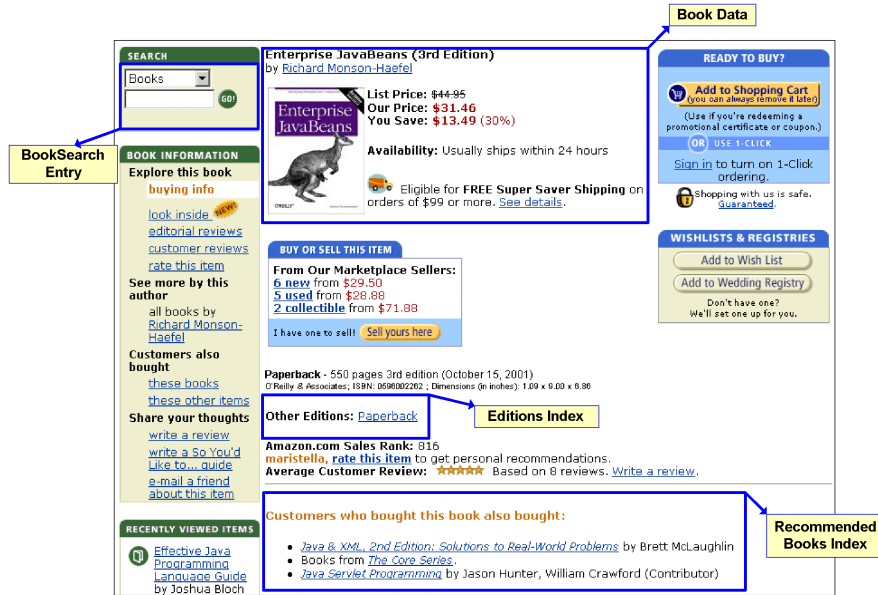


Figure 3. Book page in the Amazon Web site.

Figure 4 represents the SearchResults page with a multidata unit showing the results of a search performed over the entity book. Each book is represented by its thumbnail, title, shipping information, price, and average customers review. Note also the inclusion of a command for adding each book to the trolley, which activates a content management operation, as it will be next discussed.



Figure 4. Search results page in the Amazon Web site.

2.3 Write Access and Content Management Operations

WebML also offers operations units for invoking external operations for managing and updating content, such as creating, deleting or modifying an instance of an entity, or adding or dropping a relationship between two instances [12]. The visual notation and a brief description for operation units are reported in Table 2.







UNIT NAME	VISUAL NOTATION	DESCRIPTION
Create		Creates a new instance of an entity.
Delete		Deletes an instance of an entity.
Modify		Modifies an instance of an entity.
Connect		Creates an instance of relationship.
Disconnect		Drops an instance of relationship.
Generic Operation		Invokes a generic operation, possibly implemented by externally available Web services.

Table 2. WebML operation units.

Figure 5 shows a simplified hypertext fragment in WebML that models the creation of order lines, their insertion into the user's trolley, and their deletion and modification. From the Book page, users select the particular edition in which they are interested (e.g., hardcover or paperback), and then go to the NewOrderLine page, where they enter information about the book purchase (e.g., the order quantity). Once the data entry is completed, a chain of "Create and Connect" operations takes place, whose effect is to create an order line, connect it to the book edition, and also connect it to the user's trolley. After completion of these operations, the Trolley page is displayed, which contains the overall information already accumulated about the trolley, including the name of the user (User Data), the trolley total price (Trolley Data), and the order lines already entered (Orders Multidata). The quantity of an order line can be modified (via the chain of QuantityEdit Entry and ModifyOrder units). Order lines can also be deleted from the trolley (via the DeleteOrder unit).

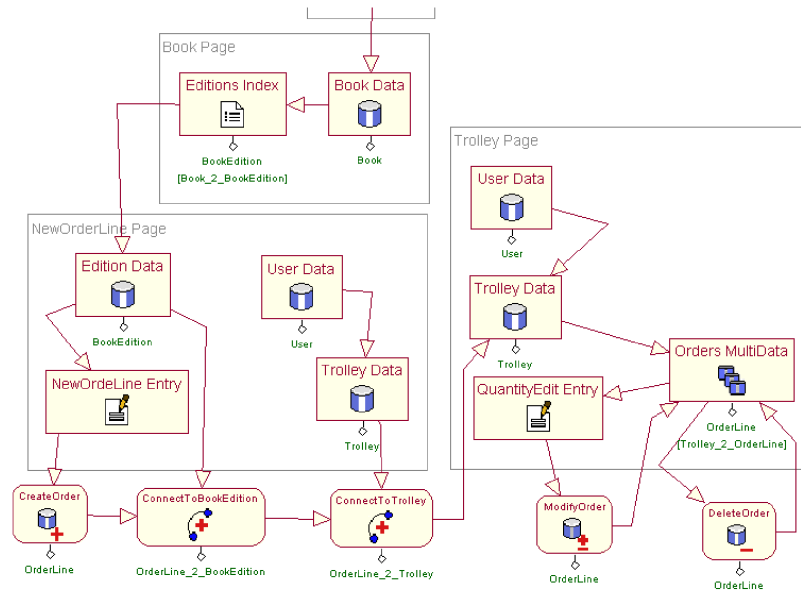


Figure 5. WebML hypertext fragment, specifying the insertion of a new order into the user’s trolley.

Figure 6 shows the trolley page within the Amazon Web site, which displays some order lines created by the user Maristella Matera. In this page, data entry units and operations allow users to modify or delete the created order lines. Data entry units enable the insertion of a new quantity for the ordered books (the default quantity is 1). The “update” button then activates the modify operation. The “delete” button, displayed in correspondence of each order line, enables the order deletion, by activating a delete operation.

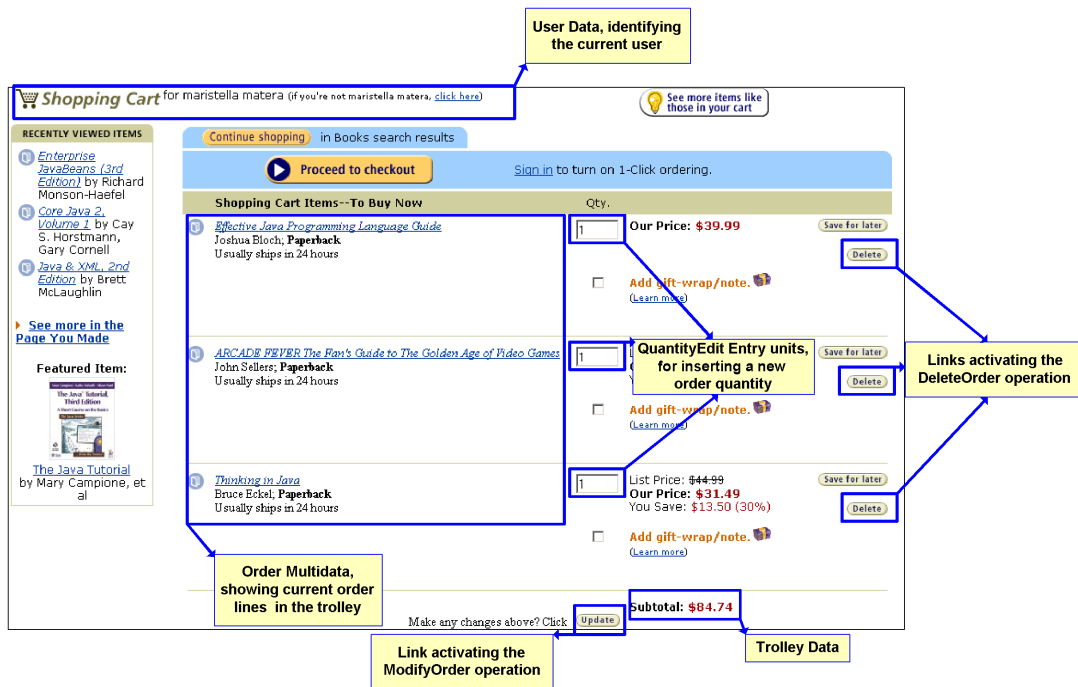


Figure 6. Trolley page in the Amazon Web site.

3 RECOGNIZING BASIC INFORMATION CONCEPTS

An important step for the development of skeletons is the classification of the role that concepts may play in the data schema. We distinguish three types of concepts:

- *Core Concepts* are the central concepts of the application, i.e., those ones that form the main asset and express the mission of the Web application, such as the product sold in a B2C site, or the personal message published in a community Web board. Core concepts form the backbone of the Web application content.
- *Access Concepts* support the location of core concepts in two ways:
 - By superimposing over them a categorization, which can be used to express index hierarchies, progressively leading the user from the home page to the core content, or search mechanisms focused on well-defined classes of core concepts.
 - By defining subsets of representative core concepts as meaningful collections (e.g., the “pick of the day” or the “site’s best choices”). Collections of core concepts are typically presented in the home page to let user have a preview of the most attractive core contents.
- *Connection Concepts* interconnect core concepts, and are typically expressed by means of relationships, which the user can navigate to move the focus from one core concept to another related one. In some cases, entities may also be used for representing interconnection concepts.

The distinction between core and access concepts is not a dogmatic one, but depends on the application domain and on the mission of the specific Web application. For instance, in an e-commerce Web application selling books, the Book Author concept associated to a Book concept could be considered either as an independent piece of core content, or as a property of the Book. A concept can be considered core if it independently contributes to the achievement of the application mission. For example, book authors may qualify as core concepts if the site can be accessed for obtaining information about authors, regardless of books; in this case, the site should treat them as first-class information assets, e.g., by publishing their bio sketch, interviews, and so on.

Reasoning as described above is very useful for abstracting the information content of a Web site and decomposing its E/R schema into well-identified sub-schemas (see Figure 7):

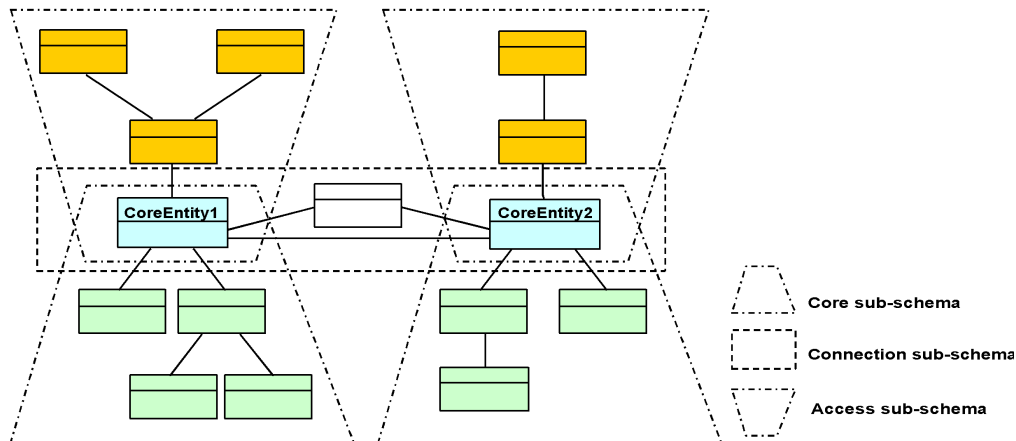


Figure 7. Three basic dimensions that characterize the Web site information content.

- *Core sub-schema*, collecting entities and relationships denoting core concepts.
- *Access sub-schema*, collecting entities and relationships playing the role of access facilitators.
- *Connection sub-schema*, including the entities and relationships that connect core entities.

The above classification is the starting point for the identification of WebML **skeletons**, i.e., pairs of E/R sub-schemas and WebML hypertext diagrams, describing structures that are recurrent within Web applications.

Skeletons represent a simplification of real cases. In fact, very often in real Web applications, pages have extra information, such as advertising or ornamental HTML markups. Since these additional elements are not relevant for understanding the structure of content and navigation, they are not addressed by our WebML skeletons; they of course can be modeled in WebML. Also, real Web applications often cluster a lot of information within pages, possibly collapsing a large number of units within a single page. Our skeletons feature instead simpler compositions, which highlight the most significant content units within pages. Of

course, our skeletons can be easily adapted to more complex compositions. However, such a simplification helps recognizing the regular structure of many Web applications, thus facilitating their understanding, and consequently their design, reverse engineering, and integration.

An important empirical finding that many Web applications have a regular structure has been recently presented in [13]. The main result of that paper is that the Web can be decomposed into cohesive collections of pages, tightly and robustly connected via a “navigational backbone” that supports their strong connectivity, with several other pages pointing *into* the backbone or being reachable *out* of the backbone. Such a result is surprisingly coherent with our framework above illustrated: the backbone represents a connection schema including all the core entities and the relationships between them, while the *into* pages represent the access schema and the *out* pages represent the core schema. Thus, the illustration of Figure 3 in [13], showing a graph of connections between backbone nodes, *into* nodes, and *out* nodes, has several aspects in common with Figure 7 of this paper, although the two figures were independently developed. Such a structure is fractal in nature and thus repeated at various levels (the Web “at large” presents a structure which is similar to the structure of its cohesive portions), thereby suggesting that “to design effective algorithms for data services at various scale of the Web it is sufficient to understand the structure that emerges from one fairly simple stochastic process.” The strength of this study is that the resulting properties about Web hypertexts connectivity have been experimentally derived through the analysis of a large number of Web pages (about 60M). However, such work does not provide any semantic classification of the emerging Web structures.

Figure 8 shows a simplified E/R diagram of the Cisco institutional Web site (<http://www.cisco.com>), which will be used in the following sections to illustrate Core, Access, and Connection skeletons. The goal of this Web site is to publish contents about the company products, and related technologies and solutions that the company is able to offer to their customers. The E/R diagram therefore includes three core entities, describing the core concepts Product, Solution, and Technology; each of them has several attributes showing their properties. Products are classified by means of categories and subcategories; the subclass HighlightProduct groups all bestseller products. Each solution is related to several business and technical resources, as well as to successful stories and possible applications of the solution. Many-to-many relationships link core entities to each other. In the next sections, we will identify some access, core, and connection skeletons extracted from this E/R diagram.

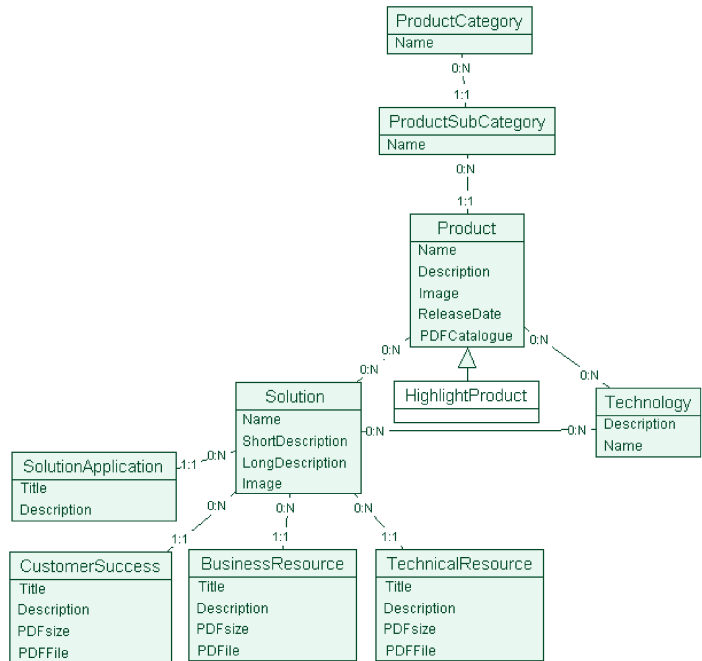


Figure 8. Simplified data schema for the Cisco Web site.

4 CORE SKELETON

Core skeletons support the display and management of the most relevant “business objects” of the site. Figure 9a illustrates a typical data sub-schema for the core skeleton, which normally has a hierarchical structure. The schema is centered on one **Core Entity**, representing the core application object, typically with many attributes representing its descriptive properties. In many cases, additional descriptive properties featuring multiple or structured values need to be sub-structured into internal **Components** of the core entity. These may be, e.g., the multiple variants or multiple pictures of a given product, or the documentation associated with it. The connections of component entities to core entities are mandatory (in relational terms, they include “foreign keys” of a referential integrity constraint). As better described in Section 5, The core entity is normally connected to one or more access entities, belonging to access skeletons defined over it.

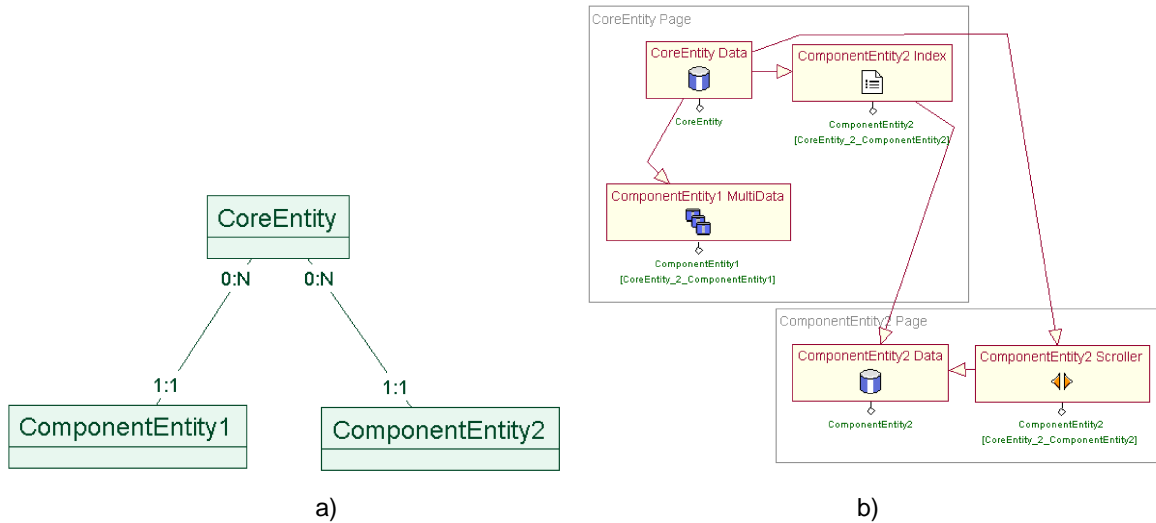


Figure 9. Data diagram (a) and hypertext diagram (b) for the Core skeleton.

The main function of the core schema is to support **content browsing**, i.e., the user’s wish to gather more and more information about the main application objects presented by the Web site. Normally, users select core instances by means of the access schema, and then navigate within the core schema for accessing its features and/or accessing related instances. A typical hypertext for content browsing, illustrated in Figure 9b, provides independent contextual accesses from the core entity to the component entities. By following such contexts, users explore those features of the core concepts in which they are mostly interested. The hypertext shown in Figure 9b uses WebML multidata and index units for showing multiple instances of the two component entities. Instances of ComponentEntity2 can be browsed in a different page through an “Indexed Guided Tour” navigation, obtained through an index unit for accessing instances of ComponentEntity2 and a scroller unit providing commands for scrolling instances back and forth.

4.1 Example

Figure 10a shows one page from the Cisco Web site, supporting content browsing for a specific solution, “Cable Service Provider”. This solution is illustrated within a data unit (A) through its textual property. A multidata unit (B) also illustrates the multiple applications associated with the solution, while three links (C1, C2, C3) connect the solution page to other pages displaying customer successes, business resources, and technical resources.

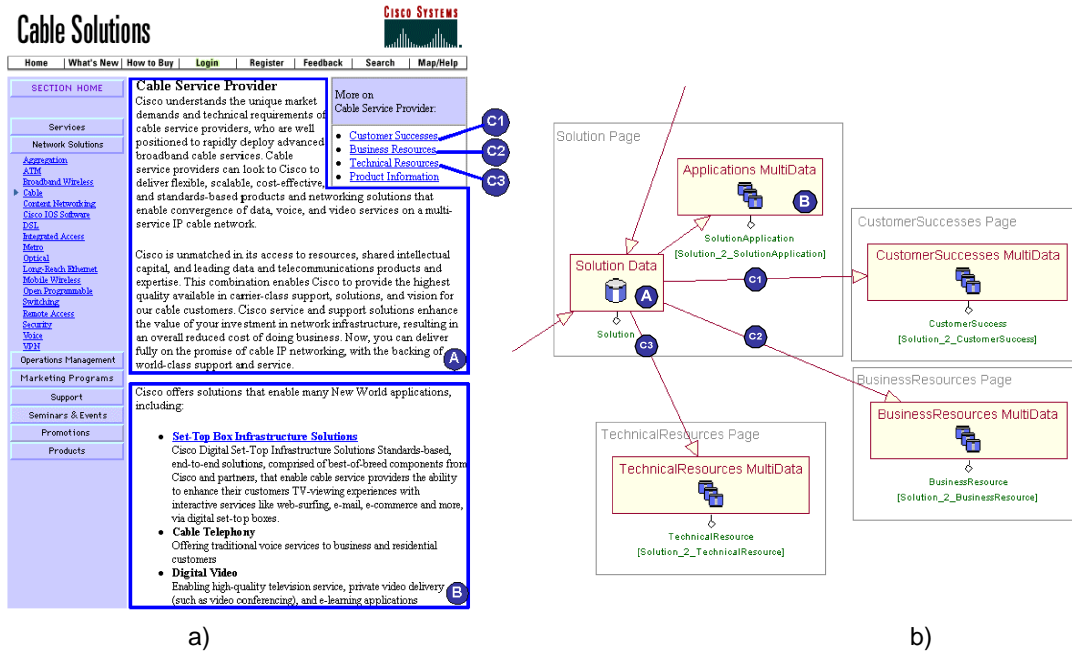


Figure 10. Content browsing for the Cisco Web site (a) and its WebML specification (b).

The core skeleton for Solution addresses entities Solution, SolutionApplication, CustomerSuccess, BusinessResource, and TechnicalResource of Figure 8. The corresponding hypertext is described in Figure 10b. In particular, a WebML data unit (A) shows solution data, and a multidata unit (B) shows the successful applications of the solution. Links C1, C2, and C3 point to pages where multidata units show customer successes, technical resources, and business resources of a given solution.

5 ACCESS SKELETONS

Access skeletons are used for selecting core concepts. Entities of an access skeleton typically represent categorization domains of the core concept. Note that categories must be treated as first class concepts, and not only as a property value of the core entity, because a category may itself store several pieces of information, for example, a representative image, some descriptive text and so on, which illustrates the common features of core concepts in the category.

Figure 11a illustrates a WebML data subschema for the access skeleton, where the central entity **CoreEntity** represents the core concept, which is surrounded by two access entities, **AccessEntity1**, and **AccessEntity2**, denoting alternative categorization concepts. The diagram contains also a sub-entity of the entity Core, **SpecialCollection**, which denotes a collection of representative core concepts.

From the data standpoint, the organization of a typical access skeleton recalls the classical schema of data marts [14], where core data (facts) are the center of several “access dimensions”. Each access dimension can be arranged as a single entity or as a hierarchy of interrelated entities. In the former case, the schema takes the structure of a “star schema”, in the latter it extends into the so called “snowflake schema”. In a similar way, within access skeletons core entities are at the center of several categorizing entities. The analogy is not only in the topology of the schema, but also in the use paradigm: in data warehouses, software is provided for effectively selecting facts, according to several dimensions, thus performing data analysis. In the Web, categorizing entities are used for dynamically accessing core objects, according to several alternative categorizations.

The hypertext expressing an access skeleton provides pages and content units dedicated to the provision of methods for better locating core concepts. Such access methods can be categorized in two broad paradigms:

- *Contextual access.* It occurs when the hypertext supports a form of navigation that consists of zooming in along a sequence of category-subcategory concepts, progressively “moving” from broad categories to narrower ones, until the core concept of interest is located.

- *Non-contextual access.* It occurs when users directly search for the desired concepts, by submitting their keywords. In this case, no context is preserved between search and target pages.

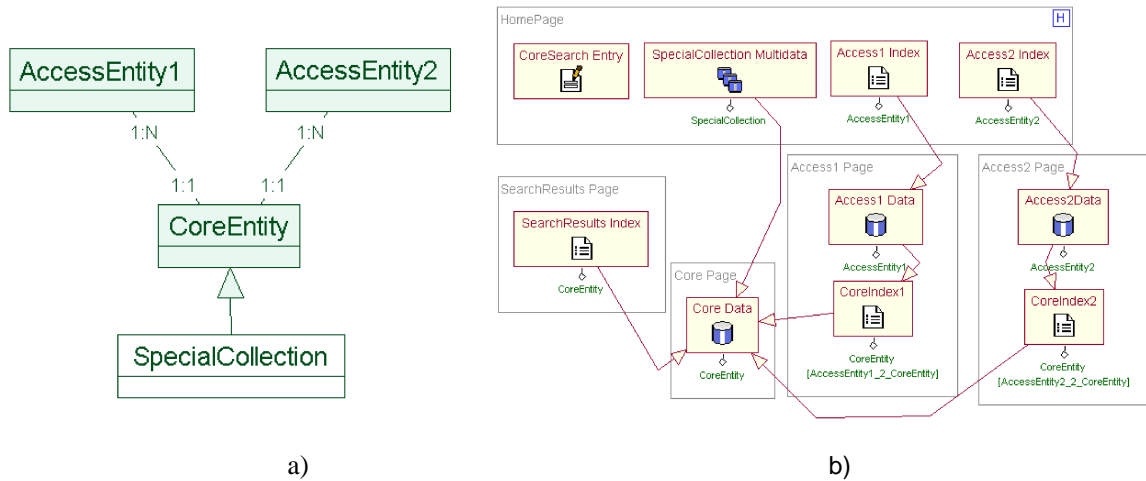


Figure 11. Data diagram (a) and hypertext diagram (b) for the Access skeleton.

Figure 11b shows the typical topology of a hypertext fragment for an access skeleton, where both contextual and non-contextual access methods are present. The Home Page includes two indexes over the categorization entities Access1 and Access2, which are the starting point of two independent contextual navigation chains. The home page also contains an entry unit (CoreSearch Entry), which denotes a search form for starting non-contextual access. Finally, the home page contains a multidata unit showing a collection of representative core concepts, from which it is possible to jump to the page showing the full details of one of the concepts in the collection.

5.1 Example

Figure 12a shows one of the pages supporting the access to products of the Cisco Web site. The page includes an entry unit for keyword-based search on Cisco products (denoted with label A), a link to a compact index on all Cisco products (B), a link to new products (C), and an index of the product categories (D). Therefore, the page supports one non-contextual access and the start points of three contextual accesses.

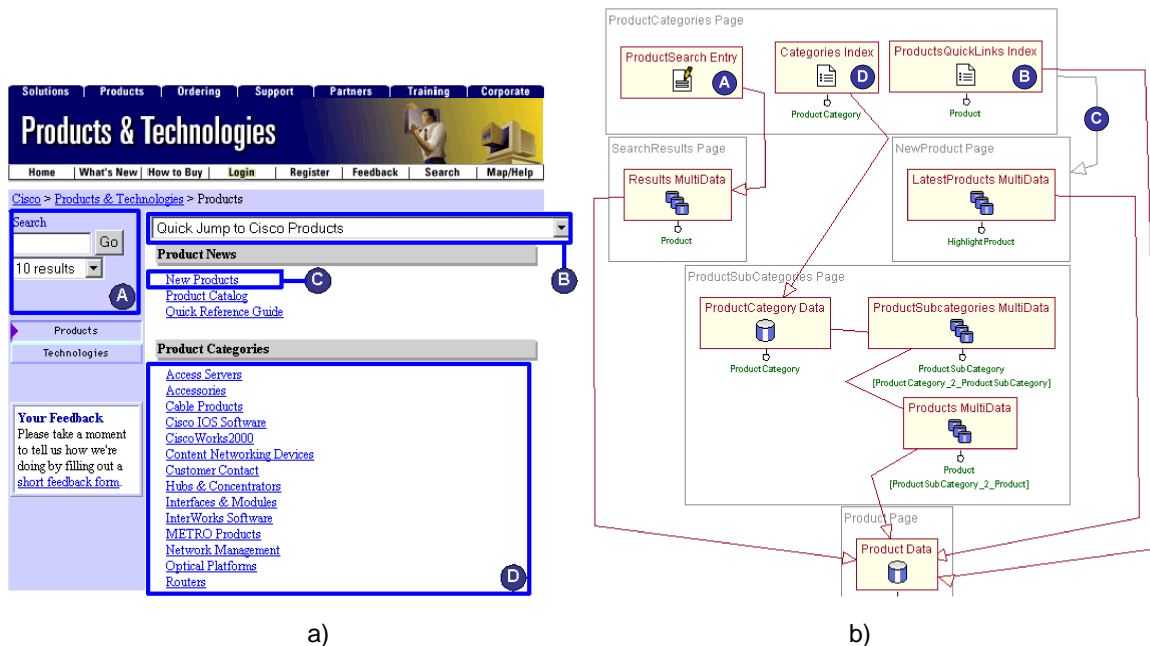


Figure 12. Cisco page for accessing products (a) and its WebML specification (b).

The access skeleton for Product addresses entities ProductCategory, ProductSubCategory, Product, and HighlightProduct of Figure 8. The corresponding hypertext is described in Figure 12b. In particular, Figure 12a corresponds to the ProductCategories page of Figure 12b, which includes the three WebML units (A, B, D) and one link (C), corresponding to the four access mechanisms. The entry unit (A) allows inserting a search condition for selecting several products, displayed as a multidata unit in the SearchResults page, from which users can select one single product. The quick index (B) points directly to products data, shown in the Product page. The multidata unit in the NewProducts page reached by following the link (C) shows the latest released products; users can select one of them, thus getting to the Product page. Finally, the Categories index (D) points to a page showing the ProductCategory data unit and a ProductSubCategories multidata unit, thus enabling a two-step contextual access to products.

6 CONNECTION SKELETON

Connection skeletons support the definition of links between the most relevant “business objects” of the site. Connection skeletons may not be relevant when each core concept is independent. Their definition is instead required when core concepts relate to each other.

In the most general data skeleton, illustrated in Figure 13a, each core entity relates to every other. In such case, there is a binary relationship between each pair of core entities, yielding to a fully connected graph; unless otherwise specified, relationships are many-to-many. The corresponding hypertext, illustrated in Figure 13b, has three identical pages, one for each core entity. Each page includes one data unit for displaying one instance of the core entity, previously selected by using the access skeleton, plus two index units for displaying related concepts of the other two core entities.

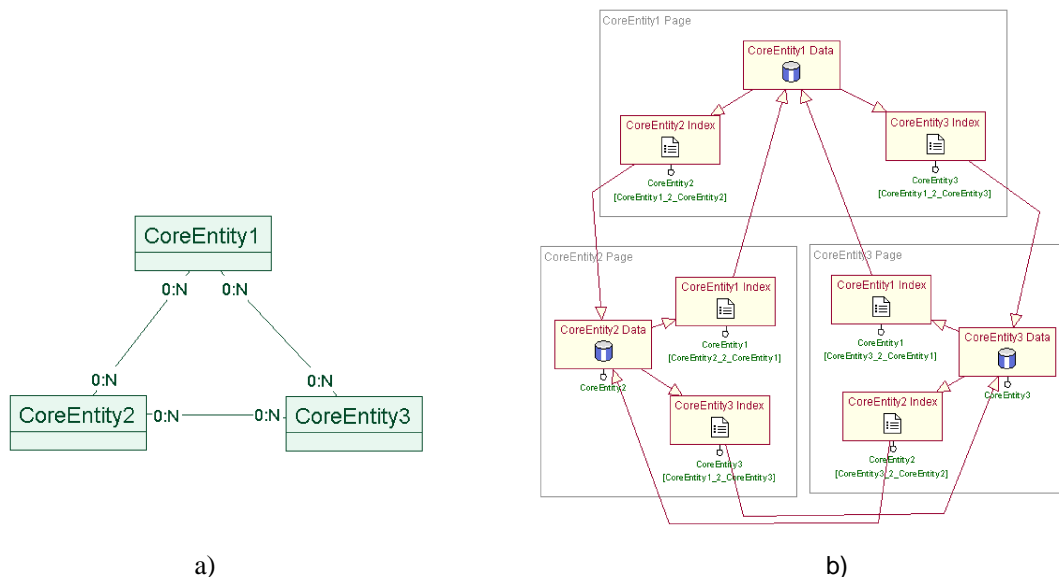
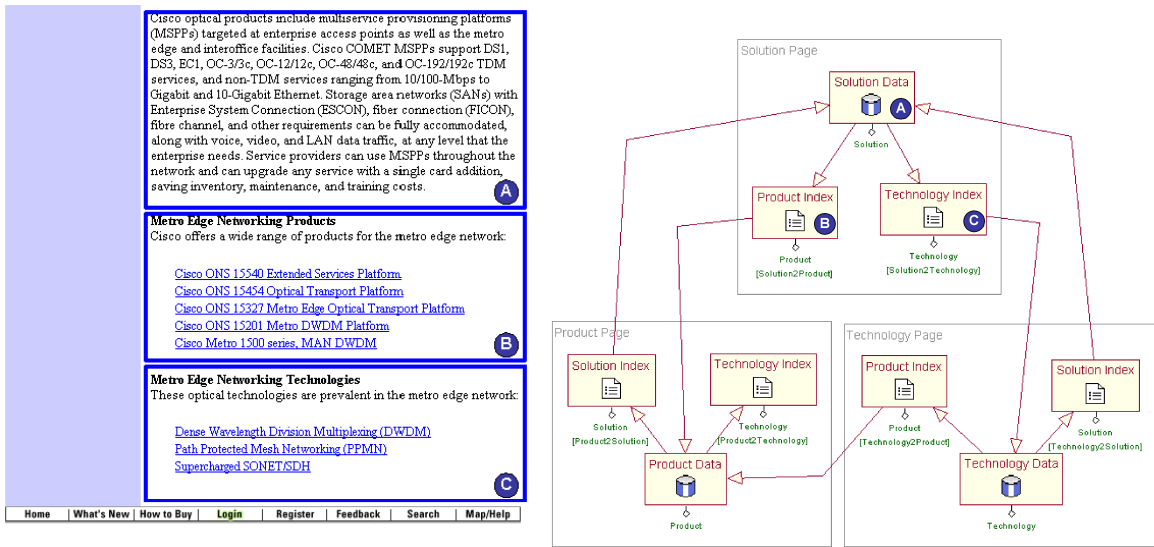


Figure 13. Data diagram (a) and hypertext diagram (b) for the Connection skeleton.

6.1 Example

Figure 14a shows the page of the Cisco “Metro Edge Networking” solution. Solution data (A) are shown in the upper part of the page (in part omitted); the lower part of the page includes two indexes (B, C), respectively pointing to the metro edge networking products and technologies.

The connection skeleton this page refers to includes entities Product, Technology, and Solution of Figure 8, together with the relationships interconnecting them. The hypertext in Figure 14b includes a WebML data unit (A) for solutions and two index units (B, C) showing indexes of related products and technologies.



a) b)
Figure 14. Cisco solution page (a) and its WebML specification (b).

7 CONTENT MANAGEMENT SKELETON

In addition to the previous data access skeletons, many Web applications support **content management**, either restricted to administrators and content owners (e.g., for updating product data in a corporate Web site), or exposed to end-users (e.g., for inserting messages in Web forums or managing user profile data).

In general, the content to be managed, which is the one to be dynamically published within the Web application, does not include the entities of the access schemas, which are predefined and stable information for core content categorization. Content management is instead concerned with the insertion, deletion, or modification of core entities and of the related components (according to the core sub-schema), and with the connection of such entities to access categories (according to the access sub-schema), and to other core entities (according to the interconnection sub-schema). Therefore, typical hypertexts for content management offer the possibility of creating one core and several component entity instances, and then relating them to entity instances of the access and interaction schema. Such hypertexts are largely based on a typical WebML pattern of operations, called “Create and Connect” chain, for generating a new entity instance and connecting it to one or more existing instances.

Figure 15 shows a content management hypertext in which users can select a category for a given core object, then create a core object instance, and connect it to that category. The category is selected via an index (not shown), and some aspects of the category are displayed. The information about the new core entity is entered in a data entry unit (NewCoreEntity Entry), and a simple Create and Connect chain is activated, which generates the data for the core object and connects it to its category. The operation goes back to the CoreEntity page, where the index displays all the items for the current category; one object selected from the index can then be modified (by invoking a chain of data entry and modification units) or deleted (by invoking a delete operation).

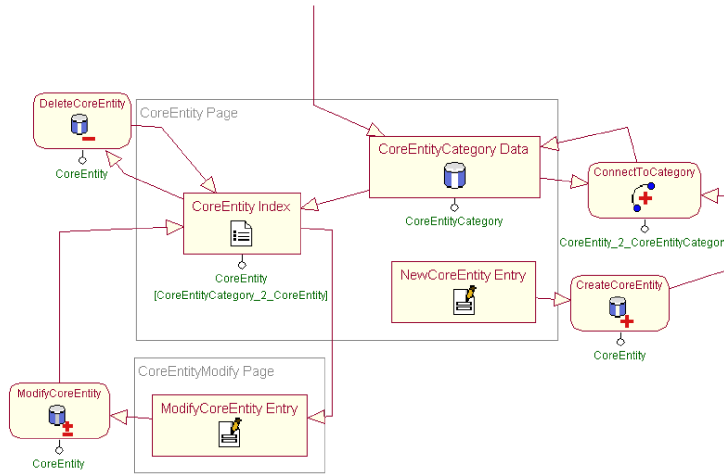


Figure 15. Hypertext diagram for content management of a core object.

7.1 Example

In order to show the application of the content management skeleton, we use an example from the Acer–Euro Web application. The goal of this application is to serve the needs of both internal personnel and customers of the Acer European branch, by organizing, collecting, managing and publishing on the Web content about the Acer products.

The page in Figure 16a is available to the application administrators for inserting data to be published on the Web about a new product family. The left side of the page shows a multidata unit (A), including product families (Series 210, Series 350, etc.) in the Travelmate category. The right part of the page (B) is a data entry unit for entering information about a new product family, with several fields available to administrators. The “Create Family” button (C) activates a chain of create and connect operations that creates the new entity instance and connects it to the appropriate category. In addition, from each product family instance, two buttons, (D) and (E), respectively activate a delete unit, for disposing an existing family, and a chain of data entry and modify units, to change its content.

The WebML content management hypertext is shown in Figure 16b. Once a category of product families is selected, its elements are shown by means of a multi-data unit (A). Two buttons (D and E) activate a delete unit or a chain of a data entry and a modify operation, respectively. The NewProductFamily entry unit (B) includes a button (C) for activating a Create and Connect chain, which generates a new product family and includes it in the previously selected category.

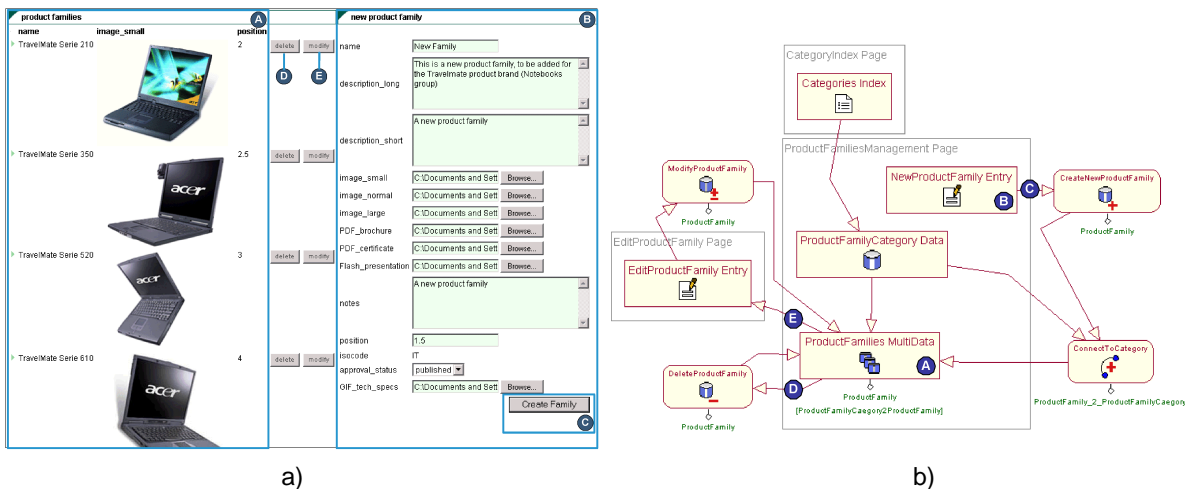


Figure 16. Content management page for the Acer Web site (a) and its WebML specification (b).

8 USE OF SKELETONS FOR DESIGNING, RE-ENGINEERING, AND CLASSIFYING SITES

The main purpose of this paper is to propose skeletons as a reasoning paradigm for understanding the deep structure of Web sites. A simple design method for data-intensive applications, which takes advantage of skeletons, may consist of the following steps:

- Determine the core concepts which need to be supported – the main application objects.
- Build the overall Web site data schema - which includes access and core schemas for each core concept and one connection schema to relate core concepts when needed.
- Build the required hypertexts - by developing the access and core hypertext fragments for each core concept and by linking them through the connection hypertext fragment.

Skeletons are also useful for application re-engineering, which is required when an existing application becomes too inefficient or cumbersome to be managed and need to be redesigned; re-engineering is also the first step of Web integration, when several applications, separately designed, need to be accessed as a single one; in such case, it is very useful that the applications follow the same “logical” organization.

The proposed framework can also be useful for classifying Web applications. We may regard Web sites as *access-centered* when their focus is on supporting a rich variety of access dimensions, and *core-centered* when their focus is on the support and management of core information. Focusing instead just on core objects may help in identifying the Web site’s *business model*, distinguishing the following five kinds of sites:

- Commerce sites*, whose business model is to sell the core object (B2B, B2C, e-shops, e-malls, virtual marketplace, e-auctions).
- Content sites*, whose business model is to give users information about the core object (digital libraries, on-line magazines, recommending systems).
- Service sites*, whose business model is to offer the core object as a service (Webs for order tracking).
- Community sites*, whose business model is to build "socially shared" core objects (forums, chat rooms, newsletters, reselling systems).
- Context sites*, whose business model is to help locating core objects (directories, search engines over local data).

The above classes are almost orthogonal and can be combined; Table 3 shows the URLs of one representative site for some of the combinations.

A modeling exercise with WebML has been successfully proposed to hundreds of graduate students in our laboratory as a means for becoming aware of the deep organization of real-life Web applications. Readers interested in this experience can start with recognizing the skeletons in the Amazon site subset provided in Section 2, and then apply a similar exercise to the sites of Table 3.

	Access-centered	Core-centered
Commerce	http://www.cdnw.com/	http://www.e-bay.com/
Content	http://www.elmundo.com/	http://www.epinion.com/
Service	http://www.dhl.com/	http://www.hotmail.com/
Community	http://developer.java.sun.com/developer/community/	http://www.quote.com/
Context	http://directory.google.com/	-

Table 3. Examples of data-intensive Web applications, classified according to the proposed taxonomy.

9 CONCLUSIONS

This paper has shown that data-intensive Web sites can be abstracted in terms of generic skeletons. This powerful conceptualization has applications in the design, re-engineering, and classification of Web sites. It can be supported by design tools, thus enabling an effective development of Web applications. However, the main merit of this approach is of intellectual nature, and can be appreciated even regardless of the use of specific design environments.

The paper has also discussed an analogy between access schemas and the star or snowflake schemas of data warehouses, and a strong analogy between our approach and empirical findings on the Web topology, reported in [13]. We believe that the process of abstracting core, access, and connection concepts, as described in this paper, is one of the steps of the process described in [13].

Most of the concepts on which skeletons are founded derive from the concrete application of the WebML model and approach to the design of an experimental version of Cisco Web application and the development of the commercial Acer-Euro Web application. WebML is currently used within the WebRatio tool suite (<http://www.webratio.com/>). Current research activities on WebML include the integration of WebML with workflow management and the development of vertical application frameworks, based on the skeletons presented in this paper, for a virtual campus project and B2B applications (e-logistics).

10 ACKNOWLEDGEMENTS

We like to thank the huge team of WebML designers and developers, in particular those people who have cooperated with us in the development of applications. The classification based on business models was inspired by a seminar given by A. Rangone at Politecnico di Milano in December 2000.

11 REFERENCES

- [1] P. Fraternali. "Tools and Approaches for Developing Data-Intensive Web Applications: a Survey". *ACM Computing Survey*, 31(3), pp. 227-263, 1999.
- [2] S. Ceri, P. Fraternali, A. Bongio. "Web Modeling Language (WebML): a Modeling Language for Designing Web Sites". *Computer Networks*, 33, pp. 137-157, 2000 (also in Proc. of WWW9 Conference, Amsterdam, May 2000).
- [3] F. Garzotto, P. Paolini, D. Schwabe. "HDM – A Model-based Approach to Hypertext Application Design". *ACM Transactions on Information Systems*, 11(1), pp. 1-26, 1993.
- [4] T. Isakowitz, W. Sthor, P. Balasubramanian. "RMM: a Methodology for Structured Hypermedia Design". *Communications of ACM*, 38(8), 1995, pp. 34-44.
- [5] J. Nanard, M. Nanard: "Hypertext Design Environment and the Hypertext Design Process". *Communications of the ACM*, Vol. 38(8), pp. 49-56, 1995.
- [6] D. Schwabe, G. Rossi. "The Object-Oriented Hypermedia Design Model". *Communication of ACM*, 38(8), pp. 45-46, 1995.
- [7] P. Fraternali, P. Paolini. "Model-Driven Development of Web Applications: the Autoweb System". *ACM Transactions on Information Systems*, 18(4), pp. 323 – 382, 1998.
- [8] J. Conallen. *Building Web Applications with UML*. Addison Wesley, Reading, MA, 2000.
- [9] P. Atzeni, G. Mecca, P. Merialdo. "Data-Intensive Web Sites: Design and Maintenance". *World Wide Web*, vol. 4, pp. 21-47, 2001.
- [10] G. Rossi, D. Schwabe, L. Esmeraldo, F. Lyardet. "Engineering Web Applications for Reuse". *IEEE Multimedia*, 8(1), pp. 20-31, 2001.
- [11] V. Crescenzi, G. Mecca, P. Merialdo. "RoadRunner: Towards Automatic Data Extraction from Large Web Sites". Proc. of VLDB 2001, Rome, September 2001, pp. 109-118.
- [12] S. Ceri, P. Fraternali, A. Bongio, A. Maurino. "Modeling data entry and operations in WebML". Proc. of WebDB 2000, Dallas (USA), May 2000, LNCS 1997, Springer Verlag, pp. 201-214.
- [13] S. Dill, R. Kumar, K. McCurley, S. Rajagopalan, D. Sivakumar, A. Tomkins. "Self-similarity in the Web". Proc. of VLDB 2001, Rome, Italy, September 2001, pp. 69-78.
- [14] R. Kimball. *The Data Warehouse Toolkit*. John Wiley and Son, 1996.

12 URLs OF THE MENTIONED WEB SITES (IN ALPHABETICAL ORDER)

1. Acer. <http://www.acer-euro.com>
2. Amazon. <http://www.amazon.com>
3. Cisco. <http://www.cisco.com>
4. WebRatio. <http://webratio.org/>

AUTHORS' PROFESSIONAL BIOGRAPHIES

Stefano Ceri is full professor of Database Systems at Politecnico di Milano. His research interests focus on: data distribution, deductive and active rules, object-orientation, and design methods for data-intensive Web applications. He is and has been responsible of several projects at Politecnico di Milano, including W3I3: "Web-Based Intelligent Information Infrastructures" (1998-2000). He was Associate Editor of ACM-Transactions on Database Systems and is currently an associated editor of several international journals, including IEEE-Transactions on Software Engineering.

Piero Fraternali is associate professor of Software Engineering at Politecnico di Milano. His research interests focus on active rules, object-orientation, design methods for data-intensive Web applications, CASE tools for automatic Web site production, and wireless applications. He is author of several articles on International Journals and Conference Proceedings. He was the technical manager of the W3I3 Project, "Web-Based Intelligent Information Infrastructures", (1998-2000).

Maristella Matera is researcher at Politecnico di Milano, where she teaches Computer Science Fundamentals. Her research interests span design methods for Web and Hypermedia applications, Web and Hypermedia usability, and formal specification of interactive systems. She has been awarded several fellowships for supporting her research work at Italian and foreign institutions; in particular, she has been visiting researcher at the Graphics, Visualization, and Usability Center, Georgia Institute of Technology, Atlanta, US.