ABSTRACT
This paper describes an environment specifically tailored for the design and the rapid prototyping of Web applications. The environment, named WARP (Web Application Rapid Prototyping), offers a set of online software tools, which assist the designer and the user browsing of a Web application, in all its different aspects.

The environment is based upon models and techniques already used in the hypermedia, information systems, and software engineering fields, adapted and blended in an original mix.

The foundation of the proposal is the conceptual design of Web applications, starting from HDM, a notation for the specification of structure navigation, and presentation semantics.

Compared to existing tools, WARP is an aid for analysis and design of Web Application because it introduces a new approach completely on-line (both the design and the execution environment). So the author can immediately formulate and evaluate requirements, specification and designs.

A real-life experience of the use of the methodology and of the WARP in academic context are reported.

Keywords
HDM, Prototyping Web Application, Conceptual Design

1. INTRODUCTION AND BACKGROUND
In general a prototype is a simplified model of a proposed system that is built for a specific purpose [1], such as:
- Formulating and evaluating requirements, specifications, and designs.
- Demonstrating feasibility, system behavior, performance, etc.
- Identifying and reducing risks of system mis-development.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage, and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Copyright 2004 ACM 1-58113-812-1/03/04...$5.00
in the early stage, when the design needs to be checked, tested, discussed and revised several times. Actual development, instead, requires all the design details to be laid down.

- in the Multideivery, when it happens that the same content must be delivered off-line, on-line, with different levels of depth and completeness, etc. In this case, it is clear that design must be completed independently from any implementation that, afterward, can proceed in several different ways.

- in the Technological flexibility, more and more delivery environments for the web are emerging. Given this situation it is necessary to choose a specific delivery environment, or risk cutting out new exciting possibilities, as it would inevitably happen in a full scale development environment.

- in the design advanced options, when sophisticated applications require advanced design features, for which fully automated implementation could not be devised. It would be negative to not include those features in the design, just because automatic implementation could not support them.

- to the design incompleteness, for advanced applications, there are specific features, which can not be described by even the best design model. These features are therefore not modeled in the schema, but they are rather described through words, diagrams and ad-hoc means. It is clear that these features will be left out by any automatic implementation based on the design model.

Prototyping means more than rapidly developing user interfaces; prototyping is a central part of a development strategy; it can be considered as:

- an approach based on an evolutionary view of software development, affecting the development process as a whole.

- the production of early working versions (“prototypes”) of the future application system and experiments with them.

In our work we prefer to consider prototyping in the evolutionary way; in the following the main concern is the definition of WARP (Web Application Rapid Prototyping), an environment to support design and prototyping of Web Applications completely on-line.

WARP is based upon an evolution of the HDM2000 model, which has been selected because it is very well documented referring to a number of complex projects, such as UWA [19].

The related works are illustrated in section 2. In section 3 we describe the main features and the overall architecture of WARP. In section 4 we shortly examine an application for which WARP is being used. In section 5 conclusions and future work are described.

2. RELATED WORK

Researchers have proposed several models and tools to support the different phases of Web applications development.

Among the abundant efforts, WEBRatio[9] is an environment based on the WebML language to describe presentation and navigation aspects and to generate the web application. Conallen[18] extended UML notation to model web specific elements, and thus made it possible to model the whole web application with UML. RMM is a database-driven methodology for structured hypermedia design. Its principle idea is to provide a visual representation of the system to facilitate design discussions. It suggest an iterative process to decompose visual components to database relationships.


Gaedke utilized WebComposition and WebComposition Markup Language (WCML) [17] to present a systematic approach for code reuse in component-based web applications. IIPS [16] modeled navigational structure, compositional structure, and user interface through ontology, and provided tools for code generation.

MODFM [15] is an architecture-based methodology to guide developers quickly constructs a prototype of web application. MODFM applies a top-down approach to functionally decompose a system into web pages organized in a menu system, while generating the codes from front-end all the way to backend, to bottom-up integrate codes into a running mockup system.

All of these efforts offered facilities to speed up design and development of web applications.

However, none of them provide efficient support for the end to end process of generation and maintenance of whole web applications.

Our investigation indicate that the most thorough methodology is, in our opinion, UWA [19] (and the related HDM2000 model), due to its ability to integrate customization aspects, requirements elicitation, operational aspects, transactional aspects which arise when a complex Web application is designed. These aspects are generally not taken into account, by tools like Araneus or WebRatio, which stress the distinction between data structure, navigation, and presentation instead.

JWeb [13] provide a design/prototyping tool, integrating XML technology with HDM to help the design the conceptual schema.

WARP is a natural evolution of JWeb based on the new evolution of methodology and on fast-prototyping completely on-line.

3. THE WARP PROJECT

The WARP project, presented in this paper, proposes a methodology and a development environment for fast-prototyping of Web Applications, which support the whole development cycle, from the first phase of requirement analysis and design of the main information and system features, till the detailed definition of these schemas and the production of applications (coding, database creation, database filling, creation of interfaces, etc.).

3.1 Goals

WARP is based on an evolution of HDM2000 model and the related UWA methodology.

In brief, UWA methodology requires that after the indispensable Requirement Analysis phase, conducted in accordance with a goal-oriented approach, the following steps should be performed:
- **Information Design**: the goal is to describe the information with which the application is going to deal, giving it a structured organization. The schema is organized into two different parts: the hyperbase schema and the access schema. The hyperbase schema describes the basic navigational capabilities offered by the application whose components are "Entities" (organized into Entity types) and "Links" (organized into Link types). The access schema describes the organization of the access structures ("Collections" in HDM2000 terminology). Both for the hyperbase schema and the access schema there is a sharp distinction between design in-the-large, where the general features of the design are defined, and design in-the-small, where the details are provided.

- **Navigation Design**: this makes clearer the most important aspect of hypermedia applications, reconsidering the information and its organization more typically from the viewpoint of its fruition and defining the navigational paths the user can follow.

- **Publishing Design**: what emerges from the previous steps must be completed with considerations on the desired presentation strategy, and organized into “pages” and “fruition units”.

- **Operations Design**: this is the step in which all the functional and transactional features (such as “register”, “submit”, etc.) beyond the pure hypermedia paradigm are modeled. Here the model allows the user to invoke the “functionalities” of the application.

In the design process all the schemas have to be examined, discussed and revised. Because of the methodology is yet young and not firm, it is important to have a real-time feedback on the design choices. So the designer needs a good support to check his ideas.

Starting from schemas, the designer should use a sample of actual data, to “visualize” the outcome of the design being developed.

At this stage different options are possible:

- In terms of visual communication (interface), a “functional” interface (i.e. an interface simply holding the different information items, the buttons, etc. without any pretense of visual impact) could be developed, leaving for the future the development of an actual interface. Another possibility is to plug in the prototype and to test several different high quality interfaces options.

- In terms of access structures and hyperbase, different alternatives could be tried in order to compare different solutions and to choose the most appropriate ones.

- In terms of detailed design, the individual information items could be completely identified, or they could be just sketched, leaving their specification to a later stage.

The combination of the different options actually defines a number of different scenarios; we are convinced that an effective supporting environment should not enforce a specific way to proceed, assuming that the “best practice” exists, but rather it should allows a great level of flexibility. It is our experience, in fact, that in different situations, for different projects, we have used different strategies and different tactics.

Our main goal is therefore to create a design environment with fast prototyping capabilities, to support HDM2000 modeling language. In summary the WARP project’s goals are:

- to adapt the HDM2000 model to the needs of fast prototyping;

- to create a suite of tools to support the design of complex Web applications with HDM2000.
- the prototype is modifiable according to the evolutionary approach which allows to obtain the final version of the Web application.

### 3.2 Warp Development Process

The process of developing a Web Application with WARP is shown in Figure 1.

The initial step is the collection of requirements and their formalization as a set of conceptual schemas in HDM2000. This phase is human-intensive, but it is supported by a tool of the WARP System called MODE[14], which permits the editing, archiving, and evolution of HDM2000 schemas.

The second phase is the generation of the supporting database; this phase takes as input the HDM2000 conceptual schema and produces as output a relational database (Editorial DB). The database consists of two parts, a part containing a representation of the structure, navigation, and presentation (called metaschema database), and an empty database for storing the application contents. This phase is totally automated by MODE.

The third phase is the management of the contents. There are two scenarios for this phase:

- the application contents do not exist: the empty database produced by MODE[13] is filled with fictitious content to test the navigation and presentation aspects by the tool WFeeder. It allows subsequently to manage all the real contents;
- the application content (or part of it) already exists and is stored in a legacy system (e.g., a relational database): in this case, the database schema produced by WARP must be mapped onto the legacy data sources in order to integrate the existing content into the page production process. This activity, although not supported by a specific tool of the WARP architecture, can be performed with the help of commercially available data replication tools, which allow the data administrator to map a database schema on top of a set of heterogeneous distributed data sources, using such techniques as views, triggers, and automatically executed data conversion programs.

In both of the above cases, the fourth step is the definition of presentation aspects. There are two scenarios again:

- it is possible to choose a presentation model among existing model set;
- it is possible to define new presentation aspects or to customize the exist models using the tool WPD (WarpPageDesigner). All the information about the publishing are stored in the Editorial Db.

The last phase is the generation (using the Generator tool) of the Runtime DB that will be used from the WEngine to allow at the user to browsing the Web Application. The Generator can produce different applications from the same Editorial DB according to users, languages and devices.

### 3.3 Overall Architecture

The WARP system is made up of two main parts as shown in Figure 1:

- **development**: for defining and customizing the application and the data for a particular set of users and/or a particular presentation using HDM2000, translating the detailed design into a relational database schema and inserting a sample of actual data in order to create an instance of the application;
- **execution**: to deliver the application on the web starting from the HDM2000 schema and from the content repository defined in the previous steps.

The Development Environment consists of four tools, as shown in Figure 1.

#### 3.3.1 MODE

MODE[14] allows the designer to define HDM2000 schemas of the application, by providing design in-the-large and by providing a full design in-the-small. To facilitate visualization and editing, two different perspectives are offered, which isolate and display separately the requirements schemas from the Information and Navigation aspects.

The designer can work simultaneously at different levels and on different parts of the design in accordance with HDM2000 methodology.

All information on the created models is stored in a relational database. Because of this, the tool allows the designer to summarize the information according to different criteria. Thus we can find out how many collections an Entity belongs to, or which requirements are satisfied by an element in the Requirements Elicitation. As soon as a change occurs, the resulting consequences for the entire project are known. In the other existing editor such as Araneus, WebRatio, and others we can not find this innovative features. In addition to this innovation, we have introduced the concept of "family of schemas" which is particularly useful to generate different design for different business strategies starting from the same business context.

Two different version of MODE are available: the first is off-line based on Microsoft Visio and the second is an editor on-line. Both these versions allow the export of the schema in the XML format in order to have the complete independence from the WARP environment.

#### 3.3.2 WFeeder

The tool is not for content management, but it is complementary to this kind of tools. It uses a content management tool to acquire the final version of the content we want to publish and stores it in the relational database. Our innovation lies in taking into account first the communication aspect of the content and then the presentation needs. It is independent from the application because it takes in input the HDM2000 schemas. No other tools have this feature, but they have to develop the application for the data management (e.g., WebRatio makes available some primitives to define the interface with contents).

One of the key factors for making model-driven design effective is the possibility of delivering fast prototypes of conceptual schemas in the early stages of development, when the final architecture of the site may not be in place yet. To enable fast prototyping, WFeeder includes one feature that can be invoked to produce “in one click” a sample of actual data, or a simplified version of them, in order to create an instance of the application. The goal of sample actual data is to validate the navigation aspects, so adopted techniques are not so complex as natural language techniques.
3.3.3 WPD
Starting from the HDM2000 schemas the publishing designer has to think how contents must be grouped and presented to the user. The Warp Page Designer has to be invoked to visually define the presentation of the Web Application. In the Browser, the Presentation page can be defined with the help of commands for drag and drop, resize or insert/delete Publishing Object (related to HDM2000 objects).

The input to design the description of presentation pages in terms of spaces and selected contents (in XML format) is provided by the HDM2000 schemas.

Once the Presentation Page is consolidated, a Style Sheet can be defined using a graphic tool (e.g. DreamWeaver, FrontPage and so on). So it is possible to evolve the initial prototype until the final application.

3.3.4 Generator
It creates a Runtime DB, if necessary, physically building the hyperview to have better performance for on-line application (with few changes during the time).

The Runtime Environment implements the logical-to-physical mapping and it is responsible of managing WARP applications and serving page requests coming from remote users.

Its main tool is WEngine.

3.3.5 WEngine
The WEngine dynamically delivers application pages constructed from the hyperbase objects, the metainformation about navigation, and the presentation style sheets.

The input to build the page request by the user is provide by the schemas, contents and layouts stored in the runtime database. So it is a very reusable component.

Also the WEngine can be used independently from the WARP environment.

It use the Event Recorder to store the informations about the user, their navigation and their collections (e.g. shopping bag).

3.4 From the Schemas to the Prototype
The passage from the conceptual schema of the site to the actual pages that constitute the application requires three mappings, which progressively transform high-level specifications into physical-level pages.

The conceptual-to-logical mapping takes as input the conceptual schema of the site, expressed by the structure, navigation, and presentation schema, plus the definition of a fixed database schema for storing metadata. The output is twofold: a metaschema database containing a description of the structure, navigation, and presentation of the site; and an empty database schema ready to store the structured part of the application content, namely, the slot values of components. The Database is named Editorial DB.

The logical-to-physical mapping is an original technique for the production of pages from conceptual schemas and content. The transformation requires the following steps:

- the user-command is interpreted by the WEngine;
- the WEngine, according to the call coming from User, updates the navigation state of the user and, using the run-time database, builds the presentation page (XML format plus XSL for Style Sheet informations).

One of the relevant features of the WARP environment, in order to keep modularity and flexibility, is that all the exchange of information among the different tools is based upon XML. XML is used to exchange schemas, parameters and also instance data. This technical choice allows a very easy strategy of enhancement, addition, substitution or replacement of all the different tools. It could be also envisioned the exchange of information (export and import) with other design and development environments.

Actually we use a proprietary format to exchange the information but we have planned to use the emerging standard as XMI.

From the technical point of view all the tools has been designed using UWA methodology. The current version of WARP has been implemented and tested using Microsoft VisualStudio.net environment and the DBMS SQL Server.

The on-line development environment (i.e. MODE, WFeeder and WPD) has been realized whit vb.net language mixed to Javascript client-side functions to implement all user-friendly features.

The execution environment is a web service and it uses SOAP to communicate with the client.

4. AN EXAMPLE APPLICATION
WARP has been used since last year to develop several applications such as fashion catalogs.

In this paper we will describe a typical use of the innovative environment and we will show the results of the UWA project.

Using the described architecture an application for the sale of credit cards by “Banca 121” have been developed in the last year at SET-Lab as part of UWA European project [19].
The application was selected for the following reasons:
- we are dealing here with three quite distinct types of application user (Multi-user).
- the application needs to be accessible from more than one type of device so as to maximize the facility of access for all user types (Multi-Device).
- this application satisfies the criterion of ubiquity because in many ways it is important for the user to be able to access the system in a variety of situations (Ubiquity).
- some of the users of the application, generally bank customers, have a profile which allows them to personalize the content of the application and the functions available to them (Profiling).

The content management phase is shown in Figure 3. On-line we can insert, modify or delete the contents of the Debit Card Entity Instances (in this case we can see the image slot).

The following step illustrated in the Figure 4 is the definition of “Debit Card” Presentation Page. It is designed in a visual way positioning the publishing Object on the sheet. Then the prototype can be generate and the user can immediately browsing.

The prototype, result of WARP development process, is shown in Figure 5.

The stakeholders we identified are: Generic User, Bank Customer without Card, BANCA 121 strategy manager, Bank Customer with Card, Salesman, and Product Manager.

The editing of Information model, in particular the description of “Debit Card” Entity Types in-the-small is shown in Figure 2.

5. CONCLUSION AND FUTURE WORK
We believe that fast prototyping is an additional important feature of a design environment; fast prototyping means that the application somehow gets implemented, more or less directly
from design, but in a fashion not binding for the final implementation, and not necessarily in an efficient manner.

Once created the prototype must be evolved to create the final web application.

In our opinion, WARP can be used very effectively as it is, for both introductory and advanced training of design, and fast prototyping. WARP is available on http://warp.unile.it/warp and the access can be requested from mario.bo chicchio@unile.it.

The next steps in the development of WARP are:
- Adding help facilities to explain modeling concepts or to suggest design examples in a context-dependent way
- Adding Operation aspects.
- Adding User Visibility on information model
- Adding Multidevice and Multilanguage supports
- Creating of semi-automatic documentation facilities

WARP is the first environment for fast-prototyping completely on-line.

6. REFERENCES


Mario Bochicchio (mario.bochicchio@unile.it) Head of the Software Engineering & Telemedia Lab (SET-Lab) Innovation Engineering Department - University of Lecce, Italy. More than 10 years of teaching experience ("Hypermedia Design", "Database Applications", "Educational Technologies") in various Italian universities. Currently involved in various national and European projects about "Ubiquitous Web Applications", "Educational Technologies" and "ICT Technologies for Cultural Heritage".

Nicola Fiore is a Ph.D. student in information engineering at the Department of Innovation Engineering - University of Lecce - Italy. His research activity concerns in modelling methodologies and their formalisation (HDM,W2000 and UWA) for hypermedia applications, and the design and development of tools for rapid prototyping. Contact him at nicola.fiore@unile.it