WEB APPLICATIONS: OUR IDEA FOR A DEVELPOMENT FRAMEWORK

Luca Contursi
Dipartimento di Ingegneria dell’Innovazione – Università degli Studi di Lecce
Via per Arnesano, 73100 – Lecce, +39 0832 320229
luca.contursi@unile.it

Roberto Paiano
Dipartimento di Ingegneria dell’Innovazione – Università degli Studi di Lecce
Via per Arnesano, 73100 – Lecce, +39 0832 320229
roberto.paiano@unile.it

Andrea Pandurino
Dipartimento di Ingegneria dell’Innovazione – Università degli Studi di Lecce
Via per Arnesano, 73100 – Lecce, +39 0832 320229
andrea.pandurino@unile.it

ABSTRACT
The increased complexity of Web applications makes us sure of the importance of making, before starting the development, a suitable organic design of such an application. Most conceptual design technologies are powerful analysis tools but are too far from the implementation and result almost useless to produce a complete work. We want to build a set of tools, able to fill the gap, letting the developer use the conceptual result of the W2000 analysis as a “ready to produce” methodology.
In this Work In Progress, we refer about the first step of our research, in which, trying to verify the effectiveness of a methodology like W2000 [1] [2], based upon the well-known HDM model [3], we produce a pilot application using a standard technology.
This experience will be the base for the framework we are thinking of, in order to guide the developer along all the development process, making the phases as automatic as possible. We describe the reasons and the goals of our work, during its working out, as well as the results we already got. We also express which will be the guidelines we are going to follow, moving on the second step.

KEYWORDS
Hypermedia, W2000, Implementation, Model

1. INTRODUCTION
“The more complex is the task, the more attention we have to pay for it”; these are words we must be aware of during a very complex process, such as the production of hypermedia applications in general, and of web applications in particular. The availability of commercial products, able to help the developer in the direct production of what the Market demands, and the particular features of the applications themselves made the design phase less considered, if not judged useless.
Most developers refused the already firm design methodologies of the Software Engineering, considering them not suitable for the new media, and too expensive in resources, time in particular.
As long as people dealt with simple presentation applications, small in size, problems didn’t rise in a worrying way. Nevertheless, as they grew in dimensions and complexity, the applications started to have features similar to the ones the traditional software had, combining data-intensive and transactional matters with purely navigational ones.
The consequences of such a situation are showing out and forcing the whole IT community to a reflection. It’s therefore clear that new methodologies are needed, and these may arise from the traditional ones, but the new requirements have to be taken in account.
At the HOC (Politecnico di Milano) and the SetLab (University of Lecce) has been developed W2000, which tries to reach this goal, allowing an accurate conceptual modeling of the applications, before the implementation phase. Around W2000, now, we want to build a complete development framework, able to put the methodology nearer to the implementation, so that the precious design effort can quickly become a real application, as much as possible in an automatic way.

1.1 The W2000 design methodology

To understand the topics we are going to talk about, it’s necessary to know the W2000 methodology fundamentals. It assumes that it’s unavoidable a good distinction between the different aspects of the application which have to be observed during the development, in order to make the development itself a structured and easy controllable process.

After the indispensable phase of the Requirement Analysis, made following a goal oriented approach, the methodology suggests a sequence of steps briefly summarizable in:

- **Information Design**: it’s goal is to describe the information the application is going to deal with, giving it a structured organization. Important features of this phase are that, during the construction of the information structure, the user point of view [4] is held as fundamental.
- **Navigation Design**: it 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 results after the previous steps must be completed with pure presentation reflections and organized in “pages” and “fruition units”.
- **Operations Design**: it’s the step in which all the functional and transactional features (such as “register”, “submit”, etc.), being beyond the pure hypermedia paradigm, are modeled. Here the model lets the user invoke the “functionalities” of the application.

Moreover, operations are the “building blocks” to be used to support complex transactions: conveniently combining them the designer must pay attention to consistency of the application state and also of the behavior of the users. The “customization methodology” [5] [6] comes together with the hypermedia design, to manage the different user profiles and the various devices in many contexts.

2. CREATING THE APPLICATION

Before we could face up the construction of a complete development framework around the methodology, we had to properly understand the amount and the nature of the gap between the result of the analysis (with the described steps) and the starting point of the actual production. It was necessary to verify that the W2000 designer point of view agreed with the one of who was intended to think about the software architecture and the implementation.

That’s why we decided to produce a pilot application, choosing it for its own features of being general, transverse among information, navigation and functional aspects.

Our choice went on a bank web application to manage the debit cards of the bank, though for different delivery devices and addressed to various kinds of users. The features of the application were suitable to fully verify the completeness of the methodological support, but also to give a good sense of the problems concerning the pure implementation. Our attention, therefore, dwelled upon the following elements:

- to verify which logical steps had to be taken to obtain, from the conceptual model, a true implementation.
- to verify the support that the model gave, first to the designer of the software architecture and then to the developer.

2.1.1 The architecture

According to the goals of validating the methodology, we chose to go directly on the development of the specific application, using existing market standards, and so a multi level architecture, immediately recognizing the presentation, business and data levels, each one with its own features and functionalities. W2000, focusing on the design at a conceptual level, leaves complete freedom to choose the particular technology for the implementation. So we decided to use the Microsoft development framework .NET.
This choice doesn’t affect the generality of the results we obtained and is perfectly in accordance (as we will explain later) with the object-oriented nature of the W2000 model.

2.1.2 The three levels

Under the choice we made, the “Debit Card” application has been developed using the three levels architecture, such as:

- **Data access**: it has access to the application data.
- **Business**: it holds all the business logic behind the application; it is concerned in the support of the multi-user access, making clearer the users’ roles and the relative behaviors of the application.
- **Presentation**: it’s the level which, using the data supplied by the data access level, filtered according to business logic, adjusts the presentation for the various devices, composing the pages for the Web, PDA, WAP clients, etc.

As the model suggests, all the information content of the application must be described during the Information Design. As a result we have a series of diagrams, in which the reality, as it’s perceived by the various users, is represented in terms or primitives (entity types, component types, slots, etc.).

Turning from the conceptual model of the information into the implementation in a database, we created a classic Entity-Relationship schema, starting from the description of the application domain and of the data structure already available from the model, using it as it was an highly detailed document for the requirement analysis.

The database we obtained in such a way is very easy to read and the effort needed to turn to a classic E-R from a W2000 Information schema has been very poor: the diagrams resulting from the Information Design are easy to be turned into E-Rs, removing the redundant links.

Once the data structure was well defined, using stored procedures and appropriate views, all the other structures of the model could be reproduced, keeping a good degree of flexibility when accessing the data.

A peculiar feature of the modern applications is that they are getting even more multi-device and multi-user. Thanks to the Object oriented approach of the chosen development framework and to the property of polymorphism [8] of the objects, it’s possible to implement and get, with a little effort, different behaviors, depending on the role of the user who asks for a service.

According to the model resulting from the Publishing Design and to the kind of implementation we chose, it’s possible to make an accurate mapping of the model structures on the .NET framework ones.

The Publishing Model allows the production of pages moving from page templates. Then the pages are modeled as unions of Sections and these as aggregations of Publishing Units, in a hierarchical structure, which suitably describes the page.

In a similar way, .NET, using ASP.NET allows to create UserControl classes, already provided with rendering interfaces (HTML, WML, etc.), we can compose a presentation page with.

In order to make the concepts clearer, we can take as example the structure of the home page for the pilot application:

![Publishing Model of the Homepage for the pilot application](image-url)
Correctly explaining the UML diagram of the structure for the page, we are able to make an exact mapping inside the framework .NET and to obtain the web page.

Figure 2 - Mapping of the Homepage for the pilot application

In figure 2, can be remarked that the various UserControls inside the page are instances of the class below; this is made clear using appropriate names: CatalogueIndex1.ascx, in fact is an instance of the class (UserControl) CatalogueIndex. These features of the .NET framework perfectly agree with the goals of the W2000 model, giving us a good feedback on the choices we made.

2.2 The Second Step

Starting from the results we obtained, we are now planning to build the complete W2000-based development framework. Obviously the same application we made in a traditional way will be the first one we will validate the different approach on.

From the new point of view, we are now giving the specific application a structure that can be already suitable for the framework. This structure must be the input for the engine which will be able to automatically prototype application itself but also all the applications in the same family.

Such a development engine would be an immediate result, but it would be also greatly “re-usable” for other applications. This solution will focus on the implementation problems of the whole model, not only of the pilot application.

All the W2000 design steps must be considered as real developing steps, specific tools must be created and right new solutions must be found. For example, regarding the information design and the relative piece of the framework, this is the way we are going to work. Instead of creating an E-R schema for the single application, we now are thinking about a database which represents the information structures of the model in a completely general way, describing the primitives themselves and the links among them. The specific application data would fill the database up, according to their classification in the model.

In other words, turning from the model to the implementation of the data, it’s time to describe the container of the information (the structures) and not to make any re-engineering effort and to describe the information itself anymore. The resulting data structure will be completely independent from the application domain (the database describes a meta-model of the data) and does not require any effort to the developer. He can avoid any design of the data (for example in an E-R schema).

3. CONCLUSION

All the time we spent in developing a real application, starting from its W2000 model, but using standard technologies, has been of great use as long as all the development phases were made having in mind the new point of view. Now we know more deeply the advantages and the problems coming up when turning from the conceptual model to a real implementation; we have a complete knowledge of the application domain, of the requirements and of which behaviors it should have. We are now aware of the limits that W2000 still shows and gave a good feedback about it. But, most of all, we have already made a lot of choices and collect a lot of ideas for the present and future work about the framework itself.
REFERENCES