Supporting Web User Interface Prototyping through Information Modeling and System Architecting

Xiaoying Kong1, Li Liu2, David Lowe1

1 Faculty of Engineering, University of Technology, Sydney, Australia
2 Project Management Graduate Programme, Faculty of Engineering, The University of Sydney, Australia
xiaoying.kong@uts.edu.au, l.liu@staff.usyd.edu.au, david.lowe@uts.edu.au

Abstract

Existing design methods used for developing web-centric systems are mostly adapted from methods for designing traditional software systems. Web-centric systems however differ from traditional software systems, in terms of both organizational and technical characteristics. Effective design methods for web-centric systems need to address these characteristics specific to web-centric systems. This paper proposes a design method for web-centric systems. The design process comprises three steps: prototyping, information modeling and system architecture design. The method is differentiated from existing design methods in that the design process commences from user interface prototyping. Information modeling activities are further enhanced in this method. To cope with the complexity of web systems, each design step is partitioned into both structural modeling and behavioral modeling. The design method is illustrated by applying the method to the design of a commercial web application.

1. Introduction

The nature of web systems is very different from conventional software systems. On the technical side, web systems typically have a tighter linkage between the business model and the technical architecture; have more pronounced open and modularised architectures; use technologies that change rapidly; demand effective information design and content management; call for more emphasis on the user interface; and place increased importance on quality attributes in mission critical applications that are directly accessed by external users. On the organizational side, web systems typically face a high level of client uncertainty of their needs and also in understanding whether a design will satisfy their needs; have high levels of volatility in requirements and project scope due to the evolution of the business model; have shorter delivery timelines; demand fine-grained evolution and maintenance with an ongoing process of content updating, editorial changes and interface tuning [1]. These unique web characteristics demand tailored development processes and design methodologies.

To effectively cope with these organizational challenges, practitioners have increasingly adopted iterative and agile methodologies in web development processes. These methodologies include the Rational Unified Process [2], eXtreme Programming (XP), Adaptive Software Development, Cockburn's Crystal Family, Scrum, Feature Driven Development, Dynamic System Development Method and Lean Development [3], and Macromedia’ Fusebox Lifecycle Process (FLiP) [4]. There are also numerous design methodologies for web systems that focus on technical challenges – particularly information design - such as RMM (Relationship Management Methodology) [5], OOHDM (Object-Oriented Hypermedia Design Model) [6], and WebML [7]. Nevertheless, as discussed below, most of these methodologies do not fully address common web characteristics and often do not align with commercial web development processes. This paper presents a web design method that supports typical commercial web development practices and takes into account the challenges of web system characteristics. The design method is based on a web system trace model [8] and a modified FLiP process. It deals with the complexity of web systems by separating the concerns of different stakeholders of the web systems. The perspectives of each group of stakeholders are then modeled into a structure model and a behavior model. The traceability among the analysis and design artifacts between different stakeholders is specified.

In the following sections, literature is reviewed in Section 2. Section 3 presents the design method. Finally, the conclusions are drawn in Section 4.
2. Literature Review

Over the last decade numerous approaches to web system design have emerged. Early approaches evolved from Entity-Relationship modeling in the Hypertext community including structured analysis and design in the information domain, and object-oriented analysis design. RMM provides a structured design methodology for hypermedia applications. Its focus is on modeling the underlying content, the user viewpoints on this content, and the navigational structures that interlink the content. OOHDM is a similar approach based on object-oriented software modeling, though somewhat richer in terms of the information representations. Other similar design methods include EORM [9] and work by Lee [10]. WSDM [11] attempts to take these approaches one step further, by beginning more explicitly from user requirements. In general, these techniques were either developed explicitly for modeling information in the context of the Web, or simply adapted from conventional software domain to the Web domain. More recently, work on WebML has begun to amalgamate these concepts into a rich modeling language for describing Web applications. The focus of WebML is very much on content modeling rather than describing the functionality that is a key element of most current commercial Web systems. One of the few approaches that attempt to integrate content representation with functionality is [12].

Research results have shown that in commercial web development, clients’ understanding of their needs evolve as a system evolves and thus design artefacts play a crucial role in the development of the clients’ understanding [1]. A typical web development process includes a prototyping phase. User interfaces (UI) are prototyped before the system design commences [1,4,13]. Agile development methodologies, which are very popular in the Web development community, emphasize the value of communication between the clients and the developers. Web system prototyping is commonly adopted as a vehicle for such communications. Web systems place a strong emphasis on user interface. Significant input is made into the “look and feel” of a web site. Usability is of paramount importance and there is considerable artistic input into page appearance. Fusebox [4,13] suggests that a user is not concerned with any part of the web system apart from the front end. What happens behind the front-end is of no direct concern to the users. The basic requirement for the back end is to support all the functionalities initially mocked up in the front end “prototype”.

The existing web design approaches such as RMM, OOHDM and WebML run counter to the emphasis on user interface and prototyping from the common web development processes. The user interfaces of their presentation design are typically derived from the design.

The method we are proposing here addresses this abnormally by placing system design after the user interface prototyping. The system design artifacts are derived from the user interface prototype. The new method is derived by imposing the design approach from Web Application Architecture Framework (WAAF) and its trace model [8,15] to a modified version of FlIP.

The FLiP process consists of a number of steps: Personas and goals; Wireframe; Prototype or Front-end development; Architecting; Fusecoding; Unit testing; Application Integration; Deployment. “Personas and goals” identify web application users and their goals. “Wireframe” models the proposed actions that will be performed by the application. A “prototype” of FLiP is a clickable model of the finished application with no backend behind it. In prototyping step, what the client expects from the application is revealed. Once the prototype is finished, the application architects construct the application design in the “Architecting” step. The architects identify fuseactions and organize them into circuits. Each fuseactions behavior is broken down into a set of fuses (code), the architects write a Fusedoc and a test harness for each fuse. In step “Fusecoding”, the coders write the fuses according to its Fusedoc. As each fuse is coded, it is unit tested against its test harness in step “Unit testing”. The architects integrate completed fuses into the final application in “Application Integration” step. The prototype is gradually transformed into a working application based on daily builds. The final product is deployed from the testing server or the staging server to the production server in step “Deployment” [4]. FLiP is platform dependent process and therefore is not a methodology for all types of web development. In addition, developers also need tools such as “DevNotes” for prototyping and “Mind Mapping” or “Fuseminder” for site construction, to support the development. Nevertheless, the most visible deficiency in FLiP is its lack of modeling methodology comparing with other common design methodologies. As software development continue to grow in complexity, and developers are used to work at high levels of abstraction to cope with this complexity, modeling software is – and will continue to be – a key aid to developers to work at high levels of abstraction [14]. Without an explicit modeling method, it will be difficult to trace artifacts throughout the development process.
The design method proposed here aims to enhance the modeling and architecting ability and design traceability to the commercial web design process. It focuses on the design phase of the web development life cycle since this a major point of departure from conventional software development. The methodology is platform-independent and does not require any particular tools. The process includes the following steps: Analysis; Prototype; Information Modeling; Architecting; Implementation and Deployment. The “Analysis” step is equivalent to “Personas and goals, and Wireframe” in FLiP. “Implementation and Deployment” corresponds to “Fusecoding, Unit testing, Application Integration and Deployment” in FLiP. The modification of the process from FLiP is in the following 3 steps: Prototype; Information Modeling and Architecting. In Web system design, “content is the King of the web” [1]. Accordingly, information modeling is a treated as a critical step in the design process. It could be iteratively applied until a satisfactory outcome has been achieved. The design method is discussed in detail in the following section.

### Table 1. Development process

<table>
<thead>
<tr>
<th>Process</th>
<th>FLiP process modified</th>
<th>Process</th>
</tr>
</thead>
<tbody>
<tr>
<td>Step</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Analysis</td>
<td>FLiP process</td>
<td>Analysis</td>
</tr>
<tr>
<td></td>
<td>FLiP process</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Prototyping;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Information modeling;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>System architecting;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Implementation and</td>
<td></td>
</tr>
<tr>
<td></td>
<td>deployment</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Fusion coding;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Unit testing;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Application</td>
<td></td>
</tr>
<tr>
<td></td>
<td>integration;</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Deployment</td>
<td></td>
</tr>
</tbody>
</table>

### 3. The Design Methodology

The key steps of the proposed development process are presented in Table 1 in contrast with the FLiP process. The focus of this design methodology is the design stage including 3 steps: prototyping; information modeling; and system architecting.

Although the development is an iterative process, the artifacts of the design are initially transformed from the analysis step. At “analysis” step, the analysts and the client explore the business process for the web application being built. Potential application users and their goals are identified. Tasks to achieve the goals are discovered. Tasks are clustered to scenarios through their semantic cohesions. The artifacts of this step include list and relationship of the user groups and their goals; list of the tasks and the description of their flows; list of the scenarios.

The artifacts of analysis step are then transformed into design activities. The design artifacts will be transformed into implementation step. The artifacts of each step are tailored from WAAF [15]. WAAF concerns of a web system into two dimensions. It established a two dimensional matrix. The horizontal dimension concerns the different perspectives of the participants in the web application development process. The perspectives include those from the viewpoints of planner, business owner, web system users, information architects, system architects, developers and testers. The vertical dimension classifies the architectures into four abstract categories, namely: structure (what); behavior (how); location (where); and pattern. Each cell in the framework is a model, a description, or an architecture as appropriate.

In practice, WAAF could be tailored to fit the needs of a specific web project without using all the defined cells. In this paper, the design method is developed by applying six cells of the WAAF matrix (user interface structure and behavior, information structure and behavior, system architecting structure and behavior) to a modified version of FLiP. Figure 1 illustrates the activities and associated artifacts in the 3 steps of the design process. Each step is discussed below.

![Figure 1. Activities and artifacts of design process](image-url)
3.1 Prototyping Step

A prototype is a full-scale front-end model of a web application. It is web users’ perspective. Prototyping step formulizes the front-end presentation of what need to be built. We consider the structure and behavior dimension of the front-end model at prototyping step. The structure cell and behavior cell of the User Interface Architecture in WAAF matrix are applied as two basic components for prototyping.

The behavior of the prototype models the user navigation path to access and operate the application. The navigation paths are derived from the business process scenarios described in the analysis step. The behavior can be modeled using user interface flow diagram [16] which considers the flow and state transition of user interfaces.

All the user interfaces on the navigation paths are collected in the prototype structure. The structure of the prototype consists of the collection of the user interfaces in the entire application and the layout and the content of each user interface.

To demonstrate the design method, we use examples of a commercial web application of an Australian company that specializes in matching investors to entrepreneurs seeking investment capital. For confidentiality reasons, we use a fictitious name for the company “XYZ-Match”. The prototyping process is illustrated using a “capital solicitation” scenario of XYZ-match. In this scenario, entrepreneurs submit business proposals and the XYZ-match approves the proposals. The whole prototype behavior of the application will include all the business process scenarios of XYZ-match described in the analysis step.

![Figure 2. User interface flow diagram](image)

The navigation paths of this scenario and the individual user interfaces to present these paths are identified and modeled in a user interface flow diagram in Figure 2. In UI flow diagram, each UI is represented using a box which only includes the name of the UI and exit points. The contents of each UI are presented in the prototype structure, not considered in the prototype behavior. Figure 3 shows the content on one page on the navigation path. The contents are what the user will see in the final product.

3.2 Information Modeling Step

Information is “the interpretation of data within a context set by a priori knowledge and the current environment” [17]. In information modeling step, information architects model the prototype into information architecture (IA). We consider two cells of WAAF Matrix information structure and information behavior as two basic components for information modeling.
Information architects classify and construct the structure and the relationship of the information in information structure. The information being modeled depends-on and satisfies the need in the UI contents in the prototype.

For example, the information needed in the capital solicitation scenario of XYZ-match can be categorized into user profile, entrepreneur profile, VC seeking criteria, publishing permission selection, etc. Under the category “user profile”, the information can be labeled as “user log in ID, user log in password, user name, user address, user email, user URL” etc. (See Figure 4).

**Information behavior** is modeled as information flow. Information architects derive the user interface flow diagrams into information flow diagrams to model the information creation, exchange, process and consumption that support the user interface flows. Information diagrams include information, information source and destination, information flow direction, and information processing unit. Figure 5 is a part of an information flow diagram of the capital solicitation scenario derived from the user interface flow in the prototyping behavior model.

### 3.3 System Architecting Step

Once the information model is finished, the system architects begin “System Architecting” to construct the web application design to modules and break down the modules into source code. Modeling **System Architecture Structure** and **System Architecture Behavior** are the major activities in this step.

In **System Architecture Structure**, system architects cluster the user interfaces into modules. If the modules are complex, break down the modules into submodules. Within each module and submodule, source code (or server pages) are included to realize the user interfaces in **User Interface Structure** and to generate the information (or content) in the **Information Structure**. **System Architecture Behavior** specifies workflow or business logic within the modules and submodules. It realizes each process in the information flow in the **information behavior** through source code flow (or server page flow). Each source code flow generates the user interface flow in the **user interface behavior** and the information flow in the **information behavior**.

In the system architecting of XYZ-Match application, the modules are clustered by user groups: investors, entrepreneurs, business executives and site administrator. User groups are identified in analysis step. The **system architecture structure** is as Figure 6. For example, the module “Investment opportunities” is broken down to submodules “investment opportunity listing”, “entrepreneurs submit business proposals” and “Business executives’ approval”. The submodule “entrepreneurs submit business proposals” includes a set of source code to generate the user interfaces and information that modeled in the prototyping and information modeling steps. The flow of this set of source code is modeled in the **system architecture behavior** to realize the user interface flow and the information flow for “entrepreneurs submit business proposals” scenario. The source code flow diagram is shown in Figure 7. Detailed algorithms within each source code are not considered in the system architecting step, they will be developed in the implementation step by coders.
The artifacts of the design steps, “user interface structure”; “user interface behavior”; “information structure”; “information behavior”; “system architecture structure” and “system architecture behavior” will be transformed into the implementation and deployment step.

4. Conclusions

We have proposed a web design method in this paper to address the evolving nature of user needs and strong emphasis of user interface of web-centric systems. The method is partially based on a widely used and effective development process - FLiP while removing its dependence on ColdFusion platform and the need for special support tools. The lack of abstract design modeling is overcome by applying a tailored version of Web Application Architecture Framework to the modified FLiP process. The new method focuses on 3 steps: prototyping, information modeling and system architecting. The design process commences from prototyping. Back-end design activities such as “information modeling” and “system architecting” support the front-end prototyping. Information modeling is made explicit by modeling information structure and information flow. The design method guides the activities through the artifacts modeling and transformation at each step. To cope with the system complexity, design activities at each step are partitioned into structural design and behavioral design. The traceability of the design artifacts is specified to improve the design completeness and coverage. The proposed design method has been used by one of the authors in developing commercial web projects. Future studies are needed to further examine the effects of this design method.
on Hypermedia Development Processes, Methods and Models (Hypertext'98), Pittsburgh, USA, 1998.


