Submitted to: *Requirements Engineering Journal*

Web System Requirements: An Overview

Associate Professor David Lowe

University of Technology, Sydney

P.O. Box 123

Broadway NSW 2007

Australia

Phone: +61 2 9514 2526

Email: david.lowe@uts.edu.au

Web System Requirements: An Overview

Abstract

The development of online and Web-based systems is becoming increasingly critical to the business strategy of many organisations. Although this development can be viewed as a form of software development, it has various unique characteristics. One of the most significant is the uncertainty in domain understanding by both clients and developers. Another is the way in which Web solutions typically lead to changes in business models and hence requirements. These are manifested as significant volatility in the requirements of these systems. In this survey paper we investigate the handling of requirements for Web systems including looking at both commercial practice and current research. We report on the outcomes of a comprehensive set of interviews and follow up surveys, and argue that the Web system requirements process needs to be fundamentally different from more conventional systems, incorporating the design process into the identification of requirements resulting in a design-driven requirements process.

Keywords

Web systems, online systems, requirements, volatility, domain uncertainty

Introduction

As user demand for the provision of online services grows, the ability to develop internet-enabled systems is becoming increasingly crucial [1, 2] to both business

success and to the provision of social and government services. Investment in these systems is growing at a phenomenal rate (see http://cyberatlas.internet.com/ for example statistics illustrating this growth). The systems being developed leverage the infrastructure of the Internet and an increasingly complex set of Web standards, protocols and technologies to provide sophisticated business solutions that merge Web-based front-ends with complex back-end software.

The nature of online and Web-based applications tends to be extremely diverse, covering applications such as information provision, e-commerce, collaborative and community building, communications, etc. Many (though certainly not all) Web-based applications share a common characteristic: they have a significant impact on the nature of the interaction that an organisation has with its clients and other external stakeholders (such as business partners, suppliers, etc.). This impact will typically lead to a change in both the fundamental business processes, and in the business model that relies on these processes. Whilst this is true of some non-Web systems (such as Customer Relationship Management systems), most “conventional” software systems can have significant internal impacts on the organisation, but do not substantially change the way in which the organisation interacts with its clients, and hence is unlikely to affect the core business model. In effect, we see Web-based systems as (typically, though admittedly not always) being an exemplar of those systems where the nature of the solution being developed fundamentally changes the problem being addressed and hence affects the requirements of the solution! In effect the problem and the solution are mutually constituted, as they are both inter-dependant. Later in the paper we revisit this issue, and describe the supporting development process as a design-driven requirements process.

In this paper when we discuss Web-based systems, we are predominantly referring to that class of systems where the system changes the nature of the external business interactions and hence the business model. This will encompass e-commerce applications, but also applications such as customer management, online inventory and supply-chain management, etc. Much of what is discussed will also be relevant to non-Web based applications that share this core characteristic.

Returning to the issue of the growth of these applications, and the obvious commercial impacts of failure to deliver effective solutions, we argue that little research has focused on understanding the processes for undertaking development of these web-based systems. Similarly, there has been a lack of attention on improving the cost effectiveness of these development processes. This is despite the fact that there are (as we shall discuss later) significant differences between Web systems and conventional software systems (discussed below).

As we shall consider in some length later in the paper, development practices from related domains do not typically address these differences particularly well. Irrespective of this there has been a tendency to adopt specific development practices from various constituent domains that contribute to Web projects (the choice often being dictated by the disciplinary background of the project management). For example, often Web development companies adapt approaches from software engineering (such as Rapid Application Development [36]), graphic design, or marketing [7], etc. - with only minor attempts to integrate all of these elements.

One of the key areas for which this often creates problems is the handling of requirements. Web projects often have a considerable degree of domain uncertainty

and requirements volatility. Both the rapid pace of technological change and the significant impacts of these systems on business models and processes lead to a lack of clarity from both developers and clients. The result is that clients not only have difficultly articulating their needs, but difficulty in even understanding what are their needs – particularly given that the introduction of these systems often leads to changes in business processes or models, and hence leads to subsequent changes in the requirements.

Conventional development processes do not handle this issue well. Typical requirements processes are largely designed to elicit requirements rather than to either support the development of domain understanding or to assist in understanding the impacts of a particular design. This is not a criticism, as this is not something that is intended to be the focus of requirements elicitation processes (and is often not a necessity in conventional development). It does however present difficulties when the solution under development affects the nature of the problem being addressed. Whilst work on domain understanding (particularly approaches such as Soft Systems Methodology [3] which attempt to support a systems analysis of processes in which technological processes and human activities are interdependent) assists in understanding the nature of the problem, it still does not clarify the impact of a solution on the problem domain.

Even incremental and iterative development processes that provide client feedback mechanisms have difficulty handling a situation where the requirements are simply not known, or are understood initially but subject to change as a result of a particular solution. This is largely because the feedback is predominantly focused on determining whether the emerging system meets the known requirements. They

also largely fail to manage the requirements volatility that results from an evolving client understanding of needs.

In this paper we consider these issues, and how requirements might be managed for Web projects. We begin by looking at existing research, including casting the net rather widely into the literature from various related disciplines. We then move on to consider current commercial practice, exploring the results of a wide-ranging set of industry interviews and surveys, as well as looking at commercial development guides and descriptions of best-practice. We discuss the view that there is a strong division between current research and commercial practice.

We conclude with two key points. The first is to propose the hypothesis that the design process can, and should, play a much greater role in managing the domain uncertainty and requirements volatility. The second is the claim that in considering requirements we need to develop a much clearer understanding of the relationship between business architectures, information architectures and technical architectures.

Web Requirements Literature

Although there is a significant volume of research on requirements management and requirements engineering approaches, little of this has filtered through into a specific consideration of requirements for Web projects. The small amount of research that does exist tends to come from a very diverse set of disciplines: software engineering; publishing; marketing; business information systems; etc. This tends to complicate the analysis of existing work, as these different discipline areas typically have distinct terminology, notations, and models.

As a result similar concepts are described quite differently in the literature belonging to different disciplines, making analysis of this area difficult.

One common theme is that the adoption of a structured, well-organised approach is important for ensuring a predictable, repeatable, manageable and cost-effective development – though this has emerged more clearly in industry publications than in the research literature (see, for example [4]).

A logical place to commence understanding the Web requirements process is to understand the differences between Web systems and more conventional software systems, particularly with respect to the impacts on the development process.

Unique Characteristics of Web Systems

There is a growing body of literature regarding the differences between Web systems and conventional software systems. In general, this literature identifies unique characteristics of internet-enabled systems that are both technical and organisational. For example, the technical structure of Web systems merges a sophisticated business architecture (that usually implies significant changes to the business model of the client) with both a complex information architecture and a highly component-based technical architecture [2].

The distributed nature of the internet, the visibility of Web systems to external stakeholders, the rapidly changing nature of the underlying technologies, and the lightweight component-based structure of most Web-systems, often means that the technology is much more visible to the users of the systems. This also means that the linkage between the business architecture and the technical design of the system

may be much tighter than for conventional software systems. Similarly, the information architecture, which covers aspects such as the content viewpoints, interface metaphors, and navigational structures, is substantially more sophisticated than conventional software systems.

More fundamental however are some of the organisational characteristics that are either unique or heightened in Web systems [5]. Two of these are uncertainty in the project domain and volatility of the client needs and available technology. With internet-based systems, the technology, development skills, business models, and competing systems are changing so rapidly that the domain is often not only poorly understood, but also constantly evolving [6]. This provides a substantial degree of uncertainty in the project context, and consequently makes resolving requirements very problematic and the requirements that do exist tend to be very volatile. Specifically, clients' understanding of not only the technology's capabilities but also their own needs typically changes dramatically during the course of a project. In particular, many web projects are vision-driven rather than needs-driven leading to an initial lack of clarity. This is also coupled with business models that are evolving rapidly as organisations migrate to an increased reliance on Internet technologies [1]. These issues are exacerbated by the fact that Web projects tend to have a short time-to-market, and are not currently well supported by development tools and techniques.

A number of additional specific differences have also been highlighted in the literature. These are discussed in more detail in [5, 7, 8]:

Short time frames for initial delivery: Web development projects often have delivery schedules that are much shorter than for conventional IT projects. This is

partly a consequence of the rapid pace of technological development, and partly related to the rapid uptake of Web systems.

Highly competitive: Web projects tend to be highly competitive. This is, of course, not new – being typical of the IT industry in general. The nature of the competitiveness is however somewhat different. There is regularly a perception that with simple Web authoring tools anyone can create an effective site. This creates inappropriate expectations from clients coupled with numerous small start-up companies claiming to be doing effective Web design but in reality offering little more than a combination of HTML skills and rudimentary graphic design. The result is a highly uninformed competitiveness.

Fine-grained evolution and maintenance: Web sites typically evolve in a much finer-grained manner than conventional IT applications. The ability to make changes that are immediately accessible to all users without their intervention means that the nature of the maintenance process changes. Rather than a conventional product maintenance / release cycle, we typically have an ongoing process of content updating, editorial changes, interface tuning, etc. The result is a much more organic evolution.

Increased emphasis on user interface: With conventional software systems users must make an (often considerable) investment in time and effort to install and learn to use an application. With web applications however, users can very quickly switch from one web site to another with minimal effort. As such the need to engage users and provide much more evident satisfaction of users needs and achievement of their objectives becomes critical. The result is an increased emphasis on the user interface and its associated functionality.

Increased importance of quality attributes: Web systems represent an increase in mission critical applications that are often directly accessible to external users and customers. Flaws in applications (be they usability, performance or robustness) are therefore much less able to be "hidden" and hence much more problematic.

Open modularised architectures: Although not unique to web applications, it is still worth mentioning the emphasis that is typically placed on open and modularised architectures for web systems. They are often constructed from multiple COTS (commercial off-the-shelf) components that are adapted and integrated together. Indeed, strong integration skills become much more critical in most Web projects.

Rapidly changing technologies: The technology that underpins most web systems is changing very rapidly. This has several consequences. The first is that it increases the importance of creating flexible solutions that can be updated and migrated to new technologies with minimal effort. For example, the need for reusable data formats (such as XML) increases substantially. A second consequence is that developer's understanding of these technologies is often restricted, increasing project risks.

Highly variable client understanding: It is extremely common for clients' understanding of the system capabilities to evolve and improve during a project. This typically means that the clients understanding of their own needs, and their expectations of the project, grow during the course of the project. This can exacerbate problems with requirements "creep" and make developing a product that satisfies user expectations extremely difficult. A related problem is the poor understanding that is often typical of clients early in a project, making initial system

definition difficult. This increases the importance of incremental and prototyping approaches.

Recognising that Web systems have some unique characteristics, a number of researchers have researched specific developmental methodologies. The methodologies have, however, largely focused on specific elements of the system; information modelling, functionality, business models, etc. We can begin by considering some of these approaches.

Information Modelling

One of the most obvious differences between Web systems and more conventional software systems (i.e. those where the system does not substantially change the way in which the organisation interacts with its clients) is the way in which information is managed and utilised. It is not surprising therefore that much of the earliest work on Web development techniques focused on information modelling and structuring.

Early approaches in this area evolved out of work on Entity-Relationship modelling – and applied this to modelling the information domain associated with applications. Indeed much of this work predates the Web and focused on hypermedia design. For example RMM (Relationship Management Methodology [9]) claims to provide a structured design methodology for hypermedia applications. In reality, the focus is very much on modelling the underlying content, the user viewpoints onto this content, and the navigational structures that interlink the content. OOHDM (Object-Oriented Hypermedia Design Model [10]) is a similar approach, though somewhat richer in terms of the information representations and based on

object-oriented software modelling approaches. Other similar examples include EORM [11] and work by Lee [12]. WSDM [13] attempts to take these approaches one step further, by beginning more explicitly from user requirements, but these are only addressed in a very rudimentary fashion. In general, these techniques were either developed explicitly for modelling information in the context of the Web, or have been adapted to this domain.

More recently, work on WebML (Web Modelling Language [14]) has begun to amalgamate these concepts into a rich modelling language for describing Web applications. However, despite its aim to support comprehensive descriptions, the focus (as with the above techniques) is very much on content modelling rather than describing the functionality that is a key element of most current commercial Web systems. One of the few approaches that attempts to integrate content representation with functionality is [15].

Further, these approaches generally provide reasonable modelling notations for representing the information architectures but rarely succeed in providing effective support for identifying the requirements that drive this architecture. The modelling approaches are largely design-focused. An exception to this is some of the work extending OOHDM, to include tools such as user scenarios and use cases which look at how a system is likely to be used [16] – and develops content models that support this usage. Another exception is work that focuses on supporting maintenance processes – such as [17].

There has been some limited work on specific approaches to formally representing certain aspects of Web applications – though this has tended to focus again on content and navigational issues to the exclusion of functionality. For

example, Hadez [18, 19] looks at the use of formal methods (using the Z notation) to specify conceptual, structural and perspective schemas. Other approaches have focused on specification of timing constraints [20, 21] rather than content structure. Again, however, the focus is very narrow and fails to couple the specifications with broader application requirements.

Software Engineering Methods

Whereas information modelling approaches focus on content modelling, software engineering approaches tend to focus on understanding required functionality and how systems to support this functionality can be designed. As with information management techniques, a number of researchers have looked at the adaptation of conventional software engineering to support Web development.

Much of this work has looked at the extension of modelling languages to support Web concepts – most notably work on adapting UML (Unified Modelling Language – an emerging industry standard for modelling Object-Oriented systems). Often this work simply uses the notation to effectively model information structures (such as [22]) and doesn't integrate this with the modelling of website functionality. One exception is the work by Conallen [23] which attempts to look at the link between content and functionality – in particular, how the software components generate content and are triggered by navigational events. Again, the approach is restricted though. The modelling is not particularly scalable, nor are effective abstractions provided for modelling the content (the content modelling is essentially restricted to representing individual Web pages). Even more problematically, the

modelling allows representation of low-level designs, but does not provide any indication of how this design is developed nor its link to client requirements.

Other researchers have used UML in rather different ways. For example, Vilain et al have adapted UML to representing user interactions [24]. Although this work is not explicitly addressing Web requirements or design, user interaction is a key element of Web systems. Indeed work such as this is moving closer to providing tools for understanding Client requirements. The authors of this work see it as supporting use cases – a UML tool aimed at understanding how a system will be utilised. Other examples of work that has adapted UML include [2, 25, 28]. In effect, the approaches based on UML provide a useful notation, but still must be integrated into a broader framework.

The above work has emphasised relatively detailed modelling of designs. An alternative thread of research has looked at modelling Web systems at a more abstract level. In particular, there is a growing body of work – largely emerging from large technology vendors such as IBM, Sun and Microsoft – that considers supported business functions and the broad technical architectures required to support these. Probably the most mature of these approaches is the patterns for e-Business work being developed by IBM (see http://www.ibm.com/framework/patterns/). This work provides a framework for identifying common patterns of business models. As stated in [25]:

> "The paths to creating e-businesses are repeatable. Many companies assume that they are unique and that therefore every creation of an e-business has to be learned as you go. … in fact, there are lessons and architectural paths or patterns that can be discerned from all these

engagements. Whether your company is a startup or has extensive legacy applications, these patterns allow you to reuse existing technologies so that your projects can be completed quickly."

For each business pattern, a number of logical architectures (or topologies) are defined. These topologies provide a mechanism for fulfilling a particular business need. In terms of requirements, the business patterns can help clients understand potential applications and the associated functionality. The approach does not, however, provide an explicit process for supporting the development of this understanding. Another problem is that the architectures tend to emphasise functionality, with little consideration of the information architecture. In particular aspects such as content modelling, information viewpoints, etc. are not addressed.

All of the above techniques support the design of Web systems – albeit at significantly varying levels of abstraction. They do not provide supporting processes for handling the requirements. Conventional software processes tend to assume that requirements are known to clients, and simply need to be elicited and analysed. These processes usually differentiate between user requirements (often formalised in a URD – or User Requirements Definition) that capture the user understanding of their needs, and the system specification (SRS – or system requirements specification) that represents the system that will meet these needs [26]. In effect the two documents are different representations of the same concepts. A typical process will be to elicit requirements (which are documented in the URD), and then analyse these requirements to construct the SRS, iterating to refine the URD as necessary.

One significant difficulty with this paradigm is that it presumes that clients either understand their requirements, or at the very least understands the problem that is being addressed. Even when the client is not able to articulate their requirements precisely, they are at least able to understand whether a given design will address their needs. In cases such as these, the design may commence prior to full resolution of requirements. The design will then be used to ascertain (from client feedback) whether the proposed solution addresses the identified need. This is problematic for Web projects, where many clients not only have a poor understanding of their requirements, but they also have a poor understanding of the problems being addressed by the new system. In these circumstances, simply using a design to clarify whether it addresses the problem will be insufficient, as the problem itself is only poorly understood.

As an illustrative example of these issues, consider the increasing use of lightweight development processes for software projects [27-29]. One of the approaches receiving the most attention is the use of XP (eXtreme Programming) [30]. XP is based on the incremental development of partial (or “spike”) solutions that are used to subsequently resolve specific requirements. These spike solutions are then integrated into the evolving system through refactoring of the current solution to incorporate these components. When used in conventional software development XP has proven to be effective for projects that are initially ill-defined [30] – a characteristic of many web projects. As a result, many of the proponents of XP and similar approaches see it is an ideal approach to be adopted for Web development [31].

There are however certain problems that restrict approaches such as these to Web projects. The first is that a number of studies (see, for example, [32, 33]) that have shown that approaches such as XP only work effectively for projects that have cohesive development teams. This is often not the case with Web projects, which often lack cohesiveness between the technical development and the creative design as a result of the disparate disciplinary backgrounds of the development team members. XP can also result in a brittle architecture and poor documentation, which makes ongoing evolution of the system difficult – something that is important for Web systems. Finally, and perhaps most fundamentally, XP utilises partial solutions to resolve uncertainty in requirements, but does not inherently handle subsequent changes in these requirements (i.e. requirements volatility) as the system evolves. This creates problems for web systems, where the emerging design results in an evolving client understanding of their needs, and hence volatile requirements [34].

In effect, conventional software engineering processes see requirements as preceding and driving the design process. Even where an iterative or incremental approach (such as XP) or a spiral approach (involving multiple feedback loops) is adopted the design is viewed as a way of assisting in the identification and validation of requirements, but rarely does it help the client to actually formulate their needs. In Web development, the situation is fundamentally different. The design process not only helps developers and clients articulate the needs, but also helps clients understand the system domain and formulate their understanding of their needs. In effect, the design allows developers to manage the inherent uncertainty and volatility.

There has been significant work on considering requirements volatility within conventional requirements engineering (see, for example [35]) and especially on

tailorable systems where requirements change leads to the need to evolve systems (see [36]). This work whilst useful, does not explicitly consider the ways in which design itself changes the nature of the requirements and hence leads to a closed loop in the development.

Approaches that focus on domain understanding have certain merit in the context of the changing requirements of Web-based systems. For example, Soft Systems Methodology [3, 37] attempts to support a systems analysis in which technological processes and human activities are interdependent. This however still emphasises design of systems (albeit composite technological / human systems) but doesn't explicitly address the issue of the way in which the system will ultimately change the nature of the business problem.

One attempt to integrate support for Web projects directly into the development process is work on WebOPEN. OPEN (Object-oriented Process, Environment, and Notation) is a third-generation, public domain, approach that was designed for the development of software intensive applications, particularly object-oriented and component-based development [38, 39]. The OPEN process framework (OPF) supports the instantiation of specific development processes from a set of work units (activities, tasks and techniques), work products and producers. The work on WebOPEN extends OPEN to include additional tasks [40] and roles and techniques [41] that are applicable to Web projects. Although a significant step forward, this work is limited insofar as it does not inherently provide guidance in structuring the process to deal with the requirements volatility. This remains an open issue within the software engineering literature.

Further Afield

A number of related areas can potentially provide some interesting insights into the specification and design of web systems. These include areas as disparate as publishing, marketing, architecture and gardening.

Beginning with publishing, we can investigate aspects such as publishing guidelines and what these say about the development (i.e. authoring) focus. Whereas software processes emphasise an understanding of (and subsequent addressing of) required functionality, publishing looks more at characterising audiences and their reaction to the material being authored. For example, the following fragments, describing what should be included in book proposal, are extracted from a number of publishing guidelines made available by numerous publishers:

A one-page overview of the book	*A chapter-by-chapter outline*
Author's background/credentials	*The audience you wish to reach*
Market/audience information	*A list of competing works*
Who is the audience	*Competing books and how this differs*
How does the book help the reader	*Expected delivery date of manuscript*

If we replace book by site, reader by user, and author by developer then most (if not all) of the above would be directly relevant to understanding the focus of a web site. It is interesting to note that at the proposal stage there is almost universal absence of any consideration of aspects such as detailed content, stylistic considerations, and production processes.

Areas such as marketing and advertising are also likely to be able to provide useful insights – focusing on designing explicitly for user engagement, conveying a desired message, and managing user or consumer reactions.

One unusual area that has been used as an analogy for web development is landscape gardening. Website development is often about creating an infrastructure (laying out the garden) and then 'tending' the information that grows and blooms within this garden. Over time the garden (i.e. Website) will continue to evolve, change, and grow. A good initial architecture should allow this growth to occur in a controlled and consistent manner. This analogy has been discussed in terms of providing insights into how a site might be managed [42].

Current Commercial Practice

The above discussions indicate that the research literature is extremely fragmented, with few attempts to draw the work together into a cohesive picture. Nevertheless, Web development is carried out – and carried out successfully in at least some cases. It is therefore worthwhile looking at what actually occurs in commercial practice. We begin by looking at the results of a broadly focused set of industry interviews and follow-up surveys.

Surveys and Interviews

In order to develop a strong understanding of commercial practice, and in particular to investigate the ways in which commercial practice in Web-based projects differed from commercial practice in more conventional software systems development (particularly as it relates to management of requirements) we

undertook a comprehensive set of industry interviews and follow up surveys. These focused on the issues that are raised during the early stages of Web development projects, and the characteristics of systems that are resolved during these stages. Detailed results of these surveys have been published elsewhere [43-46]. We shall provide an overview of the results here.

Background. A significant volume of data was collected in the form of both interviews and surveys. The interviews were intended to identify general perceptions and qualitative trends and were conducted primarily over a period of 6 weeks during April/May 2000. The interviews were typically 20-40 minutes and involved a series of questions focusing on interactions with clients and the processes for understanding client needs. The surveys were intended to capture more quantitative information. The survey was either completed as a follow–up to the interview or as a stand-alone survey. The survey data was collected during the period April-June 2000. The survey consisted of fifteen sets of detailed questions ranging from company profiling, project profiling, client relationships, and development practices.

The respondent companies that took part in the interviews and surveys covered a broad spectrum of development areas: multimedia development, Internet development, and intranet development. Similarly, their core business varied considerably, from consulting to contract development to in-house development. The application domains also covered a broad spectrum: financial institutions, medical practitioners, general SME's, top 500 corporations, finance, travel and tourism, legal, manufacturing, government, and consumer advisory services.

Cost, complexity and scale of projects. The projects of individual companies varied greatly in complexity and cost (a reflection of the range of scale of applications

being developed for the Web). In terms of cost the range was from $2000 to $50million with the average company working on contracts ranging from $100,000 to $1 million. In correlating cost to other measures or project scale, various factors had poor correlation. In particular, both number of source documents and number of resultant site pages were very poorly correlated to overall cost. The frequency of content changes had a surprisingly high correlation.

Numerous respondents felt that hidden costs were a major issue in projects (average of 3.4 on a scale of 0=strongly disagree to 5=strongly agree), and that clients have difficulty understanding the implications or details of project bids (average of 3.4)

The duration of projects also varied considerably - from 2-3 days on small projects, 2-4 months on medium sized projects and 9-12 months on large-scale projects. The average expected lifetime of systems was marginally under 18 months, with few projects expected to have a lifetime of over 24 months. Most respondents felt that projects were generally completed on time – though a number of factors impacted on this. In particular, the elements which most affected the ability to adhere to a schedule were: the existence of a detailed risk analysis; good project management; staff stability; clients meeting milestones; scope creep.

There was a high level of outsourcing amongst most of the respondents. This extended to aspects such as graphic design, usability evaluation, functionality testing, scripting or application development and development of specific functional areas (database development and administration, server management, etc.). Outsourcing supported the gaps in respondent companies' areas of expertise and

tended to be the areas where contractual staff and strategic alliances were called upon.

Client/developer interaction. Many of the respondents' companies aim towards having only one point of client contact (56%) and using a designated project leader rather than having multiple points of contact (44%). Almost all of these companies also stressed the importance of teamwork. The one client liaison seemed to be important though the respondents who opted for several leaders or team focusing had strong team processes in place.

There was a very strong feeling (average rating 4.4 on a 0-5 scale) that clients did not understand well the capabilities of the technologies. Similarly it was felt (average rating 4.2) that clients did not understand their own needs. Anecdotal comments indicate that clients are also unaware of aspects such as the workload involved in developing internet sites and the implications on their companies. If the clients' liaison was from their marketing team then the IT side was often poorly comprehended (and hence articulated – and therefore specified). If the clients' liaison was from the organisations' IT team then the marketing or business side was not well understood.

Perhaps surprisingly, respondents felt that clients had a low understanding of their own organisations and existing processes (most of the time undocumented) that need to be changed to allow for the effective integration of the new system. Other concerns were the implications of an incorrect specification and that the consequences of the fact that the client is often responsible for content provision. There was a consensus from the majority (76%) of the respondents that there

needed to be a process at the beginning of the projects focused on educating their clients.

Identifying requirements. Almost all respondents interviewed clients as part of the process for identifying requirements. A much smaller number felt that interviewing potential users was important. The majority of respondents (84%) found that the look and feel and content issues were secondary to the business case. 11% of the remaining 16% of respondents are primarily involved in the front end of multimedia development and either contract or are contracted to complement backend work. Perhaps surprisingly, critical success factors (i.e. acceptance criteria or essential requirements) were brought up only by 5% of respondents as a vehicle for capturing the business case.

The majority of respondents (76%) indicated the importance of getting the requirements and specification correct – though there was considerable divergence of opinion as to when this should occur. Some other important points that contribute to the success of a Web project are senior executives and management commitment to the project, early user analysis and usability evaluation, and recognising the concomitant changes to the organisations workflows and business practices.

The majority of respondents (84%) felt that there was a major shift in the level of awareness from the beginning of a project to the end. The extent to which this was considered a problem was largely correlated to the scope of the project. For small projects, the project duration and budget left little room for changes in client expectations and hence requirements during the project. It was viewed by a number of respondents that it was therefore critical that clients for small projects be educated as early as possible in the development cycle.

There was reasonable consistency in views about the elements that should be captured in specifications. Key elements that reoccurred consistently included: current systems; vision for the project; business strategies; business drivers; business case; budget; stakeholders.

Development processes. Most of the respondents attempt to capture requirements before they sign final contracts. It was also recognised that initial tendering often occurs before this point – leading to a two-step contract negotiation. Some clients were seen to be happy to commit to a budget (during the tender process) based on broad business objectives, and then finalise the contract at a later stage based on specific analysis of the detailed requirements. The scope of the contracted requirements is typically constrained by retaining a focus on the business case and establishing that there is good basis for specific detailed requirements.

The majority of respondents (84%) had a standard pro-forma for documenting requirements and contracts, and the majority (76%) believed that they adhered to it consistently. The majority of respondents (68%) carried out formal evaluation, though this was primarily on the final product – and typically against the specification. Several respondents indicated that due to the evolution of the specifications (which were typically not maintained in a coherent documented form) direct testing against the specification was not feasible.

All the respondents felt that they carried out some level of maintenance of the project files – though the elements that were considered important to maintain varied considerably (for example, often the design prototypes were considered to be a better level at which to maintain the documentation of requirements than the original specifications).

All the respondents had an initial signoff on either a project brief or proposal. Only one respondent pointed out that an initial signoff was difficult when they were working off a concept. There seemed to be two distinct methods of specifying. The first was the traditional software specification methods – essentially:

a. Requirements analysis

b. Functional specification

c. Technical Specifications

d. Testing specification

Or alternatively:

a. Initial brief

b. Design documents

c. Content information

d. Technical specifications (including testing).

It was recognised by essentially all respondents that client needs will change and evolve over the life of a project, despite a well-written requirements specification document. Most respondents had some form of change management process, though few companies had formal tool-level support for change or configuration management of applications. Most respondents document all changes whether minor or major in formal proceedings mainly to justify their reasons for working out of scope.

Of the respondents that didn't indicate having in place formal procedures there seemed to be a feeling that business cases change during the development phase and that to support new media development these changes need to be allowed rather than a fixed price paradigm be set and adhered to.

Primary divergences of views. There were a number of areas where there was divergence of views. In particular, there were significant differences in the levels of agreement with the following statements (from the survey):

General Relationship With Client

We interview the clients to determine requirements

We interview intended users to determine requirements

It is important to respond to changes in user requirements as they occur

Changes in the user requirements require the site to be renegotiated.

Clients needs evolve considerably during the lifetime of a project

Development Processes

It is important to identify technologies to use as soon as possible

It is important to be able to modify the system once it is completed

Analysis of Specification Documentation

Based on the primary data collected and analysed during the interviews and surveys, we were able to gain a broad picture of the approaches taken to resolving client needs in Web development projects. In order to focus this onto more specific detail, particularly with regard to those elements that tend to be identified as important to specifying a system, we followed up the interviews and surveys by analysing a significant number of commercial Web specifications.

These specifications covered a broad range: from initial expressions of interest to responses for requests for tender and to full contractual specifications. They also varied from specifications for completely new systems to specifications for redevelopment or modifications to existing systems.

It was interesting to note that there were both significant areas of overlap in the aspects considered in these documents, as well as significant areas of divergence. The primary areas of divergence were as follows:

Consideration of the underpinning business model. The specifications varied widely in the extent to which they actively identified and documented the business model that drove the development of the system. Initial indications are that those that had a clear business model were usually much more cohesive, though this requires further investigation for confirmation and quantification.

More interestingly, the results indicate that commercial practice has tended to emphasise business modelling as a driver for structuring requirements, coupled with the utilisation of designs in assisting clients to develop an understanding of the problem and potential solutions, and especially the way in which proposed solutions may lead to consequent changes in business practices or models (and hence requirements). We would argue that this stronger emphasis on business modelling is, in effect, a tacit acknowledgement by developers of the impacts that the solution has on the nature of the problem, and the consequent need to therefore understand at least the shape of the solution space when clarifying the problem.

Extent of emphasis on site look and feel. The stage at which look and feel issues (both navigational, and graphic design) are introduced varied considerably. Some specifications focused on this very early in the process, to the detriment of other aspects.

Consideration of changes to organisational work flows. There was significant variation in the way in which specifications addressed potential impacts

outside the immediate technical issues related to the site. These were most noticeable with regard to impacts on organisational processed.

More interesting than the areas of divergence were the areas where the specifications had a degree of consistency, but where this was at odds with practice in other domains, especially conventional software development. The most noticeable of these areas is the extent to which strict user and client needs are mixed with design information. This is particularly noticeable once an initial expression of client needs evolves into a full specification. These full specifications almost uniformly mixed design information with client requirements. Indeed the specifications often used designs as a way of representing requirements, bypassing a formal requirements specification.

Organisation Specific Methodologies and Best Practice Guidelines

Finally, we considered process documentation from various companies. This includes aspects such as descriptions of methodologies, best-practice guidelines, development standards, etc.

The dominant trend was towards the utilisation of documentation, not to support development processes, but as a marketing tool to illustrate claimed capabilities. In numerous cases this documentation was relatively superficial, and in those cases that were investigated in more detail, often did not map to more detailed process documentation. This is not to say that these organisations did not adhere to the processes described, rather that it was often not documented in any substantial detail. This is an area that requires further investigation.

Analysis

Bringing all the above information together, we can draw some interesting conclusions. The first, and most obvious, is that there is a significant gap between current research and commercial practice. It is unclear whether this is a consequence of a lack of awareness of research outcomes by practitioners, or research outcomes that are not yet suited to practical development. Indications are, however, that the core focus of much commercial work does not map well to research emphases.

For example, research has tended to emphasise detailed design issues – both in terms of content and functionality, albeit using discrete approaches that are not well linked – and has barely considered approaches to understanding clients' needs and how they are formulated and represented. Conversely, commercial practice places significantly more emphasis on understanding client needs, and in particular how business models and processes are impacted and how they are linked to system architectures (both informational and functional).

The second major observation is with regard to the role of the design process. Conventional requirements engineering and management processes see the requirements as preceding and driving the design process. Even where an iterative or incremental approach (such as XP), or a spiral or prototyping approach (involving multiple feedback loops) is adopted the design is viewed as a way of assisting in the elicitation, clarification, and validation of requirements. In some cases the client understands very well their needs and these approaches are aimed at assisting the developer to identify and formalise these needs. In other cases the client has trouble

formulating or understanding their needs and the design process is aimed at facilitating an understanding of the needs.

With Web development, the design process will support the above role, but can also play a very different role. The design process not only helps developers and clients articulate the needs, but also helps clients understand the system domain and how it changes with the introduction of a new system. The result is that the design process can assist clients to formulate needs that are interdependent on a given design [34]. In effect, the design allows developers to manage the domain uncertainty and volatility that is a consequence of the introduction of different designs.

A useful comparison at this point is with the approach that is often referred to as Ready-Fire-Aim [47]. This essentially is referring to approaches where the design is commenced prior to a full understanding of the requirements (or coding commenced prior to a full design, depending on the interpretation) as a way of informing clients in the presence of uncertainty. In contrast, commercial Web development is typically about developing prototype solutions as a way not of resolving uncertainty, but rather to understand the impact of a given solution. This is a little bit like saying “Well, if we fire there, then it will have this impact, but if we fire there it will have that impact”. i.e. Possible solutions are jointly investigated by the developer and client (typically, through a design prototyping approach – but *prior* to committing to a specific solution) in terms of their impact on the problem domain and hence the requirements, with the ultimate result that a solution is identified that matches a problem that has been changed by that solution.

Although this concept is well accepted within industry, very little consideration has been given to it within the research literature. Research has been limited to empirical work using scenario-based redesign of partially developed sites [48] though this work has at least recognised the importance of designs in assisting clarification of client needs:

> "We practice a revised method of scenario-based design inferred from a theoretical perspective which treats design as inquiry, inquiry as dialogue and dialogue as the source of all tools, including mental constructs. The result is a set of techniques for using structured dialogue between users and designers to increase designers' understanding of specific domains of users' work."

In commercial Web projects, these concepts (and particularly the mutial interdependence of requirements and design) are typically reflected in the absence of separate requirements and design documents. Rather, developers tend to create a hybrid specification that blends design and requirements (something that is usually viewed as anathema in conventional software engineering).

In other words, system design allows stakeholders to understand technical possibilities and limitations, and hence improve their understanding of the development context. The result is a vehicle for reducing the underlying uncertainty. For this to be effective, however, we need to develop a suitable model of the relationship between system design, client requirements, and uncertainty within these requirements. This uncertainty model can then be used to adapt the requirements engineering process – resulting in a design-driven requirements process. This is the focus of our ongoing research.

Finally, there are a number of minor issues that are highlighted by either commercial practice or current research:

It is important to establish a client liaison who understands both the IT/Technical aspects and the business domain. Although such a person may not be in a position within the client's organisation that allows them to readily interact with developers or contractors, there is much to be gained from encouraging the client to bring such a person into the liaison, particularly in terms of reducing the uncertainty discussed above.

Establish effective client education mechanisms. A major problem in many projects appears to be the lack of understanding by the client. This extends from an understanding of the capabilities of the technologies to an understanding of their own business processes and how these should possibly be modified with the introduction of the new Web system(s). It is important that clients accept the need to develop a clearer understanding, and be willing to work with the developers in building up this knowledge.

Clarify business cases as early in the process as possible. The business case forms the foundations on which the requirements, specifications and subsequent development are grounded. Conventional development often is based on an implicit understanding of possible business cases – something that is not true in Web development projects. Without a clear understanding of the clients' business case a project can waste considerable effort going around in circles – proposing designs that subsequently require considerable change.

Adopt early evaluation: A primary characteristic of many Web projects is the lack of client understanding with regard to desired attributes of the system being developed. A key way to handle such uncertainty is to adopt early evaluation of potential designs. This can be through usability evaluations, focus groups, etc.

Conclusions and Further Work

In this paper we have analysed the current situation with regard to the management of requirements for Web projects. In particular, we have contrasted the research work documented in the literature with existing commercial practices. There is a significant discrepancy between the foci in these areas. Research has tended to focus on design methods, but largely without considering how these design processes contribute to a domain understanding and clarification of the requirements.

A key paragraph from the analysis is worth repeating, given the crucial significance that it has. Commercial practice has tended to emphasise business modelling as a driver for structuring requirements, coupled with the utilisation of designs in assisting clients to develop an understanding of the problem and potential solutions, and especially the way in which proposed solutions may lead to consequent changes in business practices or models (and hence requirements).

This has many superficial similarities to prototyping, insofar as both approaches use partial solutions to improve understanding. Indeed, Web development makes heavy use of prototypes. However, whereas conventional prototyping is largely focused on assisting developers to understand the clients' requirements, web development uses prototypes to inform clients of the implications of potential

solutions on their business processes and the subsequent possible changes to their requirements. We can view the problem domain and solution domain as being mutually constituted or interdependent, and neither can be resolved in the absence of the other.

Considerable work still remains to be carried out in this area. There are two key areas that we are investigating. The first is the development of a design-driven requirements process that structures the way in which design activities can be linked to the clarification of requirements through an appropriate model of domain uncertainty.

The second is an improved linkage between the architectural elements of a Web system. In particular, given the evolving role of designs, it is important that the business architecture be able to be linked to both the information architecture and the technical architecture of a system. Indeed a fruitful area for further investigation is to look at how the informational and functional aspects of the architecture can be coupled appropriately during the design. This is particularly important in terms of understanding how changes to the technical architecture affect the nature of the business architecture. We are just beginning work that address this by integrating concepts from pattern theory, Jackson's problem frames, and Alexander's work on architecture.

One element driving both these research areas is our ongoing refinement of a characterisation model that captures the various elements that emerge in the specification and design of Web sites. This model provides a basis for understanding which elements should be clarified at which point in the process, and the linkages between these elements.

Acknowledgements

The authors wish to acknowledge the collaborative funding support from the Australian Research Council, Access Online Pty Ltd and Allette Systems Ltd. Under grant no. C4991-7612. In particular we wish to thank John Eklund, Ross Jeffery, Louise Scott, Lucila Carvalho, and John D'Ambra for their contributions to this research project, and Brian Henderson-Sellers and Brendan Haire, for the related discussions on WebOPEN and the differences between Web development and conventional software development.

We also wish to acknowledge the valuable contributions of the numerous companies and individuals who participated in the industry interviews and surveys.

Finally, I would like to acknowledge the valuable feedback, advice, and suggestions of the reviewers of this paper.

References

[1] L. D. Stein, "Profit, the Prime Directive," *WebTechniques*, vol. 5, pp. 14-17, 2000.

[2] P. Russell, "Infrastructure - Make or Break your E-Business," presented at TOOLS-Pacific 2000: Technology of Object-Oriented Languages and Systems, Sydney, Australia, 2000.

[3] P. Checkland and J. Scholes, *Soft systems methodology in action*. Toronto: Wiley, 1990.

[4] L. Gates, "Analysis and Design: Critical yet Complicated," in *Application Development Trends*, vol. February 2001, 2001, pp. 40-42.

[5] J. Burdman, *Collaborative Web Development*: Addison-Wesley, 1999.

[6] G. Sinha, "Build a Component Architecture for E-Commerce," *e-Business Advisor*, vol. March, 1999.

[7] E. England and A. Finney, *Managing Multimedia: Project Management for Interactive Media*, 2nd ed: Addison-Wesley, 1999.

[8] S. Overmyer, "What's Different about Requirements Engineering for Web Sites?," *Requirements Engineerng Journal*, vol. 5, pp. 62-65, 2000.

[9] T. Isakowitz, E. Stohr, and P. Balasubramanian, "RMM: A Methodology for Structured Hypermedia Design," *Communications of the ACM*, vol. 38, pp. 34-44, 1995.

[10] D. Schwabe and G. Rossi, "The Object-Oriented Hypermedia Design Model," *Communications of the ACM*, vol. 38, pp. 45-46, 1995.

[11] D. Lange, "An Object-Oriented Design Method for Hypermedia Information Systems," presented at HICSS-27: Proc of the Twenty Seventh Hawaii International Conference on System Sciences, Maui, Hawaii, 1994.

[12] S. C. Lee, "A Structured Navagation Design Method For Intranets," presented at Third Americas Conference on Information Systems, Association for Information Systems (AIS), Indianapolis, 1997.

[13] O. De Troyer and C. Leune, "WSDM: A user-centered design method for Web sites," presented at 7th International World Wide Web Conference, Brisbane, Aust, 1997.

[14] S. Ceri, P. Fraternali, and A. Bongio, "Web Modeling Language (WebML): a modeling language for designing Web sites," presented at Proceedings of WWW9 Conference, Amsterdam, 2000.

[15] K. Takahashi and E. Liang, "Analysis and Design of Web-based Information Systems," presented at 7th International World Wide Web Conference, Brisbane, Aust, 1997.

[16] N. Guell, D. Schwabe, and P. Vilain, "Modeling Interactions and Navigation in Web Applications," presented at World Wild Web and Conceptual Modeling'00 Workshop - ER'00 Conference, Salt Lake City, USA, 2000.

[17] H.-W. Gellersen, R. Wicke, and M. Gaedke, "WebComposition: An Object-Oriented Support System for the Web Engineering Life Cycle," *Computer Networks and ISDN Systems*, vol. 29, pp. 8-13, 1997.

[18] D. M. German, "Hadez, a Framework for the Specification and Verification of Hypermedia Applications," University of Waterloo, 2000.

[19] D. M. German and D. D. Cowan, "Formalizing the specification of Web applications," *Lecture Notes in Computer Science, Springer Verlag*, vol. 1727, pp. 281–292, 1999.

[20] J. P. Courtiat, M. Diaz, R. C. D. Oliveira, and P. Senac, "Formal Methods for the description of timed behaviors of multimedia and hyper-media distributed systems," *Computer Communications*, vol. 19, pp. 1134–1150, 1996.

[21] F. B. Paulo, M. Augusto, S. Turine, M. C. F. d. Oliveira, and P. C. Masiero, "XHMBS: A formal model to support hypermedia specification," presented at Ninth ACM Conference on Hypertext, 1998.

[22] H. Baumeister, N. Koch, and L. Mandel, "Towards a UML Extension for Hypermedia Design," presented at <<UML>> 1999: The Second International Conference on The Unified Modeling Language, Fort Collins, Colorado, USA, 1999.

[23] J. Conallen, *Building Web Applications with UML*: Addison-Wesley, 1999.

[24] P. Vilain, D. Schwabe, and C. S. d. Souza, "A Diagrammatic Tool for Representing User Interaction in UML," presented at <<UML>>2000: The Third International Conference on The Unified Modeling Language, York, U.K., 2000.

[25] J. Lord, "Patterns for e-business: Lessons learned from building successful e-business applications," IBM, 2000, pp. 4.

[26] I. Sommerville and G. Kotonya, *Requirements Engineering : Processes and Techniques*: John Wiley and Sons, 1998.

[27] E. Angelique, "A Lightweight Development Process for Implementing Business Functions on the Web," presented at WebNet'99, Honolulu, Hawaii, USA, 1999.

[28] R. Fournier, *Methodology for Client/Server and Web Application Development*: Yourdon Press, 1999.

[29] M. Haggard, *Survival Guide to Web Site Development*: Microsoft Press, 1998.

[30] K. Beck, *Extreme Programming Explained*: Addison-Wesley, 1999.

[31] D. Thomas, "Managing Software Development in Web Time Software," presented at XP2000, Cagliari, Italy, 2000.

[32] R. Martin, "A Case study of XP practices at work," presented at XP2000, Cagliari, Italy, 2000.

[33] J. Siddiqi, *eXtreme Programming: Pros and Cons*: IEEE Computer Society, 2000.

[34] D. Lowe, "A Framework for Defining Acceptance Criteria for Web Development Projects," presented at Second ICSE Workshop on Web Engineering, Limerick, Ireland, 2000.

[35] D. Zowghi, R. Offen, and R. Nurmuliani, "The Impact of Requirements Volatility on the Software Development Lifecycle," presented at International Conference on Software, Theory and Practice (ICS2000), China, 2000.

[36] D. Stamoulis, P. Kanellis, and D. Martakos, "Tailorable Information Systems: Resolving the Deadlock of Changing User Requirements," *Journal of Applied Systems Studies*, vol. 2, 2001.

[37] P. Checkland, *Systems Thinking, Systems Practice*: John Wiley & Sons, 1981.

[38] I. Graham, B. Henderson-Sellers, and H. Younessi, *The OPEN Process Specification*. Harlow, U.K.: Addison-Wesley, 1997.

[39] B. Henderson-Sellers, A. Simons, and H. Younessi, *The OPEN Toolbox of Techniques*. UK: Addison-Wesley, 1998.

[40] B. Haire, B. Henderson-Sellers, and D. Lowe, "Supporting web development in the OPEN process: additional tasks," presented at COMPSAC'2001: International Computer Software and Applications Conference, Chicago, Illinois, USA, 2001.

[41] B. Haire, D. Lowe, and B. Henderson-Sellers, "Supporting web development in the OPEN process: additional roles and techniques," presented at OOIS, Montpellier, France, 2002 (Accepted).

[42] D. Lowe, "Web Engineering or Web Gardening?," in *WebNet Journal: Internet Technologies, Applications and Issues*, vol. 2, 2000.

[43] D. Lowe and B. Henderson-Sellers, "Impacts on the development process of differences between web systems and conventional software systems," presented at SSGRR 2001: International Conference on Advances in Infrastructure for Electronic Business, Science, and Education on the Internet, L'Aquila, Italy, 2001.

[44] D. Lowe and B. Henderson-Sellers, "Web Development: Addressing Process Differences," *Cutter IT Journal*, 2001.

[45] D. Lowe and J. Eklund, "Commercial issues in specification of Web systems," presented at AWRE'2001: 6th Australian Workshop on Requirements Engineering, Sydney, Australia, 2001.

[46] D. Lowe and J. Eklund, "Client Needs and the Design Process in Web Projects," presented at WWW'2002: The Eleventh International World Wide Web Conference, Hawaii, USA, 2002.

[47] J. K. Holtzman, "Ready. Fire!! Aim???," presented at Proceedings of the 11th annual international conference on Systems documentation, Waterloo, Canada, 1993.

[48] L. Erskine, D. Carter-Tod, and J. Burton, "Dialogical techniques for the design of web sites," *International Journal of Human-Computer Studies*, vol. 47, pp. 169-195, 1997.