A Framework for Defining Acceptance Criteria for Web Development Projects

A/Prof David Lowe [1]

[1] Faculty of Engineering, University of Technology, Sydney
P.O. Box 123 Broadway NSW 2007 Australia
david.lowe@uts.edu.au

Abstract. Despite the rapid evolution of Web technologies and development tools and skills, most Web sites fail (to varying degrees) to achieve their true business goals. This is at least partially due to our inability to effectively define Web acceptance criteria (from both a client perspective and a developers perspective). These criteria cover those characteristics that the final system must possess, and against which the development can be carried out. Examples include broad business objectives, and detailed content and functional descriptions, but also navigability, user engagement, site evolvability, and especially site maintenance. In this paper we consider the need for an improved ability to define acceptance criteria for Websites as a target for the design and maintenance process. We describe a framework that includes dimensions covering both product criteria and organisational elements. We also discuss how the various dimensions within this framework can be represented using various existing techniques.

1 Introduction

There has been a phenomenal recent growth in the development of interactive media applications. This is especially true of Web-based development. Despite this rapid growth - or possibly as a partial consequence of it - our understanding of the purpose and design goals of Websites during development is typically very poor [1,2]. This problem is most noticeable in the difficulties that are typically encountered during contract and tendering negotiations for the outsourced development of Web-based projects. It is not uncommon to find Web projects that are poorly understood, resultant sites or applications which do not come close to achieving their desired aims, project bids which vary in estimated cost by anything up to an order of magnitude, and significant conflicts or dissatisfaction between developers and clients over the costs and development results [2].

Many of these problems can be traced to an inability to accurately define the needs of clients in a way that is both understandable to developers and expressed in a way that allows them to be tied to the specific technical constructs which underpin the Web [1,3]. If we consider other developmental domains then we can see the difference more starkly. For example, in software engineering, it is common practice to develop - often in collaboration with the client - a set of "acceptance criteria" which define the goals of the project. These criteria then provide a target against which the development must be carried out. Additionally, these acceptance criteria and the resulting specifications are key elements in techniques for determine the scope and costs of projects - such as Boehm's development of COCOMO [4] and Basili's work on the TAME resourcing model [5]. In these cases, although the specific form of the acceptance criteria may vary - for example they may be represented as a user requirements document or as a contracted statement of work - the language is which they are couched is well understood within the profession. Indeed most professions have developed specific "languages" that are commonly understood and which can be used to define the needs and scope of a project. This is not yet true of Web development. Although the technical aspects of the Web are well understood, and methods for expressing client objectives are evolving, the two have not yet been reconciled.

This is not to say that it is appropriate or even desirable to create a pro-forma and associated specification language for defining "Web Requirements Specification". Indeed, as we shall discuss later in this paper, the nature of Websites and hence the development process is such that this may be counter-productive. We do however need to be able to define a target against which the design, implementation and maintenance of Web sites can be carried out. For want of a better term, we have in this paper referred to this target as the site acceptance criteria - i.e. those elements that specify what will result in a Web site or system that is acceptable to the client.

In the following section we consider in more detail the need for acceptance criteria and how they are handled in other domains. We also look at the current situation in Web development and comment on the problems that are arising as a result of a lack of a common language for defining acceptance criteria.

We then move on to looking at how such a language might be constructed and what form it might take for Web development. In particular we emphasise that for Web development the language needs to consider both the product characteristics (where research on areas such as usability analysis can be beneficial) as well as organisational processes. These processes need to be put into place to cope with the incremental and ongoing nature of website evolution. Including these processes in the initial consideration, and hence into acceptance criteria, is crucial. We illustrate this point by looking at some analogous domains (landscape gardening, city planning, etc.) where the goal is not just a "product", but also includes the ongoing process for coping with the evolution of the "product".

We then consider how we tie these elements together into an overall framework for defining Website projects. This framework makes use of various disparate areas, including work on hypermedia and Web evaluation such as SUE [6], the Technology Assessment Method, [7] and conventional software engineering processes and standards.

We finish by acknowledging that we have raised many more questions than answers, but have provided an initial research agenda and begun pointing the way towards possible resolutions of some of these questions. Even at this early stage, it is possible to utilise these ideas in defining Web specifications that are clearer and more likely to result in improved systems.

2 Background

Web development encompasses the creation and maintenance of an increasingly wide range of applications, covering a diverse set of needs and potential clients. They extend from content-rich Web sites to E-commerce systems; from flexible document management systems to workflow and process tools. Although the domains of application are very diverse, these applications all have some common characteristics. For example, they all utilise rapidly evolving Web technologies to provide solutions which evolve over time in a significantly more fine-grained, even organic, manner than is typical of more conventional information or software systems. An important question that this raises is how exactly do we develop an understanding of exactly what form these solutions should take.

2.1. Understanding Client Needs in Web Development

It is commonly accepted in commercial Web development that the determination of the purpose and scope of Websites is typically very poorly understood. This is for a variety of reasons, but includes a lack of understanding of the Web development process [8], an evolving understanding of the potential of Web technologies, and communication breakdowns between clients and developers. This generates a wide range of potential problems, including:

- ***Poor quality and unmaintainable applications***. If the developers are unable to understand and/or express the needs of clients in a way that allows translation into specific design solutions, then the resultant Web applications or systems will be inherently unable to accurately address these needs. The result will be applications which have low quality (in the sense that quality equates to "fitness for purpose"). They are also more likely to be difficult to maintain, given that the initial structure will be less well suited to the initial needs.
- ***Poor scoping and hence planning of development projects***. When developers do not have a good grasp of the needs of the clients it becomes substantially more difficult to determine the scope of development projects, and hence to resource and plan the projects.
- ***Increased difficulty in providing competitive bids***. The vast majority of Web development is carried out on a commercially competitive basis. Competitive bidding on these projects relies heavily on accurate cost estimation which in turn is dependant upon an accurate understanding of the client needs and how these relate to the technical foundations upon which the application will be built. Without an understanding of clients' needs there will be substantial risks of both underbidding and overbidding. Conversely, an ability to accurately define client needs is a critical element of improved models for resource allocation and cost breakdown - which are in turn important for bid preparation, and in assisting cli-

> ents to evaluate bids more objectively and comprehensively. Greater budgetary detail will also enhance the effectiveness of contract negotiation.

It is worth noting in passing that many of the more successful commercial Web development organisations have addressed this issue by taking a strongly collaborative approach to the identification of client needs. They will often work very closely with a client during not only the initial discussions, but often well into the design stages of a project.

Having accepted the importance in the development process of understanding client needs, it is useful to consider how client needs are elicited in other development domains.

2.2 Software Specification Process

In software engineering there are well-established mechanisms for identifying and recording user needs. For example, the process might typically involve moving from a set of client needs to a formalised expression of these as a set of user requirements (often recorded as a URD - User Requirements Document). The URD can be analysed, and subsequently refined, using a variety of analysis techniques, tools and methods to determine possible flaws, missing requirements or ambiguities. The result of this is a refined URD that captures the clients view of their needs, and a Software Requirements Specification (SRS) which captures a technical expression of the requirements that can be used as the basis for development. If developed correctly, the URD is understandable and acceptable to the client, and the SRS is a consistent technical representation of the URD.

Although the specific process will often vary from this, the basic activities of the process and the outcomes are relatively well established. Similarly, we can go one step further and look at the elements that are typically considered in a URD, SRS or equivalents: functional requirements, performance requirements, interfaces and behaviours, and non-functional requirements such as robustness and maintainability.

The notations and terms for discussing these elements vary, but the terms are still couched in a common language that is relatively well understood by both clients and developers. Similarly, there are common expectations about the process even though the specific activities may vary (being incremental, involving prototyping etc). The result is a common expectation that the initial stages of the project will result in a project specification that can be used as a target for a well defined, and well-bounded, development project. It is also worth noting that the same requirements are typically used at the end of the project as the basis for a set of acceptance tests that determine the acceptability or otherwise of the product that has been developed. This essentially closes the development loop, allowing both clients and developers to close the development (or specific stages of the development - something that is typically very important for contractual development.

Table 1. SUE Hypermedia-Specific Usability Attributes

General Principles	Criteria	Attributes
Efficiency	Accessibility	Access layer soundness Navigational richness
	Orientation	Session history soundness Context observability Reuse soundness
	User control availability	Media control availability Navigational control availability
Learnability	Consistency	Structural Consistency Dynamic Consistency
	Predictability	Regularity Media Interface Soundness Navigation Interface Soundness Collection Ordering Coherence User’s Knowledge Conformance

2.3 Web Specification

Unlike software development, and development in most other domains, Web development is typically lacking numerous aspects. These include: a well-established process for developing an understanding of client needs; a language which is common between clients and developers for communicating and representing these needs; and a clear technique for closing the loop and providing closure for development effort. It should be recognised that in each case there are gradually appearing commercial approaches to addressing these - though not in a consistent or cohesive way (which is critical for a clients understanding of, and ability manage a project). Each of these elements is critical for an effective and manageable development process.

We can gain some insights into this by looking at existing research directions. Consider the above discussion about elements of a software specification: functional, performance, behavioural, user-interface, non-functional. This categorisation is not necessarily appropriate for Web applications - given the inherent differences between most Web applications and other types of software applications or systems. We can start to identify a parallel set of requirements categories by looking at the nature of Web applications and how they might be evaluated.

For example, work on the SUE methodology [6] has provided a systematic approach to the evaluation of hypermedia (including Websites). This method provides a broad multi-dimensional analysis of different elements of usability, considering criteria such as accessibility, orientation, user control etc. These criteria, and the associated evaluation activities, are aimed at identifying possible problems in applications, rather than quantifying client needs. They do however provide guidance on those aspects of development related to creating a usable application and which are worthy of consideration during the initial determination of client needs. Table 1 shows the usability attributes defined by SUE.

SUE however only addresses a very specific hypermedia-related set of usability attributes. It does not address issues related to the extent of the required content, ways in which this content might be maintained over time, the expected functional behaviour, issues such as security and access control, etc. Similarly, the concepts are phrased in the language of evaluation and not in a 'client' understandable language. This needs to be addressed before we can begin to effectively develop a basis for specifying Web application acceptance criteria.

3 The Need for a Specification Language

One of the biggest problems currently facing web developers and their clients is the lack of an established language or vocabulary for describing Web systems. Much of the vocabulary of the developer is tied to specifics of tools and technologies. This is quite necessary for following new technological trends and advancements in the field. Unfortunately, the client for whom a web application is being developed is not likely to have the same vocabulary as the developers for discussing systems and technology. They will describe their web requirements in terms that are specific to their particular domain of interest - often a particular business domain. These descriptions will often rely heavily for their meaning on the nature of the domain, and without a detailed knowledge of this domain, the developer runs the risk of misinterpreting these descriptions. In addition, such descriptions often do not lend themselves to quantifiable objectives, something a developer should seek to establish in order to define the scope of the project. An ambiguous, unquantifiable system description is a project disaster waiting to happen.

For a developer, this ambiguity is an unavoidable consequence of trying to map client needs in a specific domain into acceptance criteria using a universally understood language (i.e understandable by both client and developer) and then into particular implementation technologies. This problem is not specific to web developers. It is faced in many other domains where clients' needs must be interpreted in order to specify a product. Two often cited examples are software development, and graphic design. Techniques for describing user requirements developed in many of these fields are applicable to web development. Unfortunately, on their own they are not sufficient. The fact that many different disciplines provide input the Web development process means we cannot rely exclusively on any particular one, and hence we must borrow from them wherever possible to develop a specification language for web projects.

Of perhaps greater concern in Web development is the rapidity with which technology and tools evolve. In the lifecycle of a single project, a technology can become obsolete, or a preferred look-and-feel can become "yesterdays news". Such a dynamic environment introduces new complexities into a specification language. If a project’s scope is not carefully specified then changes in technology or customer expectations can lead to difficulties in ever establishing that a project has met the stated specifications. For example a non-exact specification such as "must look good" is very subjective, and can easily change as tastes change over the period of the development lifecycle.

A similar problem is the changes that occur in the clients’ technical understanding and level of expectation over the life cycle of a project. The rapid growth of the web and its related technologies makes it difficult even for people in the field to keep abreast of all that is going on. For clients’ whose core interest is elsewhere, it is well nigh impossible. However, the popularity of the Web means that increasingly diverse groups of people are being drawn into discussions about how the technology can be applied to benefit them. As these people are exposed to the technology their knowledge grows, and with it their expectations and requirements also grow. What, at the beginning of the project, seemed quite impressive to a client has, by the end of the project, become unsatisfactory. From the developers' perspective, it is essential to describe their specification in a way that clearly defines their responsibilities in regard to the scope of the project.

Before we consider the dimensions of this specification language there is one final set of observations that are important to make. This is with regard to the nature of the Web development process and how it might influence what we wish to express.

4 The Form of Web Development

Much of the language of software specification has an implicit assumption about the way in which software development is carried out. Specifications are typically predicated on an understanding that there will be a particular software "release" which is the target of development. This in turn means that it is reasonable to define just the nature of this release (or a specific set of releases).

Web development typically has a very different development cycle, and as such the underlying assumptions regarding how systems are specified needs to be carefully examined in the context of this changed process.

4.1 Web Development: Organic rather than Defined Releases

The similarities of web development to fields such as software development and graphic design can mask the rather significant differences. As was mentioned briefly earlier in the paper Web development tends to differ greatly in that we are no longer aiming to develop a "finished" product. Rather, we are aiming to create an organic entity that starts with an initial consistent structure, but continues to grow and evolve over time. This evolution is much finer-grained than the maintenance changes that occur with more traditional software products, and tends to be an integral part of the life cycle of the product. Compare this to conventional software maintenance, which tends to be a coarse-grained response to errors in the products, changes in requirements, or a changing environment.

A major consequence of this is that it becomes no longer appropriate to define a set of acceptance criteria for a fixed product. Indeed, the concept of defining a static development target is no longer relevant. Yet, despite this, we still need to have a basis for both development and evaluation, and probably also contract negotiation. How do we define acceptance when there is no stationary target against which we can design? To answer this, let us look a little deeper still.

Most Web development typically involves the establishment of an initial information architecture [9] that then supports the evolution of the site. This evolution (at least when it is successful) includes a comprehensive integration of the content maintenance into the organisational processes of the client. Where this integration does not occur, the site rapidly stagnates and ceases to serve a valid function. As discussed in [9], in this context a successful development effort would cover:

- "Clarifies the mission and vision for the site, balancing the needs of its sponsoring organization and the needs of its audiences.
- "Determines what content and functionality the site will contain.
- "Specifies how users will find information in the site by defining its organization, navigation, labeling and searching systems.
- "Maps out how the site will accommodate change and growth over time."

It is the last point that is the element which provides the fundamental difference between Web development and development of conventional software systems. It implies that the project does not have a point of closure. Rather, the criteria for development cover both the initial framework that must be established, and procedures for ongoing development that must be put in place.

As a simple example consider the situation where a business sells many different types of widgets. They currently have well-established, effective, and well-understood processes for managing stock levels, product catalogues, supplier ordering etc. They decide that they need to "sell via the internet" in order to remain competitive. They contract out the development, specifying the content, look and feel, marketing focus and potential target group of the site. The developers create and deliver a site that is initially very effective, but rapidly becomes extremely difficult to maintain. The product lists and details rapidly change and the site requires continual development to keep up to date. The ordering processes are such that the company now has two different and incompatible sets of business processes. The problem is exacerbated when 6 months later the company decides to change it's marketing campaign and wishes to modify the entire look and feel of the site. In other words, the entire site has not been developed to integrate cleanly with existing content databases, business practices and workflows, nor with an understanding of the potential for significant changes.

The implications of this can be better understood by looking at some development domains that involve similar evolutionary or organic development.

4.2 Web Engineering or Web Gardening?

In order to understand the evolutionary nature of the web development lifecycle it is useful to move away from viewing it in relation to software engineering, graphic design, or marketing, as is often done. Although perhaps not obvious at first sight, parallels can be drawn between web development and areas such as town planning and landscape gardening. Let’s explore each of these further.

Software engineering is about adopting a consistent and scientific approach, tempered by a specific practical context, to development and commissioning of systems or applications. Website development is often much more about creating an infrastructure (laying out the garden) and then 'tending' the information which grows and blooms within this garden. Landscape gardening involves the creation of a structure that is consistent with the initial objectives, but also takes into account the way in which the garden will change over time. A good initial design for a garden will allow this growth to occur in a controlled and consistent manner. The evolution of Web applications is analogous to a garden changing as a natural part of its cycle of growth. We have inherent growth (changes in the content), day to day maintenance of the garden (updating links, compressing databases, regenerating dynamic pages) and very occasional redesigns. In both cases, we are constantly working with a changing evolving system.

We can also draw comparisons with town planning. This emerged out of the chaos that resulted as large numbers of people came to occupy relatively small areas of land. Without decent roads, water and plumbing such places were prone to regular disasters. While populations were small, a haphazard approach to organisation was sufficient, and problems could be addressed as they arose. For larger populations, problems quite readily became disasters. A well thought out, organised town tends to be less prone to problems, and, when problems do arise, they can be addressed more readily.

But town planning goes further than organising the town as it stands now. It recognises that towns and cities are dynamic entities that tend to grow and change, and that this growth, if not managed carefully, will rapidly result in significant problems. Growth is planned for, and the town infrastructure is continually expanded to properly support future growth. This approach of the town planner holds a valuable lesson for the web developer. Web structures are rarely static. The client will want to continually add and change content, modify the look and feel, or enhance the functionality long after the site has been initially ‘commissioned’. Users will want new ways of accessing and navigating through the changing information. Technologies will evolve and become more sophisticated. In other words, the development of a web application does not halt when it is initially commissioned. Rather, its growth has just begun. The ability of the information architecture to cope with this growth is a significant factor in the perceived success or failure of a web site - and should therefore be an integral part of any set of acceptance criteria.

Table 2. Acceptance Criteria Framework

Dimension	Possible Representations	Example Elements
Client/User		
Client problem statement	(Natural language)	
Product vision	(Natural language)	Client needs and business objectives
Users	(Natural language)	User descriptions and models
Application		
Content modelling	Structured language, hypermedia / information modelling languages (OOHDM, HDM, entity modelling, etc.)	Existing content structure, Information views, Navigational structures, Required content
User interaction	Modified TAM	Usability and usefulness metrics
	Structured language, hypermedia modeling, HCI models, etc	Access mechanisms, user control behaviour, user orientation, search requirements, security control
Development Constraints	Natural language, standards	Adherence to corporate policies, Resource availability
Non-functional requirements	Natural language, quality metrics, adherence to standards	Reliability of content, Copyright constraints
Application Evolution		
Evolution directions	(Natural language)	Expected content changes
Client adoption/ integration of Web	Business Process Reengineering	Information dissemination paths, Workflow changes
Maintenance processes	Natural language, process models	Content maint. responsibility, Web management

		cycles

4.3 Delivering Product and Process

The upshot of the above discussions is that it implies that, unlike more conventional development, the development of Web applications needs to consider much more actively both the initial product and the process by which this will be managed and maintained. As borne out by current practice, the focus of successful Web development is not only the particular content, nor is it only the development of an architecture for organising and maintaining this content. Rather it extends to cover the integration of these elements into the activities and culture of the client organisation. This integration will need to involve both designing technological solutions which are suited to the business processes, and elements of BPR (Business Process Reengineering) [10] where the business processes are adapted to the constraints or requirements of the technologies. These key observations give us clear pointers to the form which acceptance criteria must take.

5 An Acceptance Criteria Framework

Drawing the above diverse discussions together, we have developed an initial framework for specifying acceptance criteria for Web sites. This framework, shown in Table 2, identifies the key dimensions that should be covered in defining a "target" against which development can be carried out.

Several important observations need to be made. First, unlike more traditional application development, these dimensions not only define a specific product, but also expectations about how that product should be able to evolve over time. Second, the three top-level categories (client/user, application framework, and application evolution) are tightly interrelated and cannot be treated - or specified - in isolation. Finally, the dimensions should not be mistaken for specifying possible designs, or an information architecture, or implementation constructs. They are solely intended to identify those aspects that need to be specified in order to define expectations of the outcomes of a Web development project.

For each of the acceptance criteria dimensions we have provided some initial (though as yet unproven) representations which can form the basis of a specification language. These representations need to be understandable by both clients and developers - i.e. they must form a common language. This language provides the basis for the expression of clients' needs, and the basis for developers design and implementation. We have not yet validated these, other than through some initial unqualified studies and using anecdotal information.

In order to demonstrate the applicability of this framework to the specification of websites and web-based systems we have developed several examples. A simple partial example of a Web specification that utilises these concepts is shown in Appendix A. This illustrates how the various elements of the framework can be applied. We are currently undertaking work looking at how existing Web specifications map to this framework, and evaluating the use of the framework in commercial development projects.

6 Conclusions and Future Work

Based on a detailed consideration of how Web applications are being developed, and in particular the organic nature of Website evolution and maintenance, we have proposed a framework for defining acceptance criteria for Websites. These acceptance criteria can provide a basis for contract negotiations between clients and developers, but even more significantly, they can provide a basis against which the design can be carried out. The framework includes both the dimensions required to define acceptance criteria, as well as an initial tentative identification of potential representations that can be used for documenting the different dimensions of the acceptance criteria. These representations form a common language that enables clients, users and developers to effectively discuss the requirements of Web applications.

The work to date has provided a justification for the need for an acceptance criteria framework, and the basis of a research agenda in this area. Future work will focus on developing a greater degree of rigor in the dimensions of the framework. We shall then clarify the possible representations that can be used to capture each of these dimensions. Indeed further research is likely to emphasise not the particular representations, but the constraints that the representations must meet to be valid for that dimension.

A parallel stream of research is to correlate these dimensions and representations to empirical data on Web project specifications and development contracts. This will help us determine those aspects that have proven to be most useful for practical development, and their relative importance. Most significantly, this work will look at how user acceptance (measured using an adapted version of Davis' technology assessment model [7]) relates to the significance of various application characteristics and hence potential acceptance criteria.

Developing effective techniques for creating Web application acceptance criteria will have several significant benefits. These include: more effective and better managed negotiations between clients and Web developers; more clearly defined applications and hence higher quality applications; and a significantly improved ability to understand the scope of development early in the project and hence improved management, resource and costing.

Acknowledgements

The author would like to thank Richard Webby, Ross Jeffery, Nick Carr, Marcus Carr, Malcolm Raymond and John Eklund for the valuable input provided in this research.

References

1. Lowe D.B.; "Engineering The Web"; In *WebNet Journal* Vol. 1, No. 1
2. Lowe D.B. and Hall W.; Hypermedia and the Web: An Engineering Approach; Wiley, 1999
3. Bieber, M. & Isakowitz, T. "Designing Hypermedia Applications", *CACM*, August 1995, 38(8):26-29
4. Boehm, B.W. *Software Engineering Economics*, 1981, Prentice Hall, Englewood Cliffs, NJ.
5. Basili, V.R. & Rombach, H.D. "The TAME project: Towards Improvement-Oriented Software Environments", *IEEE Trans. Software Eng*. 1988, 14(6):758-773.
6. Garzotto F and Matera M (1997) "A Systematic Method for Hypermedia Usability Inspection", *The New Review of Hypermedia and Multimedia*, Vol 3, pp39-65
7. Davis, F.D. "Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology", *MIS Quarterly*, 1989, 13(3):319-342.
8. Lowe D, Webby R and Bucknell A (1999) "Improving Hypermedia Development: A Reference Model-Based Process Assessment Method", *ACM Hypertext '99: 10th International Conference on Hypertext and Hypermedia*, Darmstadt, Feb 21-25
9. Rosenfeld L and Morville P (1998) *Information Architecture for the World Wide Web*, O'Reilly and Associates.
10. Hammer & Champy (1993) *Reengineeing the Corporation: A Manifesto for Business Revolution*, Harper Business, New York
11. Garzotto F., Mainetti L., and Paolini P.; Hypermedia Design, Analysis, and Evaluation Issues". Communications of the ACM, 38(8).
12. Mukhopadhyay T., Vicinanza, S. & Prietula M.J. "Estimating the Feasibility of a Case-Based Reasoning Model for Software Effort Estimation", *MIS Quarterly*, 1992, 16(2): 155-171.
13. Nielsen J. and Mack R.L.; "Usability Inspection Methods", New York: John Wiley & Sons,1994

Appendix: Sample Specification

The following is a collection of fragments from a typical Web specification. (Note: In the final version of this paper, a link will be provided to a full online version of a typical specification).

XYZ Widget Company
Web Acceptance Criteria

Document number: ZYZWC-34TR4
Version: 0.3a (draft)   Date: 17th November 1999

Author: David Lowe, University of Technology, Sydney
Distribution: uncontrolled

Overview

This document contains the specification of the Web-based E-commerce system to be developed for XYZ Widget Company. It outlines key issues, development constraints and site requirements. This document contains:

1. Client Problem Statement
2. Site Vision
3. User models
4. Required content
5. User interactions
6. Non-Functional requirements
7. Development constraints and technical restrictions
8. Support for site maintenance
9. Integration into client organisation
10. Development schedules and deliverables
11. Acceptance mechanisms and client liaison

1. Client Problem Statement

XYZ Widget Company is the leading European distributor of quality commercial-grade widgets. They have been in operation for over 70 years and have established an international reputation for quality products and efficient and effective service.

In line with this emphasis on providing service to our clients we wish to extend the distribution channels to include the internet. Several of our competitors in the highly competitive widget industry have established a Web presence and we see this as both a significant threat and a huge opportunity to broaden our client base.

In this context we see a Web presence as providing both a new channel providing access to similar services to those we currently provide, as well as a vehicle for extending the range of services.

more details here

2. Site Vision

Client Needs: The clients of XYZ-WC are extremely diverse, coming not oly from the commercial sector, but also from on-sellers, individual contractors, and government agencies. A full list of clients, along with general characteristics and a client needs analysis, is given in Appendix A. more details here

Business objectives: The purpose of the site if two-fold. Firstly, the site must support the maintenance of the existing client base, by providing an enhanced service. Specifically, the site should not only support existing ordering processes, but also provide access to information not currently available. For example, the provision of Widget data sheets will facilitate the retention of clients. Secondly, the site must be able to support (and be consistent with) the active marketing campaigns currently carried out by XYZ-WC. *More details here*

3. User models

Appendix B contains full scenarios (diagrammed using object-oriented UML notation) that illustrate usage patterns that the site must support.

more details here

4. Required content

The content that must be accessible from the site includes the following:

- Full product catalogue, including product specifications, an image of each widget, product data sheets, cost and more details here. Note that users must be able to identify themselves and the relevant costing model utilised (including different currencies, and tax schemes). An example of the existing product information contained in the master product catalogue is shown in Appendices C and D.
- Contact information for XYZ-WC
- Corporate information on XYZ-WC
- Information on the correct usage and installation of widgets
- *more details here*

Note that the site should be developed in a way which is consistent with the assumption that the information to be provided in the site will change regularly.

Appendix E of this specification provides a model of the current information sources, represented using OOHDM.

more details here

5. User interactions

The site must support effective interaction with users. Appendix B detailed typical usage scenarios that must be available. In addition to this the site must support the following interaction mechanisms:

- Every page must contain the primary site menu, and an identification of the current location within the site structure.
- A site map
- A search engine that allows clients to search for Widgets by name, part number, clasification, manufacturer, more details here
- Users must be able to register with the system and then log in at a later date. Once logged in the system will utilise information stored on company, deliver details, payment schemes, costing models, more details here
- *more details here*

Utilising TAM (Technology Acceptance Model) the developed site must rank at least 9.0 on the "perceived usefulness" scale and at least 6.0 on the "percieved ease-of-use" scale.

more details here

6. Non-Functional requirements

more details here

7. Development constraints and technical restrictions

The site must be:

- usable on all browsers from IE3 and Netscape 3 onwards
- No page must be larger than 30k to download (including images) unless the user is explicitly warned of the large size.
- *more details here*

8. Support for site maintenance

Appendix G of this specification details the current workflows used to maintain the product catalogues, ordering databases, invoicing and supply processes, and supplier purchases. The site must be integrated into these processes so that there is no impact on the ability to carry out, or cost in carrying out, these activities. The content in the site must be automatically maintained from the existing databases and integrate with these flows.

More details here

9. Integration into client organisation

More details here

10. Development schedules and deliverables

more details here

11. Acceptance mechanisms and client liaison

more details here

Appendices

A. XYZ Widget Company Client list and characterisation
B. Usage models: Scenario diagrams and use cases
C. Example XYZ Widget product catalogue
D. Sample of master product database
E. OOHDM model of information sources
F. Sample screen design
G. Current workflow processes