
Introduction

The World Wide Web (Web) was originally conceived in 1989 as an environment to allow for the sharing of information (e.g., research reports, databases, user manuals) amongst geographically dispersed individuals. The information itself was stored on different servers and was retrieved by means of a single user interface (Web browser). The information consisted primarily of text documents inter-linked using a hypertext metaphor (<http://www.zeltser.com/web-history/>) [1].

Since its original inception, the Web has changed into an environment employed for the delivery of many different types of applications. Such applications range from small-scale information-dissemination-like applications, typically developed by writers and artists, to large-scale commercial, enterprise-planning and scheduling, collaborative-work applications. The latter are developed by multidisciplinary teams of people with diverse skills and backgrounds using cutting-edge, diverse technologies [1–3]. The increase in the use of the Web to provide commercial applications has been motivated by several factors, such as the possible increase of an organisation's competitive position, and the opportunity for small organisations to project their corporate presence in the same way as that of larger organisations [4]. Numerous current Web applications are fully functional systems that provide business-to-customer and business-to-business e-commerce, and numerous services to numerous users [1].

Industries such as travel and hospitality, manufacturing, banking, education and government utilised Web-based applications to improve and increase their operations [3]. In addition, the Web allows for the development of corporate intranet Web applications, for use within the boundaries of individual organisations [5]. The remarkable spread of Web applications into areas of communication and commerce makes it one of the leading and most important branches of the software industry [1].

Web development is a relatively new and rapidly growing industry, with e-commerce alone weathering the recession and growing 11 % in the United States

in 2009, with similar growth in 2010.¹ This continued growth makes it worthwhile to conduct research that enables Web development companies to make more efficient managerial decisions [6].

To date, the development of Web applications has generally been ad hoc, resulting in poor-quality applications, which are difficult to maintain [1]. The main reasons for such problems are unsuitable design and development processes, and poor project management practices [3]. A survey on Web-based projects, published by the Cutter Consortium in 2000, revealed a number of problems with outsourced large Web-based projects [3]:

- 84 % of surveyed delivered projects did not meet business needs.
- 53 % of surveyed delivered projects did not provide the required functionality.
- 79 % of surveyed projects presented schedule delays.
- 63 % of surveyed projects exceeded their budget.

As the reliance on larger and more complex Web applications increases, so does the need for using methodologies and best practice guidelines to develop applications that are delivered on time, within budget, have a high level of quality and are easy to maintain [7–9]. To develop such applications Web development teams need to use sound methodologies, systematic techniques, quality assurance, rigorous, disciplined and repeatable processes, better tools, and baselines. Web engineering² aims to meet such needs [11].

Web engineering is described as [12]:

the use of scientific, engineering, and management principles and systematic approaches with the aim of successfully developing, deploying and maintaining high quality Web-based systems and applications.

This is a similar definition to that used to describe software engineering; however, both disciplines differ in many ways. Such differences are discussed next.

Web Applications Versus Conventional Software

An overview of differences between Web and software development with respect to their development processes, technologies, quality factors, and measures is presented here. In addition, this section also provides definitions and terms used throughout the book (e.g., Web application).

¹ <http://blogs.wsj.com/digits/2010/03/08/e-commerce-growth-slows-but-still-out-paces-retail/>

² The term “Web engineering” was first published in 1996 in a conference paper by Gellersen et al. [10]. Since then this term has been cited in numerous publications, and numerous activities devoted to discussing Web engineering have taken place (e.g., workshops, conference tracks, entire conferences).

Web Hypermedia, Web Software or Web Application?

The Web is the best-known example of a hypermedia system. To date, numerous organisations world-wide have developed a vast array of commercial and/or educational Web applications. The Web literature uses numerous synonyms for a Web application, such as Web site, Web system, Internet application. The IEEE Std 2001–2002 uses the term Web site defined as [13]:

“A collection of logically connected Web pages managed as a single entity.”

However, using Web site and Web application interchangeably does not allow one to differentiate between the physical storage of Web pages and their application domains.

The Web has been used as the delivery platform for three types of applications: Web hypermedia applications, Web software applications, and Web applications [14].

- *Web hypermedia application*—a nonconventional application characterised by the authoring of information using nodes (chunks of information), links (relations between nodes), anchors, access structures (for navigation), and delivery over the Web. Technologies commonly used for developing such applications are HTML, XML, JavaScript and multimedia. In addition, typical developers are writers, artists and organisations who wish to publish information on the Web and/or CD-ROMs without the need to know programming languages such as Java. These applications have unlimited potential in areas such as software engineering, literature, education and training.
- *Web software application*—a conventional software application that relies on the Web or uses the Web’s infrastructure for execution. Typical applications include legacy information systems such as databases, booking systems, knowledge bases, etc. Many e-commerce applications fall into this category. Typically they employ development technologies (e.g., DCOM, ActiveX, etc.), database systems, and development solutions (e.g., J2EE). Developers are in general young programmers fresh from a Computer Science or Software Engineering degree course, managed by a few more senior staff.
- *Web application*—an application delivered over the Web that combines characteristics of both Web hypermedia and Web software applications.

Web Development Versus Software Development

Web development and software development differ in a number of areas, which will be detailed later. However, of these, three such areas seem to provide the greatest differences and to affect the entire Web development and maintenance processes. These areas encompass the people involved in development, the intrinsic characteristics of Web applications and the audience for which they are developed.

The development of conventional software remains dominated largely by IT professionals, where a sound knowledge of programming, database design, and project management is necessary. In contrast, Web development encompasses a

much wider variety of developers, such as amateurs with no programming skills, graphics designers, writers, database experts and IT professionals, to name but a few. This is possible as Web pages can be created by anyone without the necessity for programming knowledge [15].

Web applications by default use communications technology and have multi-platform accessibility. In addition, since they employ a hypermedia paradigm, they are non-sequential by nature, using hyperlinks to interrelate Web pages and other documents. Therefore, navigation and pluralistic design become important aspects to take into account. Finally, the multitude of technologies available for developing Web applications means that developers can build a full spectrum of applications, from a static simple Web application using HTML to a fully-fledged distributed e-commerce application [7]. Conventional software can be developed using several programming languages running on a specific platform, components off the shelf (COTS), etc. It can also use communications technology to connect to and use a database system. However, the speed of implementing new technology is faster for Web development relative to non-Web-based applications.

Web applications are aimed at wide-ranging groups of users. Such groups may be known ahead of time (e.g., applications available within the boundaries of an intranet). However, it is more often the case that Web applications are devised for an unknown group of users, making the development of aesthetically pleasing applications more challenging [16]. In contrast, conventional software applications are generally developed for a known user group (e.g., department, organisation) making the explicit identification of target users an easier task.

For the purpose of our discussion, we have grouped the differences between Web and software development into 12 areas, which are as follows:

1. Application characteristics
2. Primary technologies used
3. Approach to quality delivered
4. Development process drivers
5. Availability of the application
6. Customers (stakeholders)
7. Update rate (maintenance cycles)
8. People involved in development
9. Architecture and network
10. Disciplines involved
11. Legal, social and ethical issues
12. Information structuring and design

(1) *Application Characteristics*

Web applications are created by integrating numerous distinct elements, such as fine-grained components (e.g., DCOM, OLE, ActiveX), interpreted scripting languages, components off the shelf (COTS, e.g., customised applications, library components, third-party products), multimedia files (e.g., audio, video, 3D objects), HTML/SGML/XML files, graphical images, mixtures of HTML and programs, and databases [16–18]. Components may be integrated in many

different ways and present different quality attributes. In addition, their source code may be proprietary or unavailable, and may reside on and/or be executed from different remote computers [17]. Web applications are, for the large part, platform-independent (although there are exceptions, e.g., OLE, ActiveX), and Web browsers in general provide similar user interfaces with similar functionality, freeing users from having to learn distinct interfaces [16]. Finally, a noticeable difference between Web applications and conventional software applications is in the use of navigational structures. Web applications use a hypermedia paradigm where content is structured and presented using hyperlinks. Navigational structures may also need to be customised, i.e., by the dynamic adaptation of content structure, atomic hypermedia components, and presentation styles [10].

Despite the initial attempt by the hypermedia community to develop conventional applications with a hypermedia-like interface, largely conventional software applications do not employ this technique.

Again in contrast, conventional software applications can also be developed using a wide variety of components (e.g., COTS), generally developed using conventional programming languages such as C++, Visual Basic, and Delphi. These applications may also use multimedia files, graphical images and databases. It is common that user interfaces are customised depending on the hardware, operating system, software in use and the target audience [16]. There are programming languages on the market (e.g., Java) that are intentionally cross-platform; however, the majority of conventional software applications tend to be monolithic, running on a single operating system.

(2) *Primary Technologies Used*

Web applications are developed using a wide range of diverse technologies, such as the many flavoured Java solutions (Java servlets, Enterprise JavaBeans, applets, and JavaServer Pages), HTML, JavaScript, XML, UML, databases and much more. In addition, there is an increasing use of third-party components and middleware. Since Web technology is an area that changes quickly, some authors suggest it may be difficult for developers and organisations to keep up with what is currently available [17].

The primary technology used to develop conventional software applications is mostly represented by object-oriented methods, generators and languages, relational databases, and CASE tools [18]. The pace with which new technologies are proposed is slower than that for Web applications.

(3) *Approach to Quality Delivered*

Web companies that operate their business on the Web rely heavily on providing applications and services of high quality so that customers return to do repeat business. As such, these companies only see a return on investment if customers' needs have been fulfilled. Customers who use the Web for obtaining services have very little loyalty to the companies they do business with. This suggests that new companies providing Web applications of a higher quality will most likely displace customers from previously established businesses. Further, that quality is the principal factor that brings repeated business. For

Web development, quality is often considered a higher priority than time to market, with the mantra “later and better” as the mission statement for Web companies who wish to remain competitive [17].

Within the context of conventional software development, software contractors are often paid for their delivered application regardless of its quality. Return on investment is immediate. Ironically, they are also often paid for fixing defects in the delivered application, where these failures principally exist because the developer did not test the application thoroughly. This has the knock-on effect that a customer may end up paying at least twice (release and fixing defects) the initial bid in order to make the application functional. Here time to market takes priority over quality, since it can be more lucrative to deliver applications with plenty of defects sooner than high-quality applications later. For these companies the “sooner but worse” rule applies [17].

Another popular mechanism employed by software companies is to fix defects and make the updated version into a new release, which is then resold to customers, bringing in additional revenue.

(4) *Development Process Drivers*

The dominant development process drivers for Web companies have three quality criteria [17]:

- reliability,
- usability and
- security,

followed by:

- availability,
- scalability,
- maintainability and
- time to market.

Reliability: applications that work well, do no crash, do not provide incorrect data, etc.

Usability: an application that is simple to use. If a customer wants to use a Web application to buy a product on-line, the application should be as simple to use as the process of physically purchasing that product in a shop. Many existing Web applications present poor usability despite the extensive range of Web usability guidelines that have been published. A Web application with poor usability will quickly be replaced by another more usable application as soon as its existence becomes known to the target audience [17].

Security: the handling of customer data and other information securely so that problems such as financial loss, legal consequences and loss of credibility can be avoided [17].

With regards to conventional software development, the development process driver is time to market and not quality criteria [17].

(5) *Availability of the Application*

Customers who use the Web expect applications to be operational throughout the whole year (24/7/365). Any downtime, no matter how short, can be detrimental [17].

Except for a few application domains (e.g., security, safety critical, military, banking) customers of conventional software applications do not expect these applications to be available 24/7/365.

(6) *Customers (Stakeholders)*

Web applications can be developed for use within the boundaries of a single organisation (intranet), a number of organisations (extranets) or for use by people anywhere in the world. The implications are that stakeholders may come from a wide range of groups where some may be clearly identified (e.g., employees within an organisation) and some may remain unknown, which is often the case [4, 16, 17, 19]. As a consequence, Web developers are regularly faced with the challenge of developing applications for unknown users, whose expectations (requirements) and behaviour patterns are also unknown at development time [16]. In this case new approaches and guidelines must be devised to better understand prospective and unknown users such that quality requirements can be determined beforehand to deliver high-quality applications [19]. Whenever users are unknown it also becomes more difficult to provide aesthetically pleasing user interfaces, which are necessary to be successful and stand out from the competition [16].

Some stakeholders can reside locally, in another state/province/county, or overseas. Those who reside overseas may present different social and linguistic backgrounds, which increases the challenge of developing successful applications [4, 16]. Whenever stakeholders are unknown it is also difficult to estimate the number of users an application will service, so applications must also be scalable [17].

With regards to conventional software applications, it is usual for stakeholders be explicitly identified prior to development. These stakeholders often represent groups confined within the boundaries of departments, divisions, or organisations [16].

(7) *Update Rate (Maintenance Cycles)*

Web applications are updated frequently without specific releases and with maintenance cycles of days or even hours [17]. In addition, their content and functionality may also change significantly from one moment to another, and so the concept of project completion may seem unsuitable in such circumstances. Some organisations also allow non-information-systems experts to develop and modify Web applications, and in such environments it is often necessary to provide an overall management of the delivery and modification of applications to avoid confusion [4].

The maintenance cycle for conventional software applications complies with a more rigorous process. Upon a product's release, software organisations usually initiate a cycle whereby a list of requested changes, adjustments or

improvements (either from customers or from its own development team) is prepared over a set period of time, and later incorporated as a specific version or release for distribution to all customers simultaneously. This cycle can be as short as a week and as long as several years. It requires more planning as it often entails other, possibly expensive activities such as marketing, sales, product shipping and occasionally personal installation at a customer's site [11, 17].

(8) *People involved in Development*

The Web provides a broad spectrum of different types of Web applications, varying in quality, size, complexity and technology. This variation is also applicable to the range of skills represented by those involved in Web development projects. Web applications can be created, for example, by artists and writers using simple HTML code or, more likely, one of the many commercially available Web authoring tools (e.g., Macromedia Dreamweaver, Microsoft Frontpage), making the authoring process available to those with no prior programming experience [4]. However, Web applications can also be very large and complex, requiring a team of people with diverse skills and experience. Such teams consist of Web designers and programmers, graphic designers, librarians, database designers, project managers, network security experts, and usability experts [17].

Web designers and programmers are necessary to implement the application's functionality using the necessary programming languages and technology. In particular, they also decide on the application's architecture and applicable technologies, and to design the application taking into account its documents and links [16]. Graphic designers, usability experts and librarians provide applications that are pleasing to the eye, easy to navigate and provide good search mechanisms to obtain the required information. This is often the case where such expertise is outsourced, and used on a project-by-project basis.

Large Web applications most likely use database systems for data storage, making it important to have team members with expertise in database design and the necessary queries to manipulate the data. Project managers are responsible for managing the project in a timely manner and allocating resources adequately such that applications are developed on time, within budget and are of high quality. Finally, network security experts provide solutions for various security aspects [3].

Conversely, the development of conventional software remains dominated by IT professionals, where a sound knowledge of programming, database design, and project management is necessary.

(9) *Architecture and Network*

Web applications are typically developed using a simple client-server architecture (two-tier), represented by Web browsers on client computers connecting to a Web server hosting the Web application, to more sophisticated configurations such as three-tier or even n -tier architectures [17]. The servers and clients within these architectures represent computers that may have different operating systems, software, hardware configurations, and may be connected to each other using different network settings and bandwidth.

The introduction of more than two tiers was motivated by limitations of the two-tier model (e.g., implementation of an application's business logic on the client machine, increased network load as any data processing is only carried out on the client machine). In such architectures the business logic is moved to a separate server (middle-tier), which services client requests for data and functionality. The middle-tier then requests and sends data to and from a (usually) separate database server. In addition, the type of networks used by the numerous stakeholders may be unknown, so assumptions have to be made while developing these Web applications [16].

Conventional software applications either run in isolation on a client machine or use a two-tier architecture whenever applications use data from database systems installed on a separate server. The type of networks used by the stakeholders is usually known in advance since most conventional software applications are limited to specific places and organisations [16].

(10) *Disciplines Involved*

A team of people with a wide range of skills and expertise in different areas is required to develop large and complex Web applications adequately. These areas reflect distinct disciplines such as software engineering (development methodologies, project management, tools), hypermedia engineering (linking, navigation), requirements engineering, usability engineering, information engineering, graphics design and network management (performance measurement and tuning) [3, 11, 19].

Building a conventional software application involves contributions from a smaller number of disciplines than those used for developing Web applications; these include software engineering, requirements engineering and usability engineering.

(11) *Legal, Social, and Ethical Issues*

The Web as a distributed environment enables a vast amount of structured (e.g., database records) and unstructured (e.g., text, images, audio) content to be easily available to a multitude of users worldwide. This is often cited as one of the greatest advantages of using the Web. However, this environment is also used for the purpose of dishonest actions, such as copying content from Web applications without acknowledging the source, distributing information about customers without their consent, infringing copyright and intellectual property rights, and even, in some instances, identity theft [16]. The consequences that follow from the unlawful use of the Web are that Web companies, customers, entities (e.g., W3C), and government agencies must apply a similar paradigm to the Web as those applied to publishing, where legal, social and ethical issues are taken into consideration [19].

Issues referring to accessibility offered by Web applications should also take into account special user groups such as the handicapped [16].

Conventional software applications also share a similar fate to that of Web applications, although to a smaller extent, since these applications are not so readily available for such a large community of users, compared to Web applications.

Table 2.1 Comparison between Web-based and traditional approaches

	Web-based approach	Traditional approach
Estimating process	Ad hoc costing of work, centred on input from the developers	More formal costing of work based on past experience from similar projects and expert opinion
Size estimation	No agreement upon a standard size measure for Web applications within the community	Lines of code or function points are the standard size measures used
Effort estimation	Effort is estimated using a bottom-up approach based on input from developers. Hardly any historical data is available from past projects	Effort is estimated using equations built taking into account project characteristics and historical data from past projects
Quality estimation	Quality is difficult to measure. Need for new quality measures specific for Web-based projects	Quality is measurable using known quality measures (e.g., defect rates, system properties)

(12) *Information Structuring and Design*

As previously mentioned, Web applications have structured and unstructured content, which may be distributed over multiple sites and use different systems (e.g., database systems, file systems, multimedia storage devices) [10]. In addition, the design of a Web application, unlike that of conventional software applications, includes the organisation of content into navigational structures by means of hyperlinks. These structures provide users with easily navigable Web applications. Well-designed applications should allow for suitable navigation structures [19], as well as the structuring of content, which should take into account its efficient and reliable management [16].

Another difference between Web and conventional applications is that Web applications often contain a variety of specific file formats for multimedia content (e.g., graphics, sound and animation). These files must be integrated into any current configuration management system, and their maintenance routines also need to be organised, as is likely that they will differ from the maintenance routines used for text-based documents [15]. Conventional software applications present structured content that uses file or database systems. The structuring of such content has been addressed by software engineering in the past so the methods employed here for information structuring and design are well known by IT professionals [16].

Reifer [18] presents a comparison between Web-based and traditional approaches that takes into account measurement challenges for project management (Table 2.1). Table 2.2 summarises the differences between Web-based and conventional development contexts.

As we have seen, there are several differences between Web development and applications and conventional development and applications. However, there are also similarities that are more evident if we focus on the development of large and complex applications. Both need quality assurance mechanisms,

Table 2.2 Web-based versus traditional approaches to development

	Web-based approach	Traditional approach
Application characteristics	Integration of numerous distinct components (e.g., fine-grained, interpreted scripting languages, COTS, multimedia files, HTML/SGML/XML files, databases, graphical images), distributed, cross-platform applications and structuring of content using navigational structures with hyperlinks	Integration of distinct components (e.g., COTS, databases, graphical images), monolithic single-platform applications
Primary technologies used	Variety of Java solutions (Java servlets, Enterprise JavaBeans, applets, and JavaServer Pages), HTML, JavaScript, XML, UML, databases, third-party components and middleware, etc.	Object-oriented methods, generators, and languages, relational databases, and CASE tools
Approach to quality delivered	Quality is a higher priority than time to market	Time to market takes priority over quality
Development process drivers	Reliability, usability and security	Time to market
Availability of the application	Throughout the whole year (24/7/365)	Except for a few application domains, no need for availability 24/7/365
Customers (stakeholders)	Wide range of groups, known and unknown, residing locally or overseas	Generally groups confined within the boundaries of departments, divisions, or organizations
Update rate (maintenance cycles)	Frequently without specific releases, maintenance cycles of days or even hours	Specific releases, maintenance cycles ranging from a week to several years
People involved in development	Web designers and programmers, graphic designers, librarians, database designers, project managers, network security experts, usability experts, artists, writers	IT professionals with knowledge of programming, database design and project management
Architecture and Network	Two-tier to n -tier clients and servers with different network settings and bandwidth, sometimes unknown	One- to two-tier architecture, network settings and bandwidth are likely to be known in advance
Disciplines involved	Software engineering, hypermedia engineering, requirements engineering, usability engineering, information engineering, graphics design and network management	Software engineering, requirements engineering, and usability engineering
Legal, social, and ethical issues	Content can be easily copied and distributed without permission or acknowledgement of copyright and intellectual property rights. Applications should take into account all groups of users including those handicapped	Content can also be copied infringing privacy, copyright, and IP issues, albeit to a smaller extent
Information structuring and design	Structured and unstructured content, use of hyperlinks to build navigational structures	Structured content, infrequent use of hyperlinks

development methodologies, tools, processes, techniques for requirements elicitation, effective testing and maintenance methods, and tools [19].

Conclusions

This chapter discussed differences between Web and software applications, and their development processes based on the following 12 areas:

1. Application characteristics
2. Primary technologies used
3. Approach to quality delivered
4. Development process drivers
5. Availability of the application
6. Customers (stakeholders)
7. Update rate (maintenance cycles)
8. People involved in development
9. Architecture and network
10. Disciplines involved
11. Legal, social, and ethical issues
12. Information structuring and design

Acknowledgements We would like to thank Tayana Conte for her comments on a previous version of this chapter.

References

1. Murugesan S, Deshpande Y (2002) Meeting the challenges of web application development: the web engineering approach. In: Proceedings of the 24th international conference on software engineering, Orlando, FL, pp 687–688, May 2002
2. Gellersen H, Wicke R, Gaedke M (1997) WebComposition: an object-oriented support system for the Web engineering lifecycle. *J Comput Netw ISDN Syst* 29(8–13):865–1553 [also (1996) In: Proceedings of the sixth international world wide web conference, pp 429–1437]
3. Ginige A (2002) Workshop on web engineering: Web engineering: managing the complexity of Web systems development. In: Proceedings of the 14th international conference on software engineering and knowledge engineering, New York, NY, pp 721–729, Jul 2002
4. Standing C (2002) Methodologies for developing Web applications. *Inf Softw Technol* 44(3): 151–160
5. Collins English Dictionary (2000) HarperCollins
6. Azhar D, Mendes E, Riddle P (2012) A systematic review of web resource estimation. In: Proceedings of promise'12, New York, pp 49–58
7. Taylor MJ, McWilliam J, Forsyth H, Wade S (2002) Methodologies and website development: a survey of practice. *Inf Softw Technol* 44(6):381–391
8. Ricca F, Tonella P (2001) Analysis and testing of Web applications. In: Proceedings of the 23rd international conference on software engineering, pp 25–34
9. Lee SC, Shirani AI (2004) A component based methodology for Web application development. *J Syst Softw* 71(1–2):177–187
10. Fraternali P, Paolini P (2000) Model-driven development of Web applications: the AutoWeb system. *ACM Trans Inform Syst* 18(4):1–35
11. Ginige A, Murugesan S (2001) Web engineering: an introduction. *IEEE Multimed* 8(1):14–18

12. Murugesan S, Deshpande Y (2001) Web engineering, managing diversity and complexity of web application development, Lecture notes in computer science 2016. Springer, Heidelberg
13. IEEE Std. 2001-2002 (2003) Recommended practice for the internet web site engineering, web site management, and web site life cycle. IEEE
14. Christodoulou SP, Zafiris PA, Papatheodorou TS (2000) WWW 2000: the developer's view and a practitioner's approach to web engineering. In: Proceedings of the 2nd ICSE workshop on web engineering, Limerick, pp 75–92
15. Brereton P, Budgen D, Hamilton G (1998) Hypertext: the next maintenance mountain. *Computer* 31(12):49–55
16. Deshpande Y, Hansen S (2001) Web engineering: creating a discipline among disciplines. *IEEE Multimed* 8(2):8–87
17. Offutt J (2002) Quality attributes of Web software applications. *IEEE Softw* 19(2):25–32
18. Reifer DJ (2000) Web development: estimating quick-to-market software. *IEEE Softw* 17(6): 57–64
19. Deshpande Y, Murugesan S, Ginige A, Hansen S, Schwabe D, Gaedke M, White B (2002) Web engineering. *J Web Eng* 1(1):3–17

<http://www.springer.com/978-3-642-54156-8>

Practitioner's Knowledge Representation
A Pathway to Improve Software Effort Estimation

Mendes, E.

2014, XI, 211 p. 84 illus., Hardcover

ISBN: 978-3-642-54156-8