Journal of Web Engineering, Vol. 1, No.1 (2002) 018-022

A PROPOSED CURRICULUM FOR A MASTERS IN WEB ENGINEERING

E. JAMES WHITEHEAD, JR.
University of California, Santa Cruz

Received August 2, 2002

To address the significant technical demand for trained Web Engineers, and to raise the current state of the practice of Web Engineering, a Master's level degree program in Web Engineering is proposed. The most significant research disciplines for Web Engineering are Network Engineering, Software Engineering, Databases and Storage Systems, and Hypermedia. Important aspects of these disciplines are distilled into key knowledge areas for Web Engineering. A curriculum organization is proposed that consists of coursework that covers the key knowledge areas, along with a multi-semester Web design project that synthesizes and applies this knowledge on a real-world Web application. The aim of the paper is to begin a dialog on the characteristics and aims of graduate-level Web Engineering degree programs.

Key words: Web Engineering education, Master of Web Engineering, Web Engineering curriculum
Communicated by: B. White

1 INTRODUCTION

In the span of a decade, the Web has created billions of dollars of wealth, transformed entire industries, and entered the mass culture. It has created new expectations on ease of access, freshness, and relevance of information accessed via the Internet from commercial, governmental, and academic Web sites. Meeting these growing expectations requires the construction of large-scale Web applications, in turn fuelling the demand for engineers who have the necessary skills and background.

As the Web continues to be employed across multiple domains, there is a demand for technical personnel that have a strong *theoretical* background in Web Engineering. Since the Web is a new technology, many people involved in Web Engineering have only a portion of the desired theoretical background. This is a contributing factor for the current problems with Web-based systems development, which has been characterized as frequently being ad-hoc, and lacking "rigour, a systematic approach, and quality control and assurance." [3] While the collapse of the speculative .com bubble has created an oversupply of people with Web Engineering experience, this is a temporary phenomenon. The Web is a relatively new technology, still in the beginning phases of a long period of adoption. As Web technology continues to diffuse into myriad mission-critical functions, there will be a steady and growing need for Web Engineers, and for educational programs tailored to develop skills needed by Web Engineers. These programs will satisfy the demand for trained engineers, as well as raise the state of the practice in Web Engineering.

This paper provides an initial proposal for a graduate, Masters level curriculum in Web Engineering, with the intent of opening a dialog on what ideally should be the components of a Master's degree in Web Engineering. Though this paper is focused on the characteristics of a Masters level curriculum in Web Engineering, undergraduate degree programs in this field are also extremely important, their composition just as worthy of dialog, planning, and implementation.

The paper begins by providing an overview of the key knowledge areas that comprise Web Engineering, followed by a discussion of how this knowledge can be conveyed in the form of courses, and a design project. The paper concludes by raising a set of challenges facing Masters Degree programs in Web Engineering.

2 KEY KNOWLEDGE AREAS IN WEB ENGINEERING

The principles underlying the creation of large-scale Web and Internet Applications derive from the disciplines of Network Engineering, Software Engineering, Databases and Storage Systems, and Hypermedia. As such, we would expect a Web Engineering curriculum to cover the following core set of knowledge areas.

Network Engineering is concerned with the theory, analysis, and construction of computer communication networks. In order to understand the interaction of Web applications with the broader Internet, Web engineers need a solid foundation in networking, encompassing such topics as:

- Physical hardware layer. The types, capabilities, and limitations of the physical devices that propagate computer network signals. Includes such areas as fibber optics, wireless and wired networks, and error control coding.
- Internet layer. The Internet Protocol (IP), and the design and analysis of routing algorithms, including congestion control.
- Transport layer. The design and analysis of reliable transport protocols, such as TCP. Properties of broadcast-style transports, including multicast and session initiation protocols.
- Application layer. Key principles in the design of application layer protocols, such as interaction styles, namespaces, scalability, and separations of concerns among architectural elements. This is complemented by the study of a broad range of significant deployed application layer protocols (e.g., HTTP/DAV, SMTP, POP, SOAP, FTP, LDAP) to understand their design.
- Performance. Analysis of the interactions of protocols operating at several layers and how to design protocols to avoid poor layer interactions. Network traffic analysis for performance tuning.

Software Engineering has as a focus the processes and key practice areas used in the creation of large software systems. Web applications involve significant amounts of software development in the Web server, and frequently in the Web browser as well. Far from being collections of static pages, large Web applications are complex software development activities. However, the large quantities of user-visible content, and the strong emphasis on presentation issues, distinguish Web application engineering from traditional Software Engineering. Software Engineering includes such topics as:

- Process. What is the best sequence of steps for creating a specific software artefact? Involves formal and informal modelling and analysis of software processes, and tailoring of processes to meet specific business needs, such as the rapid schedule of Web application development.
- Requirements. Best practices for the elicitation of system requirements from multiple sources, including customers, standards, and users. Analysis techniques for prioritizing requirements, and determining conflicting requirements.

- Architecture. Modelling of software systems at a high level of abstraction. Analysis of the architecture such that it meets system requirements and achieves strategic objectives. Domain specific architectures, where a software architecture is developed and fine-tuned for a specific use scenario (such as Web applications), are especially relevant for Web engineering because of Web Application Servers, a Web server that has been fine tuned for the domain-specific needs of Web application development. Detailed case studies of several Web application servers can be used to illustrate architectural concepts and tradeoffs.
- Design. Methodologies, such as Object-Oriented Design, that permit a large, loosely structured problem to be decomposed into modules capable of formal description using a programming language.
- Testing. Techniques for exercising a software system to determine whether it meets its requirements. Testing issues specific to Web applications, including load testing, cross-platform testing of dynamic Web pages (e.g., JavaScript across multiple browser platforms), and usability testing.
- Environments. The software tools employed by developers in the construction of software systems. Configuration management systems are an important class of tool, since they provide configuration control over the rapidly changing files that comprise a large development effort.

Databases and Storage Systems, storehouses for large quantities of information, are the foundation of many Web applications. In a typical e-commerce Web site, databases store such items as product information and current inventory status, as well as customer data ranging from name and address to prior shopping history. Deep understanding of the principles and application of database and storage systems allows Web engineers to design their data repositories to scale as the application grows. Database and Storage System principles include:

- Data design, including the entity relationship and semantic data models, and their translation into relational database schemas. Representation of data using tree-based representation schemes, such as XML. Also includes relational algebra, and relational calculus, and optimization of database schemas.
- Database query languages for retrieval of information, including commercial search query languages such as SQL and QBE. Techniques for efficiently querying tree-structured XML data.
- Theory of Internet search engines. Foundations of information retrieval, and recent work on search engine hit ranking. Efficient spidering and representation of Web pages. Information and schema integration from heterogeneous, network-accessible databases.
- Physics of storage devices, including disk, tape, and memory-based systems. Principles of filesystem design, including interactions with the physical storage device. Advanced topics, such as storage area networks, and distributed filesystems.

Hypermedia is concerned with linked information objects. Specifically, hypermedia is interested in the architectural properties of systems (such as the Web) that support hypermedia linking, and hypermedia link traversal, the design of large hypermedia linked corpuses of information (such as a Web site), as well as the rhetorical and narrative properties of specific hypermedia link structures. Hypermedia areas of particular interest for Web engineers include:

- Design. Techniques for structuring and decomposing an information space into individual hypermedia pages. Decomposition of hypermedia pages into more fine-grain objects that can be reused across multiple pages, and their reintegration.
- Visualization. Graphical techniques for conveying an overview image of a Web site.
- Usability. Design and analysis techniques for ensuring a Web site can be efficiently used to accomplish a specific task.
- Collaboration tools. Software tools that allow a hypermedia network to be developed, simultaneously, by multiple people, including those from multiple organizations.

3 CURRICULUM ORGANIZATION

The Web Engineering key knowledge areas given in the previous section outline the scope of knowledge that should be taught in a Master's curriculum in Web Engineering. A fairly straightforward way of teaching this information is in the form of a sequence of courses. One of the advantages of a Web Engineering program is that it can build upon existing courses that are typically offered within Computer Science and Computer Engineering graduate programs. One possible subdivision of material into courses has each of the major disciplines (Network Engineering, Software Engineering, Databases and Storage Systems, and Hypermedia) having 25% of the total coursework dedicated to covering it.

Additionally, since this is an engineering discipline, taking coursework alone is insufficient to ensure that all the knowledge needed to develop an industrial quality Web application can be synthesized and applied. As a result, a large design project, lasting multiple semesters if possible, is an essential part of a Web Engineering curriculum. The design project would involve the construction of a non-trivial, Internet-scale database-backed Web Application. This project would synthesize the information from the coursework, and cause it to be applied on a real project, one that meets the needs of some business unit within the university, or that is suggested by a local government office, or by a local corporation. This project could have associated with it a lecture component that teaches some of the specific tools and technologies used in the development of Web Applications.

In overall organization, this arrangement of coursework plus multi-semester capstone project is very similar to (and is motivated by) existing Masters programs in Software Engineering, such as those at Carnegie-Mellon University [1], and the Oregon Masters in Software Engineering [2].

4 ISSUES FACING A MASTERS OF WEB ENGINEERING

While the rapid technological evolution of the Web platform has led to the significant demand for engineers with a deep background in the skills necessary to build and field large-scale Web applications, this same rapid change presents unique challenges to a Web Engineering degree program.

If the degree program is focused on working professionals who might perform their coursework at night, it might take as much as three years to complete a Masters program. Yet this is inconsistent with such a fast-moving field. True, the knowledge provided by a Masters in Web Engineering would be more theoretical in nature, and wouldn't immediately be irrelevant. The challenge lies in finding some way to expedite the delivery of the Master's degree so it doesn't take three years. Courses offered on the weekends, or in intensive bursts of 1-2 weeks, might be one way of addressing this. Of course, the program could be offered full-time, and then it would be possible to finish in a year. However, this would undoubtedly exclude a large number of working professionals who could not afford to forgo their current salary for a year.

Another issue facing the degree program is how best to incorporate some of the content generation skills that are also an important aspect of designing a Web site. Graphic arts skills are very important in determining the graphic design and layout of Web pages, and the graphic design of a site can have a significant impact on its success, and its usability. As a consequence, graphic arts seem to have a place within a Web Engineering curriculum, and should belong within the set of key knowledge areas. However, this raises problems as well. It is uncommon to find graduate-level courses introducing graphic design to people with technical backgrounds. Introductory courses in the graphic arts tend to be found at the undergraduate level, yet an undergraduate course is not appropriate for a Masters curriculum. As well, adding an arts course to a technical Masters, one that presumably would be offered out of a School of Engineering, or School of Computer Science, is bound to raise questions by other faculty concerning its role in the curriculum. None of these problems seem insurmountable: a special graphic design course could be created for this degree, and might fit well in electronic media/arts departments or research groups. Additionally, the content-driven nature of Web development provides a ready rationale for the inclusion of graphic arts into the curriculum.

4 SUMMARY

This paper has presented the outline of a degree program in Web Engineering, giving a series of key knowledge areas that should be covered by such a program, and a proposed organization of the program as a set of courses and a multi-semester design project. It is hoped that this paper will stimulate further discussion on what a graduate-level Web Engineering degree program should cover, and how it should be structured.

ACKNOWEDGEMENTS

A prior version of this paper appeared in Proceedings of the Fourth Workshop on Web Engineering, held with the 10th International World Wide Web Conference, in May, 2001. I would like to thank Philip Greenspun for his feedback on this proposal.

REFERENCES

[1] Carnegie Mellon University, “Master of Software Engineering Degree Requirements,” (2001). Web page, accessed February 21, 2001. http://www.cs.cmu.edu/afs/cs.cmu.edu/project/mse/www/information/description.html.

[2] S. Faulk, “Achieving industrial relevance with academic excellence: lessons from the Oregon Master of Software Engineering,” *Proc. ICSE 2000*, 2000, pp. 293-302.

[3] S. Murugesan, Y. Deshpande, S. Hansen, and A. Ginige, “Web Engineering: A New Discipline for Development of Web-based Systems,” *Proc. First ICSE Workshop on Web Engineering*, Los Angeles, CA, May 16-17, 1999.