

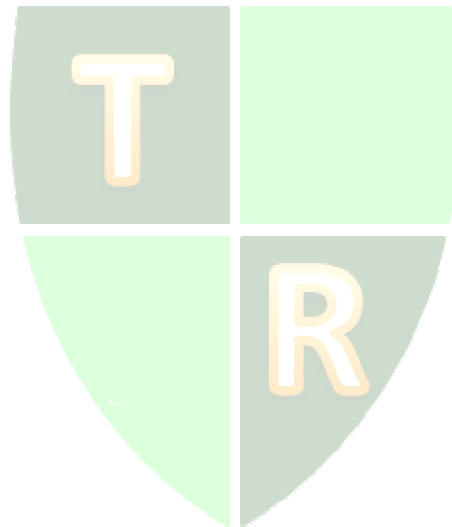
Methodologies, tools, and techniques in practice for Web application development

Monica Lam
California State University, Sacramento

Abstract:

This exploratory study investigates what methodologies, tools, and techniques are used by Web developers. The survey results from over 300 Web developers in USA show that RAP, XP, WSDL, and WebML are used by practitioners. The PCA indicates that WebML is uniquely associated with HIPO Chart, WAE, Decision Table, Program Flowchart, Deployment Diagram, and Pseudocode for Web application development. Guidelines for determining when to use what methodology are needed.

Keywords: Methodologies, Web Application Development, Explorative Study, WebML



INTRODUCTION

Web presence is nowadays essential for business operation, marketing, and strategy. There is no doubt that the majority of information systems to be developed in the future will be Web-based even for internal purposes. Strategic collection and utilization of information via different kinds of Web-based applications can determine whether a business will be successful. This study defines a Web application as a software system that relies on the Web as its interaction medium with the end-users to create, exchange, and modify data for transaction requirements. As Web applications have become more and more important, how to ensure the success of their development is an issue of interest and practical value to practitioners, educators, and researchers. Though Web applications fall under the umbrella of software systems, they are unique in terms of user recognition, user environment, communication control, testing requirement, security issues, interface requirements, feedback mechanism, functionality design, and life cycle.

Methodologies for application development are step-by-step procedures to carry out the development activities in different phases of a system development life cycle. A methodology has its own assumptions about the reality, its own techniques to support working principles and enforce discipline, and its own tools to generate the deliverables for activities. In other words, there are a collection of specific techniques and tools for a certain development methodology. A search of the literature reveals that although there are many suggested methodologies, tools, and techniques for Web application development, they have not been consolidated into a few proven approaches for Web developers. What are the methodologies being used by practitioners to develop their Web applications when there are no solid guidelines? Are practitioners using some heuristic methods developed by themselves, some modified methodologies from the literature, or not using any methodologies at all? If practitioners are using some methodologies for Web application development, are there any problems? What are those problems? What do practitioners need in order to solve those problems? This explorative survey study attempts to answer the above questions.

LITERATURE BACKGROUND

The literature of Web application development is mainly divided into the two groups of technology focus and empirical study focus. For technology focus, Conallen (1999, 2003) discusses modeling Web application using Web Application Extension (WAE) to Unified Modeling Language (Booch 1999). Web Modeling Language (WebML) is developed as a graphic notation and a textual XML syntax for specifying complex Web sites at the conceptual level (Ceri 2000). Chen and Heath (2001) present a modified prototyping method (MPM) that captures Web application components, client-side processing, and server-side processing to address scalability, load balancing, security, maintainability, interfacing with legacy systems, and fast development. Standing (2002) proposes the Internet Commerce Development Methodology (ICDM) that provides a management strategy and a development strategy covering meta-development, user involvement, site and component development, requirements analysis, functional framework, physical architecture, and evolution. Avison and Fitzgerald (2003) classify system methodologies into pre-methodology era (no methodology), early methodology era (waterfall life cycle and flowchart), methodology era (structured design, entity-relationship diagram, prototyping, CASE), and post-methodology era. We are now in the post-methodology

era where methodologies are ad-hoc and trial-and-error based on developers' personal knowledge and experience. Object-oriented technology and application frameworks are the dominant programming concepts and tools used by Web developers nowadays.

For the empirical study focus, Taylor et al. (2001) classify the skill sets required for Web site development into the three overlapping sets of technical, business, and analytical based on cases studies in twenty UK organizations. Kautz et al. (2004) identify the following characteristics of methodology utilization from three different projects in a major Danish software development company. First, there is no universally applicable methodology for system development. Due to the lack of knowledge or time, developers are usually forced to begin the actual physical modeling very early in the development process without taking advantage of conceptual and logical analysis advocated by methodology. Second, methodologies are used symbolically to provide comfort and confidence to developers and end-users. Third, developers prefer incremental methodologies to sequentially organized methodologies. As functional requirements are guaranteed to change during the development process, a formal method that iterates is necessary. Fourth, methodology adoption depends on management support, explicit adoption, and cooperation agreement among all parties. Lang and Fitzgerald (2006), in a survey of 164 companies, identify the hybrid, customized, or proprietary in-house method or approach as the most adopted methodology in Web/hypermedia system design (23%), followed by traditional SDLC (22%), and rapid or agile development methods (15%).

In terms of development techniques, screen prototypes (97%), flowcharts (95%), 2D site mapping techniques (91%), storyboards (85%), and entity-relationship diagrams (74%) are the top five techniques used by Web developers. Masrek et al. (2008), in a survey of 66 companies in Malaysia, found that the mostly adopted development methodologies for Web applications are in-house methodology (48.48%), rapid application development (36.36%), and Unified Modeling Language (34.85%). The most popular techniques used are data flow diagrams (59.09%), flowcharting (57.58%), prototyping (50%), entity-relationship diagrams (48.48%), and project management (42.42%). Smolander and Rossi (2008), based on a case study, conclude that the benefit level provided by UML for enterprise-wide e-business architecture is medium for technical and language requirements but low for organizational requirements.

Overall the literature has provided some background information regarding what practitioners are using for methodologies, tools, and techniques. However, the results from available studies are limited to case study, small sample size, non-USA company basis, or simple ranking analysis. This explorative study focuses on the importance of methodologies, tools, and techniques for the success of Web application development as well as their relationship using a large sample of over 300 respondents in USA. Comments are also collected to address the deficiency and problems of existing methodologies as well as the requisite skills and knowledge for Web developers.

SURVEY INFORMATION

A preliminary questionnaire was developed and submitted to ten Web developers for pretest. Feedback from pretest was used to revise the questionnaire in terms of terminology clarification, format, ordering of questions, and addition of questions. The survey was administered by an Internet research company to a sample of 1500 contacts in a period of 4 weeks. There were one initial invitation email and one follow-up reminder email to contacts. To

provide incentive for participation, for each completed survey, a donation of \$10.00 is made to a charity organization of the respondent's choice. The initial collection of responses was filtered using a reliability test based on respondents' answers to multiple pairs of variables measuring the same concepts. If a respondent's answers to those pairs of variables differ more than 3 points in a scale of 1-7, that respondent is deleted from the final data set for analyses. The reliability test generated a total of 312 valid responses for analyses. The valid response rate is 21% of the survey sample. Appendixes I - III have the main questions for methodologies, development phases, and tools and techniques in this survey.

Respondents were asked to evaluate the significance (in a scale of 1-7 from not significant at all to very significant) of methodologies, development phases, and tools and techniques for Web application. Tables 1 - 5 respectively show the company type, number of employees, annual sales, job title of the respondent, and the number of Web applications developed by the company in the past 3 years. The remaining of this paper includes the discussion of evaluation results of the significance of methodologies, development phases, and techniques; the discussion of principle component analysis results for the winning combination of methodologies, development phases, and development techniques for Web application; summary of open-ended comments from respondents; and finally the conclusion with recommendation.

Evaluation Result of Significance of Methodologies, Development Phases, and Techniques

Tables 6 – 8 list the percentages of "very significant" for the success of Web application development for methodologies, development phases, and tools and techniques respectively. In terms of methodologies, the highest is rapid application prototyping (12.8%), followed by extreme programming (6.7%), waterfall system development life cycle (3.5%), Web modeling language (2.9%), and Rational Unified Process (2.6%). In addition to the rapid application prototyping and waterfall system development life cycle as identified in earlier studies (Fitzgerald, 2006), this survey shows that extreme programming and Web Modeling Language (Conallen 2003) have emerged as very significant methodologies for Web application development. As the development life cycle of Web applications is shorter than other applications, extreme programming as an agile method can react quickly to requirement changes and verify assumptions. Web Modeling Language is especially designed for developing Web applications. Though the learning curve of Web Modeling Language can be high, it is a comprehensive tool to document functionality and communication for Web applications. Though waterfall system development life cycle was developed well before the Web age, it remains to be highly applicable especially for complex and mission critical system.

For development phases, the top five are testing (49.8%), functionality requirements, (40.1%), launch (32.3%), application coding (29.4%), and code review (29.4%). It seems that the basic development phases such as testing and application coding are more significant than the Web-application specific phases such as page layout design, component design, and Web service design. This tells us that the core phases to be mastered for all kinds of application developments are still the basic ones that every application must have.

For tools and techniques, the top five are application framework (21.3%), review/staging Web site for communication purposes (18.9%), workflow analysis (16.8%), entity relationship diagrams (12.9%), and story boarding (12.4%). Entity relationship diagrams (ERD) is still

among the popular development tools in the Web age, which validates the classical value of this database design tool. Application framework has become an integrative platform for developers to carry out different phases of system development. Story boarding is a new tool that is important for documenting functionality in Web applications. On the other hand, various UML diagram tools such as object diagrams and class diagrams are not considered as very important.

Overall, Web application developers are using new methodologies such as WebML and extreme programming from the past decade to facilitate web application development. Basic development phases such as testing, requirement analysis, and launch are still essential. While classical tools such as ERD are still commonly embraced, integrated tools such as application framework and project management software are indispensable.

Results from Principle Component Analysis

Principal component analysis is next applied to group methodologies, development phases, and development techniques into different factors, which answer the question of whether some combinations of methodologies, development stages, and tools and techniques are critical success factors for Web application development. Principle Component Analysis (PCA) is a factor analysis method for exploratory study (Meyers et al. 2006).

Table 9 shows the total variance of the concept of success of Web application development explained by principle component analysis using the rotation method of Varimax with Kaiser Normalization in SPSS. The selection criteria for components are: (1) the eigenvalue of a component is greater than 1, and (2) the total variance of all components is greater than 50%. We identify 11 components according to the above selection criteria. Table 10, the rotated component matrix, shows the correlation of variables with the 11 extracted components. A variable is included in a component if its correlation with that component is greater than 0.5. Note that the label of ST means development tool and technique, SM means methodology, and SP means development phase. Please refer to Tables 6-8 for specific meaning of each variable.

Table 11 provides the interpretation of the 11 extracted components. Figure 1 is a graphical representation of the 11 components contributing to successful Web application development. A very unique component is component #2, which is named as the methodology and tools for structure and processing of the entire Web application. Component #2 is unique because it is the only component that has a methodology (SM4) as a contributing variable. None of the other methodologies is captured in any other component. Among practitioners, SM4 tends to be associated more with Hierarch-Input-Process-Output (HIPO) Chart, Web Application extension (WAE) to Unified Modeling Language (UML), Decision Table, Program Flowchart, Deployment Diagram, and Pseudocode. The variables in Component #2 represent a mixture of new (WAE, Deployment Diagram) and old (HIPO, Decision Table, Program Flowchart, Pseudocode) development tools and techniques governed by the overarching methodology of WebML. This is a unique finding that can be investigated more in future research. While a few new development tools and techniques are useful, old tools and techniques still provide important utility to developers. WebML seems to be the winning methodology developers are embracing to guide and integrate Web development activities.

Component #1 is diagrammatic tools for object structure, object behavior, and process interaction including Class Diagram, Sequence Diagram, Activity Diagram, Object Diagram,

StateChart Diagram, Use Case Diagram, Component Diagram, Collaboration Diagram, and Entity-Relationship Diagram.

Component #3 is tools for communication and productivity including Periodic/Standardized Progress Report, Review/Staging Web Site for Communication Purposes, Project Management Software, and Diagram Generation Software. The first three components form the development core consisting of WebML, diagrammatic tools for objects, logic, and processes, and tools for communication and productivity in Figure 1.

Development phases that contribute to the concept of Web application development success include the phase of designing pages and their relationship (component #4), the phase of quality control (component #5), the phase of before and after development (component #6), the phase of data, technical specification, and functional specification (component #7), and the phase of integrating external operations (component #8). Overall, development phases that address page design, data design, pre-development, launch, and integration are important to Web developers. Components #9 - #11 are standalone methodologies including RAP, WSDL, and XP, which are being used by Web developers to different extent to guide and control the development cycle. In Figure 1, the development phases and standalone methodologies are factors interacting with the three core circles of tools and techniques to generate Web applications.

Respondent Comments

Because of the explorative nature of the study, this survey has open-ended questions for respondents to provide comments in addition to the structured questions. Respondents have taken a lot of effort to provide a wealth of information regarding the problems and deficiencies of existing methodologies as well as requisite knowledge and skills for successful Web application development. This section summarizes and discusses respondents' valuable input as follows.

Problems and Deficiencies of Methodologies, Tools, and Techniques for Web Application Development

Methodology Overhead vs. Development Parameters

Formal methodologies tend to be heavy in documentation and labor intensive. Developers need to evaluate whether it is feasible in terms of time and budget to apply a certain methodology. For small to medium projects which are not mission critical, the cost of errors is usually low. In those scenarios, when meeting timeline for marketing purposes is more important than 100% error free, methodological procedures such as validating functional specification, documentation, and quality control are usually sacrificed. When there are competing objectives, methodology has a low priority. Moreover, due to a lack of understanding regarding complex methodologies, developers cannot properly apply them to produce the desired outcome. When one step of a methodology is not applied correctly, the domino effect can lead to inconsistency and erroneous results. The perception of the incapability of methodologies to produce desirable results may be due to the misapplication or operator errors rather than the innate flaws of methodologies.

Developer Quality vs. Methodology Merit

One respondent's feedback: "Regardless of which methodology you use, you still need good people, good communication, succinct requirements, reasonable deadlines, adequate funding, group-wide discipline, and a lot of luck." To realize the benefits provided by methodology, we need developers who understand how it works, can communicate it to the entire team, and have the discipline to maintain consistency and surface problems. Whereas competent developers can succeed without formal methodology, methodology cannot shine in the hands of mediocre developers. Yet a suitable methodology can facilitate and support good developers' planning, goal setting, resource acquisition, and negotiation.

Top-Down vs. Bottom-Up

Some organizations do not apply formal methodology because top management does not require them. When top management does not impose standards, it is difficult for project managers to enforce methodology across different developers in a project. In an environment of no standard, developers generally use what they know to accomplish tasks assigned to them in the shortest period of time without much consideration for high level functionality and long-term maintainability. In addition to standard enforcement, top management has to allocate funding to provide training for developers so that all can be on the same page for communication and problem solving.

Speed, Accuracy, and Efficacy of Methodologies

The number one complaint from developers is the ambiguous requirement statement and constant requirement changes from end-users. Existing methodology does not provide an interactive tool that can define functionality in an understandable format for end-users to review, revise, and approve. End-users demand functionality faster than any methodology can efficiently document, verify, and prototype. Scope creeping has become the major failure factor for system development because of the lack of methodology to control it. It is imperative that top management has a policy to lock down requirement specifications after a reasonable period of iterative design and validation. Impact analyses for requirement changes are necessary for evaluating whether to accept requirement changes from end-users. At the end, it is a trade-off between speed and quality. Top management should guide prioritization and resolve power struggle among key players in a project. For methodology to be realistically applicable and useful, in addition to technological support, methodology needs to provide procedure and methods to really simplify the innate complexity of system development, achieve effective communication among team members, resolve conflicts between end-user requirements and developer constraints, and align all activities toward the same goals of the organization.

Emerging Methodologies for Web Applications

Web applications are subject to changing requirements more frequently than other systems. The development cycle of Web applications is also shorter and more volatile than others. Is there a need for Web application development methodology? Judging from respondents' comments from this survey, the answer is a definitely yes. Methodology is needed

for maintainability, scalability, and upgrading for Web applications, especially for large-scale and mission critical projects. For those projects, it seems that developers fall back to the Waterfall Development Life Cycle for its reliability and consistency, supplemented by ad-hoc rapid prototyping as demanded by situations to provide a review document to end-users. As one respondent suggested, the ideal methodology is the one that can guide when to switch to another methodology or tool during the development cycle, which generates a hybrid approach consisting of the most appropriate tools and techniques from different methodologies.

Emerging Development Phases and Tools for Web Applications

Considering that Web applications frequently use third-party Web services and components, rely on off-shore contracting, and consist of multi-company projects, there is the need for development phases and tools to test multi-tier interoperability and integration, and to ensure security among all system parts in all possible platforms by all possible users. Peer evaluation and end-user in-progress evaluation can be instrumental to surface problems as early as possible. Feedback from Web application users is constant and fast due to the Web's operation characteristics. The maintenance phase of Web application development is continuous and compressed, which requires tools for quick diagnosis and delivery for problem solving.

Requisite Knowledge and Skills for Successful Web Application Development

One striking comment regarding requisite knowledge and skills for successful Web application development from respondents is developers' soft skills including listening, ethics, critical/analytical/logical thinking, verbal/written communication, interpersonal/diplomatic maneuver, leadership, desire to learn and understand, time management, understanding of and adaptability to different stakeholders, and ability to align and grow with organizational mandates and objectives. In terms of technical skills and knowledge, developers need to know front-end, back-end, database, and architecture conceptualization. Front-end expertise includes programming (defensive, extreme), scripting, prototyping, user-interface layout design, methodologies, diagrammatic tools, and framework. Back-end expertise includes server-side programming, enterprise level development tools, multi-tier integration and testing, collaboration tools, and version control. As Web sites today are database-driven, knowledge of database and their connectivity, manipulation, and maintenance are critical. Architecture and infrastructural knowledge for multi-platform, multi-system, and multi-organization systems consisting of third-party components and Web services are also essential. As methodologies, tools, and techniques are changing and volatile, it is more important for developers to understand their function and objectives than to learn all of them. As there can be multiple ways to achieve an objective, understanding of the objective can provide flexibility to developers to evaluate new methodologies and tools and their values. Whereas technical expertise can get the job done, soft and people skills make it successful. Confidence, trust, and buy-in from end-users especially power end-users are critical for successful development and implementation of Web applications. This is a very good reminder for what to be included in curricula for Web developer training. It is usually not the methodology or the tool that fails but the people who fail it.

CONCLUSION AND RECOMMENDATION

The survey results from this exploratory study show that RAP, XP, WSDL, and WebML are used by practitioners to different extent depending on the complexity, scope, and nature of the system. New methodologies such as XP tend to be used for smaller and less complicated system. In terms of development phases, testing, requirement analysis, coding, and review are important. In terms of development tools and techniques, application framework, communication Web site, workflow analysis, and RED are important. WebML is uniquely associated with HIPO Chart, WAE, Decision Table, Program Flowchart, Deployment Diagram, and Pseudocode. This unique association seems to show that a mixture of new and old techniques is necessary to satisfy all different activity needs during the development life cycle for Web application. Respondents' comments support the above statistical finding that there is the need to use a hybrid model of methodology, tools, techniques, and development phases to support a variety of activities. If practitioners use a certain methodology, they usually supplement it with in-house tools or tools developed for other methodologies.

Respondents identify scope creeping, changing user requirements, lack of development standards supported by top management, fixed and complicated methodologies, user errors, lack of people and diplomatic skills, lack of effective communication tools to interact with users and development team members, lack of common objectives, and lack of time as the common development problems. There are a few suggestions to help with Web application development problems. First, practitioners need guidelines for determining when to use what methodologies and tools. As different methodologies and tools have their advantages and disadvantages, the knowledge of when to use what will reduce development time and cost. Under the hybrid model, practitioners also need to know when to switch to another methodology to achieve the optimal performance. Second, to communicate with different stakeholders, practitioners need to have effective communication tools. They especially need a tool that can capture end user requirements in an understandable, correct, and technically sound format. That communication tool can allow end users to revise functional requirements and also see the associated cost. Third, top management has to set and enforce objectives, settle conflicts, and enforce standards. Fourth, we need to train developers not only with technical skills but also people skills. Fifth, integration for third-party components and Web services, architectural support, and data standards require careful planning and consolidation.

References:

- Avison, D.E., Fitzgerald, G. (2003). Where Now for Development Methodologies? *Communications of the ACM*, 46(1), 79-82.
- Booch, G., Rumbaugh, J., & Jacobson, I. (1999). *The Unified Modeling Language user Guide*. Reading, MA: Addison Wesley Longman.
- Ceri, S., Fraternali, P., & Bongio, A. (2000). Web Modeling Language (WebML): A Modeling Language for Designing Web Sites. *Computer Networks*, 33(1-6), 137-159.
- Chen, J.Q. & Heath, R.D. (2001). Building Web Applications: Challenges, Architectures, and Methods. *Information Systems Management*, 18(1), 68-79.
- Conallen, J. (2003). *Building Web Applications with UML*. Boston, MA: Pearson Education.
- Conallen, J. (1999). Modeling Web Application Architectures with UML. *Communications of the ACM*, 42(10), 63-70.
- Kautz, K., Hansen, B., & Jacobsen, D. (2004). The Utilization of Information Systems Development Methodologies in Practice. *Journal of Information Technology Cases and Applications*, 6(4), 1-20.
- Lang, M. & Fitzgerald, B. (2006). New Branches, Old Roots: A Study of Methods and Techniques in Web/Hypermedia Systems Design. *Information Systems Management*, 23(3), 62-74.
- Masrek, M.N., Hussin, N., & Tarmuchi, N. (2008). An Exploratory Study on Systems Development Methodologies for Web-Based Applications. *Information Management & Computer Security*, 16(2), 137-149.
- Meyers, L.S., Gamst, G., & Guarino, A.J. (2006). *Applied Multivariate Research: Design and Interpretation*. Thousand Oaks, CA: Sage Publications.
- Smolander, K. & Rossi, M. (2008). Conflicts, Compromises, and Political decisions: Methodological challenges of Enterprise-Wide E-Business Architecture Creation. *Journal of Database Management*, 19(1), 19-40.
- Standing, C. (2002). Methodologies for Developing Web Applications. *Information and Software Technology*, 44, 151-159.
- Taylor, M.J., Ebngland, D., & Gresty, D. (2001). Knowledge for Web Site Development. *Internet Research*, 11(5), 451-461.

Table 1. Respondent Company Type

Company Type	Percentage
Multi-National Company	22.43
Public Limited Company	7.22
Small/Medium Enterprise	54.75
Federal/State Government Type	2.66
Others	12.93

Table 2. Number of Employees in Respondent Company

Number of Employees	Percentage
Under 500	62.74
500 – less than 1000	10.65
1,000 – less than 10,000	17.11
10,000 – less than 50,000	6.08
50,000 – less than 100,000	1.52
100,000 or more	1.90

Table 3. Annual Sales in Respondent Company

Annual Sales	Percentage
Under 10 m	31.18
10 – less than 100 m	27.38
100 – less than 1,000 m	12.55
1,000 – less than 10,000 m	4.56
10,000 – less than 50,000 m	3.04
50,000 – less than 100,000 m	3.04
100,000 m or more	18.25

Table 4. Job Title of Respondent

Job Title	Percentage
Computing Infrastructure Manager	1.14
Database Administrator	0.38
Editor/Copywriter	0
Graphic Designer	0.76
HTML Developer	1.52
Information Architect	1.90
IT Executive (CIO, VP, Director)	71.10
Network Manager	0.38
Producer	0
Project Manager	9.13
Project Stakeholder/Client/Business Owner	3.42
Programmer/Code Writer	4.18
Quality Assurance Engineer	1.14
Tech Lead	4.94

Table 5. Number of Web Applications Developed by Respondent Company in the Past Three Years

Number of Web Applications	Percentage
0 – 1	4.94
2 – 5	24.71
6 – 10	29.66
11 – 15	11.41
16 – 20	7.98
21 or more	21.29

Table 6. The Percentage of "Very Significant" for Methodologies

Item	Methodology	Percentage
SM3	Rapid Application Prototyping	12.8
SM2	Extreme Programming	6.7
SM5	Waterfall System Development Life Cycle	3.5
SM4	WebML (Web Modeling Language)	2.9
SM1	Rational Unified Process	2.6
SM6	Compuware's UNIFACE	0.3

Table 7. The Percentage of "Very Significant" for Development Phases

Item	Development Phase	Percentage
SP18	Testing	49.8
SP5	Functionality Requirements	40.1
SP19	Launch	32.3
SP15	Kickoff Meeting to Review Functional and Technical Specifications	29.4
SP16	Application Coding	29.4
SP14	Technical Specifications	28.3
SP7	Operations and Business Process Design	26.5
SP1	Creative Brief/Concept Creation	25.8
SP4	Generation of Project Plan: Mission, Objectives, Targeted Users, Scope, Budget, Web Teams	24.4
SP9	Presentation/Page Layout Design	19
SP17	Code Review	19
SP6	Data Storage and Access Design	18.6
SP8	Navigation Design	18.6
SP13	Infrastructure Configuration	17.9
SP2	Functional/Technical/Operational Feasibility Studies	15.8
SP12	Component Design	15.4
SP11	Web Service Design	15.1
SP10	Page Communication/Relationship	12.5
SP3	Cost/Benefit Analysis	9

Table 8. The Percentage of "Very Significant" for Tools and Techniques

Item	Tool/Technique	Percentage
ST23	Application Framework	21.3
ST18	Review/Staging Web Site for Communication Purposes	18.9
ST17	Workflow Analysis	16.9
ST1	Entity Relationship Diagrams (ERD)	12.9
ST2	Story Boarding	12.4
ST19	Periodic and standardized Progress Reports	12.4
ST20	Project Management Software	12
ST22	Code Generation/Review/Testing Software	11.6
ST13	Program Flowcharts	10.4
ST3	Use Case Diagrams	8.4
ST5	Object Diagrams	6.8
ST4	Class Diagrams	6
ST14	Decision Tables	6
ST6	Sequence Diagrams	5.6
ST10	Component Diagrams	5.6
ST11	Deployment Diagrams	5.6
ST9	Activity Diagrams	5.2
ST16	Pseudocode	5.2
ST21	Diagram Generation Software	4.8
ST15	Hierarchy-Input-Process-Output Charts (HIPO)	4.4
ST12	Web Application Extension to Unified Modeling Language	3.6
ST7	Collaboration Diagrams	3.2
ST8	Statechart Diagrams	2.4

Table 9. Total Variance Explained by Principal Component Analysis with Rotation

Component	Eigenvalue	% of Variance	Cumulative %
1	6.34	13.20	13.20
2	4.59	9.56	22.76
3	3.09	6.43	29.19
4	2.55	5.30	34.50
5	2.48	5.17	39.67
6	2.42	5.05	44.72
7	2.25	4.69	49.41
8	2.22	4.62	54.03
9	1.83	3.81	57.84
10	1.63	3.39	61.24
11	1.53	3.18	64.42

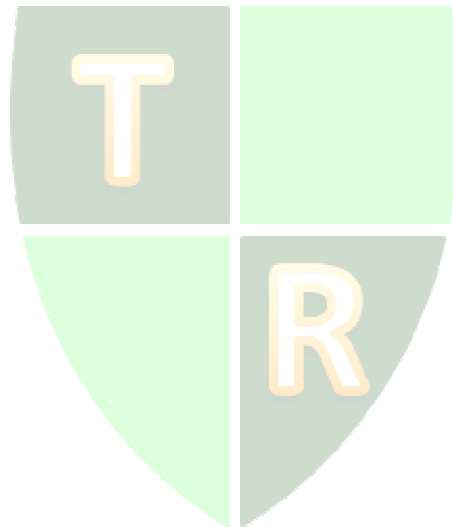


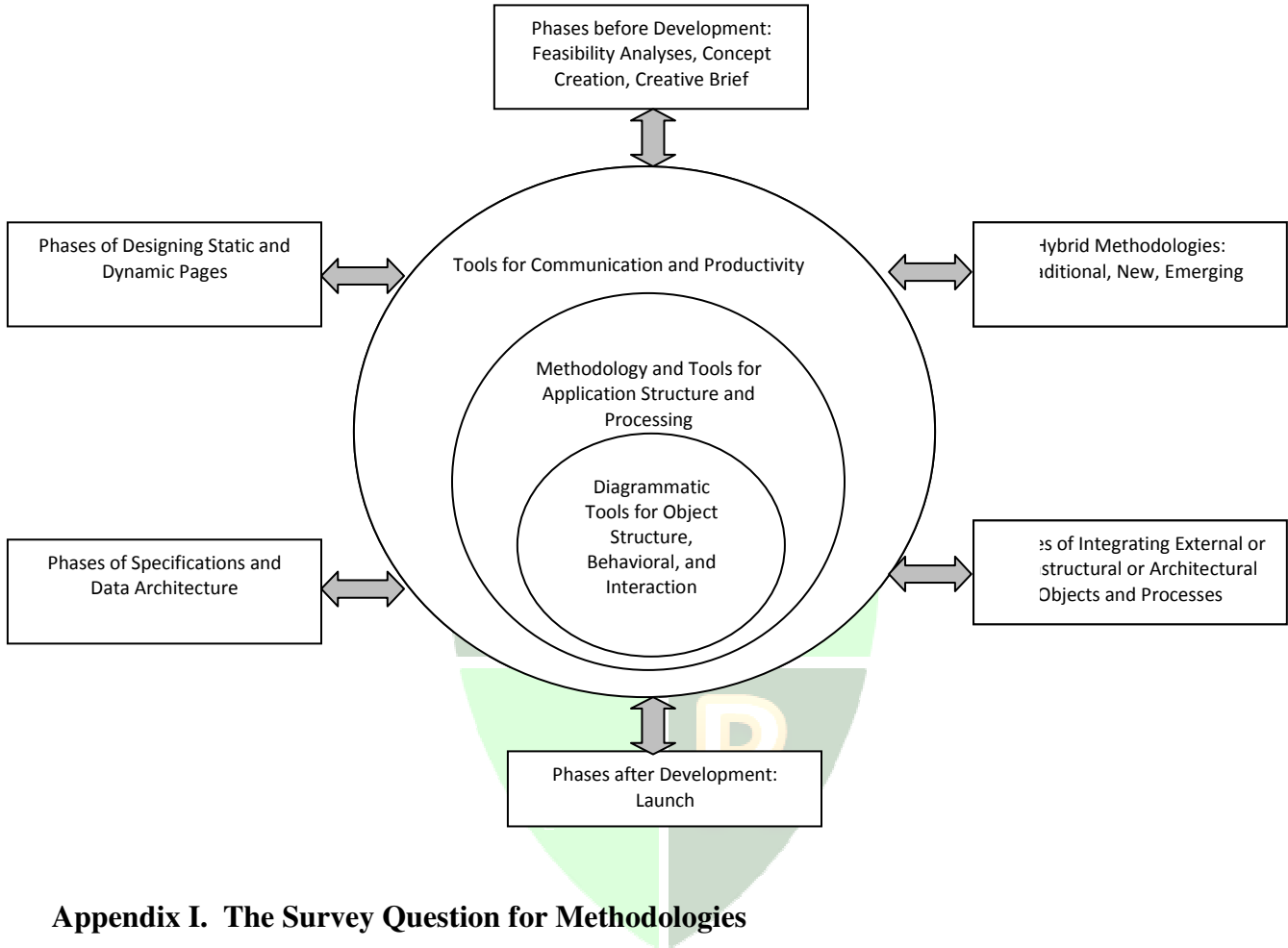
Table 10. Rotated Component Matrix

	1	2	3	4	5	6	7	8	9	10	11
ST4	0.75	0.19	0.05	0.10	0.17	0.06	0.03	0.16	0.19	0.19	-0.01
ST6	0.75	0.23	0.10	-0.02	0.11	0.08	0.14	0.10	0.02	0.08	0.10
ST9	0.70	0.27	-0.01	0.01	-0.12	0.12	0.19	0.14	0.10	-0.12	0.10
ST5	0.70	0.25	0.09	0.10	0.20	0.04	0.04	0.17	0.16	0.18	-0.07
ST8	0.69	0.28	0.17	-0.02	0.00	0.11	-0.02	0.01	0.01	0.08	0.22
ST3	0.67	0.03	0.29	0.17	0.15	0.03	0.07	0.00	-0.06	0.14	0.08
ST10	0.63	0.44	0.07	0.05	0.01	0.10	0.14	0.11	0.13	-0.09	-0.07
ST7	0.63	0.40	0.21	-0.05	0.04	0.20	0.11	0.05	-0.08	0.06	0.17
ST1	0.57	0.07	0.10	0.18	0.03	0.09	0.04	0.08	0.35	0.12	0.00
ST15	0.26	0.80	0.17	0.05	0.08	0.09	-0.01	0.01	0.10	0.04	0.06
ST12	0.29	0.72	0.13	0.07	0.02	0.10	-0.04	0.00	0.02	0.09	0.18
ST14	0.33	0.69	0.17	0.08	0.06	0.11	0.13	0.11	0.10	-0.09	-0.05
SM4	0.22	0.61	-0.03	0.16	0.02	0.19	-0.02	0.08	0.02	0.27	0.27
ST13	0.27	0.59	0.12	0.01	0.02	0.05	0.24	0.14	0.12	0.04	-0.35
ST11	0.54	0.57	0.13	0.11	0.06	0.10	0.15	0.09	-0.04	-0.18	0.03
ST16	0.23	0.52	0.11	-0.05	0.03	-0.05	0.01	0.03	0.30	0.21	0.02
ST19	0.14	0.16	0.75	0.16	0.09	0.11	0.02	0.16	-0.04	0.00	0.02
ST18	0.07	-0.03	0.66	-0.01	0.07	0.03	0.34	0.04	0.25	-0.08	0.15
ST20	0.18	0.28	0.60	0.02	-0.01	0.14	0.08	0.16	-0.09	0.11	0.04
ST21	0.49	0.29	0.51	0.07	0.06	0.00	-0.01	0.07	0.11	0.10	-0.02
ST23	0.36	0.04	0.48	-0.04	0.16	-0.01	-0.05	0.15	0.38	0.01	0.09
ST17	0.20	0.34	0.45	-0.01	0.13	0.27	0.15	0.06	0.36	0.01	-0.12
ST22	0.35	0.30	0.44	0.09	0.22	-0.20	-0.30	0.26	0.07	0.13	-0.09
SP9	0.16	0.01	0.03	0.83	0.18	0.01	0.23	0.03	0.05	-0.02	0.08
SP8	0.03	0.11	0.03	0.80	0.15	0.15	0.10	0.06	0.18	0.00	-0.04
SP10	0.06	0.09	0.13	0.76	0.09	0.08	0.03	0.34	0.00	0.10	0.00
SP18	0.08	-0.08	0.08	0.17	0.70	0.21	0.11	0.08	0.04	-0.03	-0.06
SP16	0.07	0.02	-0.05	0.07	0.68	-0.15	0.28	0.16	0.13	0.02	0.11
SP17	0.17	0.20	0.19	0.17	0.67	0.07	0.10	0.24	0.03	0.14	-0.03
SP2	0.25	0.14	0.08	0.04	-0.03	0.69	0.18	0.11	0.11	0.03	-0.03
SP3	0.08	0.11	0.01	0.04	0.03	0.68	-0.09	0.27	-0.03	0.23	0.01
SP19	0.06	0.16	0.19	0.15	0.41	0.52	0.06	0.07	-0.09	-0.18	0.15
SP1	0.08	0.06	0.02	0.17	0.34	0.52	0.17	-0.08	0.28	-0.24	0.06
SP4	0.18	0.09	0.29	0.19	0.04	0.43	0.33	-0.09	-0.21	0.27	-0.22
SP5	0.08	0.00	0.17	0.23	0.20	0.08	0.70	0.06	0.09	0.14	-0.08
SP14	0.24	0.13	0.07	0.05	0.23	0.08	0.57	0.22	-0.06	0.10	-0.14
SP6	0.12	0.13	-0.04	0.17	0.17	0.07	0.54	0.26	0.16	0.02	0.21
SP15	0.11	0.04	0.24	0.11	0.41	0.22	0.42	0.02	-0.20	-0.09	0.00
SP11	0.15	0.07	0.16	0.34	0.15	0.15	0.11	0.70	-0.15	0.03	0.13
SP12	0.25	0.09	0.25	0.12	0.20	0.26	0.13	0.67	0.18	-0.03	-0.07
SP13	0.22	0.11	0.18	0.09	0.23	0.07	0.28	0.62	0.02	-0.02	0.06
SM3	0.20	0.18	0.01	0.17	0.00	0.03	-0.01	-0.02	0.61	0.02	0.26
SP7	0.02	0.21	0.19	0.10	0.12	0.35	0.21	0.33	0.43	0.12	-0.23
ST2	0.37	0.19	0.23	0.12	0.13	-0.01	0.13	-0.18	0.40	-0.11	0.04
SM5	0.20	0.07	0.03	0.05	0.00	0.04	0.13	0.00	0.04	0.76	-0.05
SM6	0.06	0.44	0.04	0.03	-0.08	0.08	0.04	0.07	-0.01	0.49	0.44
SM1	0.38	0.25	0.09	-0.05	0.19	0.21	0.02	-0.02	-0.03	0.38	0.38
SM2	0.30	0.09	0.15	0.04	0.04	-0.06	-0.03	0.09	0.26	-0.06	0.68

Table 11. Interpretation of Components

Component	Variables	Interpretation
1	(1) ST4 – Class Diagram (2) ST6 – Sequence Diagram (3) ST9 – Activity Diagram (4) ST5 – Object Diagram (5) ST8 – Statechart Diagram (6) ST3 – Use Case Diagram (7) ST10 – Component Diagram (8) ST7 – Collaboration Diagram (9) ST1 – Entity Relationship Diagram	Diagrammatic tools for object structure, behavior, and interaction
2	(1) ST15 – Hierarchy-Input-Process-Output Chart (2) ST12 – Web Application Extension (WAE) to Unified Modeling Language (UML) (3) ST14 – Decision Table (4) SM4 – Methodology for Web Modeling Language (WebML) (5) ST13 – Program Flowchart (6) ST11 – Deployment Diagram (7) ST16 – Pseudocode	Methodology and tools for structure and processing of the entire Web application
3	(1) ST19 – Periodic/Standardized Progress Report (2) ST18 – Review/Staging Web Site for Communication Purposes (3) ST20 – Project Management Software (4) ST21 – Diagram Generation Software	Tools for communication and productivity
4	(1) SP9 – Presentation/Page Layout design (2) SP8 – Navigation Design (3) SP10 – Page Communication/Relationship	Development phase of static and dynamic page design
5	(1) SP18 - Testing (2) SP16 – Application Coding (3) SP17 – Code Review	Development phase of quality control
6	(1) SP2 – Functional/Technical/Operational Feasibility Studies (2) SP3 – Cost/Benefit Analysis (3) SP19 – Launch (4) SP1 – Creative Brief/Concept Creation	Development phase of before and after development
7	(1) SP5 – Kickoff Meeting to Review Functional and technical Specifications (2) SP14 – Technical Specifications (3) SP6 – Data Storage and Data Access Design	Development phase of specifications and data architecture
8	(1) SP11 – Web Service Design (2) SP12 – Component Design (3) SP13 – Infrastructure Design	Development phase of Integrating External Operations
9	(1) SM3 – Rapid Application Prototyping	RAP methodology
10	(1) SM5 – Waterfall System Development Life Cycle	WSDL methodology
11	(1) SM2 – Extreme Programming	XP methodology

Figure 1. Core and Supporting Methodologies, Tools, and Techniques for Web Application Development in Practice



Appendix I. The Survey Question for Methodologies

	How do you attribute the following methodologies to the success of Web application development if they are used in your organization?	N/A	not significant at all ----- very significant						
			0	1	2	3	4	5	6
SM1	Rational Unified Process	0	1	2	3	4	5	6	7
SM2	Extreme Programming	0	1	2	3	4	5	6	7
SM3	Rapid Application Prototyping	0	1	2	3	4	5	6	7
SM4	WebML (Web Modeling Language)	0	1	2	3	4	5	6	7
SM5	Waterfall System Development Life Cycle	0	1	2	3	4	5	6	7
SM6	Compuware's UNIFACE	0	1	2	3	4	5	6	7
SM0	Others (please specify):								

[SM] What problems or deficiencies do you see in methodologies for Web application development in your organization?

Appendix II. The Survey Questions for Development Phases

	How do you attribute the following development phases to the success of Web application development if they are used in your organization?	N/A	no significant at all ----- very significant						
SP1	Creative Brief/Concept Creation	0	1	2	3	4	5	6	7
SP2	Functional/Technical/Operational Feasibility Studies	0	1	2	3	4	5	6	7
SP3	Cost/Benefit Analysis	0	1	2	3	4	5	6	7
SP4	Generation of Project Plan: Mission, Objectives, Targeted Users, Scope, Budget, Web Teams	0	1	2	3	4	5	6	7
SP5	Functionality Requirements	0	1	2	3	4	5	6	7
SP6	Data Storage and Access Design	0	1	2	3	4	5	6	7
SP7	Operations and Business Process Design	0	1	2	3	4	5	6	7
SP8	Navigation Design	0	1	2	3	4	5	6	7
SP9	Presentation/Page Layout Design	0	1	2	3	4	5	6	7
SP10	Page Communication/Relationship	0	1	2	3	4	5	6	7
SP11	Web Service Design	0	1	2	3	4	5	6	7
SP12	Component Design	0	1	2	3	4	5	6	7
SP13	Infrastructure Configuration	0	1	2	3	4	5	6	7
SP14	Technical Specifications	0	1	2	3	4	5	6	7
SP15	Kickoff Meeting to Review Functional and Technical Specifications	0	1	2	3	4	5	6	7
SP16	Application Coding	0	1	2	3	4	5	6	7
SP17	Code Review	0	1	2	3	4	5	6	7
SP18	Testing	0	1	2	3	4	5	6	7
SP19	Launch	0	1	2	3	4	5	6	7
SP0	Others (please specify):								

[SP] What problems or deficiencies do you see in phases of Web application development process in your organization?

Appendix III. The Survey Questions for Development Tools and Techniques

	How do you attribute the following tools/techniques to the success of Web application development if they are used in your organization?	N/A	not significant at all ----- very significant						
			1	2	3	4	5	6	7
ST1	Entity Relationship Diagrams (ERD)	0	1	2	3	4	5	6	7
ST2	Story Boarding								
ST3	Use Case Diagrams	0	1	2	3	4	5	6	7
ST4	Class Diagrams	0	1	2	3	4	5	6	7
ST5	Object Diagrams	0	1	2	3	4	5	6	7
ST6	Sequence Diagrams	0	1	2	3	4	5	6	7
ST7	Collaboration Diagrams	0	1	2	3	4	5	6	7
ST8	Statechart Diagrams	0	1	2	3	4	5	6	7
ST9	Activity Diagrams	0	1	2	3	4	5	6	7
ST10	Component Diagrams	0	1	2	3	4	5	6	7
ST11	Deployment Diagrams	0	1	2	3	4	5	6	7
ST12	Web Application Extension to Unified Modeling Language	0	1	2	3	4	5	6	7
ST13	Program Flowcharts	0	1	2	3	4	5	6	7
ST14	Decision Tables	0	1	2	3	4	5	6	7
ST15	Hierarchy-Input-Process-Output Charts (HIPO)	0	1	2	3	4	5	6	7
ST16	Pseudocode	0	1	2	3	4	5	6	7
ST17	Workflow Analysis	0	1	2	3	4	5	6	7
ST18	Review/Staging Web Site for Communication Purposes	0	1	2	3	4	5	6	7
ST19	Periodic and standardized Progress Reports	0	1	2	3	4	5	6	7
ST20	Project Management Software	0	1	2	3	4	5	6	7
ST21	Diagram Generation Software	0	1	2	3	4	5	6	7
ST22	Code Generation/Review/Testing Software	0	1	2	3	4	5	6	7
ST23	Application Framework								
ST0	Others (please specify):								

[ST] What problems or deficiencies do you see in tools/techniques for Web applications in your organization?

[OF] What training or skills do you believe are the most useful for Web application developers? Please also provide other feedback you have about Web application development.