Web-Based Decision Support for Software Release Planning

Jingzhou Li
University of Calgary
2500 University Dr NW
Calgary, AB, Canada, T2N 1N4
jingli@ucalgary.ca

Guenther Ruhe
University of Calgary
2500 University Dr NW
Calgary, AB, Canada, T2N 1N4
ruhe@ucalgary.ca

Abstract

Web technology and Web Services represent a great opportunity for improving knowledge and experience exchange. In this paper, the focus is on intelligent decision support for software release planning. We first characterize the problem of software release planning, and then review Web technology and Web-based Decision Support Systems. By deriving some major requirements on Web-based decision support for release planning, we then put forward a suggestion for an architectural design. We discuss the first steps of its realization and future directions of its real-world application.

1. Motivation

Software development is a multi-disciplinary, knowledge-intensive, and human-centric process in which decisions about business, technology, market, software quality and process, distribution of resources, and so on, must be effectively and efficiently made in order to achieve the software product goals and eventually the business goals. Software Engineering Decision Support (SEDS) [1] considers the software planning, development and evolution process as a continuous problem solving and decision making activity. The expected result is a better understanding, management and control of software process, products, resources, tools, and technologies. SEDS will employ possible theories and technologies to fulfill its goals.

Web technology is now applied in various applications as the Internet provides oceans of information and knowledge, and more and more effective information publishing and retrieval techniques. In addition to information, there are services available on the Web, i.e. Web services [2][3]. With the availability of both information and services, Web technology will have a more profound impact on our society.

Web-based Decision Support Systems (DSS)[4][5] try to take the advantage of the Web technology to facilitate the support for decision-making. Some pilot development and application of Web-based DSS has been proved to be effective in many application domains [6].

In this paper, we will use the body of SEDS as a basis to explore the role of Web technology as a support mechanism for software engineering decision-making. Particularly, we will take the software release planning process as an example to illustrate this idea. We first introduce software release planning in Section 2, then the new development of Web technology in Section 3. After deriving some major demands of software release planning on Web technology in Section 4, we put forward an architectural design for Web-based decision support system for software release planning. Finally the conclusions and future work are presented.

2. Software release planning

Incremental software development replaces monolithic-type development by offering a series of increments with additive functionality. The process is a central part of planning in incremental software development. Typically each increment is a complete system that is of value to the client. This means that each new increment can be evaluated by the client. The results feed back to the developers, who then take that information into account when implementing subsequent phases. This feedback may introduce changes to requirements or new requirements, priorities, and constraints.

Software release planning determines which customer gets what features and quality at what point in time. It is a generalization of the “requirements triage” [21] defined as the process of determining which requirements a product should satisfy given the time and resources available. Simultaneously, it is one of the most brilliant examples of decision-making as part of requirements engineering activities. Without good release planning ‘critical’ features are jammed into the release late in the cycle

without removing features or adjusting dates [22]. This might result in unsatisfied customers, time and budget overruns, and a loss in market share.

Incremental development has many advantages over the traditional waterfall approach. First, prioritization of requirements ensures that the most important requirements are delivered first. This implies that benefits of the new system are realized earlier. Consequently, less important requirements are left until later and so, if the schedule or budget is not sufficient, the least important requirements are the ones more likely to be omitted. Second, customers receive part of the system early on and so are more likely to support the system and to provide feedback on it. Thirdly, the schedule/cost for each delivery stage is easier to estimate due to smaller system size. Fourth, user feedback can be obtained at each stage and plans adjusted accordingly. Fifth and most importantly, an incremental approach is sensitive to changes or additions to requirements.

The process of requirements engineering of software systems is a complex problem solving activity involving many stakeholders and many decisions. As a key activity in requirements engineering, software release planning is a complex, human-centric process in which intensive knowledge and intelligent support are heavily needed. Release planning for incremental software development assigns requirements to releases such that all technical, resource and budget constraints are met. Achieving an optimal balance of conflicting stakeholder opinions requires a prioritization of these opinions. For example, it has to be decided what sort of requirements will go into products, which releases the requirements are put into, and when they will be implemented. [7][8][9].

To summarize main characteristics of software release planning:

- **Requirements are not well specified and understood**: There is usually no formal way to describe the requirements. Non-standard format of requirement specification often leads to incomplete descriptions and makes it harder for stakeholders to properly understand and evaluate the requirements.
- **Stakeholder involvement**: Stakeholder examples are user (novice, advanced, expert), shareholder, project manager, and developer. In most cases, stakeholders are not sufficiently involved into the planning process. This is especially true for the final users of the system. Often, stakeholders are unsure why certain plans were suggested. In the case of conflicting priorities, knowing the details of compromises and why they were taken would be useful. All these issues add to the complexity of the problem at hand and if not handled properly, they create a huge possibility for project failures. On the other hand, how the stakeholders can easily participate the planning process is also a big problem. It is impossibe to have all the stakeholders gathered together to have a discussion. Wired or wireless network support is necessary for stakeholders to be involved easily and frequently in the release planning process.
- **Change of requirements and other problem parameters**: Requirements always change as the project progresses. If a large number of requirements increase the complexity of the project, their dynamic nature can pose another challenge. Other parameters such as the number of stakeholders, and their priorities, etc., also change with time - adding to the overall complexity.
- **Size and complexity of the problem**: Size and complexity are major problems for project managers when choosing release plans - some projects may have hundreds or even thousands of requirements. The size and complexity of the problem, and the tendency for not involving all of the contributing factors, makes the problem prohibitively difficult to solve by individual judgment or trial and error type methods.
- **Uncertainty of data**: Meaningful data for release planning are hard to gather and/or uncertain. Specifically, estimates of the available effort, dependencies of requirements, and definition of preferences from the perspective of involved stakeholders are difficult to gauge.
- **Availability of data**: Different types of information is necessary for actually conducting release planning. Some of the required data are available from other information sources within or external to the organization. Ideally, release planning is incorporated into existing Enterprise Resource Planning or other organizational information systems.
- **Constraints**: A project manager has to consider various constraints while allocating the requirements to various releases. Most frequently, these constraints are related to resources, schedule, budget or effort. Some of the constraints are hard constraints, others are soft ones.
- **Unclear objectives**: “Good” release plans are hard to define at the beginning. There are competing objectives such as cost and benefit, time and quality, and it is unclear which target level should be achieved.

- **Efficiency and effectiveness of release planning**: Release plans have to be updated frequently due to changing project and organizational parameters. Ad hoc methods help determine solutions but are far behind objective demands.
- **Tool support**: Currently, only general-purpose tools for requirements management are available. None of them focuses on the characteristics of release planning.

Some tasks to be done and questions to be answered for software release planning:

- **Ensure the release plans satisfy the precedence constraints**: technical (e.g. software quality especially reliability and maintainability, system architecture), logical (e.g. business process, functional relations, market policy and competition), and resource.
- **Determine the requirements priorities**: by different groups of stakeholder such as users, developers, managers, and shareholders, depending on their different opinions of requirements.
- **Effort estimation**: when new or additive requirements added, explore the ripple effects: change of data structure, system architecture, etc., which may affect the effort.
- **Resources allocation**: person, hardware, software, COTS, and technology when making release plan.

Some of these characteristics and tasks have been partially addressed in some literature. In [7], a method based on genetic algorithm called EVELOVE is used to generate the releases incrementally with the inputs of effort, stakeholder priorities, precedence and coupling. However, determining the value of these inputs still requires a great deal of information and effort. In [8], release planning is discussed from a perspective of market driven requirements engineering processes. Requirements dependency is described as a crucial task of release planning. Unfortunately, no practical methods were proposed. Paper [9] mainly discussed the requirements interdependencies in release planning, without considering other factors.

From the main characteristics and the above preliminary literature review, we conclude the relevance of support for the procurement of information, knowledge, and services (such as remote collaboration and negotiation over the Web either wired or wireless) are also crucial in decision-making support for release planning. Web technologies offer a great opportunity to achieve these goals.

3. Web Technology

Web (or Internet/Intranet/Extranet) technology with its rapid development has been providing great benefit for the whole world. Known for its ability to organize and traverse oceans of information as a "web", Web technology now has a new member—Web services which are a new breed of Web application. Web services are self-contained, self-describing, modular applications that can be published, located, and invoked across the Web. Web services perform functions, which can be anything from simple requests to complicated business processes.

Now that we have both information and services available on the Web, it really becomes true that "network is computer." Therefore, the concept of Internet Operating System (IOS) or Grid [10] has emerged as a way to generalize the view of the Web. Taking IOS as a platform, we are now able to assembly an application from the Web using its contents (or information) and services. The contents on the Web are scattered across the Web sites over the world. They are multi-sources, multi-forms, and different in management mechanisms, to name a few. To use the information on the Web effectively, the Virtual Database (VDB) technology [11] has come into being. By combining the Semantic Web technology and Web services [2][12], discovery of Web services can be performed at a high level in accordance with business requirements [13].

Compared to traditional network applications and the early-age applications of the Internet, the Web-based applications will benefit from the following aspects which are also the challenges the new Web technology faces:

- Grid computing and peer-to-peer systems applications
- Distributed and replicated file system management
- Content caching, replication, and distribution; content delivery networks
- Secure and reliable messaging based on HTTP, SOAP, and XML and also the security and confidentiality of Web-based system
- Directories of distributed and replicated content and services
- Ubiquitous user interfaces aggregating Web applications
- Management of distributed processes and orchestration of services
- Monitoring and management of applications based on Web contents and services
- The orchestration of services and their related data

Because of DSS's characteristics, Web-based DSS [14][15] [16] seems promising in that Web technology provides much support that the traditional technologies could not, e.g. the availability of both internal and external information, and simple yet powerful user interface. In [17], guidance for building Web-based DSS is provided.

Although there are web services already available, their practical use still does not exploit the potential benefit.

4. Demands of Web-based decision support for software release planning

There is some obvious evidence about the potential benefits of web-based DSS, but what are the demands in more detail, especially when focusing on web-based DSS for software release planning?

- **(R1) Different decision support functionality:** inclusion of data driven, document driven, communication driven, model driven, and knowledge driven; or the combination of the above functionality; other intelligent support components, such as reasoning and analysis methods, interface component, and explanation component.

- **(R2) Interaction** among and **integration** of various DSS components to allow information exchange and sharing in order to fulfill their functionality individually and as a whole.

- **(R3) Multi-sources and multi-levels abstractness** of information and knowledge: internal/external, domain-dependent/domain-independent, structured/unstructured, business/technical, for decision-makers and decision support components. For example, decision-makers may ask how a result data comes and where it comes from for explanation.

- **(R4) Negotiation among stakeholders**: One example in software release planning is that stakeholders need to negotiate the priority of requirements to be released according to their opinions of the requirements.

- **(R5) Collaboration and coordination** among stakeholders.

- **(R6) Integration** with existing information systems and resources. The decision support process will share data with existing information systems and some decision support functions may need to interact with some components of existing information systems.

- **(R7) Friendly user interface**: users should be able to access all the information on the web from as many devices as possible at any time, for example, wireless equipment.

In dependence of the problem, different aspects of these requirements may become more important than others.

5. Web-based decision support for release planning

From the above discussion we know that the Web essentially offers to the user services and contents through a unified interface tool – the web browser. This is illustrated by Figure 1.

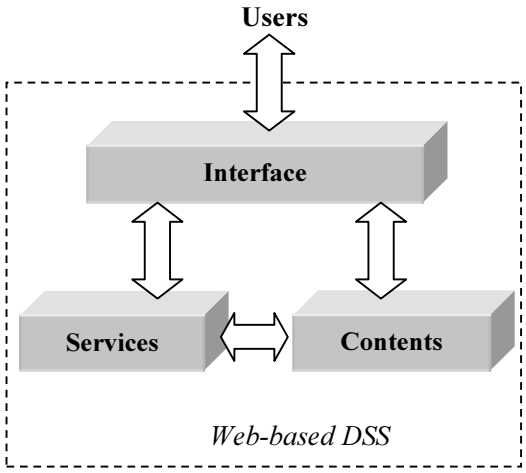

Figure 1. Conceptual Components of Web-based DSS

Here the services may be functions of local and legacy information systems, web-based or not, and Web services on both Intranet and Extranet. The functionality of the whole decision support system is thus divided into the aforementioned services. What are the advantages of using the concept of providing services?

- Services can be (self-) described, published and discovered on the web so that they can be invoked as needed;

- Services may be implemented and invoked platform-independently;

- Services can be published by providers, discovered and invoked by consumers. The two sides communicate with each other using protocols, such

as HTTP and SOAP for web services: thus the services are loosely coupled by the Web, which increases the independency of system components;

- Systems with service architecture that are loosely coupled by the Web can be easily integrated;
- With the rapid development of Web service technology, there will be an increasing variety of services available on the Web, which makes the development of new applications increasingly easier than before.

The contents components in Figure 1 may be any kind of data or knowledge existing on the Web, both Intranet and Extranet, ranging from structured data stored in databases to text documents and multimedia files. So far, the most distinctive advantage of Web is that it may contain multi-sources, multi-forms, and distributed information.

Combining the services and contents on the Web, or consuming the Web contents by the services on the Web, Web-based applications will be easily constructed, and easily operated by users through the unified Web browser interface.

Corresponding to the characteristics of the release planning problem and the demands of Web-based decision support for release planning, the following aspects of decision support for release planning are further discussed based on the notion illustrated by Figure 1. The requirements in Section 4, R1 to R7, covered by the aspects are indicated in the parenthesis.

- **DSS components (R1, R2)**: can be organized and implemented as services. For example, if general models and analysis methods for release planning in a model centered component are implemented as services and probably provided by some producers on the web, then decision alternatives can be obtained from different services, evaluated, and chosen. The construction of this component becomes relatively easy. Take EVOLVE [7] for release planning as another example. The method may be realized wholly or partially as Web services and put on the Web so that it can be implemented and invoked easily. Meanwhile, the interaction between components becomes simple since the services are loosely coupled by the Web. So are similar situations for the other components.
- **Web contents (R3)**: the problem of multi-resources and multi-forms of data has been tackled by Web contents that encompass any kind of data, documents, or even knowledge [18] that is distributed on the Web.
- **Collaboration/negotiation (R4, R5)**: over the Web [19], easy communication with both wired, such as Netmeeting, and wireless equipment, such as Smartphone and PDA, especially when using Web services, so that stakeholders may communicate and negotiate over the Web. Likewise, the developers may collaborate over the Web.
- **System integration (R6)**: decision support system can be easily integrated with the existing information systems under the service architecture: information sharing, document formalizing/editing/searching/management based on XML technology.
- **User interface (R7)**: Easy-to-use and easy access from the Web browser on thin clients. Wired or wireless devices can also be used from anywhere at anytime.

We propose a more detailed composition of Web-based decision support components as shown in Figure 2.

There are vertical components in Figure 2 (a), which are specific functions, and horizontal components in Figure 2 (b), which are commonly used by all the vertical components.

Horizontal components include, for example, negotiation, collaboration/coordination support over the Web as a common means for all vertical components. The explanation component is used for tracing back the solutions the vertical components provided to help the decision makers to understand the problem and problem-solving process.

Vertical components include, for example, COTS DSS [20] for COTS-based software development; communication-driven support mainly for negotiation, collaboration, and coordination among stakeholders; document-driven support for managing a variety of e-document formats, keeping track of textually represented knowledge for decision-making; model-driven support for modeling and simulation; knowledge-driven support for management of “expertise”, “experience”, and “skill” for specialized domain or problems. The functionality of these components is normally implemented and used separately under traditional DSS architecture. However they can be integrated under the Web-based and service centered architecture.

The contents, on the other hand, can be put under the management of Virtual DB, Web knowledge management,

or simply left as normal Web contents for browsing using the Web browser.

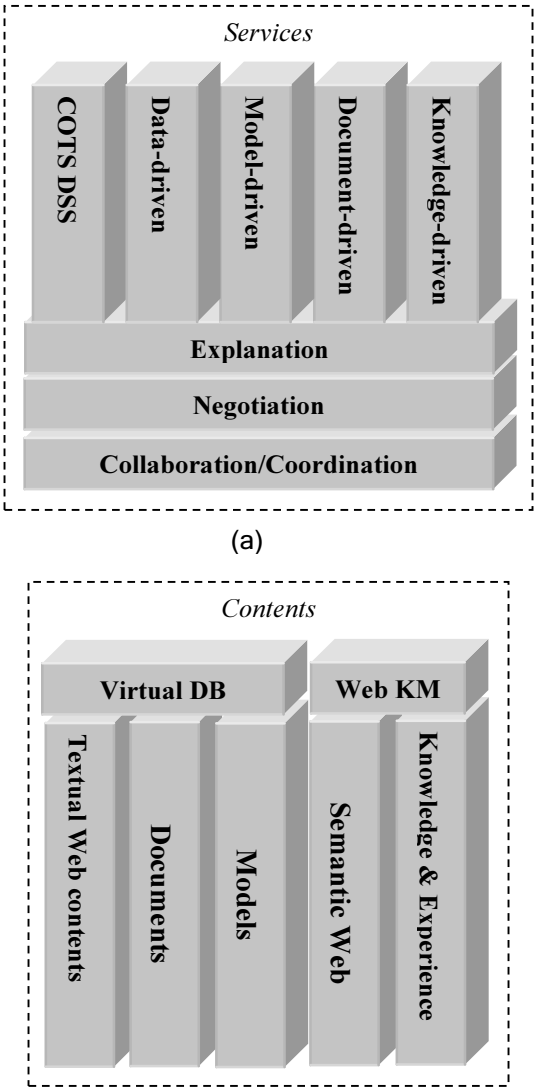

Figure 2. Detailed components of Services and Contents

Putting all the considerations discussed above together, we propose an architectural design for Web-based decision support for software release planning, as illustrated by Figure 3.

6. Conclusions and future work

In this paper, we have presented an architectural design for web-based decision support for release planning based on the characteristics of release planning and the demands on its Web-based decision support Although it has not been implemented and applied in the real world, it provides an insight view on the possibility and benefit of employing Web-based technology for software release planning. In fact, the architectural design can also be used to support other kinds of release planning such as project or product planning.

The Web-based and service centered architecture has some advantages over the traditional ones:

- Multi-sources and multi-formats of information: This is the most advantageous aspect for decision support where a great amount and variety of information is needed. For XML-based Web contents, it will be easier to retrieve both information and knowledge.

- Loosely coupled service architecture: Services can be provided, discovered, and invoked over the Web. They are loosely coupled by the Web via standard languages, such as XML, WSDL, OWL, and protocols, such as HTTP, or SOAP [2][3]. This makes it easier for interaction between components, and the integration with the legacy information systems.

- Easy collaboration and involvement over the Web: Stakeholders can be easily brought together on the Web for collaboration and negotiation purpose, either synchronously, e.g. through Netmeeting, or asynchronously, e.g. e-mail.

- Easy access through Web interface: Users can easily access the Web contents through the Web browser, especially non-computer professionals who have been accustomed and always willing to use the simple yet powerful Web browser. On the other hand, users may access the Web at anytime from anywhere through wired or wireless, simple or complex devices.

Our future work will focus on the following aspects:

- Develop a prototype to evaluate the proposed notions and architecture in cooperation with industry in order to make it more practical.

- Find ways to facilitate the integration of legacy information systems with the Web.

- Explore the application of Web services in order to get the first-hand experience of creating and using services on the Web.

- Develop a high-level description of Web services and more general concept of services in order to map the high-level business

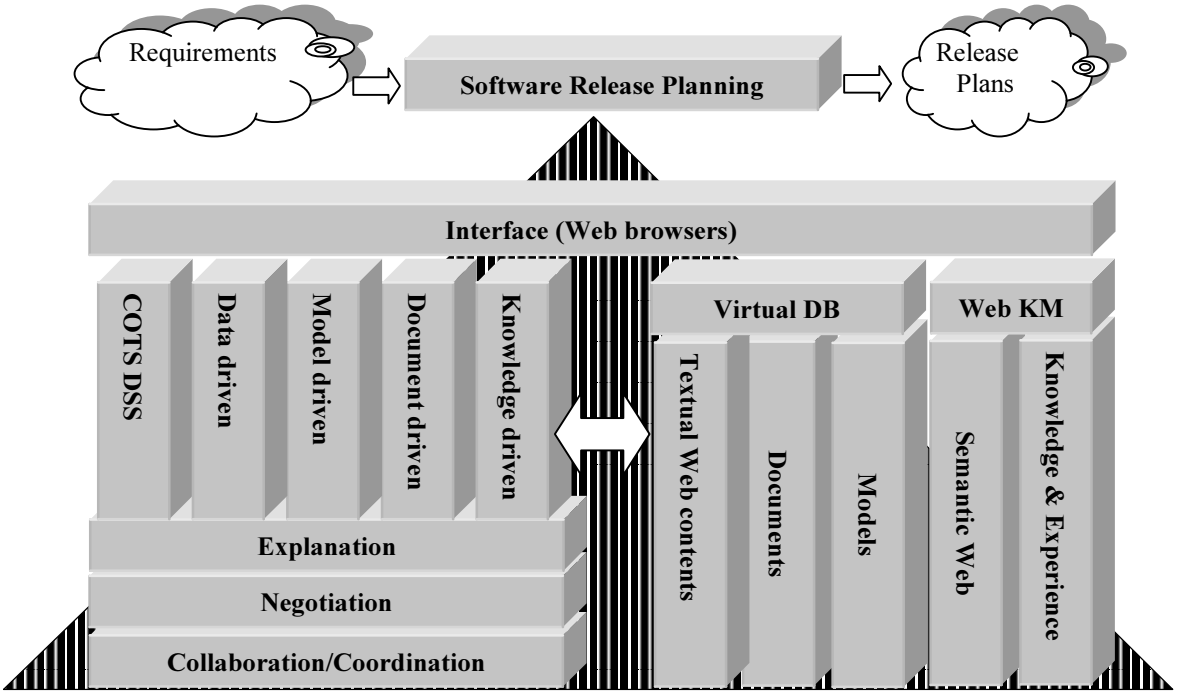

Figure 3. Architectural design for Web-based DSS for release planning

requirements to the low-level implementation of services such as the Web services at present.

- Web-based knowledge management: knowledge representation, discovery, and storage on the Web.
- Put more semantics into both the contents and services on the Web for "meaningful" discovery using semantic Web and intelligent Web technologies.
- Incorporate available and effective Web related technology into decision support for release planning first, then the other software engineering activities.
- Build a prototype of Web-based DSS supporting wireless and mobile devices that can be used by stakeholders to participate in the planning process through simple Web browser.

Acknowledgements

The authors would like to thank the Alberta Informatics Circle of Research Excellence (iCORE) for its financial support of this research.

References

[1] Guenther Ruhe, "Software Engineering Decision Support—Methodology and Applications", *chapter 5 in Innovations in Decision Support Systems*, Graziella Tonfoni, Lakhmi Jain (Eds.), International Series on Advanced Intelligence, Volume 3, 2003, pp. 143-174.

[2] Michael P. Papazoglou, "The World of e-Business: Web-Services, Workflows, and Business Transactions", *Web Services, e-Business, and the Semantic Web*, Bussler et al. (Eds.), CAiSE 2002 International Workshop, WES 2002, Toronto, Canada, May 2002.

[3] Paul Cowles, "Web Services and the Semantic Web", *Web Services Journal*, Volume 02, Issue 12, 2002.

[4] J.P. Shim, et al., "Past, present, and future of decision support technology", *Decision Support Systems*, Issue 33, 2002, pp. 111-126.

[5] C. Carlsson, et al., "DSS: directions for the next decade", *Decision Support Systems*, Issue 33, 2002, pp. 105-110.

[6] Allan Leck Jensen, Iver Thysen, Peter S. Boll and B.K. Pathak, "Pl@nteInfo: A World Wide Web based Decision Support System for Crop Production Management in

Denmark", *Agricultural Information Technology in Asia and Oceania 1998*, pp.125-125.

[7] D. Greer, Guenther Ruhe, "Software Release Planning: An Evolutionary and Iterative Approach", *Information and Software Technology* (Accepted).

[8] P. Carlshamre, B. Regnell, B., "Requirements lifecycle management and release planning in market-driven requirements engineering processes", *Proceedings of 11th International Workshop on Database and Expert Systems Applications*, Sept. 2000, pp. 961-965.

[9] Carlshamre P., et al., "An industrial survey of requirements interdependencies in software product release planning", *Proceedings of Fifth IEEE International Symposium on Requirements Engineering*, August 2001, pp. 84-91.

[10] Dmitri Tcherevik, "Internet Operating System", *Web Services Journal*, Volume 02, Issue 09, 2002.

[11] Anand Rajaraman, Peter Norvig, "Virtual Database Technology: Transforming The Internet Into A Database", *IEEE Internet Computing*, Vol. 2, Issue 4, July/August 1998, pp. 55-58.

[12] Joram Borenstein, Joshua Fox, "Semantic Discovery for Web Services", *Web Services Journal*, Volume 03, Issue 04, 2003.

[13] Andrew Bibby, Cesar Brea, "Business Reengineering Through Web Services", *Web Services Journal*, Volume 02, Issue 09, 2002.

[14] Benjamin Khoo, Guisseppe Forgionne, "Web-Based Decision Making Support Systems", *Hawaii International Conference on Business, June, 2003*, Hawaii, USA.

[15] Helen S. Du, Xiaohua Jia, "A Framework for Web-based Intelligent Decision Support Enterprise", *IEEE Proceedings of the 23rd International Conference on Distributed Computing Systems Workshops (ICDCSW03)*, 2003.

[16] Suresh Sridhar, "Decision support using the Intranet", *Decision Support Systems*, 23 1998, pp. 19-28.

[17] Daniel Power, Shashidhar Kaparthi, "Building Web-based Decision support Systems", *Studies in Informatics and Control*, Vol. 11, No. 4, December 2002, pp. 291-302.

[18] Sergio A. Alvarez, et al., Introduction to Part IV: Data Mining and Web Knowledge Management, *Web Knowledge Management and Decision Support*, Oskar Bartenstein, et al. (Eds.), 14th International Conference on Applications of Prolog, INAP 2001 Tokyo, Japan, October 2001. LNAI 2543, Springer, 2003.

[19] Gregory E. Kersten, Sunil J. Noronha, "WWW-based negotiation support: design, implementation, and use", *Decision Support Systems*, 25 1999, pp. 135-154.

[20] Guenther Ruhe, "Intelligent Support for Selection of COTS Products", *Proceedings of the Net.ObjectDays 2002*, Erfurt, Springer 2003, pp. 34-45.

[21] Davis, A.M., The Art of Requirements Triage, IEEE Computer 36(3), Mar 2003, pp 42- 49.

[22] Penny, D., An Estimation-Based Management Framework for Enhancive Maintenance in Commercial Software Products, Proc. International Conference on Software Maintenance, 2002.