Development of Non-functional Requirements for Decision Support Systems

Kassem Saleh

***Abstract*—**Decision Support System (DSS) are interactive software systems that are built to assist the management of an organization in the decision making process when faced with non-routine problems in a specific application domain. Non-functional requirements (NFRs) for a DSS deal with the desirable qualities and restrictions that the DSS functionalities must satisfy. Unlike the functional requirements, which are tangible functionalities provided by the DSS, NFRs are often hidden and transparent to DSS users but affect the quality of the provided functionalities. NFRs are often overlooked or added later to the system in an ad hoc manner, leading to a poor overall quality of the system. In this paper, we discuss the development of NFRs as part of the requirements engineering phase of the system development life cycle of DSSs. To help eliciting NFRs, we provide a comprehensive taxonomy of NFRs for DSSs.

***Keywords*—**Decision support system, Development, Elicitation, Non-functional requirements, Taxonomy

I. Introduction

WITH the increasing growth of enterprise information and the complexity of systems to be managed, the demand and need for Decision Support Systems (DSSs) assisting the management in the process of decision making is growing [1].

NFRs are requirements that are related to the quality aspects of the system being developed or the functionalities provided by the system [2]. The quality-based requirements cover all levels and phases of the DSS life cycle including, pre-development and development, operations, and maintenance and evolution phases. While eliciting the functional requirements using the use case modeling approach [3,4], the analyst must also elicit the NFRs that are associated with each use case. In addition to use case-specific NFRs, the analyst must also identify and elicit generic DSS-wide NFRs. Generic NFRs are use case independent and apply to the entire DSS. For example, security requirements can be either use case-specific or generic. However, cultural, political, and standards conformity requirements are mainly generic requirements since they normally apply to the DSS as a whole. NFRs can be either technical or non-technical. Technical NFRs, such as performance requirements, are quantifiable and possibly automatically verifiable requirements [7]. These types of requirements affect the whole software architecture and can impose or limit the possible design or architectural choices.

Non-technical NFRs, such as standards conformity requirements, are mostly non-quantifiable and only verifiable using a non-automated review process. The proper elicitation of NFRs is critical. The types of requirements can conflict or affect each other; in addition, they can affect other functional requirements. For example, a security requirement can affect the performance of the DSS negatively, requiring, for example, the additional exchange of messages.

Kassem Saleh is with Kuwait University, Dept. of Information Science, Box 5969, 13060 Safat (phone: 965-6649-8883; e-mail: kassem.saleh@ku.edu.kw).

Moreover, a security requirement can require additional functional requirements. NFRs can also be a main concern either to the users or to the developers. For example, testability is a NFR that affects the maintenance predominantly because it requires that the developer makes some additional effort to make the DSS testable. However, making the DSS testable has a positive and indirect effect on the reliability of the DSS. Reliability is a technical NFR of particular interest to the users. On the other hand, performance is a NFR that mainly affects the user because response time delays and lack of memory are noticed by the user. However, the developer is also affected because meeting performance requirements must be dealt with at various phases of the software development process.

Normally, non-technical NFRs are generic ones. However, technical NFRs can be either system-specific or use case-specific. For example, various performance requirements can be attached to multiple use cases, unlike, for example, standards conformity requirements that are typical system-wide requirements. Some of the NFRs can also be considered DSS quality attributes that are mainly of interest to the developer but can have a tangible impact on the users. NFRs must not attempt to prescribe or impose a technical solution. They are essentially technology independent. The types of NFRs that might be needed to constrain the DSS under development and its environment are introduced in the following sections. In [2], non-functional properties in service-oriented architectures for web services were identified and assessed based on a questionnaire filled by web service users.

In this work, we place the NFRs under three categories related to the DSS life cycle, starting from DSS development NFRs, to DSS operations NFRs and ending with DSS maintenance and evolution NFRs. Non-technical NFRs are mainly constraints that must be considered when the DSS as a product is being developed.

To elicit the NFRs, the appropriate stakeholder must consider each type of requirement thoroughly. Depending on the type of DSS being developed, it is determined whether the requirements are mostly DSS-wide or not. In the first case, the DSS-wide constraints and requirements must be considered first. Then, when considering each use case of the DSS separately, it is decided whether to relax or modify the DSS-wide requirements. In another approach, we first consider the types of requirements that are mostly DSS wide requirements, such as usability and maintainability requirements. Then, we proceed to those requirements types that can be partly use case dependent, such as security and performance. In this way we attempt to minimize the rework of the requirements from generic to specific ones. NFRs can be documented separately in the DSS requirement document if they are generic. However, if they are specific to a particular use case, they can be attached closer to the use case description. The constraints section of the use case can include specific NFRs applicable to that use case. Further work on automating the capturing and management of NFR is needed.

The development of a DSS follows the same development process of a typical software product starting from requirements specifications, design, to implementation, testing and deployment. The stakeholders in the development process for DSSs in general and their functional and non-functional requirements in particular, include: (1) the DSS users including the manager as a decision maker and the DSS manager responsible for the system management and maintenance, (2) the representative of the institution requesting and financing the development of the DSS, (3) the DSS analyst representing the DSS development team, and (4) legal, regulatory, professional and standard authorities related to the application domain. The rest of the paper is organized as follows. Section II presents the DSS development and pre-development NFRs. Section III presents the DSS operations NFRs. Section IV presents the DSS maintenance and evolution NFRs. Finally, Section V concludes the paper and provides directions for future work.

II. Development and pre-development NFRs

To elicit the NFRs that are most relevant prior and during the development of the DSS, the appropriate stakeholders such as the client, developer and user representatives must consider each of the following requirement types thoroughly. In the following, we describe each type of development and pre-development requirements. Requirements are listed in alphabetical order.

Accessibility requirements impose access related features of the DSS, such as web based intranet access, or limited closed system access.

Cost and budget requirements are typical pre-development requirements needed for the initial planning for the DSS development. These requirements constraint the number and quality of the features provided by the DSS.

Cultural and political requirements address the cultural and political constraints that have to be considered when developing the DSS. The analyst and client must be aware of the cultural sensitivities of the countries in which the DSS will be deployed and used. These could include language use issues, use of symbols, and politically offensive contents, among others. Failure to properly elicit the requirements can affect its acceptability and market penetration. Should the requirements conflict in other countries or societies, several versions of the DSS might have to be developed and deployed.

Design and implementation requirements impose constraints on some design and implementation features of the DSS, like the algorithms and architectural and design patterns used, the programming language and development environment used, and the operating systems and hardware platform used.

Documentation requirements address the documents needed as deliverables of the DSS development process. These documents are either internal technical documents needed as part of the adopted development life cycle process, or user-related documents such as user and installation manuals.

Interface requirements identify all the external interfaces of the DSS such as external information sources, and other hardware systems, like sensors, devices among others.

Legal and regulatory requirements address the legal and regulatory issues related to the DSS. Adherence to local and international laws and regulations have to be observed by the DSS and the process by which it is developed. For example, for national security concerns, the development team must not include people that have resided in certain countries. Other requirements related to the DSS itself can require that the DSS is in line with copyright laws and regulations. If overlooked, the requirements can lead to lawsuits and criminal investigations affecting the economic feasibility and reputation of the development company.

Look and Feel requirements provide the general guidelines and constraints related to the user experience. The look and feel could be a novel approach to the user interface interactions or could be an adaptation of an existing and proven look and feel used by the software industry.

Personnel requirements are pre-development requirements providing some constraints related to the personnel engaged on the DSS development project. These requirements include personnel security procedures and recruitment requirements in terms of qualifications and experiences of the personnel.

Standards conformity requirements indicate the standards that must be followed while developing the DSS. There are different types and levels of standards. Internal standards that are developed by the DSS development company can include coding, testing, and documentation standards and templates. Specific country standards might have to be followed and the developer and client must be aware of them. Military standards exist and need to be followed if the military is one of the DSS stakeholders. Industry standards have to be considered. Specific standards exist for the health, finance, banking, and education sectors, among others. Professional standards can also be referred to in this type of requirements. Finally, international standards, such as those developed by the International Standardization Organization (ISO) and the International Telecommunications Union (ITU), can also be used as part of the standards conformity requirements. The DSS analyst representing the developer and the client representing the users must be aware of the standards related to the application domain of the DSS under development. Non-adherence to known and relevant standards and the failure to meet them can lead to unusable DSS or a delayed release of the DSS. Therefore, the requirements must be elicited carefully.

III. Operations NFRs

To elicit the NFRs that are most relevant during the DSS operation, the appropriate stakeholders such as the client, developer and user representatives must consider each of the following requirement types thoroughly. Requirements are listed in alphabetical order.

Auditability requirements impose constraints on the type and granularity of the audit logs recorded during the DSS operation for accountability reasons among others.

Availability requirements are used to impose an upper limit on the downtime of the DSS, indicating an acceptable level of failures. Availability requirements are also considered part of the security requirements are discussed later.

Deployability requirements impose constraints on the way the DSS is deployed. Installability and configurability requirements, referring to the ease of installation and configuration, can also be related to deployment constraints.

Interoperability requirements impose constraints on the types of systems with which the DSS has to interface. A DSS might have to interface and communicate with other DSSs, software systems and hardware devices. This type of requirements may impose some dependability requirements on the proper specification and implementation of the DSS interfaces. Consequently, the requirements are indicative of some potential external risks to the development process.

Operational requirements impose constraints and capacity requirements on the environments in which the DSS under development will operate. The constraints can include the characteristics in which the servers are physically located, minimum speed of the network connections, the year-to-year growth in the number of DSS users, specific quality of service parameters, the operating systems that are compatible, and the hardware devices with which the software interfaces.

Performance and efficiency requirements impose some technical constraints on the response time delays, startup and shutdown times, throughput of the system, and memory requirements. Response time delays can be either system-wide or use case specific. For example, putting an upper limit on the maximum response time delay for all system functions is a system-wide performance requirement. However, we can also require different upper limits, depending on the individual function or use case. Response time-related performance requirements can also be stated as throughput requirements, putting upper and lower bounds on the acceptable rate of completed transactions.

Reliability requirements impose some values related to the reliability of the DSS under development. Typical values include the mean time between failures or the maximum time allowed for failures over a period of time, in addition to some quality of service requirements.

Robustness requirements impose constraints related to the way the DSS handles abnormal and erroneous inputs and operational conditions. The requirements address the DSS behavior with respect to error recoverability and fault tolerance and can be system-wide or use case-specific requirements. Robustness requirements should not state how to achieve DSS robustness or suggest a particular technological solution to do it.

Safety requirements are needed when the DSS being developed deals with safety-critical issues such as an DSS controlling a chemical production plant. The safety requirements must address and require the enforcement of safety standards that are known in the particular application domain. Eliciting safety requirements requires some expertise in the domain of the safety-critical application to avoid costly legal consequences should any of the requirements be omitted.

Scalability requirements impose constraints on how the DSS should scale up to a high user input load, i.e., concurrent number of users, at all its interfaces. This requirement can force specific architectural design choices on the development team. Ideally, scalability requirements are quantifiable and are normally verified during load and stress testing.

Security requirements are critical for a successful DSS and need to be identified early in the DSS development process. A security requirement can be either a system-wide or use case-specific requirement. Security requirements address four main security concerns: (1) confidentiality, (2) integrity, (3) availability, and (4) accountability. The concerns are dealt with by imposing and adhering to identification, authentication, authorization, integrity, immunity, privacy, non-repudiation, survivability, physical protection, and security standards conformity requirements.

Access control-related requirements, including identification, authentication, and authorization are used to address confidentiality concerns. Also, physical protection requirements can be useful in addressing confidentiality concerns at the physical level. Identification, authentication, and authorization requirements can be system-wide or use case specific. The client can require that certain functionalities are only accessible by identified, authenticated, and authorized users, and different access rights can be assigned based on the user's role in the organization.

Integrity concerns are addressed using integrity, immunity, and privacy requirements. An integrity requirement can be system-wide or use case specific.

Availability issues are dealt with using the survivability and physical protection requirements. A survivability requirement can be applicable system-wide. It can also be function- or use case-specific.

Accountability concerns are dealt with using the non-repudiation and standards conformity requirements. Accountability requirements can be either system-wide or applicable to specific functions or use cases of the system and are dealt with at different levels of granulatities (see auditability requirements). However, standards conformity requirements are normally system-wide. Although security requirements are considered to be NFRs, some of them must be implemented by first identifying the appropriate security-related functional requirements. For example, identification, authentication, and authorization requirements are considered by introducing a logon use case as a functional requirement. These use cases are called security use cases.

Supportability requirements are related to constraints on the available support of the DSS after its deployment. User training requirements and user documentation requirements can be included in the supportability requirements.

Usability and understandability requirements address the constraints imposed by the client in its representation of the user community. The objective for the constraints is to make the DSS easy to use by the various users interacting with it. The usability requirements are normally elicited by first knowing the intended users and their backgrounds and characteristics. This step would be the first to undertake in a usability engineering process. Look and feel requirements imposing a standard look of the user interface of the DSS can be included in the usability requirements.

User-friendliness requirements impose some constraints on the user experiences when interacting with the DSS. The constraints can have implications on the functional requirements and the graphical user interface design decisions.

The availability of context-sensitive help versus generic help, a forgiving and courteous interface, and the ease of navigability are examples of user-friendliness requirements. Usability and user-friendliness requirements complement each other and are often seen as equivalent. A system can be easy to use but might not be friendly. Ideally, a highly-usable system is normally a user-friendly system.

IV. MAINTENANCE AND EVOLUTION NFRS

To elicit the NFRs that are most relevant during the DSS maintenance and evolution, the appropriate stakeholders such as the client and developer representatives must consider each of the following requirement types thoroughly. Requirements are listed in alphabetical order.

Maintainability and modifiability requirements impose constraints related to the ease with which the DSS can be modified, fixed, adapted to new environments and technologies, or expanded. To meet the requirements, there can be various technical and managerial measures that need to be taken. Technical measures can be related to the development environment and tools used as well as the development methodologies and models adopted. Managerial measures include hiring decisions and appropriate developers training programs. Technical documentation requirements and understandability requirements can be part of the maintainability requirements. Similarly, adaptability requirements for customization, personalization, internationalization and localization should be considered for DSSs operating in various cultural, social and political contexts.

Portability requirements impose some conditions related to the future deployment of the DSS. DSS portability is defined as the ease with which the DSS can be modified to run on a different hardware platform or software environments. To meet this requirement, many constraints related to the development environment, design and implementation methodologies used have to be imposed.

Retirement requirements impose some conditions related to the decision and processes needed for retiring or decommissioning the DSS, such as the procedure to inform the users and the disposition of the collected personal user information and audit logs.

Reusability requirements impose constraints on the development of the system related to the degree of reuse. There are two types of reuse: development *with* reuse and development *for* reuse. Development with reuse aims at producing the DSS faster by using existing software components. A development with reuse requirement helps the reliability of the overall system provided we are reusing good quality components. Development for reuse aims at producing highly maintainable DSS that is typically made of reusable components. The components can be reused in future projects by the same team or other development teams. A development for reuse requirement helps the maintainability of the DSS and imposes specific decisions related to the software development methodologies used.

Testability requirements impose constraints on the future testing of the DSS during development and maintenance. Testability is defined as the ease with which testing can be performed. Observability, controllability and diagnosability requirements are normally part of the testability requirements. Testability requirements are somehow related to the maintainability requirements and the ease of DSS maintenance activities. Traceability requirements impose some constraints on the ease with which the DSS is traceable. This requirement can be related to the traceability of the different parts of the software development process deliverables or related to the traceability of the DSS during its execution. In the first type of traceability, the requirement refers to the ability to link every aspect of the DSS deliverables forward and backward. For example, each module in the DSS design can be traced backward to a requirement specification element, or forward to a particular piece of code or test cases, hence, enhancing the DSS maintainability and meeting the maintainability requirements. In the second type of traceability, traces collected during the execution of the DSS can be needed. DSS execution traces are normally used for testing and debugging purposes, thus, enhancing the DSS testability to meet the DSS testability requirements. In addition, this aspect of traceability helps to meet the auditability and non-repudiation aspects of security requirements.

V. CONCLUSIONS AND FUTURE WORK

Elicitation of non-functional requirements for DSSs will contribute positively to the development of higher quality DSSs and hence higher adoption of these DSSs by the users. The types of requirements must be considered early in the development process while the functional requirements are also developed. In this paper, a comprehensive list of non-functional requirement types are identified. We tried to be as exhaustive as possible to be able to capture and identify all these requirements. We are currently developing three specific case studies in the e-learning, e-health and e-banking domains. In the future, we are planning to develop tools to capture and document these requirements.

TABLE I
THE TYPES OF NFRS AND THEIR CLASSIFICATION

Category	Types of non-functional requirements
DSS Development and Pre-development	Accessibility, cost, cultural, documentation, design and implementation, interface, legal, look and feel, personnel, political, regulatory, standards conformity
DSS Operation	Auditability, Availability, deployability, efficiency, interoperability, operational, performance, reliability, robustness, safety, scalability, security, supportability, understandability, usability, user friendliness
DSS Maintenance and Evolution	Adaptability, maintainability, modifiability, portability, retirement, reusability testability, traceability

ACKNOWLEDGMENT

The author would like to acknowledge the support of this work by a Kuwait University Travel Grant.

REFERENCES

[1] R. Sprague, "A framework for the development of decision support systems", MIS Quarterly, Vol. 4, Number 4, December 1980.

[2] L. Chung, B.A. Nixon, E. Yu, J. Mylopoulos, Non-Functional Requirements in Software Engineering, Kluwer Academic Publishers, Dordrecht, 2000.

[3] H. Becha and D. Amyot, "Non-functional properties in service oriented architecture – A consumer's perspective", to appear in the *Journal of Software*, 2012.

[4] G. Booch, I. Jacobson and J. Rumbaugh, The UML User Guide, Addison-Wesley, 1999.

[5] ITU-T, Recommendation Z.150, User Requirements Notation (URN) – Language Requirements and Framework, Geneva, 2003.

[6] S. Robertson and J. Robertson, Mastering the Requirements Process, Addison-Wesley, 1999.

[7] K. Saleh, Software Engineering, *J. Ross Publishing*, USA, 2009.

[8] K. Saleh and A. Al-Zarouni, "Capturing non-functional requirements using the user requirement notation", Proceedings of the *International Research Conference on Innovations in Information Technology (IIT 2004)*, Dubai, Oct 2004, pp. 222-230.

Kassem Saleh was born in Beirut, Lebanon in 1963. After completing his technical high school in electronics in Beirut, he moved to Ottawa, Canada in 1981 where he received his BS, MS, and PhD in computer science from the University of Ottawa in Canada in 1984, 1985 and 1991, respectively. Dr. Saleh worked as a software design engineer at Northern Telecom in 1984 and then as a computer systems specialist at Mediatel, Bell Canada, from 1985 to 1991. He is currently a professor in Information Sciences at Kuwait University. Kassem was on the faculty of Concordia University during 1991-1992, Kuwait University from 1992 to 2000, and American University of Sharjah from 2000 to 2007. Dr. Saleh is also a Certified Information Systems Security Professional (CISSP) since 2005. He is a senior member of IEEE and a professional member of the ACM. The Journal of Systems and Software has ranked Dr. Saleh among the top scholars in the field of systems and software engineering in seven of its annual assessments published from 1996 to 2003. His research interests include software engineering, requirements engineering, communications protocols, and information security. Dr. Saleh has published more than 130 refereed journal and conference papers and has presented numerous tutorials and lectures at international conferences and universities worldwide. Dr. Saleh is currently editor-in-chief of the Journal of Software. Dr. Saleh founded and chaired the Kuwait Conference on e-Systems and e-Services (www.kcess.org).