Issues in the Development of a Requirements Traceability Model

Balasubramaniam Ramesh
Code AS/RA
Naval Postgraduate School
217 Ingersoll Hall
Monterey, CA 93943
E-mail: ramesh@is1.as.nps.navy.mil

Abstract

In the development of large-scale, real-time, complex computer intensive systems, it is essential to maintain traceability of requirements to various outputs to ensure that the system meets the current set of requirements. Based on an empirical study in a simulated systems development environment, this research highlights several major issues that need to be considered in the development of a model of requirements traceability.

1 Introduction

A primary concern in the development of large-scale, real-time, complex, computer-intensive systems is ensuring that the performance of system meets the specified requirements. As part of the system development and maintenance process, many decisions and trade-offs are made that affect a variety of components of the system. Further, the requirements themselves evolve and undergo many change during the development process. In such a context, it is essential to maintain traceability of requirements to various outputs or artifacts produced during the systems design process, to ensure that the system meets the current set of requirements. Maintaining consistency between the requirements and the design is especially critical in situations where an organization relies upon outside contractors for developing systems. Having a systematic way of validating that every requirement is met by the design is important, not only to ensure that the system performs correctly, but also to determine whether contractual obligations have been met.

A comprehensive scheme for maintaining traceability, especially for complex, real-time systems, requires that all system components (not just software), created at various stages of the development process, be linked to the requirements. These components include hardware, software, humanware, manuals, policies and procedures. In order to achieve this objective, it is essential that traceability be maintained through various phases of the systems development process, from the requirements as stated (or contracted) by the customer, through analysis, design, implementation and testing to the final product.

A variety of definitions of traceability have been proposed in the literature depending on the intended use of traceability information. Greenspan and McGowan [1] provide a generic definition of traceability: traceability is a property of a system description technique that allows changes in one of the three system descriptions - requirements, specifications, implementation - to be traced to the corresponding portions of the other descriptions. The correspondence should be maintained throughout the lifetime of the system.

Our goal is to develop a model that represents traceability information at the level of systems design, relating requirements to all system components. Further, such a model should not only discuss the kinds of linkages or relationships that should be maintained, but also the reasoning that can be performed with such information.

2 Towards a Model of Traceability

The need for better understanding of traceability is widely recognized. Maintenance of traceability is also mandated by several standards governing the development of systems for the U.S. Government. However,
the precise definition of the kinds of traceability linkages or relationships that must be maintained is currently lacking. A principal challenge in this research is the development of a model that represents and provides the semantics of various traceability linkages or relationships between requirements and system components.

There are a variety of stakeholders involved in systems development process, including project sponsors, project managers, analysts, designers, maintenance personnel, testing personnel, and end users. A basic premise in our research is that development of a model of traceability could be geared towards the needs of these various stakeholders at various stages in the systems development process. The first phase of our approach to this problem has been an empirical one. We have conducted a preliminary study to explore the traceability needs of various stakeholders. This study was conducted with graduate students of systems analysis in a simulated systems development environment. Prior to participation in the study the subjects designed and implemented a system using a case study drawn from a real-life large scale system. Further, the subjects had prior exposure to concepts of traceability in domains other than computer based systems development (such as aircraft maintenance, ship building). Primary data collection strategies used in this research include focus groups for idea generation and protocol analysis for studying problem solving behavior. In this paper, we present some findings from the preliminary study. The results of this study are being used in designing a comprehensive study involving stakeholders in large scale, complex, real-time systems development efforts.

3 Issues in the Development of a Model of Traceability

In this section, we discuss results from the analysis of data collected using focus groups and protocol analysis. A discussion of major issues that need to be addressed in the development of a model of traceability, and the mechanisms to support the capture of and reasoning with this information is presented. Findings from relevant literature are included to elucidate the main issues.

3.1 Bidirectional Traceability

Bidirectional traceability implies both forward and backward traceability. Forward traceability is provided if each requirement specifically references a design component. In other words, forward traceability allows one actually to see where requirements materialize in the finished system. Backwards traceability is provided when a requirement is referenced by a design element. Though one of the most critical uses of traceability is ensuring that a design element satisfies a requirement, the existence of such a link may not answer the question: are the functionalities of the design element required by requirements? To help answer this question, links need to be bidirectional in order to allow requirements to be traced forward from requirements to systems components, and backward, from systems components to requirements.

3.2 Criticality of Requirements

A useful way of identifying critical requirements is to relate them to the central mission of the system. Business processes or missions that generate requirements could be identified, and requirements evaluated with respect to such processes, to arrive at a classification. For example, traceability should address the issue of how the requirements are arrived at. This necessitates a mechanism to represent the elaboration and refinement of requirements, from the central mission or business processes that generate them. This corresponds to the concept of tracing back from requirements [2].

It may be unnecessary or even undesirable, considering the overhead involved in maintaining traceability, to maintain linkages between every requirement and every output created during the systems design process related to it. The need to relate mission criticality to a traceability scheme was considered important by many subjects in the focus groups: ("We just have to realize that it [traceability] is not necessary for mundane decisions.")

3.3 Design Rationale

Another important component of traceability is design rationale information. Traceability linkages to represent rationale would capture the why or reason for design decisions. Tracking relationships among design objects, and understanding how and which of those objects is affected by change, is vital in the maintenance of the system. The subjects were keenly aware of the need for design rationale as a component of

---

1This is a direct quote from a subject participating in the empirical study. Henceforth, all quotes from subjects are enclosed in parenthesis and quotation marks
traceability: ("The systems designer needs traceability in order to examine the logic behind the system."); "Traceability could be very useful for justifying why you did something the way you did it.").

3.4 Project Tracking and Management

Requirements traceability can be used very productively in project management and tracking. During the systems definition and subsequent phases, traceability is essential to ensure that all systems requirements have been met. Establishing all life cycle phases as complete can go a long way toward guaranteeing the ease of the verification and validation process.

The project manager can use links such as status, completion date, and authorization between various components of the system for scheduling, continuity, and security. Such information is indispensable in integrating project management into the systems development operation, and the efficient completion of project management tasks.

The subjects were very interested in project tracking and management possibilities using traceability. ("Without traceability, if you've lost a linkage you spend much valuable time tracing back to the original requirement."); ("The project manager needs traceability for tracking milestones.")

3.5 Accountability

A major use of traceability is to provide accountability. Using traceability legitimately to communicate with the original designer of a system component, or to understand the capability of a system, is an example of such potential use. However, caution must be used when employing traceability information to enforce accountability. The use of accountability information as a means for performance appraisal may be inappropriate.

Some accountability information that could be captured using traceability linkages include: design elements designed by, validated by, and modified by development personnel. The availability of such information will be indispensable in maintaining and revising a system. The focus group subjects perceived an urgent need for the accountability element tempered by constraints on its usage: ("Traceability needs to be something that humans can work with, not just a whip held over people."); ("Accountability needs to be supplemented with good communication.").

3.6 Humanware

Humanware form a critical component of any large-scale system. It just as important to capture how requirements relate to humanware as with other components of the system. This may involve tracing the responsibility for a requirement to a human. A comprehensive mechanism for traceability should link the humanware component of a system to the other components. Examples of such linkages include systems functionalities performed by humans. This information is necessary to ensure that the allocation of requirements is complete and correct.

3.7 Documents/Manuals

Document traceability determines the existence of relationships between two document segments; it means that a particular document is in accord with a previous document, with which it has some type of relationship. Document traceability also ensures that all components in one document have a related component in another document. Consistency and completeness constraints apply within a document and across documents. Inconsistent references and incomplete specifications may occur and can be checked [3]. Traceability linkages to documents include interpreted by, defined by, and consistent with. Such linkages specify how to obtain a required performance from a systems component. Our focus group subjects had considerable insight into some of the document traceability implications: ("Stakeholders are interested in having traceability to be able to write quality manuals and data dictionaries.").

3.8 Dependencies

Since complex systems are composed of interdependent components, such dependencies should be represented and maintained. Further, systems design is a complex activity involving interdependent decisions. In the absence of mechanisms to record such dependencies, over time and with changing development teams, this information will be lost. Such dependencies may span different systems components. A decision about software may be dependent on an earlier decision about hardware. As the system evolves over its life cycle, the hardware decision may be altered, leading to inconsistencies with the software that was based on the earlier hardware decision. Unless the dependencies are captured and maintained, such issues may go undetected, leading to severe system integration problems.
Another form of dependency is the fact that there may be several components needed to satisfy a requirement. As the system evolves over its development life cycle, it is desirable to identify design or implementation elements that "partially satisfy" a given requirement. For instance, a hardware/software combination is often necessary to satisfy a given requirement. When either the hardware or software component is developed, traceability information should reflect the fact that the partially satisfied requirements are fully satisfied by performance of necessary actions. Dependencies could be transitive. ("an element in the data base design depends on [an element in] the data flow diagram [which in turn] depends on [a piece of] requirement.").

3.9 Horizontal and Vertical Traceability

Vertical traceability refers to the "association of software (system) life cycle (SLC) objects of different types (typically created in different SLC processes). An example of a vertical traceability relationship would be between requirements statement and design statement" [4]. "Vertical traceability is easy because there's a 'rule'... you explode a process and either you have to or you don't."

Horizontal traceability refers to the "association of SLC objects of the same type (typically created in the same SLC process). An example of this type of traceability includes parent/child relationships among decomposed data flow diagrams and the 'derived from' relationship among requirement statements" [4]. "When you're moving horizontal is when you're analyzing what process is inside what process." Horizontal traceability equates to a ("subprocess transferring data to another subprocess like primitive levels have to talk to each other, etc."). Horizontal traceability also refers to relationships between different views of the same level of (design) representation.

3.10 Automated Support for Traceability

Automated support for traceability can be extremely valuable when systems are large and/or complex. "When performed manually, the tasks are time-consuming and error-prone; moreover, users' abilities to analyze traceability data are limited by the sheer volume of data" [4]. In such circumstances, "an automated software tool is an imperative, as the measuring process can become extremely onerous" [5]. As stated by Thayer and Dorfman, "There have been many cases where it appeared, at the outset, that it would be an easy task to keep track of it [manually], but when the system design is complete, and the customer is trying to understand whether all the test data really satisfies the original requirements they wrote, the automated traceability would be 'worth its weight in gold'"[6]. The degree of automated support can vary widely, depending on the level of sophistication warranted/desired.

4 Conclusions

Our research suggests that a comprehensive model of traceability should include semantic models that can support various stakeholders in systems development activities. Models that capture essential features of the systems development process, such as the design rationale are components of a useful traceability scheme. Further, such models should be supported by reasoning mechanisms to support various stakeholders. Development of such models and reasoning mechanisms are the primary goals of our ongoing research.

References


