7N-61
411910

Paper abstract submission to the 16th Pacific Northwest Software Quality Conference

Testing, Requirements, and Metrics

L. Rosenberg, PhD	L. Hyatt	T. Hammer	L. Huffman	W. Wilson
Unisys/GSFC	NASA GSFC	NASA GSFC	Unisys/GSFC	SATC/GSFC
301-286-0087	301-286-0087	301-286-0087	301-286-0087	301-286-0087
Linda.Rosenberg@ gsfc.nasa.gov	Larry.Hyatt@ gsfc.nasa.gov	Ted.Hammer@ gsfc.nasa.gov	Lenore.Huffman@ gsfc.nasa.gov	William.Wilson@ gsfc.nasa.gov

ABSTRACT

The criticality of correct, complete, testable requirements is a fundamental tenet of software engineering. Also critical is complete requirements based testing of the final product. Modern tools for managing requirements allow new metrics to be used in support of both of these critical processes. Using these tools, potential problems with the quality of the requirements and the test plan can be identified early in the life cycle. Some of these quality factors include: ambiguous or incomplete requirements, poorly designed requirements databases, excessive or insufficient test cases, and incomplete linkage of tests to requirements. This paper discusses how metrics can be used to evaluate the quality of the requirements and test to avoid problems later.

Requirements management and requirements based testing have always been critical in the implementation of high quality software systems. Recently, automated tools have become available to support requirements management. At NASA's Goddard Space Flight Center (GSFC), automated requirements management tools are being used on several large projects. The use of these tools opens the door to innovative uses of metrics in characterizing test plan quality and assessing overall testing risks. In support of these projects, the Software Assurance Technology Center (SATC) is working to develop and apply a metrics program that utilizes the information now available through the application of requirements management tools. Metrics based on this information provides real-time insight into the testing of requirements and these metrics assist the Project Quality Office in its testing oversight role. This paper discusses three facets of the SATC's efforts to evaluate the quality of the requirements and test plan early in the life cycle, thus preventing costly errors and time delays later. Data from the projects is used to support and clarify the concepts discussed.

The first testing metrics activities that will be discussed are the development of a tool and its application early in the life cycle in order to assess the quality of requirements. This paper describes application of the Automated Requirements Measurement (ARM) tool. ARM parses the text of the requirements into identifiable units in order to evaluate potential words or phrases that may affect their testability. Because both the software acquirer and the software provider must understand and contractually agree to the requirements, specifications are usually written in natural language. The use of natural language to prescribe complex, dynamic systems has at least two severe problems: ambiguity and inaccuracy. Many words and phrases have dual meanings which can be altered by the context in which they are used. Defining a large, multi-dimensional

capability within the limitations imposed by the two dimensional structure of a document can obscure the relationships between individual groups of requirements.

As a result of these realizations about natural language, the SATC developed the ARM tool to assess the quality of requirement specification documents and to identify poorly specified requirements that may introduce risks into the project. Since testability is one aspect of requirement quality, ARM t searches the requirements document for terms the SATC has identified as quality indicators - weakness or ambiguity - and quantifies them. From these quantifications, profiles of document structure adds to the knowledge the project management uses for its decision making. Additionally, reports produced by the tool identify the individual specification statements that need improvement. It must be emphasized that the tool does not assess correctness of the requirements specified; it does, however, assess the structure, language, and vocabulary of both the document itself and the individual requirements. [7]

The next testing metrics activities we investigate relate to the test plan links between test cases and the requirements. Using NASA project data we will look at the linkage of the requirements, the relationship between unique requirements and unique tests, and the ratios of test to requirement links. Once requirements are written, methods for ensuring that the system contains the functionality specified must be developed; this section of the paper discusses three efforts to evaluate the quality of the test plan while still in the requirements phase. [5]

In the total set of test cases, each requirement must be tested at least once, and some requirements will be tested several times because they are involved in multiple system states in varying scenarios and in different ways. But as always, time and funding are issues; while each requirement must be comprehensively tested, limited time and limited budget are always constraints upon writing and running test cases. It is important to ensure that each requirement is adequately, but not excessively, tested. In some cases, the requirements can be grouped together using criticality to mission success as their common thread; these *must* be extensively tested.[3] In other cases, requirements can be identified as low criticality; if a problem occurs, their functionality does not affect mission success while still achieving successful testing. In order to ascertain the point at which testing benefits become marginal, the SATC developed the second set of metrics based on data available in the requirements database. These metric analyses use the linking information of the requirements to the tests in three ways. The first is to verify that each requirement is tested at least once. The next two analyses characterize the depth and breadth of the test plan.[2] It is expected that each requirement will be linked to multiple test cases, and that each test case will test multiple requirements.[1,4,6]

And finally, this paper investigates the quality of the requirements management database schema as it relates to cleanliness of data and the ease with which requirement and testing metrics can be obtained. In the preparation of the database that houses the requirements and tests, both the requirement segment and the test segment must be designed with the identical schema design philosophies to enable evaluation of the test plan links to requirements as they are entered into the database. This paper briefly discusses a

requirements database schema that supports comprehensive evaluation of requirements driven testing.

There are no published or industry guidelines or standards for these testing metrics—intuitive interpretations, based on experience and supported by project feedback, are used in this paper. NASA project management has reacted favorably to these metrics and has used the analysis results to mitigate perceived risks. The SATC continues working on methods to mathematically validate the intuitive guidelines. The objective is to assist project management in producing high-quality requirements and test plans while identifying and minimizing project risks.

REFERENCES

1. Beiser, B. *Software Testing Techniques* (1983), Van Nostrand Reinhold Company.
2. Hammer, T., Huffman, L., Wilson, W., Rosenberg, L., Hyatt, L., Requirement Metrics—Value Added in *Proc. Third IEEE International Symposium on Requirements Engineering,* (Annapolis MD, January 1997) IEEE Computer Society Press.
3. Hammer, T., Rosenberg, L., Huffman, L., Hyatt, L., Measuring Requirements Testing in *Proc. International Conference on Software Engineering* (Boston MA, May 1997) IEEE Computer Society Press.
4. Kitchenham, Barbara, Pfleeger, Shari Lawrence, Software Quality: The Elusive Target, *IEEE Software 13, 1* (January 1996) 12-21.
5. NASA, *Software Assurance Guidebook* (September 1989).
6. Software Technology Support Center, *Software Test Technologies Report* (August 1994).
7. Wilson, W., Rosenberg, L., Hyatt, L., Automated Quality Analysis of Natural Language Requirement Specifications in *Proc. Fourteenth Annual Pacific Northwest Software Quality Conference*, (Portland OR, October 1996).