A DEMONSTRATION OF THREE REQUIREMENTS LANGUAGE SYSTEMS

by
Joe Celko, LTC John S. Davis, and John Mitchell
Army Institute for Research in Management Information and Computer Science
(AIRMICS)
Georgia Tech
Atlanta, GA 30332

SUMMARY:

This paper is a summary of a comparative demonstration of three systems requirements tools: the U. S. Air Forces Computer Aided Design and Specification Analysis Tool (CADSAT) with extensions developed by Logicon [1], Teledyne Brown Engineering's Input/Output Requirements Language (IORL) [2], and TRW's Software Requirements Engineering Methodology (SREM) [3]. This work is unique because the approach used is close to that of a controlled experiment.

Three teams were each given the same Detailed Functional Systems Requirement (DFSR) of an actual U.S. Army management information system. The DFSR was made up of narratives, flowcharts, decision tables, and so forth. Each of the teams acted independently of the others, and turned in a separate final report.

Information regarding the number and type of errors found by each demonstration is given.

INTRODUCTION

The Army Institute for Research in Management Information and Computer Science (AIRMICS) is the branch of the U. S. Army Computer Systems Command (USACSC) concerned with the investigation of improved techniques and methodologies for use by USACSC. One area of current investigation deals with requirements specification and analysis tools.

The prior methods for defining system requirements have produced documents in English text. Large organizations, like the United States Army, require large systems , and therefore the requirements documents tend to be large and complex.

These specification documents are so large that it is impossible for a human being to remember, much less analyze, all of their details. Indeed, in very large systems, even trying to remember where to locate the details is a major effort. Thus, errors and contradictions in the system specifications have tended to pass into the design and coding phases. It is an accepted fact that the expense of correcting errors which were made in the specification is greater than any other type of error correction [4].

There are two schools of thought on requirements specification. One school argues that the user does not really know what he wants until he sees it. Therefore, prototype systems which can be developed and changed quickly should be used to obtain the requirements. This is an evolutionary approach which begins with fielding a prototype.

The second school argues that if formal methods are used to define the problem from the start, then the implementation of the system is relatively easy. Automated requirements systems belong to the second school.

Both approaches have advocates and advantages. The method needed depends upon the problem and the degree to which it is definable at the time of the specification.

The goal of all requirements specification and analysis tools is to specify the problem in such a fashion that:

(1) The formal requirements should be the bridge between the non-computer personnel who have the problem and the computer personnel who have to implement the solution on a computer system.

(2) The software system for the formal requirements should be a tool for the computer personnel during the design, coding and testing phases of the system development process. It should be an interactive working document.

(3) The requirements can be checked for formal errors. For example, are there any undefined items in the requirements? Is an output generated before the process has the required inputs? Are there items which have multiple names? Are there separate items which have identical names?

THE DEMONSTRATIONS

AIRMICS arranged for a demonstration of three requirements engineering tools: the U. S. Air Forces Computer Aided Design and Specification Analysis Tool (CADSAT) with extensions developed by Logicon, Teledyne Brown Engineering's Input Output Requirements Language (IORL), and TRW's Software Requirements Engineering Methodology (SREM). Independent teams were each given a copy of the detailed functional system requirement for the Standard Army Maintenance System (SAMS) [5] and asked to translate and analyze it with their respective software tool. When an assumption had to be made, the teams were to document it and proceed with their translation work.

THE DEMONSTRATION PROBLEM

SAMS is a standard management information system intended for worldwide use by U. S. Army installations and organizations for maintenance management. The requirements are a two part document totaling approximately 2200 pages. The first part is the retail level Maintenance Operation Management (MOM) whch has 2100 pages and the second part is the Maintenance Program Operation Maintenance (MPOM).

MOM contains 85 narratives of systems functions and constraints witn design discussions and solutions. The balance of both MOM and MPOM contains sections which give I/O descriptions, flowcharts, decision logic tables, file descriptions, information elements, external interfaces, telecommunications requirements, and a glossary of terms.

Each demonstration team was able to refer to a section number as the source of their information concerning requirements and assumptions.

OVERVIEW OF THE THREE TOOLS

All of these descriptions are brief overviews of manuals listed in the references provide further information.

IORL DESCRIPTION:

In IORL [6],[7] the system parts are represented in a graphic format known as Schematic Block Diagrams (SBD). The interaction among SBD's is shown by arrows (interfaces) connecting the blocks. A list of parameters passed along the interface appears in the I/O Parameter Table (IOPT). The I/O Relationships and Timing diagram (IORTD) relates the input parameter and output parameter values in terms of decision logic, mathematical functions and I/O response timing.

IORL produces graphical output, with references on the drawings to the tables in the system. Utility programs which are not part of IORL were used to produce some of the final report information.

The system has been in use at Teledyne Brown Engineering for 11 years. The methodology has a strong resemblance to the data flow diagram techniques of Yourdon, IST/McAuto, and others.

SREM DESCRIPTION:

The SREM Requirements Statement Language (RSL) is the principal interface between the system and people. RSL is processed by REVS (Requirements Engineering Validation System). The RSL looks much like a structured programming pseudo-code, and can be processed by REVS to produce simulations.

REVS utility programs provide consistency and completeness checks. The user can query the systems to extract certain parts of the requirement, and generate charts known as R-Nets. Static and dynamic models of the system can be built and tested. This feature depends on an external library of Pascal modules which simulate the parts of the system.

CADSAT DESCRIPTION:

CADSAT is a version of PSL/PSA developed for the U. S. Air Forces. PSL/PSA is a well-known requirements language, first developed at the University of Michigan by Teichroew, Hershey and Sayani [8].

CADSAT is weak in graphics but has the ability to perform simple simulations from the formal language.

RESOURCES USED:

The primary resources were machinery and personnel. The composition of the demonstration teams varied due to different selection criteria, and the three systems run on different hardware, ranging from minicomputer to mainframe.

CADSAT was run on a VAX 11/780. It used 22 hours of machine time. The team consisted of:

1 Technical consultant at quarter time for 7 months.
1 Analyst at full time for 7 months.
2 Junior Analysts at full time for 7 months.

IORL was run on a dedicated single user GT44 (PDP 11/40, with graphics peripherals). Computer time used was not recorded. The team consisted of:

1 Technical supervisor at 1/6 time for 6 months.
4 Analysts at 1/3 time time for 7 months.
1 Clerk at 1/3 time time for 7 months.

SREM was run on a CDC 7600. It used 5.9 hours of machine time. The team consisted of:

1 Technical supervisor at half time for 7 months.
1 Analyst at full time for 7 months.

DOCUMENTS PRODUCED:

The output of each system is so different that the only way to compare them is to simply describe them and comment.

The SREM demonstration produced a data dictionary of 750 pages, a 1,500 page depiction of the thirty major functions, and 112 pages of documentation on SAMS problem.

IORL produced a total of 3,000 pages of documentation, much of which was graphical. This page count includes schematic block diagrams, timing diagrams and I/O parameter tables as well as English text materials.

CADSAT produced a total of 835 pages of documentation. Most output was in the formal language, rather than graphics and English text.

TRACEABILITY:

One important feature of a formal requirements system is the ability to trace a requirement back to its source. In this case, the source was a written specification. This meant that the references could be made using the chapter-section-paragraph numbers of the SAMS documents.

The IORL demonstration accounted for the source of requirements by entering comments on the graphics. The Teledyne Brown Engineering team used utility programs (which are not part of IORL) to produce cross references.

The Requirements Statement Language (RSL) in SREM accounted for the source of requirements by storing sources along with the requirements. SREM provides reports which trace source requirements to the RSL constructs, and vice versa. There is also a query ability in the system. The SREM database can be checked for completeness by finding constructs which do not match source requirements and vice versa.

CADSAT also stores sources along with the requirements, and provides reports which trace source requirements to the PSL constructs, and vice versa.

ERRORS FOUND:

Differences in error reporting terminology, methodology and assumptions made by personnel make it difficult to compare the error detection results of the three teams. In fact, the more sophisticated the package, the fewer errors it tended to report because it was better able to resolve questions that arose without referring to a human.

All of the demonstrations found numerous instances of incompleteness and inconsistency. These problems were definitely errors, not a matter of judgement or a result of incompatibility between the requirements documents and the mechanical requirements of the system. They included such things as:

(1) The same data item has several different definitions.

(2) The source of a data item on an output report is not defined.

(3) The description in the decision tables disagrees with the corresponding narrative.

SREM reported 302 problems in the following categories: Inconsistent (101), Ambiguous (70), Missing (53), Illogical (38), Incomplete (26) and Other (14). Most of the problems were detected by the analysts during construction of the R-Nets (60%) and the rest were found by the automated validation system.

IORL reported 542 problems in the following categories: Inconsistent (143), Ambiguous (126), and Other (273).

CADSAT reported 250 problems in the following categories: Inconsistent (115), Incomplete (52), Observations/Questions (79)and Typographical error (4). The approach taken with this system was to apply a completeness analysis and a separate consistency analysis.

CONCLUSIONS:

The first conclusion which can be drawn is that any tool is better than no tool at all. This is not unexpected, but it is nice to have hard data to back it up.

The next conclusion is that a more sophisticated tool will find fewer problems than a a less sophisticated tool. The more sophisticated tool is better able to resolve apparent problems without having to make a report to the human user.

IORL seemed to be the easiest to learn. (This might be the result of the close resemblance to several popular dataflow design methodologies, such as IST/McAuto [9], Yourdon [10], and SADT [11].) The team assembled by Teledyne Brown Engineering had no prior exposure to either IORL or to systems which resemble the SAMS requirement statement. It is also the cheapest to use because of the implementation on a minicomputer. IORL has strong graphics and uses the fewest computer resources, but has the weakest analysis software tools.

SREM had the most advanced features of the three, but this also made it the hardest to learn and to use. Its implementation on a large mainframe also made it the most costly. The automated analysis and the graphics were the best among the systems. A version of REVS is now available on a VAX 11/780.

CADSAT gave the best data structure representation. It was the most flexible of the three. Reports could be read even by those unfamiliar with the systems. Since CADSAT is a version of PSL/PSA, it has the largest number of users and greatest body of experience.

Though conditions for each translation were similiar, this was not a controlled experiment, and the results should be regarded with caution. Each software tool has a different philosophy and computer support. There is no clear "winner" or "loser". Instead, the proper approach is to think of a three-way trade-off. The factors are the computer resources, the level of sophistication of the requirements tools and the user experience level. No single factor can be stressed without shifting costs over to one or both of the others. The systems designer is then left with the task of optimizing among the factors to provide the best possible tool under the actual working conditions. The only solid conclusion is that any of the three tools is better than the manual requirements validation procedures.

AIRMICS hopes to use knowledge and experiences gained from these demonstrations for the improvement of ongoing research to develop more advanced systems for automating the requirements development process.

REFERENCES

[1]
K. M. Terrell and L. A. Johnson, "AIRMICS LARE/CADSAT Analysis Final Report", Logicon Report R 81003(A), April 1980.

[2]
Everhart, et al, "SAMS IORL Demonstration Final Report", Teledyne Brown Engineering Report SD80-AIRMICS-2428, September 1980.

[3]
R. P. Loshbough, et al, "Applicability of SREM to the Verification of Management Information Systems Software Requirements", TRW Report 37554-6950-001, April 1981.

[4]
Barry Böehm, "Software Engineering", IEEE Transactions on Computers, December 1976 (pages 1226-1241).

[5]
U. S. Army, TM 38-L71-2 Volume IV, Detailed Functional Systems Requirements (DFSR), Standard Army Maintenance System (SAMS) - Retail Level, Maintenance Operations Management (MOM), (SAMS-1).

[6]
Teledyne Brown Engineering, "The IORL Operators Users' Language Manual", 1980.

[7]
Teledyne Brown Engineering, "The IORL Users Language Manual, Version 1-B", 1980.

[8]
Teichroew and Chikofsky, "An Introduction to the Problem Statement Language (PSL) and the Problem Analyzer (PSA)", ISDOS Project, Department of Industrial & Operational Engineering, University of Michigan, Ann Arbor, MI 48109, August 1980.

[9]
Gane and Sarson, "Structured Systems Analysis", Prentice-Hall, New York, 1979.

[10]
Yourdon and Constantine, "Structured design", Yourdon Press, New York, 1975.

[11]
Dickover, "Principles of Coupling and Cohesion in the Practice of SADT", Technical Pub No. 039, SofTech Inc. Waltham, MA, 1976.

--END--