Prototype Educational Tools for Systems and Software (PETS) Engineering

Joseph E. Kasser
University of South Australia, Mawson Lakes, Australia
Joseph.kasser@unisa.edu.au

Xuan-Linh Tran
University of South Australia, Mawson Lakes, Australia
Xuan-Linh.Tran@unisa.edu.au

Simon P. Matisons
University of South Australia, Mawson Lakes, Australia
simon.matisons@unisa.edu.au

Abstract: *This paper briefly discusses the problem of poor requirements and how, while solutions are known, they are not being transferred from academia to industry. The paper then discusses a project to implement an approach for transferring the solution from academia to industry by using the Blackboard principles of Artificial Intelligence and software Agents to provide a suite of evolving educational tools based on an object-oriented information systems paradigm. The paper then gives an overview of the functionality provided by the tools, some early results and a list of the expected benefits of the project.*

Keywords: *Requirements engineering, postgraduate education, tools*

Introduction

The systems and software development industry is characterized by a paradigm of project failure (Standish 1995). The situation has been described by Cobb's Paradox (Voyages 1996), which stated "*We know why projects fail, we know how to prevent their failure --so why do they still fail?"* While the problem of poor requirements engineering and management has been repeatedly and widely discussed and documented for at least 10 years as a contributing cause of project failures (Hooks 1993; Kasser and Schermerhorn 1994; Jacobs 1999; Carson 2001; etc.), the continual documentation and discussion of the problem of poor requirements engineering and management has not resulted in a practical solution to the problem.

The PETS project

This paper discusses a project for the development and use of a suite of prototype educational tools for systems and software (PETS) engineering that:

- Have the potential to transfer the solution to the problem of poor requirements management from academia to industry and consequently reduce cost and schedule overruns.
- Are as simple to use as a slide rule.

- Can be used in the classroom and in the workplace.

The development of these tools and accompanying classroom materials is known as the PETS project.

Background

Research into the cause and effect of the problem of poor requirements management has shown that the current focus of systems and software engineering on improving the written text of the requirement is an important but incomplete solution to the problem. Moreover, there has been recent recognition that a requirement is more than just the imperative statement. For example both Alexander and Stevens (2002) and Hull et al. (2002) discuss additional attributes of a requirement in conjunction with improving the writing of requirements. However, in practice, there is difficulty in adding these additional attributes to the traditional requirement document or database. This is because the current systems and software development paradigm generally divides the work in a project into three independent streams – Management, Development, and Test (Quality) (Kasser 1995). Thus requirements engineering tools contain information related to the Development and Test streams (the requirements) while the additional attributes tend to be separated in several different tools, e.g. (Requirements Management, Project Management, Work Breakdown Structures, Configuration Control, and Cost Estimation, etc.).

Expanding the scope of the requirement

After research into reasons for project failures (Kasser and Williams 1998), and the management of change over the System Life Cycle (SLC), Kasser (2000) used an object-oriented approach within an information system paradigm to derive the additional attributes of a requirement needed to alleviate the expensive cost and schedule impacts and proposed the following set of Quality System Elements (QSE) as being necessary for effective system and software development:

- Unique identification number - the key to tracking.
- **Requirement -** the imperative construct statement in the text mode, or other form of representation.
- **Traceability to source(s) -** the previous level in the production sequence.
- **Traceability to implementation -** the next level in the production sequence. Thus requirements are linked to design elements, which are linked to code elements.
- **Priority -** knowing the priority allows the high priority items to be assigned to early Builds, and simplifies the analysis of the effect of budget cuts.
- **Estimated cost and schedule -** these feed into the management plan and are refined as the project passes through the SLC.
- **The level of confidence in the cost and schedule estimates** - these should improve as the project passes through the SLC.
- **Rationale for requirement -** the extrinsic information and other reasons for the requirement.
- **Planned verification methodology(s) -** developing this at the same time as the requirement avoids accepting requirements that are either impossible to verify or too expensive to verify.
- **Risk -** any risk factors associated with the requirement.
- **Keywords -** allow for searches through the database when assessing the impact

of changes.

- **Production parameters -** the Work Breakdown Structure (WBS) elements in the Builds in which the requirements are scheduled to be implemented.
- **Testing parameters -** the Test Plans and Procedures in which the requirements are scheduled to be verified.
- **Traceability sideways to document duplicate links -** required when applying the QSE to an existing paper based project.

The information in the QSE is related to all three streams of work in a project because the underlying concept for the QSE is that the three streams of work are interdependent not independent. It is a different paradigm to the one that has produced the current generation of Requirements Engineering and Project Management tools. Thus importance and value of the additional attributes of a requirement are discussed in textbooks and the set of attributes defined as the QSE are taught in the postgraduate classroom in UniSA. However, while practitioners publish, and academia teaches, what the students "should" do, the current generation of requirements engineering tools do not allow the students to use that knowledge, and hence there is no (easy) way for the students to see how the QSE improve the current paradigm, and consequently the new knowledge is not used outside the classroom which means that the problem of poor requirements management remains.

Baseline for the PETS project

The baseline for the PETS project is the need to improve the effectiveness of postgraduate education in the information technology (IT) industry and the sum of previous products and experience as described below.

The need to improve the effectiveness of postgraduate education in the corporate environment

The knowledge explosion of the early 21st century has given rise to the situation in the IT industry in which the half-life of the knowledge needed to do a particular job can be as short as two years. This has resulted in the need for continual education and life long learning. Corporate employees are thus faced with the problem of acquiring new knowledge while at the same time performing on their job, and meeting their family and other non-corporate lifestyle duties. A goal of the project is to improve the effectiveness of postgraduate education for students in the corporate environment in the IT industry. The PETS should assist in the resolution of Cobb's paradox by increasing the effectiveness of requirements engineering and management, while minimising classroom learning time.

The Requirements Workshop at University of Maryland University College

The requirements workshop was held as part of postgraduate courses in Software Engineering at University of Maryland University College (UMUC). The workshop

- Discussed the problems resulting from poor requirements.
- Provided examples of poorly written requirements.
- Provided a set of requirements for writing requirements (Kasser 1995).
- Asked the students to read and evaluate an instructor-supplied requirements document to determine the number of good and bad requirements in the document.

The Student Enrollment and Course Tracking System (SECTS)

The SECTS provided students in the requirements, design and development, independent validation and verification (IV&V), and software maintenance postgraduate courses in

Software Engineering and Computer Systems Management at UMUC with different perspectives of the same system (Kasser and Williams 1999). The approach used was to create paper documents for various aspects of the same system. Thus the requirements class created a requirements document, the design classes created design documents, the IV&V class created test documents, and the software maintenance class created a software maintenance plan. The PETs will be used to create similar documents in classes at UniSA, but will emphasize the underlying QSE by treating the documents as views or printouts of the information in the database rather than stand-alone paper-based products.

The Suite of Agents concept

The Agent based approach of using a suite of software products for rapid software evolution based on a domain model was presented as a way to develop software that is quicker and less expensive to maintain (Glover and Bennett 1996). Kasser (2000) also described the concept of a suite of tools for accessing the information in the QSE. Kasser and Cook (2003) discussed the projected implementation of the suite using a rapid incremental solution construction approach, which maps into both the Blackboard and Agent based approaches.

The First Requirements Elucidator Demonstrator (FRED)

An early prototype of FRED was presented by Kasser (2002) as a tool that ingested requirements from documents and identified potential defects in requirements by parsing the text for the presence of a set of "poor words". FRED was based on automating and improving the manual process performed during the Requirements Workshop at UMUC. FRED produced a Figure of Merit (FOM) for the document. The FOM is a simple one-dimensional measurement for the quality of a document based on the presence or absence of "poor words". The FOM allows comparisons to be made of the quality of documents of different sizes. The FOM was calculated using the formula

FOM = 100 * (1 - number of defects / number of requirements).

The prototype educational tools for systems and software (PETS) engineering

The plan is to develop and use an initial simple suite of tools with a similar user interface. The tools would then continue to evolve into intelligent agents using the Blackboard approach as more is learnt about their use. After some analysis of how, and when, the

Tool	Functionality	Course			Release
ACE	Acceptance Criteria Elucidator		T		0.8
ANT	AssistaNt for Test plan generation		T	P	0.3
BULL	BUild pLanning tooL			P	-
COCK	COnfiguration Control Keeper		T	P	-
COW	COst profiling Wizard			P	0.2
CRIP Charter	CRIP Chart generator			P	-
ET	requirement Enhancing documentation Tool	R			0.2
Max	Requirements completeness MAXimiser	R			0.1
RAT	Risk documentation And profiling Tool	R		P	0.1
RP	Requirement Priority documentation tool	R		P	0.2
SRR	System Requirements Review report generator			P	-
TIGER	Tool to InGest and Elucidate Requirements	R	T		1.15
TPA	Task Planning Assistant			P	-
WORM	Wizzard for pOor Requirement reforMatting	R	T		-

Table 1: The initial set of PETS

management and technical information in the QSE is used in the SLC, the access to information was partitioned and Table 1 shows the tools that were identified as candidates for the initial suite.
Each tool is currently targeted for use in one or more of the following postgraduate courses

- **The Requirements (R) course** covers the ingestion, writing, elucidation, and the allocation of key words to requirements for ease of database searches.
- **The Test and Evaluation (T) course** covers the addition of acceptance criteria and plans for testing the requirements.
- **The Software Engineering Project Management (P) course** covers risk, priority, cost, allocation of requirements to builds, and configuration control.

Each tool accesses the same QSE database (and, in some cases, its own database) providing generic selection, reporting, and sorting functionality. The specific functionality provided by the initial version of each of the tools is outlined below in alphabetical order. In addition, each tool provides the appropriate printed reports.

Acceptance Criteria Elucidator (ACE)

ACE allows the user to create, view, add, and modify acceptance criteria for requirements in the QSE database. ACE also provides a printout of each requirement in the database together with its acceptance criteria. The main user interface screen for the tool is shown in Figure 1.

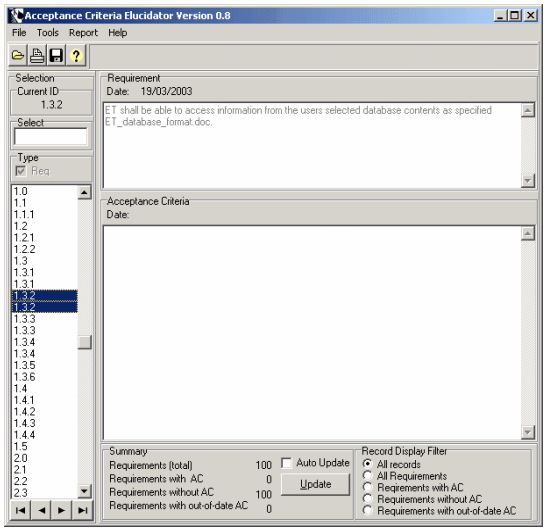

Figure 1: ACE

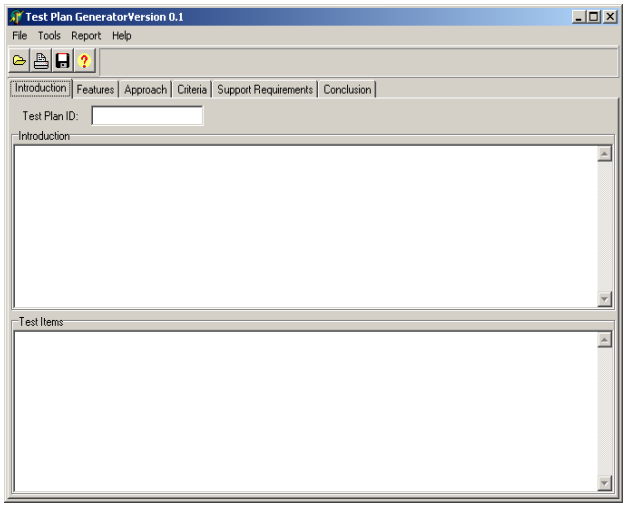

Figure 2: ANT

AssistaNt for Test plan generation (ANT)

This tool

- Creates sections of a test plan in accordance with IEEE Standard 829-1998.
- Facilitates grouping of tests.
- Links acceptance criteria to tests.
- Documents the estimated cost to perform the test.
- Documents traceability of the test to specific requirements.
- Documents the test equipment needed.
- Documents the schedule for the tests.
- The main user interface screen for the tool is shown in Figure 2.

BUiLd pLanning tool

This tool allows the user to create, view, document and modify

- The Builds to which requirements are assigned.
- The estimated cost of the Build.
- The priority of the requirements assigned to the Build.

COnfiguration Control Keeper (COCK)

This tool allows the user to view, modify and create, the following information

- Change request unique identification number.
- Criteria to show that change has been implemented.
- Product/Build Identification.
- Change request text.
- Change request Disposition.
- Change request Priority.
- Impacted requirements.
- Product/Build Number in which change will be implemented.
- Request allocated to CCB meeting date.
- Actual date of CCB Meeting.
- Reason for acceptance or rejection.
- Key words.
- Impact assessment identifier.
- Reporting functions for configuration control information.

COst profiling Wizard (COW)

This tool, shown in Figure 3, allows the user to create, view, document and modify

- An assigned cost to each requirement.
- The rationale for the cost.
- The accuracy of the cost.
- The total cost (high, mean, and low) of the set of requirements.

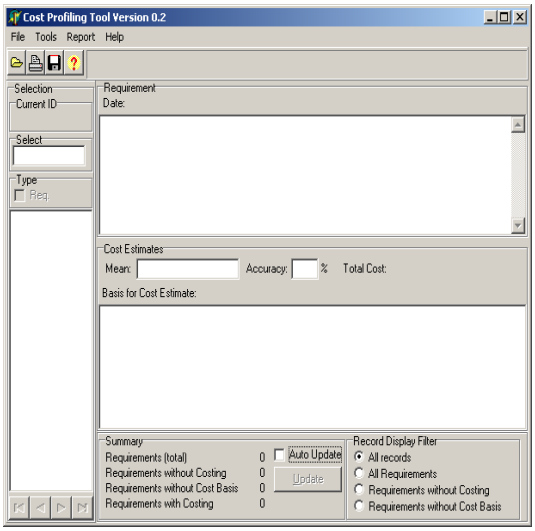

Figure 3: COW

CRIP Chart generator

This tool allows the user to view, modify and create Categorized Requirements in Process (CRIP) charts for measuring project progress (Kasser 1997). The CRIP approach is to:

- Identify a number of categories (e.g. complexity, risk, estimated cost, priority, etc.).
- Quantify each category into ranges (e.g. 1-10, A-J, etc.).
- Assign each of the requirements into one or more categories.
- Place each requirement into the assigned range in its category (e.g. complexity range 3, cost range 4, priority range 5).
- Monitor the changes in the state of the production work attributed to each of the requirements between the SLC reporting milestones and look for specific items discussed in Kasser (1997) to answer the question "how much of my project has been completed?"

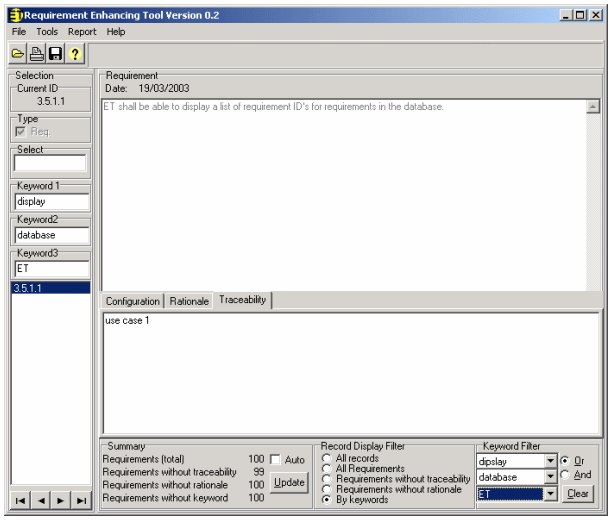

Figure 4: ET

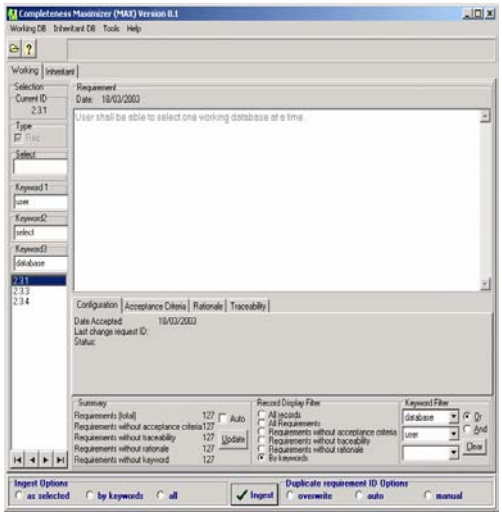

Figure 5: MAX

requirement Enhancing documentation Tool (ET)

ET shown in Figure 4, provides for the viewing, addition, modifying, and reporting, of the following QSE

- Keywords
- Rationale for the requirement
- Traceablity to source(s) and sideways

Requirements completeness MAXimiser (MAX)

The main user interface screen for the tool is shown in Figure 5. Max allows the user to add, view, and modify requirements, and to inherit requirements from a second QSE or requirements database. Max can be used to inherit requirements from a similar system. For example:

- Some of the requirements for one PET can be inherited from a previous tool database (e.g. user interface requirements).
- When developing requirements for a specific type of system (e.g. communications satellite), non-functional requirements for the object class of communications satellites can be inherited (thermal vacuum, humidity, and vibration) hence maximizing the completeness of the requirements for the specific instance being constructed.

Risk documentation And profiling Tool (RAT)

This tool allows the user to view, document and modify

- An assigned risk (1-10) to each requirement.
- The rationale for the risk.

- The risk mitigation strategy.

The tool also displays the total risk as a Pareto chart to provide a visible risk profile as shown in Figure 6.

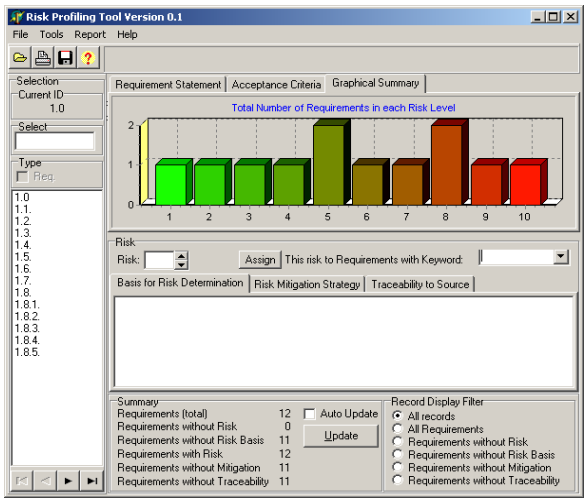

Figure 6: RAT

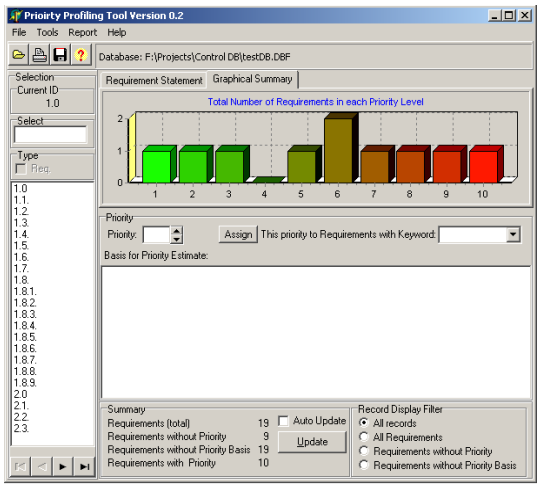

Figure 7: Requirement prioritising tool

Requirement priority documentation tool (RP)

This tool allows the user to view, document, and modify, an assigned priority (1-10) to each requirement. The tool also shows the total priority profile as a Pareto chart identifying the number of requirements in each range of priority. The main user interface screen for the tool is shown in Figure 7.

System Requirement Review (SRR) report generator

The SRR report generator provides reports on QSE information provided by other tools in the suite. These are

- Requirements and acceptance test criteria from ACE.
- TIGER Figure of Merit.
- System Risk Profile.
- Task plans.
- Test plans.
- Cost estimates.
- Build plans.
- CRIP charts.

Tool to InGest and Elucidate Requirements (TIGER)

TIGER performs the following functions

- Ingesting of requirements from text documents and the keyboard.
- Modification of existing requirements.
- Elucidating requirements based on a set of “poor words” and points out up to six types of (potential) defects in each of the requirements (multiple requirement in a paragraph, possible multiple requirement, unverifiable requirement, use of “will” or “must” instead of “shall”, and the existence of a user defined “poor word”).
- Allowing for additional “poor words” to be added as they are identified.
- Allowing for “poor words” to be used in a requirement when their use is appropriate. For example, the requirement that “the system shall display the

combined total of A and B" is a good requirement.

- Providing a built-in agent using deterministic grammar for the engineering of requirements (BADGER) that facilitates the correct format for writing requirements by, prohibiting many "poor words", and minimizing the need for retyping by the use of drop down lists (Scott 2003).
- Producing a report documenting each occurrence of a "poor word" in the requirements.
- Producing the FRED FOM.

The main user interface screen for the tool is shown in Figure 8.

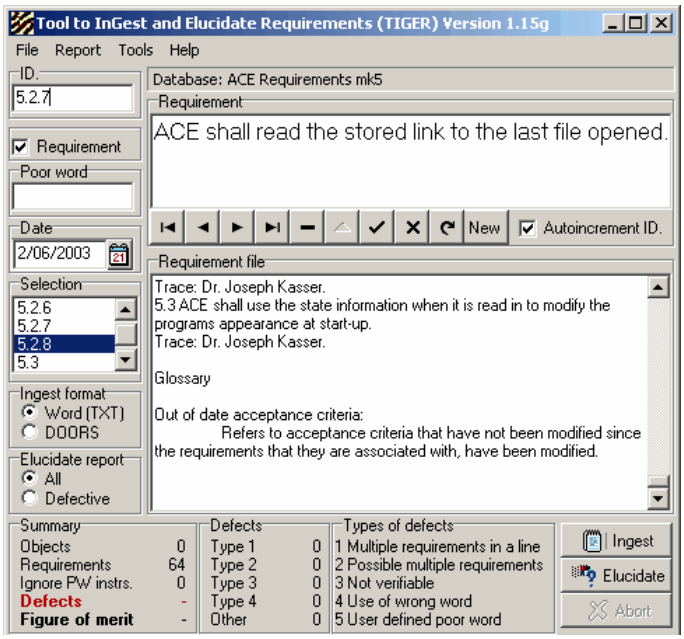

Figure 8: TIGER

Task planning assistant (TPA)

For each task in the project, this tool allows the user to create, add, view, and modify the

- Name of the task;
- WBS number;
- Requirements (identifications), upon which the completion of the task is derived;Priority of the task, based on the priority of the requirement;Name of person responsible for performing the task;WBS subtask elements within the task;
- Name of person generating the task;
- Reasons why the task is being performed;Key milestones;
- Previous (prerequisite) tasks;
- Subsequent (dependent) tasks;
- Decision points (what decisions need to be made and why);
- Risks;
- A written narrative description of what the task will accomplish and an outline of how the task will be performed;
- An estimated time frame (schedule) to perform the task (start and completion dates, minimum, most probable, and worst case times);
- A list of pre-requisites governing the start of the task (products);

- A list of personnel resources needed for the task (skills and skill-levels);
- Extent and level of effort by each personnel resource (team member);
- Cost estimates to perform the task (minimum, most probable, and worst case);
- A list of equipment resources needed to perform the task;
- A list of anything else needed to perform the task;
- A list of the products produced by the task;
- A suggested list of evaluation criteria for measuring the effectiveness of the task.

Wizard for poor Requirement formatting (WORM)

This tool assists test planning and facilitates ensuring the completeness of testing by converting written requirement paragraphs containing multiple requirements into separate requirements. As an example of the work that this tool can expedite, consider the following requirement (ST-DADS 1992):

204.1 DADS shall automatically maintain statistics concerning the number of times and the most recent time that each data set has been accessed. These same statistics shall be maintained for each piece of media in the DADS archive.

The tool splits this requirement into the following four requirements to simplify tracking the completeness of the test plans:

204.1a DADS shall automatically maintain statistics concerning the number of times ~~and the most recent time~~ that each data set has been accessed. ~~These same statistics shall be maintained for each piece of media in the DADS archive~~.

204.1b DADS shall automatically maintain statistics concerning ~~the number of times and~~ the most recent time that each data set has been accessed. ~~These same statistics shall be maintained for each piece of media in the DADS archive.~~

204.1c DADS shall automatically maintain statistics concerning the number of times~~ and the most recent time~~ that ~~each data set has been accessed. These same statistics shall be maintained for~~ each piece of media in the DADS archive [has been accessed].

204.1d DADS shall automatically maintain statistics concerning ~~the number of times and~~ the most recent time that ~~each data set has been accessed. These same statistics shall be maintained for~~ each piece of media in the DADS archive [has been accessed].

Leaving the sections of the requirement that were not being tested in place but stricken through clearly identifies which section of the requirement is being tested. An unfortunate side effect is that it also clearly shows the defects in the requirement. Note that the phrase '`has been accessed`' has been moved in the last two sub-requirements to clarify the sub-requirement.

Educational materials

The purpose for the tools is that they are to be introduced in the educational environment. Hence the following additional documentation is being generated for each tool.

- Help files in Windows 'hlp' format
- Installation software
- Sample databases and documents
- PowerPoint presentation and lecture notes for use of tools in class.

The Benefits of the PETS project

The benefits of the PETS project are, it should:

- Reinforce the concept and usefulness of the QSE by providing a way to use them.
- Provide tools for the students to use in class and in the workplace.
- Provide new views of integrated management and technical information, e.g., cost and priority.
- Allow (new) questions to be raised before major project funds are committed, e.g.:
- What is a good risk profile for a project?
- Does customer really want a low priority high cost requirement?
- Provide a baseline for a platform of opportunities for further research in areas such as complex systems, artificial intelligence, expert systems, project management and education. Follow-up on the use of the tools in the workplace by the students will be required to determine if the suite of tools makes a difference.

Early results

The tools are still under development. TIGER was the first tool to be used in a class lecture on requirements engineering in three postgraduate courses. Before TIGER, the discussions in the tutorials focussed on the structure and format of requirements. After TIGER was introduced and used to elucidate sample requirements, the focus of the in-class discussions changed to cover the difficulties of writing good requirements. Further research is in progress.

Summary

This paper briefly discussed the problem of poor requirements and how, while solutions are known, they are not being implemented in industry. The paper then discussed an approach for transferring the solution from academia to industry by using the Blackboard principle of Artificial Intelligence to provide a suite of evolving educational tools based on an object-oriented approach to requirements engineering and management within an information systems paradigm. The paper then gave an overview of the functionality provided by the tools and concluded with a list of the potential benefits of the project and some early results of the use of the TIGER tool.

Availability of the PETS

The PETS that have been tested in the educational environment, and released, can be obtained by accessing the Systems Engineering and Evaluation Centre website (http://www.seec.unisa.edu.au) and following the link to Software Tools. The PETS are provided free for educational purposes.

References

Alexander, I. F., Stevens, S. (2002). *Writing Better Requirements*, Addison-Wesley.
Carson, R.S. (2001). "Keeping the Focus During Requirements Analysis", *Proceedings of the 11th International Symposium of the International Council on Systems Engineering (INCOSE),* Melbourne, Australia.
Glover, S.J., Bennett, K.H. (1996). "An Agent-Based Approach to Rapid Software Evolution Based on a Domain Model", *Proceedings of the International Conference on Software Maintenance*, page(s): 228 –237.
Hooks, I. (1993). "Writing Good Requirements", *Proceedings of the 3rd International Symposium of the National Council on Systems Engineering (NCOSE),* retrieved from http://www.incose.org/rwg/writing.html, *on* November 1, 2001.
Hull, M.E.C, Jackson, K., Dick, A.J.J. (2002). *Requirements Engineering*, Springer.
Jacobs S. (1999). "Introducing Measurable Quality Requirements: A Case Study", *IEEE International Symposium on Requirements Engineering*, Limerick, Ireland.
Kasser, J.E., Schermerhorn R. (1994). "Determining Metrics for Systems Engineering", *Proceedings of the 4th International Symposium of the NCOSE*, San Jose, CA.
Kasser, J.E. (1995). *Applying Total Quality Management to Systems Engineering*, Artech House.
Kasser, J.E. (1997). "What Do You Mean, You Can=t Tell Me How Much of My Project Has Been Completed?", *Proceedings of the 7th Annual Symposium of the INCOSE,* Los Angeles, CA.
Kasser, J.E. (2000). "A Framework for Requirements Engineering in a Digital Integrated Environment (FREDIE)", *Proceedings of the Systems Engineering, Test and Evaluation Conference*, Brisbane, Australia.
Kasser, J.E. (2002). "A Prototype Tool for Improving the Wording of Requirements", *Proceedings of the 12th International Symposium of the INCOSE,* Las Vegas, NV.
Kasser, J.E., Cook, S.C. (2003). "Using a Rapid Incremental Solution Construction Approach to Maximize the Completeness and Correctness of a Set of Requirements for a System", *Proceedings of the 13th International Symposium of the INCOSE, Washington* D.C.
Kasser, J.E., Williams, V.R. (1998). "What Do You Mean You Can't Tell Me If My Project Is in Trouble?", *First European Conference on Software Metrics (FESMA 98)*, Antwerp, Belgium.
Kasser, J.E., Williams V.R. (1999). "The Student Enrollment and Course Tracking System Meta-Project", *Proceedings of the Portland International Conference on Management of Engineering and Technology (PICMET)*, Portland, OR.
Scott. W., (2003). Personal communications and example of prototype from Ph.D. in progress.
Standish (1995), *Chaos*, The Standish Group, *retrieved from* http://www.standishgroup.com/chaos.html, on March 19, 1998.
ST DADS (1992). Requirements Analysis Document (FAC STR-22), Rev. C, August 1992, as modified by the following CCR's:- 139, 146, 147C, 150 and 151B.
Voyages (1996), "Unfinished Voyages, A follow up to the CHAOS Report", The Standish Group, retrieved from http://www.pm2go.com/sample_research/unfinished_voyages_1.asp, January 21, 2002.

Acknowledgements

The authors acknowledge Professor Stephen Cook's suggestions for a number of the acronyms for the PETS, and William Scott's sharing of concepts from his PhD research.