

The Case for Applying an Early-Lifecycle Technology Evaluation Methodology to Comparative Evaluation of Requirements Engineering Research

Martin S. Feather
Jet Propulsion Laboratory
California Institute of Technology
Martin.S.Feather@Jpl.Nasa.Gov

Abstract

The premise of this paper is that there is a useful analogy between evaluation of proposed problem solutions and evaluation of requirements engineering research itself. Both of these application areas face the challenges of evaluation early in the lifecycle, of the need to consider a wide variety of factors, and of the need to combine inputs from multiple stakeholders in making these evaluations and subsequent decisions.

Approaches to comparative evaluation have been developed and applied by the requirements engineering community, so we should seek to learn from and, when appropriate, reuse these same approaches in the comparative evaluation of requirements engineering research.

An example of such is the quantitative early-lifecycle design evaluation methodology that we have developed and used successfully at JPL for evaluating a variety of technologies, including hardware, software and combinations of both. We briefly summarize this evaluation methodology, including the ways in which it has proven successful. We indicate how it might be adopted for the purposes of evaluating requirements engineering research products.

1. Analogy between problem solution evaluation and requirements engineering research evaluation

The workshop focus is the evaluation of requirements engineering research itself. We argue that this is analogous to one of the main themes of requirements engineering, namely early-lifecycle comparative evaluation of proposed problem solutions. Both application areas of evaluation share the following fundamental challenges that make evaluation problematic:

1.1. Temporal separation between decisions and ramifications

Decisions are far removed from the ramifications of those decisions, making customary approaches to evaluation problematic. For example, a novel technology that appears promising as the means to solve some problem may well require considerable engineering to mature it for real use. It will not exist for some time in a form upon which measurements directly reflective of its final use can be taken. Nevertheless, in the interim there is the need to make key decisions on whether to choose that technology over some other, and decisions on how to go about its maturation.

1.2. Wide variety of concerns (functional and non-functional requirements)

A wide variety of concerns must be addressed to complete the long journey from concept to implementation. In software development, we are familiar with the need to address both “functional requirements” and “non-functional requirements (NFRs)” (the multiple “-ilities”, e.g., usability, maintainability, reliability) required of successful implementations. An equally diverse mix of needs is needed for requirements engineering research to be successful. For example, the “functional” requirements of an analysis tool for checking the consistency of requirements include the requirement that it yield accurate results (identify a set of requirements as consistent if and only if they are indeed consistent), and performance measures such as speed, memory usage etc. These would be the measures expected of a typical benchmark, as proposed by the workshop’s call for papers. The preceding item raised the problematic issues of gathering such measurements from research early in its conceptual phases. Additionally, there are equivalents of “non-functional requirements” to take into account, such as usability (how much effort does it take to apply the analysis method?), trainability (what skills are required of

would-be users, and how feasible is it to instill them through training?), maintainability (what does it take to keep the analysis method up-to-date as the target language evolves?) and compatibility (in what form are inputs expected to be provided to the method, and what forms of output is it capable of producing? on what computing platform(s) does it run?).

1.3. Multiple stakeholders and their variety of interests

Many within the requirements engineering field have stressed the importance of recognizing all the stakeholders involved in the adoption of some system, and accommodating their diverse (and often contradictory) needs and wishes. The same phenomenon is equally prominent in determining the successful uptake of research of all kinds. The researchers themselves have their own skills and motivations (e.g., to develop a tenure-quality track record through publication). The funding agencies supporting the research may be prepared to support financially only a portion of the lifecycle phases that lead from research concept to usable product. The end uses may wish for ancillary products (e.g., documentation, user's guides, courseware, on-line help, continuing support, users' groups) that would not normally be the outcome of a research effort.

Recognizing all these stakeholders is a necessary precursor to eliciting from them their interests in the object of the evaluation.

1.4. Sufficient competitive advantage

New products face the need to sufficiently differentiate themselves from standard practice, *as it will be at the time they are poised for adoption*. For example, large commercial investments in software engineering tools and techniques become de-facto standards, no matter the supposed advantages that a more theoretically grounded approach offers. Thus new products must either augment those emerging standards (meaning that they will be forced to live with the strictures of those standards, desired or not), or be sufficiently advantageous as to justify their adoption despite their incompatibilities with standard practices.

The same phenomenon holds true of research advances: a result of sufficient merit to lead to peer-reviewed publication may not necessarily be capable of displacing an inferior but already widely accepted approach. There is continual competition for the limited resources that can be invested in "improvements": to be viable, a candidate must be able to make a plausible case for a sufficiently impressive "Return On Investment (ROI)" that its adoption will yield.

2. Improving technology infusion at JPL and NASA through early-lifecycle evaluation

Our experience with these real-world challenges stems from ongoing work at JPL and NASA to better infuse new technologies into space missions. In this section we summarize the risk-centric basis for our approach to the study and improvement of technology infusion.

Section 3 provides the details of the evaluation process, and Section 4 then relates this approach specifically to the workshop theme of evaluation of requirements engineering research.

2.1. Use of a risk-informed decision process to aid technology infusion

Ken Hicks at JPL studied the track records of technology infusion into NASA space exploration missions, and judged that the rate at which new technology becomes employed missions has room for significant improvement. There is a recognized technology infusion "gap" between early lifecycle concepts, and mature, dependable products. Many seemingly promising technologies fail to cross this gap. Ken Hicks identified the primary impediments to stem from imperfect formulation and communication of requirements, insufficient attention paid to the stringent engineering needed to demonstrate flight readiness, and lack of consideration of competitive alternative solutions. These are clearly instances of the challenges discussed in the preceding section.

The "Technology Infusion Maturity Assessment (TIMA)" process has been developed to overcome these impediments. The success of the TIMA process hinges on the combination of (1) human experts to provide knowledge, insight and guidance, (2) an organized method for conducting the assessment effort, and (3) customized software to support the process steps and human decision making activities. The TIMA process has been used successfully at JPL for evaluating a variety of technologies, including hardware, software and combinations of both.

The TIMA process makes use of a risk-informed decision support tool, "Defect Detection and Prevention (DDP)". TIMA's technology infusion evaluation studies have been the leading users of DDP [Cornford et al, 2001]. Other applications have included overall risk management for an ongoing space flight mission, quality assurance planning, activity selection across an entire program of NASA Earth Science Missions [Tralli, 2003], and some preliminary investigations into matching practitioner needs to research activities [Feather et al, 2003]. DDP originated from Steve Cornford's vision of a structured method for quality assurance planning of

hardware systems [Cornford, 1998]. He considered quality assurance activities to be “risk filters” (i.e., the activities either reduce or remove risks that would otherwise threaten mission success). The risk-centric Defect Detection and Prevention (DDP) software tool [Feather et al, 2000] resulted from this vision. DDP, and the thinking that underlies it, is central to our studies of technology infusion. A brief summary of its key elements follows. For full details, see [Feather&Cornford, 2003].

Briefly, DDP relies on quantitative assessments of the relationships between three classes of information:

Requirements (a.k.a. “Objectives” or “Goals”) – the things the system needs to accomplish (includes constraints on its operation and development),

Failure Modes (a.k.a. “Risks”) – all the things that could occur that would negatively impact or limit the attainment of Requirements, and

Preventative Measures, Analyses, Controls and Tests (PACTs) (a.k.a. “Mitigations” or “Solution Options”) – all the things that could be done to reduce the likelihood and/or severity of Failure Modes.

The quantitative assessments are of:

Impacts – the proportions by which the Failure Modes, should they occur, will limit the attainment of Requirements, and

Effects – the proportions by which the PACTs, should they be applied, will reduce Failure Modes (and so lead to greater attainment of Requirements).

2.2. Benefits of the TIMA process for technology evaluation

The risk-informed TIMA process has proven useful for technology infusion evaluations. Generally, the results match the expectations of the experts involved in these studies. This lends credence to the validity of the TIMA process. Where the process adds value is in the surprises it has been able to reveal. In almost every study some of the results are, at first glance, surprising to the experts. For example, a Failure Mode that was not anticipated to be particularly problematic turns out to be so. Because of the detailed risk-informed underpinnings, the experts can quickly scrutinize the details and confirm that the surprise is not a mistake, but a reflection of the data that they have entered. The value of the TIMA process is that it emerges with these surprising results *at this early stage*, allowing for compensatory actions to be taken before things progress a long ways down an inferior route (after which correctios tend to be far more expensive to perform).

A typical technology evaluation involves anywhere from half a dozen to twenty experts, and is organized as three or four half-day sessions in which all those experts participate, to provide the information, and perform the decision making. Because the process is reliant upon the estimates those experts provide, it is important to get the

most qualified participants possible. This makes the process somewhat expensive to perform – if 10 people are involved in four half-day sessions, that adds up to a total of 20 work days of (skilled peoples’) time. Nevertheless, the savings that we have seen from early identification fo problems far outweigh the cost of these studies.

3. Key Steps of the TIMA process for Technology Evaluation

The key steps the TIMA process for technology evaluations are listed in this section. Note that although they are presented in sequential order, it is not necessary to strictly adhere to this order, and in practice it is in fact relatively common to return to earlier steps on recognizing that something was overlooked or needs to be explored in greater detail.

3.1. Establishing the stakeholders in the technology.

Stakeholders include those with the most to gain by infusion (e.g., flight project technologist, technology researchers themselves), and relevant subject area experts for the design, development and deployment of the technology (e.g., experts in avionics, packaging, manufacturing and test, experiment design, failure analysis, materials, quality assurance).

3.2. Identifying the requirements that the technology must meet

These are all the requirements that must be seen to be satisfiable before mission designers & managers will have adequate confidence to infuse the technology into a flight project. These requirements encompass high-level mission requirement, schedule, budget and other resource requirements, development requirements, and detailed functional requirements specific to the required technology performance for its intended mission (akin to functional requirements).

Since not all requirements are equally important, requirements are assigned “*weights*”, reflecting their *relative* importance (e.g., a requirement assigned a weight of 10 is twice as important as a requirement assigned a weight of 5).

3.3. Determining the potential, relevant “Failure Modes”

“Failure Modes”, or risk elements, are all the concerns that could negatively impact the desired performance of the technology as a result of issues that range from non-thorough definitions of design and performance requirements, to ineffective fabrication/ assembly materials and methods, to inadequate test processes for verification and validation of the specified

performance/reliability of the product, and to shortcomings of programmatic and institutional resources and/or infrastructure. Also done in this step is a quantitative assessment of *how much* each Failure Mode can *impact* the requirements (i.e., what proportion of a given requirement will be lost if the Failure Mode occurred). The aggregation of this information identifies “tall pole” Failure Modes – those that most threaten the Requirements.

3.4. Identifying PACTs that can reduce the risk of failure

PACTs include practices and procedures involving design, fabrication, assembly and functional characterization by testing or diagnostic exercises that most likely will be required for advancing the flight technology. As for Requirements and Failure Modes, these are elicited from the experts. Also done in this step is an assessment of *how effective* each PACT will be in reducing each Failure Mode (e.g., chance of detecting or preventing the Failure Mode), and *how costly* each PACT will be to implement.

3.5. Decision-making aided by the DDP tool and its risk-based calculations.

The previous steps have all been for the purpose of *gathering* information. In this step the experts use that gathered information to help them make their decisions. The risk-centric nature of DDP gives the experts insight into which are the most serious impediments to success, namely those Failure Modes with the greatest sum total impact on requirements. The experts’ primary goal is a cost-effective selection of PACTs that together will sufficiently diminish the Failure Modes, and so lead to adequate attainment of Requirements. Since PACTs cost resources, they must be chosen judiciously. To aid in this, DDP provides heuristic search to locate near-optimal PACT selections for a given cost level.

It is possible that the gathered information reveals there to be *no* satisfactory selection of PACTs. Perhaps the desired levels of risk reduction requires PACTs whose sum total costs exceed the resources available, or even that there are Failure Modes against which only weakly effective PACTs have been identified. In cases such as these the technology being evaluated may need to be rethought, perhaps giving preference to a competing technology that evaluates to be more cost-effective. We have encountered instances where the outcome has been a change in the *requirements* – while the technology may have proven unsuited to the originally posed set of requirements, by abandoning those of the requirements that emerge as the most problematic (i.e., impacted by hard-to-reduce Failure Modes), the technology might be feasible for a different range of applications to those

originally anticipated. If the impediment is primarily cost-driven (i.e., a feasible solution exists but is too costly), the information can be used to make a defensible case for a greater-than-intended investment.

3.6. Documenting and reporting

The TIMA findings and suggested recommendations for stakeholders need to be documented and communicated to readers beyond just those present in the TIMA sessions themselves. The end results of a TIMA evaluation are accompanied by the supporting DDP information. This permits subsequent review of the decision-making and offers detailed guidance on the course the technology development should follow. Should circumstances change (e.g., a PACT intended to be applied become infeasible), the decision-making can be revisited, using the DDP information to guide an alternate selection.

4. Use of a risk-informed decision process for evaluation of requirements engineering research

This section considers how the steps of the TIMA process for technology evaluation would apply to the workshop’s goal of evaluation of requirements engineering research. Illustrations are provided using a hypothetical example, that of evaluation some (unspecified) form of research into model-checking technology for requirements engineering purposes. These are to be read only as indicators of the kinds of issues that this approach deals with, and do not necessarily correspond to findings of a realistic assessment. For a more extensive examination of model checking in this manner, see [Feather, 2002].

4.1. Establishing the stakeholders in the research.

Just as the technology infusion process required identification of all the stakeholders involved in maturing that technology to mission use, evaluation of requirements engineering research necessitates identification of all the stakeholders involved in the maturation of that research from concept to ultimate application(s). These include the ultimate beneficiaries of the advance conveyed by the successful research, the end-users/operators, developers who will be required to bridge the gap from concept to product, funding sources that will pay for its maturation, and managers who will be held accountable for the decision to select the research.

Note that the workshop might not necessarily have among its participants representatives of all these stakeholders, in which case some of the evaluation steps might need to be deferred, or explored for plausibility of

the evaluation approach, but not for final decision-making.

Model-checking examples: the researchers with the model checking expertise need to be committed to carrying the research through towards a usable product, and may need to continue their involvement over a long timeframe. If the maturation requires a series of increasingly life-like case studies, applications, etc., then the domain experts in the relevant subject areas must be involved. Very likely there will need to be an ongoing and detailed partnership between these two sets of stakeholders.

4.2. Identifying the requirements that the research must meet.

There are a numerous requirements for a research concept to mature to become a viable method / tool / standard / framework /... These include details on how the end-product of the research will ultimately be used, and how the research will mature from its current status towards that end product (i.e., requirements on the end product, and requirements on the development of that end product).

Model-checking examples: requirements stem from the intended purpose of the model checking – e.g., validating requirements, finding bugs in designs, verifying properties of source code, automatic generation of test cases, etc. A particular form of model checking research will not likely lends itself equally well to all these kinds of applications. Capability requirements include: size of problem that can be analyzed, speed, platform, nature of the formalism over which model checking is performed (e.g., Java byte code). Funding requirements may necessitate a series of small steps demonstrating progress at quarterly intervals, say.

4.3. Determining the potential, relevant “Failure Modes”

These are all the potential impediments that could negatively impact the ability of the research to be successfully matured to meet the requirements. The purpose of this step is to elicit these, and related each of them to the requirement(s) it would obstruct, indicating the strength of that obstruction. Getting to the completion of this stage yields Requirements, Failure Modes, and a quantitative linkage among them. In technology infusion studies, this is often a valuable intermediate result. Its attainment forces the disciplined consideration of requirements (something often recommended by the requirement engineering community!), and of the broad range of obstacles to those requirements. The emphasis throughout is on breadth of coverage. The qualitative linking of Requirements to Failure Modes is done to only coarsely distinguish magnitude of obstacles (e.g., sufficient to characterize a Failure Mode’s obstruction of

a requirement as 100% obstruction, 90%, 70%, 30%, 10% or 0% – the default unless indicated otherwise). If it seems impossible to assign even so coarse an estimate of magnitude, then that is usually an indication of the need to subdivide either or both of the requirement or Failure Mode into greater detail. Likewise, disagreement about the value is almost always indicative different parties referring to different circumstances, the solution to which is again to subdivide.

The net result of this quantitative linkage of Requirements to Failure Modes is often a source of some surprises. While experts’ intuition is usually correct overall about what are the key Failure Modes, it is not uncommon to find that some of them turn out to be strikingly less, or more, severe than would have been anticipated. For the workshop’s purposes of research evaluation, similar such surprises would correspond to over- or under-estimated obstacles to the success of the research in question.

Model-checking examples: end-users may be unfamiliar with the model checking specification language (e.g., for SPIN this would be Promela) and the language for expressing temporal properties (e.g., Linear Temporal Logic – LTL). Researchers may not be sufficiently familiar with the application domain to know what details can safely be abstracted. Lack of predictability of how long it will take to apply model checking to a problem will be likely be a serious impediment to planning for its role in a typical development effort.

4.4. Identifying PACTs that can reduce the risk of failure

These are all the activities that could be considered for reducing the Failure Modes (i.e., overcoming those identified potential impediments to research maturation).

Model-checking examples: develop a training course to teach would-be users the model checking language. Make use of automated translation into the model checking language (e.g., there is a variety of work that is aimed at automate translation from statecharts or other design artifacts into input for model checkers). Arrange for *funded* pilot studies to involve both researchers and domain experts working in partnership. Have researchers keep careful track of the time it takes them to tackle new problems.

4.5. Decision Making

The TIMA process’ decision-making step uses the information gathered in the preceding steps to help the experts decide whether the technology is viable, what PACTs (activities) are needed to mature it to use, etc. In applying this process to evaluation of research, this decision making step is the time at which the research is

critically evaluated with respect to its requirements, and what it would take to attain those requirements.

Model-checking examples: in comparing two model checking methods, it is revealing to see what Failure Modes (obstacles) prove the most problematic for each method, and what “PACTs” (means to overcome obstacles) will be needed to render each of the methods viable.

4.6. Documenting and Reporting

The final step of a TIMA evaluation is thorough documentation of the findings. For the workshop purpose of investigating how to go about research evaluation (but not necessarily carrying any specific evaluation through to completion), this step need not be considered in much detail. Nevertheless, it is important that any evaluation method be capable of yielding the information from which thorough documentation can be generated. Better yet, it should be more than a “static” document. The TIMA process, through its use of the DDP tool, culminates in a set of information that can be viewed and manipulated using the DDP tool, allowing for “what-if” investigations, etc., long after the original evaluation was completed.

5. Workshop activity using this approach

We suggest that it is feasible to consider trying this approach *at the workshop itself*. To do this would require:

1. Selecting a research method and application area with which the workshop participants are reasonably familiar. Note that we only require that *some* of the participants have knowledge of the research method itself – these participants can serve in the role of the researchers knowledgeable in that method; other participants would be needed to represent the would-be users of the method, the (perhaps skeptical) funders who have to pick and choose with care the research that they support, the developers/trainers (as appropriate) etc. The most critical need is for coverage of the different areas of expertise that all (or as many as possible) of the different stakeholder groups would have.
2. Following several steps of the TIMA process. In the limited time available it will not be possible to complete an entire evaluation of a research method, however several of the key steps can be explored in sufficient detail to give a good feel for what is involved. In particular, it is important to experience elicitation of some number of the Requirements, Failure Modes and PACTs,

and (some of) the quantitative relationships among these.

The DDP software tool that underpins the TIMA process is designed to support on-the-fly capture of this form of information, elicited from experts in group sessions. This would be ideally suited to a workshop session. The way this is usually done is to have the DDP tool projected onto a screen visible to all. This serves as the focal point for discussions, allowing all involved to see the information gathered to date, and to see the currently active area of information elicitation or decision making. A single user “drives” the DDP tool, so the other participants are not required to be experts in controlling it through its interface. DDP’s main concepts are relatively straightforward: trees structures are used to hierarchically organize the three sets of information - Requirements, Failure Modes and PACTs; matrices between Requirements and Failure Modes, and between Failure Modes and PACTs, are used to capture the quantitative relationships among these. Simple bar charts are employed to convey the magnitude of various calculated values (e.g., for every Failure Mode, the sum total impact it has on Requirements). The net result is that with a small amount of introductory explanation, viewers of the DDP screens are quickly able to grasp the information content of those screens. See the appendix for some examples of DDP documentation of these aspects.

6. Related Work

In this section we briefly summarize some other approaches to early-lifecycle evaluation.

Early decision-making is often assisted by *qualitative* decision support methods. Quality Function Deployment (QFD) is prominent among these, and has been used in a wide variety of settings [Akao, 1990]. DDP’s effect and impact matrices are reminiscent of the Relationship Matrix used in many forms in QFD. DDP is distinguished by its foundation upon a *quantitative* risk model, which gives meaning to DDP’s cost and benefit calculations.

The requirements engineering research community has shown increasing interest in models of “goals” (roughly speaking, precursors to requirements). See the mini-tutorial [van Lamsweerde, 2001] for an overview of this area. We discuss two of these kinds of models:

The KAOS framework for goals, requirements, etc. [Bertrand et al, 1998] is used to build a logical structure of how system-wide requirements decompose to, ultimately, requirements on the individual components in a system. Models built in this framework seem well suited to exploring the functional behavior, and to some extent, non-functional aspects. DDP models are weaker in that they lack the logical structure of KAOS models, but conversely have emphasized more the quantitative aspects

that predominate in imperfect solutions.

The i* framework [Chung et al, 2000], [Mylopoulos et al, 2001] combines logical structures with *qualitative* models. Their combination rules for these qualitative models support well tradeoff analysis between major design alternatives. DDP models seem more appropriate when there are a large number of small alternatives.

7. Conclusions

We have summarized an approach to technology evaluation that has seen use over the last few years in evaluating spacecraft technology concepts and helping plan their maturation towards flight ready technologies. The key to this approach is the use of a risk-centric model that connects Requirements to “Failure Modes”, and Failure Modes to “PACTs” (activities to diminish the adverse impact and/or likelihood of those Failure Modes). This is applied in a disciplined way to gather the wide range of information that must be taken into account if a technology concept is to be carried all the way through to use.

This approach, we have suggested, is also applicable to the evaluation of requirements engineering research methods. We have described how the steps of this approach to technology evaluation carry over to research method evaluation. Finally, we have outlined how the approach could be tried out at the workshop itself.

Overall, this approach takes a much broader look at the question of research evaluation than the benchmark focus of the workshop description. We feel that benchmarks have an important role, but if used in isolation would not suffice to encompass the wide range of concerns that arise in successful utilization of research results. We hope that this paper encourages others to think on how evaluation might employ such a broader perspective.

8. Acknowledgements

The research described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology under a contract with the National Aeronautics and Space Administration.

Dr. Steve Cornford leads the DDP effort at JPL. Ken Hicks has developed and applied the TIMA process that makes use of DDP for technology evaluations.

9. References

[Akao, 1990] Y. Akao. “*Quality Function Deployment*”, Productivity Press, Cambridge, Massachusetts, 1990.

[Bertrand et al, 1998] P. Bertrand, R. Darimont, E. Delor, P. Massonet and A. van Lamsweerde. “GRAIL/KAOS: an environment for goal driven requirements engineering”, *20th Int. Conference on Software Engineering*, 1998.

[Chung et al, 2000] L. Chung, B.A. Nixon, E. Yu and J.

Mylopoulos “*Non-Functional Requirements in Software Engineering*” Kluwer Academic Publishers, 2000.

[Cornford, 1998] S. Cornford. Managing Risk as a Resource using the Defect Detection and Prevention process. *International Conference on Probabilistic Safety Assessment and Management*, September 13-18, 1998.

[Cornford et al, 2001] S.L. Cornford, M.S. Feather & K.A. Hicks. “DDP – A tool for life-cycle risk management”, *IEEE Aerospace Conference*, Big Sky, Montana, Mar 2001, pp. 441-451.

[Feather, 2002] M.S. Feather. “A Quantitative Risk-based Model for Reasoning over Critical System Properties”, Proceedings of RHAS’02, International Workshop on Requirements for High Assurance Systems, Sept. 9, 2002, Essen, Germany: <http://www.sei.cmu.edu/community/rhas-workshop/proceedings.pdf>

[Feather et al, 2000] M.S. Feather, S.L. Cornford, M. Gibbel. “Scalable Mechanisms for Goals Interaction Management”, *Proceedings 4th IEEE International Conference on Requirements Engineering*, Schaumburg, Illinois, 19-23 Jun 2000, IEEE Computer Society, pp 119-129

[Feather et al, 2003] M.S. Feather, T. Menzies & J.R. Connelly. “Relating Practitioner Needs to Research Activities”, to appear in the 11th IEEE International Conference on Requirements Engineering, 2003.

[Feather&Cornford, 2003] Feather, M.S. & Cornford, S.L.. “Quantitative risk-based requirements reasoning”, to appear in *Requirements Engineering* (Springer), published online 25 February 2003, DOI 10.1007/s00766-002-0160-y. Available from: <http://eis.jpl.nasa.gov/~mfeather/AvailablePublications/>

[Mylopoulos et al, 2001] J. Mylopoulos, L. Chung, S. Liao, H. Wang and E. Yu. “Exploring Alternatives during Requirements Analysis”, *IEEE Software* 18(1), pp. 92-96, 2001.

[Tralli, 2003] D.M. Tralli. Programmatic Risk Balancing. *IEEE Aerospace Conference*, Big Sky MT, 2003.

[van Lamsweerde, 2001] A. van Lamsweerde. “Goal-Oriented Requirements Engineering: A Guided Tour”, *Proceedings 5th IEEE International Symposium on Requirements Engineering*, Toronto, Canada, August, pp. 249-263, 2001.

Appendix – DDP guide documentation

Some pictorial guidance documentation of the DDP tools’ tree and matrix windows is shown on the next page. These are provided to give an indication for the “look and feel” of the DDP tool in the way it displays the tree and matrix information that is needed for the evaluation exercise proposed herein. They also serve to help indicate that the DDP is itself a relatively mature vehicle for conducting such a research evaluation exercise.

Tree Window

Requirements (Target icon)

Failure Modes (Skull and crossbones icon)

PACTs (Plus icon)

Indicators of displaying
Checked, UnChecked, or Both

"AddNode" button:
click to add a new item
as a root or leaf

"Layout" button:
click to set font
sizes etc.

Tree type

Slot type

Slot values

| | |
|----|------------------|
| 5 | read-only |
| 8 | editable (click) |
| 10 | edit underway |

SLOTS (drag to resize)

TREE

* to be included in consideration, a node & ALL its ancestry must be checked

Subtrees

+ collapsed

- expanded (click to switch)

Focus (left-click sets it)

● node in focus

■ ancestry of node

Check boxes(*)

checked (on)

unchecked (off) (click to switch)

Matrix Window

"Layout" button:
fonts, etc.

Left mouse click's effect

Row titles

"Alert" status of cell

Drag header cell dividers to resize

Subtrees

[+] collapsed

[-] expanded (click to switch)

Col = Interface Issues

Row = Quality Control

| PACT x FM | | Col - Interface Issues | | | | | | | [-] Fund [-] Lit | |
|-----------|------------|------------------------|--------|-------|--------|----------|--------|------|------------------|--|
| | | FMs | Misapp | Wrong | Tolera | Interact | | | | |
| PACTs | PACTs | PACTs counts | 6 | 6 | 7 | 6 | 76 | 12 | | |
| Peer | | 15 | 0.9 | 0.3 | 0.7 | 0.9 | 0.9 | 0.9 | | |
| Componer | | 9 | | | | | 0.587 | 0.9 | | |
| [-]Prev | Qualify | 3 | | | | | | 0.99 | | |
| Measu | Environme | 9 | | | | | 0.387 | 0.9 | | |
| | Quality | 6 | 0.3 | 0.1 | 0.9 | 0.9 | 0.112 | 0.7 | | |
| | Implemen | 1 | | 0.99 | | | | | | |
| | Build | 13 | 0.9 | 0.3 | 0.9 | 0.3 | 0.737 | 0.9 | | |
| | Quality | 12 | 0.9 | 0.9 | 0.9 | 0.9 | 0.35 | | | |
| | FMECA | 1 | | | | | 0.0875 | | | |
| | [-]Perform | FMECA | 1 | | | | 0.112 | | | |
| | FMECA | FMECA | 1 | | | | 0.0875 | | | |

counts of non-zero values in rows subtree aggregate values

Rqmt x FM matrix: each cell holds **impact** value i :
 $0 \leq i \leq 1$: proportion of requirement lost if Failure Mode occurs

PACT x FM matrix: each cell holds **effectiveness** value e :
 $0 \leq e \leq 1$: proportion of FM reduced if PACT applied;
 $-1 \leq e < 0$: PACT *induces* risk by likelihood $|e|$
 $e < -1$ PACT *aggravates* risk by $|e|$

Blank = 0 Non-numeric values *... or ?... ignored in calculations.

Right-click cell for property editor

| | |
|--|--|
| | |
| | |