final
CL 01-2231

Cost-Benefit Based Assurance Planning

Martin S. Feather, Steven L. Cornford, Julia Dunphy
Jet Propulsion Laboratory, California Institute of Technology
4800 Oak Grove Drive, Pasadena, CA 91109, USA
{Martin.S.Feather, Steven.L.Cornford , Julia.Dunphy}@Jpl.Nasa.Gov

Abstract

An important aspect of critical systems design is the planning of the activities needed to assure the correctness of those systems. Design is inevitably resource-constrained, and by implication, so are assurance activities. Their planning must take into account both their costs, and their benefits.

We have extended an existing risk management framework with a refined cost-benefit model. Benefits are measured in terms of reduction of risk – both risk in the development process itself, and risk in the developed artifact. Cost calculations take into account both costs of performance of the assurance activity (e.g., inspection, analysis, test) and costs of repair (work needed to repair discovered defects). The custom-built tool that supports risk management has been extended accordingly.

The net result is a framework that supports expert users in making cost-benefit based planning decisions. Details of our approach, current status, and future challenges, are presented.

Keywords: Risk, requirements tradeoffs, design, quality assurance; Validation and Verification (V&V); risk management; Return On Investment (ROI), software quality; software process improvement; optimization; genetic algorithms; learning; NASA.

1 INTRODUCTION

Design is inevitably resource-constrained. Design encompasses the activities needed to analyze, test, repair, etc., artifacts in the course of their development. Design efforts must select these activities judiciously, so as to make most effective use of limited resources. Rarely, if ever, can a project afford to apply all such possible activities to all portions of the design.

The goal of our work is to facilitate cost-benefit based planning of design, with a particular focus on the assurance aspects. Such cost-benefit based assurance planning is an enabler of the following activities:

- developing a plan that minimizes risk with the available resources;
- developing a plan that minimizes the resources needed to reduce risk to an acceptable level;
- identifying the requirements proving the most costly to attain (and therefore are leading candidates for discarding if the resources cannot be expanded, and the risk is unacceptably high).

The common theme underlying these is tradeoff, between benefits in the form of requirements attainment, and costs in the form of resources spent to mitigate risks. The focus of this paper is on our detailed cost-benefit model that makes such tradeoff possible. The rest of this paper is organized as follows:

Section 2 - background to this work, a previously developed risk management framework.
Section 3 - our refined cost-benefit model, which augments the existing risk management framework.
Section 4 - the detailed formulae of our refined model.
Section 5 - ramifications of the model.
Section 6 - status and challenges.
Section 7 - related work and conclusions.

2 BACKGROUND

Our starting point is a NASA-developed risk management framework, the Defect Detection and Prevention (DDP) tool for risk assessment, planning and management [Cornford et al, 2001]. In this section we summarize the salient aspects of the DDP risk model.

DDP deals with requirements, risks and risk mitigations. Risks are quantitatively related to requirements, to indicate how much each risk, should it occur, impacts each requirement. Mitigations are quantitatively related to risks, to indicate how effectively each mitigation, should it be applied, reduces each risk. A set of mitigations achieves benefits (requirements are met because the risks that impact them are reduced by the selected mitigations), but incurs costs (the sum total cost of performing those mitigations). The main purpose of DDP is to facilitate the judicious selection of a set of mitigations, attaining requirements in a cost-effective manner.

In more detail, DDP deals with the following concepts:

Requirements – what the design is to achieve. Requirements are assigned *weights*, representing their

relative importance.

Risks – things that, should they occur, will lead to loss of requirements. Risks are assigned an *a-priori likelihood* (the chance of the risk occurring, if nothing is done to inhibit it).

Mitigations – activities that could be done to reduce the likelihood of failure modes and/or reduce their impact on requirements. Mitigations are assigned *costs*, the costs of performing them.

Impacts – for each Requirement x Risk pair, how much of that Requirement will be lost should that Risk occur.

Effects – for each Mitigation x Risk pair, the extent to which that Mitigation will reduce the risk.

These combine in the following ways:

Each impact is expressed as a number in the range [0 – 1] representing the proportion of the requirement that is lost if that risk occurs. Impacts are additive – when several risks impact the same requirement, their impacts add together. It is possible that impacts on a requirement can add up to greater than 1, in which case the amount of attainment of that requirement is zero. Intuitively, risks are treated as completely independent. This matches the thinking that dominates early phase risk assessment and risk mitigation planning, the area we have targeted to date. However, in order to make use of the model for later phase applications, when designs are more detailed, we will need to incorporate logical fault trees ("and" and "or" combinations of risks). This extension is ongoing work. Currently, when logical combinations are needed, we work around this limitation manually.

Each effect is expressed as a number in the range [0 – 1] representing the proportion of the risk that is reduced if the mitigation is applied. Effects are multiplicative – when several mitigations are applied to reduce the same risk, their total effect is computed as: (1 – the product, for each mitigation M, of (1 – M's effect)). Intuitively, mitigations act as "filters" arranged in series, such that each mitigation filters out its effectiveness' proportion of the risks that enter it.

The costs of the applied mitigations are additive. In our NASA applications of DDP, mitigations can be assigned costs in several dimensions each of which is of importance to spacecraft, e.g., dollars, mass, power.

3 REFINED COST-BENEFIT MODEL

The focus of this paper is on our recent refinements to DDP's cost-benefit model. The key areas of refinement are as follows:

- Make a distinction between different categories of mitigations - preventions, detections and alleviations.
- Separate the cost of performing a detection-style risk mitigation from the cost of repairing the problems it detects.
- Assign detection mitigations to distinct phases of development, permitting the calculation of repair cost to take into account the phase in which the repair is conducted.

These areas of refinement are explored further in the following subsections.

3.1 Categories of mitigations:

Mitigations are subdivided into:

- *Preventions* – activities that reduce the likelihood of problems occurring, e.g., training of programmers reduces the number of mistakes they make.
- *Detections* – activities that detect problems, with the assumption that detected problems will be corrected, e.g., unit testing detects coding errors internal to the unit, which are then corrected. The net effect of detection and repair is a reduction in the likelihood of risks remaining.
- *Alleviations* – assurance activities that decrease the impact (severity) of problems should they occur, e.g., programming a module to be tolerant of out-of-bound values input to it from another module.

3.2 Costs of mitigations:

The calculations of mitigations costs are:

- *Preventions* – incur the cost of performing the prevention activity.
- *Detections* – incur two costs: that of performing the detection itself, e.g., performing the unit test, and that of repairing the problems it detects, e.g., correcting a problem found during unit test. The cost of performing a detection is a function of the detection itself, while the cost of repairing a problem it uncovers is a function the problem. DDP computes the total cost by summing the detection's performance cost and the problem

repair costs for the expected number of problems that it detects.

- *Alleviations* – incur a cost of performing the alleviation activity.

3.3 Time of detection and repair:

Assurance activities are performed at certain times in the course of a development. For our applications, we find it sufficient to make distinctions between only the major phases of development, such as requirements, design, coding, unit test, etc. These times are taken into account during the cost and benefit calculations as follows:

- *Preventions* occur first. They reduce the likelihood of the risks they effect. The risks, their likelihoods possibly reduced by preventions, go forward into the detection stages.
- *Detections* are applied in a series of stages, one for each development phase. In each stage the detections of that stage combine to reduce the likelihood of the risks they effect. This risk reduction has two consequences – reduced likelihoods of the risks going forward into the next stage, and incurred repair costs for the risks discovered in this phase. The repair cost of a risk is a function of the risk and the time at which it repaired (but, note, not the detection used to find the risk).
- *Alleviations* are applied last. They reduce the severity (impact) of the risks they effect, but not their likelihood.

The cumulative effects of mitigations on risk reduction and cost accumulation are shown graphically in Fig. 1.

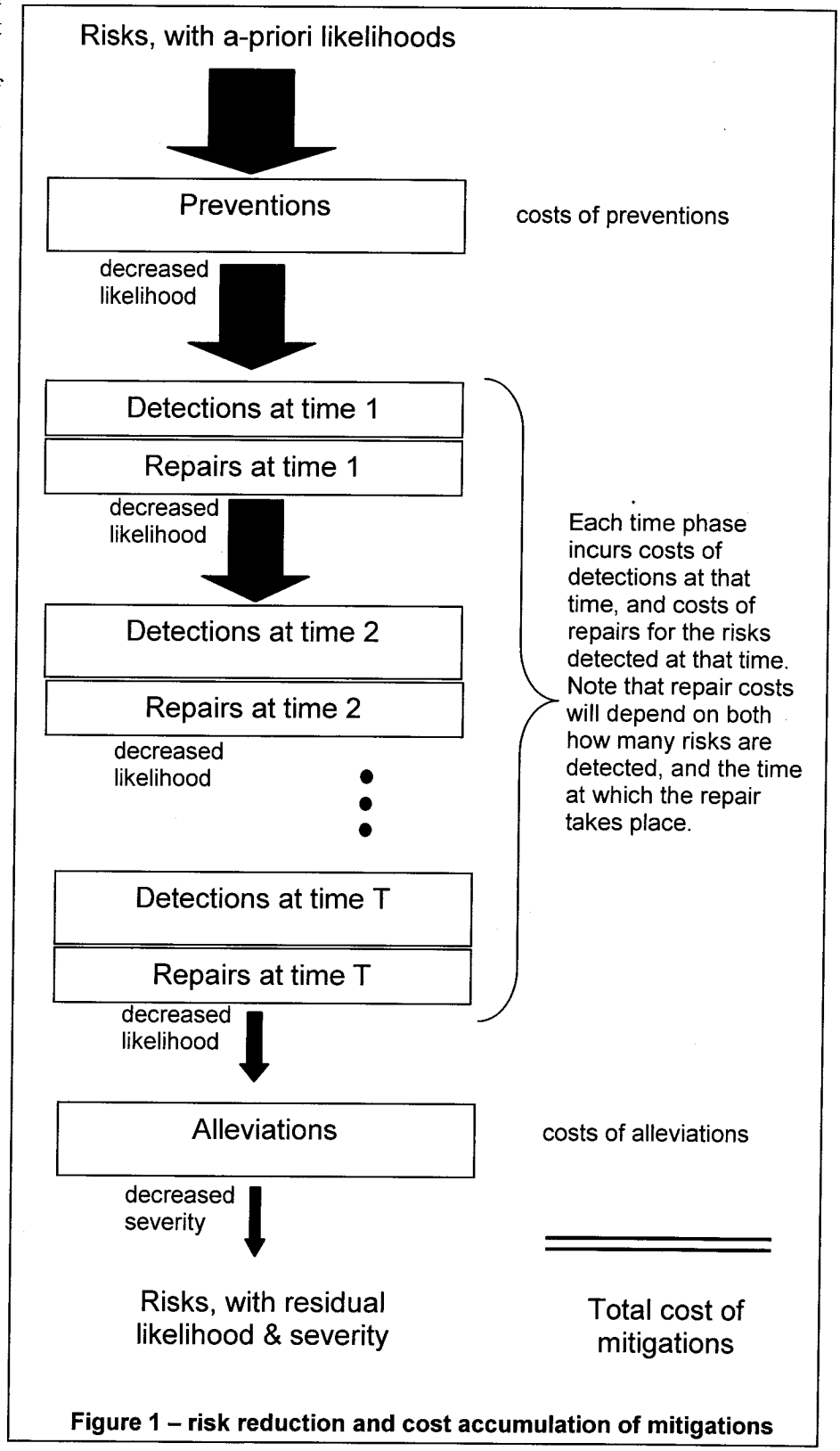

Figure 1 – risk reduction and cost accumulation of mitigations

4 COST-BENEFIT MODEL FORMULAE

This section gives the formulae by which we calculate costs and benefits.

4.1 Assigned values:

aprioriLikelihood(risk) : [0 - 1] - *the likelihood of a risk occurring if nothing is done to mitigate it.*

cost(mitigation) : real - *the cost of performing a mitigation (which could be a prevention, detection or alleviation).*

repairCost(risk, time) : real - *the cost of repairing a risk at that time.*

weight(requirement) : real - *the importance of a requirement.*

impact(risk, requirement) : [0 - 1] - *the proportion of the requirement lost if the risk occurs.*

effect(prevention, risk): [0 - 1] - *the proportion of the risk's likelihood reduced by the prevention.*

effect(detection, risk) : [0 - 1] - *the proportion of the risk's instances detected by the detection; when followed by a repair, the net effect is to reduce the likelihood of the risk remaining present.*

effect(alleviation, risk) : [0 - 1] - *the proportion of the risk's impact alleviated by the alleviation.*

time(detection) : [1 - T] - *the time at which a detection is performed, where time values are represented here as integers in the range 1 to T.*

4.2 Intermediate calculations:

unmitigatedSeverityunmitigatedSeverity(risk) : real - *the risk severity if no mitigations are applied, calculated as:*

unmitigatedSeverity (r) =
Σ (q $\in$ requirements) : weight(q) * impact(r, q)

unmitigatedRequirementAttainment(requirement) : real - *the attainment of the requirement if no mitigations are applied, calculated as:*

unmitigatedRequirementAttainment(q) =
weight(q) *
Σ (r $\in$ risks) : impact(r, q) * aprioriLikelihood (r)

preventedLikelihood(risk) : [0 - 1] - *the risk likelihood after application of preventions, calculated as:*

preventedLikelihood(r) =
aprioriLikelihood(r) *
Π (p $\in$ preventions) : (1 - effect(p,r))

detectedLikelihood(risk, time) : [0 - 1] - *the risk likelihood after application of preventions and of detections and repairs up to and including the time, calculated as:*

detectedLikelihood(r, i+1) =
detectedLikelihood(i) *
Π (d $\in$ detections: time(d) = i+1) : (1 - effect(d, r))

where

detectedLikelihood(r,0) = preventedLikelihood(r)

alleaviatedness(risk) : [0 - 1] - *the proportion of the risk's severity reduced by alleviations, calculated as:*

alleaviatedness(r) =
Π (a $\in$ alleviations) : (1 - effect(a, r))

4.3 Cost calculations:

sumPreventionCosts : real - *the total cost of applying the preventions, calculated as:*

sumPreventionCosts =
Σ (p $\in$ preventions) : cost(p)

sumDetectionCosts: real - *the total cost of applying the detections, calculated as:*

sumDetectionCosts =
Σ (d $\in$ detections) : cost(d)

sumRepairCosts(time): real - *the total cost of applying the repairs of that time, calculated as:*

sumRepairCosts(t) = Σ (r $\in$ risks) : repairCost(r, t) * (detectedLikelihood(r, t-1) - detectedLikelihood(r, t))

sumAlleviationCosts : real - *the total cost of applying the alleviations, calculated as:*

sumAlleviationCosts =
Σ (a $\in$ alleviations) : cost(a)

totalCosts : real - *the grand total costs, calculated as:*

totalCosts =
sumPreventionCosts +
sumAlleviationCosts +
sumDetectionCosts +
Σ (t $\in$ [1 - T]) : sumRepairCosts(t)

4.4 Benefit calculations:

unmitigatedRequirementAttainment(requirement) : real - *the attainment of the requirement if no mitigations were applied, calculated as:*

unmitigatedRequirementAttainment(q) =
weight(q) *
(1 - minimumof(1, Σ(r ∈ risks) :
aprioriLikelihood(r) * impact(r, q)))

mitigatedRequirementAttainment(requirement) : real - *the attainment of the requirement if mitigations are applied, calculated as:*

mitigatedRequirementAttainment(q) =
weight(q) *
(1 - minimumof(1, Σ (r ∈ risks) :
finalLikelihood(r) * alleviatedness(r) *
impact(r, q)))

totalUnmitigatedRequirementsAttained: real - *the total requirements attained if no mitigations are applied, calculated as:*

totalUnmitigatedRequirementsAttained =
Σ (q ∈ requirements) :
unmitigatedRequirementAttainment(q)

totalMitigatedRequirementsAttained: real - *the total requirements attained if mitigations are applied, calculated as:*

totalMitigatedRequirementsAttained =
Σ (q ∈ requirements) :
mitigatedRequirementAttainment(q)

totalBenefits : real - *the grand total benefits, calculated as:*

totalBenefits =
totalMitigatedRequirementsAttained -
totalUnmitigatedRequirementsAttained

If need be, we could also sum up the mitigation and repair costs on a phase-by-phase basis, so as to determine not just the total costs, but the spending profile over the course of time.

5 RAMIFICATIONS OF REFINED COST-BENEFIT MODEL

5.1 Early-lifecycle detections

The advantage of detecting and repairing problems early in the lifecycle is well known. For example, the oft-quoted statistics on the exponential increase in the cost to repair a requirements error if left until design/code/test time. Our cost-benefit model encompasses this phenomenon. Repair costs are a function of both the kind of risk being repaired, and the time at which the repair takes place. By escalating a risk's repair cost over time, the cost calculations will exhibit the phenomenon. It is possible to add an early detection for a risk, and as a result see to both a benefit increase and a cost decrease. The increased benefit comes from the further reduction of risk that the extra detection and repair provides. The decreased cost comes from the early repair of a risk that would otherwise be detected and repaired much later, when repair costs are considerably higher. Cost considerations such as these are discussed in [Kaner, 1996].

Example: suppose we have planned a software development effort in which system testing is to be applied. We may suppose that system testing will detect coding errors, and also, to some extent, requirements errors. Note that correcting requirements errors as late as system test time is likely to be very expensive. Now consider the addition to the development plan of a requirements-time inspection: if this detects some of the requirements errors that would otherwise remain undetected until system testing time, the net result may well be a reduction in overall development cost.

Also evident is a law of diminishing returns. Each additional detection of the same kind of risks reduces only the residual proportion of risks that have escaped the other detections.

Example: suppose there is in place a risk detection with effectiveness of 0.9 (i.e., detection rate for that kind of risk). Only 0.1 of the risk will escape detection. Applying a second detection with effectiveness of 0.9 will result in 0.01 of the risk escaping the detection of both. Thus the additional risk reduction from the addition of the second detection was from 0.1 to 0.01, i.e., 0.09, far less than the 0.9 risk reduction of the first one.

5.2 Return on Investment

Return-On-Investment (ROI) calculations can be derived if requirements are valued on the same scale as costs, as follows:

ROI = Benefit / Cost, *where*
Benefit =
(requirements attained with mitigations applied) - (requirements attained without mitigations applied)
Cost = total cost of mitigations and repairs

In terms of our formulae,
ROI = totalCosts / totalBenefits
Provided that weight (the attribute of a requirement representing its importance) is measured in (or converted into) the same units as cost.

In our application of these methods for software assurance planning in the NASA setting, we have a range of approaches to valuing requirements. At the conservative end, we may measure the attainment of mission success as having the value of the total development cost of the mission. More aggressively, we might attempt to place a value on the science return of a successful mission, for example, the value of discovering water available on Mars.

5.3 Simplifying assumptions

Our cost-benefit model makes some simplifying assumptions of the combined effects of mitigations on both costs and benefits. For example, two similar mitigations might discover substantially the same risks, and so their combined effect may be less than our formulae would predict. Another example is of two mitigations that share the same setup costs - the consequence might be that the cost to perform both of them is less than the sum of their individual costs.

To date, we have not elaborated our model to encompass such possibilities. Instead, when faced with instances of these, we rely on manual workarounds. For example, when we know that the combination of mitigations M1 and M2 does not match that predicted by our formulae, we manually add a third mitigation, M1&M2. We assign to this the combined effectiveness and cost values that we believe hold for the combination of the two. When selecting mitigations, we are careful to select at most one of { M1, M2, M1&M2 }. Such workarounds allow us to proceed with DDP applications, at the expense of a small amount of additional effort.

As the need for such workarounds grows, we will then consider further refinement of our cost-benefit model. One area where such a need could indeed arise is in the use of this approach to plan Independent Validation and Verification (IV&V). At NASA, IV&V is intended to supplement, not replace, Software Quality Assurance. Thus IV&V may well perform the same class of assurance activities that are already being done as part of SQA.

6 STATUS AND CHALLENGES

The status of this work is that the cost-benefit model described above is fully functional within the DDP tool, and has been tested on real datasets. In order to perform the testing, we retroactively added cost data to datasets that resulted from earlier applications of DDP (applications that took place before the refined cost model was added).

These experiments have confirmed that the implementation of the refined cost-benefit model is capable of performing the calculations required. In keeping with the DDP heritage, all of the values on which a calculation is based are accessible to, and changeable by, the end users. Such changes trigger automatic recalculation. This feature allows users to quickly tailor generic knowledge to the task at hand. It can also be used to explore "what-if" scenarios, e.g., the benefits of investing research with the goal of increasing the effectiveness of a requirements-time inspection.

We have also tested the implementation by applying it to other peoples' small-scale examples of cost-benefit calculations - see related work section (on one occasion this uncovered an arithmetic error in the other person's example!).

The DDP tool is constructed to allow a group of experts to arrive at a cost-effective risk mitigation plan for specific projects. In past DDP applications, selection of an appropriate suite of mitigations was done by hand. Experts would first populate DDP with data pertinent to the project in question. Then, they would make selections of individual mitigations to move towards an overall mitigation suite that achieved the requisite levels of risk reduction while making good use of available resources. In some

cases this led to revision of requirements, and even to the recommendation to not go ahead with the current plan. For descriptions of DDP and its usage, see [Cornford et al, 2001].

As the cost-benefit model becomes more refined, we predict a shift in the way in which DDP users arrive at their mitigation suite selections. Rather than selecting mitigations one by one, we foresee the need to employ automation that will search for alternative suites of mitigations. The motivations for this are twofold:

- Assurance planning in a resource-constrained setting is fundamentally an optimization problem. If there are, say, 50 possible assurance activities that may be selected, then the number of choices of suites of selected mitigations is 2^50. The incremental approach of seeking a near-optimal solution by addition / removal of individual mitigations is not a good use of people's time.
- The refined cost-benefit model induces more complex interactions among mitigations, most notably the influence of early detections on repair costs. These make it hard to study mitigations on a one-by-one basis. Their cost benefit ramifications can be very different depending on the other mitigations in the suite.

In response, we are exploring the use of automation for search (to find near-optimal mitigation suites) and sensitivity analysis (to know which of our data values and choices have the most influence on our final decisions). Experiments in this direction are progressing using two approaches:

- Heuristic search - we have been using a genetic algorithm to search for near-optimal mitigation suites. E.g., we search for a suite of mitigations that maximizes requirements attainment while keeping its total cost below a user-provided threshold. [Dunphy et al, 2001]
- Treatment leaning - in cooperation with Prof. Tim Menzies, we have been exploring the use of his treatment learning approach [Menzies&Singh, 2001].

The results so far are promising - we have applied them to DDP datasets of upward of 80 mitigations (which therefore have a search space in excess of 2^80). Our experiments so far suggest that the two approaches have complementary strengths and weaknesses. The heuristic search approach, based on Genetic Algorithms, has been applied to a simpler version of the cost-benefit model than described here, and upon that simpler version, appears to progress rapidly in the direction of finding increasingly optimal solutions. However, as we extend this to work with the refined cost-benefit model, we may need to adjust the internal operation of the genetic algorithm accordingly.

Conversely, the treatment learning approach works off of randomly generated mitigation selections, for each of which the cost and benefit is computed by the DDP tool with the cost-benefit formulae we have described. The advantage is that as DDP's cost-benefit model evolves, this approach remains unaffected. Currently, however, this approach suffers from the bottleneck of generation of the large numbers of random runs needed as the basis on which to draw stable conclusions.

More broadly, we see the key future challenge will be to relate the results of automated search back to users. For example, we need to be able to report to users not only the near-optimal (minimal risk) solution for the exact cost level that they stipulated, but also the options in the space "close" to that solution. If they could afford to pay slightly more, do there exist options with much improved benefit (lower risk)? Which of their requirements are proving to be the most problematic? At present, DDP makes use of several custom visualizations of the set of requirements, the set of risks, and the set of mitigations (see [Feather et al, 2000] for details). Each of these is well suited to displaying the status of a given selection. What we feel will soon be needed is an imaginative way to extend these visualizations to present not just the current status, but the primary options, as discovered through automated search. The net result will be the judicious combination of automation with expert human intuition.

7 RELATED WORK AND CONCLUSIONS

We have looked for related work in, particularly, the software arena. There we have found that cost-benefit analyses of *individual* activities have been reported (e.g., reinspections - [Biffl et al, 2000]; regression testing – [Graves et al., 1998]; inspections and structured testing – [Gack, 2000]). Studies of overall process improvement also exist (e.g., [McGarry et al, 1998] that relate software defects,

productivity, development cycle time and effort estimation to process ratings akin to the Software Engineering Institute (SEI)'s Capability Maturity Model (CMM)- [McGarry et al, 1998]). However, the middle ground, of quantitatively planning the suite of activities to apply to a given project, is relatively under explored. In general there seems a lack of data on, say, each of the key process areas of CMM level 3. We are now looking to the ongoing efforts of the consortium http://www.cebase.org to gather such data.

We see some relationship to the cost/benefit work of [Karlsson & Ryan, 1997], since embodied in a commercial tool [Focal Point AB]. 2-D graphical plots of requirements, where the dimensions indicate value of attainment against cost of attainment, appear to serve as useful guides to software release planning. Our work involves risk as the key factor that underlies all our calculations, and in this respect is quite different.

Risk estimation approaches (e.g., fault tree analysis, baysian methods) appear very well suited to the assessment of a *single* design. However, our application is to the planning of mitigations, were the driving concern is the cost-benefit-guided selection from among a large set of such mitigations.

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. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government or the Jet Propulsion Laboratory, California Institute of Technology. Contributions from, and discussions with, Burton Sigal (JPL), Patrick Hutchinson (Wofford College, Spartanburg SC), Peter In (Texas A&M), John Kelly (JPL), Tim Kurtz (NASA Glenn), James Kiper (Miami Univ., Ohio) and Tim Menzies (U. British Columbia) have been most useful in helping us formulate our ideas.

9 REFERENCES

[Biffl et al, 2000] S. Biffl, B. Freimut and O.Laitenberger. "Investigating the cost-effectiveness of reinspections in software development", *23rd Int. Conference on Software Engineering*, 2001, pp. 155-164.

[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.

[Dunphy et al, 2001] J. Dunphy, M.S. Feather & S.L. Cornford. "Optimizing the Design of end-to-end Spacecraft Systems using Risk as a Currency" (in submission - please contact the authors for an advance copy).

[Feather et al, 2000] M.S. Feather, S.L Cornford & M. Gibbel. "Scalable Mechanisms for Requirements Interaction Management", *IEEE Int. Conference on Requirements Engineering, 2000.*

[Focal Point AB] Focal Point™, a trademark of Focal Point AB http://www.focalpoint.se

[Gack, 2000] "Defect Tracking + Inspections = $ in Your Pocket", http://www.iteffectiveness.com/defecttracking.htm

[Graves et al, 1998]. T. Graves, M. Harrold, J. Kim, A. Porter and G. Rothermel. "An Empirical Study of Regression Test Selection Techniques". *20th Int. Conference on Software Engineering*, 1998, pp. 267-273.

[Kaner, 1996]. C. Kaner. "Quality Cost Analysis: Benefits and Risks", Software QA Vol 3, #1, p. 23, 1996.

[Karlsson & Ryan, 1997] J. Karlsson & K. Ryan. A Cost-Value Approach for Prioritizing Requirements. *IEEE Software*, Sept./Oct. 1997, 67-74.

[McGarry et al, 1998] F. McGarry, S. Burke & B. Decker. Measuring the impacts individual process maturity attributes have on software products. Proceedings, *5th* International *Software Metrics Symposium*, 1998, pp. 52-60

[Menzies&Hu, 2001] T. Menzies & Y. Hu. "Constraining Discussions in Requirements Engineering via Models", *1st International Workshop on Model Based Requirements Engineering*, San Diego, California, Dec 2001.