

Visualization Support for Risk-Informed Decision Making when Planning and Managing Software Developments

Martin S. Feather
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.,
Pasadena CA 91109
+1 818 354 1194

Martin.S.Feather@jpl.nasa.gov

James D. Kiper
Dept. of Computer Science
& Systems Analysis
Miami University
Oxford OH 45056
+1 513 529 5931

kiperjd@muohio.edu

Tim Menzies
Computer Science
Portland State University
Room 120, Fourth Avenue Building
1900 SW 4th Ave., Portland OR
97201
+1 503 231 5277

tim@menzies.us

ABSTRACT

Key decisions are made in the early stages of planning and management of software developments. The information basis for these decisions is often a mix of analogy with past developments, and the best judgments of domain experts. Visualization of this information can support to such decision making by clarifying the status of the information and yielding insights into the ramifications of that information vis-à-vis decision alternatives.

We illustrate this in the context of a risk-based model developed and applied at NASA for planning the development of systems that use advanced technologies, and for research and technology portfolio planning.

Categories and Subject Descriptors

C.4 [Performance of Systems] – design studies, modeling techniques, reliability, availability, and serviceability

D.2.1 [Software Engineering]: Requirements/Specifications – *elicitation methods, tools.*

D.2.9 [Software Engineering]: Management – *cost estimation, software quality assurance.*

I.2.8 [Artificial Intelligence] Problem Solving, Control Methods, and Search – *Heuristic methods*

I.5.3 [Pattern Recognition] Clustering – *similarity measures.*

K.6.3 [Management of Computing and Information Systems] Software Management – *software development, software selection.*

General Terms

Management, Design, Economics, Reliability.

Keywords

Visualization, Requirements Prioritization, Software Risk, Cost-Benefit Analysis, Decision Making.

1. INTRODUCTION

The mission statement of the ACM Symposium on Software Visualization states “Software visualization encompasses the development and evaluation of methods for graphically representing different aspects of software, including its structure, its abstract and concrete execution, and its evolution.” From the program from the 2003 conference [7] and related events, such as the ICSE Workshop on Software Visualization [18], it is evident that much of the work in the area of software visualization is focused on concerns of detailed designs and code (e.g., design structure, program understanding, algorithm animation, debugging). The complexity of many software systems offers a fertile ground for application of, and further challenges for, software visualization.

The purpose of this paper is to suggest there is also scope for application of software visualization to the earliest phases of the software lifecycle – when requirements are being determined, and planning is done for the entire software development to follow. This might appear an unlikely area for software visualization: much of the reasoning would seem to be inevitably qualitative rather than quantitative; details are sketchy at best (design concepts rather than detailed designs and code); the quantities of information do not appear to warrant sophisticated visualization.

It is clear, however, that the earliest phases of the software lifecycle are critical: key decisions are made, including determination of the purpose of the software (from which its requirements follow), planning the rest of its lifecycle (subsequent development, testing, deployment, maintenance and future upgrades), the allocation of resources to those phases (e.g., budget, schedule, testing platforms), and the determination of the architecture and early phase design. Furthermore, our experience suggest that there is a non-trivial amount of information available even during the earliest phases of the software lifecycle. This

information comes from many sources (many “stakeholders” in the parlance of the software community) – the customers of the software (or of the system of which the software is but a part); its funders, managers, developers, users, etc. Information may take the form of guidance extracted from past experience, coupled with experts’ best estimates in the face of novel aspects of the effort (new applications, new circumstances, new hardware and software resources, new development methodologies, etc).

Decision making during these earliest phases remains a predominantly human activity, but one which can and should be well-informed by this wealth of information. Visualization can play a prominent role by portraying the information in ways that make it amenable to scrutiny (so that the information basis for the decision making is clear) and amenable to extracting key insights and guidance towards such decision making.

This paper illustrates our work in this area, based on a novel but effective approach to employing quantitative reasoning during the early phases of system (software and/or hardware) developments, and technology planning. We have previously published on various aspects of this work. Here we focus on how use of (relatively straightforward) visualization mechanisms plays a crucial role throughout our work. The remainder of the paper is organized as follows:

Section 2 summarizes our approach to qualitative reasoning for early lifecycle decision making.

Sections 3 – 6 presents the visualizations we have found to be helpful, with a brief summary discussion at the end of each section. Specifically, section 3 addresses the way users can scrutinize individual problem solutions (and simple comparisons among them); section 4 looks at the understanding gained from computer-assisted search over the space of all possible solutions, section 5 looks at ways to explore within that solution space, and section 6 explores some applications of this approach to technology portfolios (rather than specific technology solutions).

Section 7 offers conclusions and discusses some related work.

2. AN APPROACH TO QUANTITATIVE REASONING SUPPORT FOR EARLY-LIFECYCLE DECISION MAKING

Our work takes place in the context of a risk-based approach to assist early-lifecycle planning of complex system developments. The approach is called “Defect Detection and Prevention” (DDP), the name reflecting its origins as a method intended for quality assurance planning of hardware systems [5]. It has been developed at JPL and NASA and applied to risk management of spacecraft and spacecraft technologies (both software and hardware) in their early phases of development. An extensive description of DDP is given in [9]. Here we summarize its key aspects.

The approach involves developing a model from which two key measures can be calculated: *benefit*, in terms of attainment of objectives, and *cost*, in terms of resources needed to achieve that benefit. The model is then used to guide decision-making, e.g., to select from among alternative plans of designs or developments.

The hallmark of DDP is its use of a quantitative risk-centric model, in which risks serve as a key intermediary in the cost-benefit calculations. A DDP model is populated by instances of three kinds of concepts: “*Objectives*” (what it is that the system or technology is to achieve), “*Risks*” (what could occur to impede the attainment of the Objectives), and “*Mitigations*” (what could be done to reduce the likelihood and/or impact of Risks)¹. In the DDP model these instances have quantitative attributes: each Objective has a *weight*, its relative importance; each Risk has a *likelihood*, its probability of occurrence, and each Mitigation has a *cost*, the cost of performing it (usually a financial cost, but other resources can also be considered, such as schedule, electrical power, mass). Quantitative relationships connect these instances: Objectives are related to Risks, and Risks are related to Mitigations. Specifically, Objectives are related to Risks to indicate how much each Risk, should it occur, *impacts* (i.e., detracts from the attainment of) each Objective. Risks are quantitatively related to Mitigations, to indicate how much of a Risk-reducing *effect* a Mitigation, should it be applied, has on reducing each Risk (either by decreasing the Risk’s likelihood, or by reducing the magnitude of the Risk’s impacts on Objectives; the nature of the Mitigation dictates which kind of reduction takes place).

Objectives can be “product” needs on the system (e.g., functional behavior, run-time resource needs, timing requirements), and/or “process” needs (on the development process itself, e.g., development environment, testing facilities, progress reporting requirements). First we give an example of a “product” Objective, followed by a “process” Objective:

- In spacecraft applications it is a requirement that software be able to operate despite memory errors (e.g., memory chips can be damaged by radiation, and a service call to replace them is not an option!). Risks to this include inability to detect memory errors (a potential mitigation for which is use of EDAC), statically linked addresses in the code (a potential mitigation for which is to follow programming practices that eschew the use of such static links), and lack of control of the process that loads software into memory (a potential mitigation for which is selection of an operating systems that provides such control).
- Spacecraft are often constrained as to when they can launch (e.g., in order that the celestial dynamics permit a rapid and fuel-efficient route to their destination), so their software must be ready on time. Risk to this may arise when using a novel software development approach (e.g., autocoding) for which there is little past experience from which to estimate feasibility of development schedules. A potential mitigation is to plan for additional early reviews specifically to scrutinize progress (early enough that remedial actions can be taken should problems be discovered).

DDP applications to NASA spacecraft technologies gather information from multiple experts – the scientists whose data

¹ On occasion we use alternate terminology such as “Requirements” in place of “Objectives”, and “Failure Modes” in place of “Risks”.

needs establish the scientific requirements for the mission, the architects, designers and programmers/engineers who must plan and code/fabricate the technologies, the testers and quality assurance personnel who help ensure their correct operation, the operations team, management, etc. In this setting DDP has been used successfully, yielding: improved insights into a variety of risks, ability to trade and calibrate risk across discipline boundaries, optimized planning of how to address risk, risk-informed comparison among design alternatives, and risk-guided descoping (strategic abandonment of objectives). We attribute DDP's successes to its ability to gather and pool non-trivial quantities of information from the multiple experts. Together they populate the DDP model of Objectives, Risks and Mitigations, and use the completed model to guide their decision making. The sections that follow illustrate how we employ simple visualization techniques to aid in this.

In summary, the space mission demands involve a combination of

- Cross-disciplinary concerns (e.g., spacecraft involves navigation, propulsion, telecommunications). These concerns are cross-coupled and interact in multiple ways (e.g., electromagnetic interference, heat transfer).
- Severe constraints on the systems being developed and on the development process itself. Time and budget pressures constrain development; operational resources constrain the resulting system (e.g., mass, volume, power).
- Mission-critical issues. Spacecraft are critical systems that must operate correctly the first time in only partially understood environments, with no chance for repair.
- Unknowns: past experience provides only a partial guide when new mission concepts are to be enhanced and enabled by new technologies of which past experience is lacking.

Because of these challenging aspects of space missions, usually no one person has expertise that spans all the disciplines, or can simultaneously juggle all the factors involved in large and complex designs. Furthermore, much of the design skill is "tacit knowledge" in the heads of spacecraft experts, so it cannot be

encoded in an automated tool. Therefore, key decision making can be enhanced by a computer-aided, human-informed process. We will illustrate how (relatively straightforward forms of) visualization play in important role in communicating information to the decision makers.

Note that while our work is within this context of space missions, the above concerns – cross disciplinary, resource constraints, critical and novel – are common to many domains, so we believe the applicability is broad.

3. SCRUTINIZING THE RISK STATUS OF INDIVIDUAL SOLUTIONS

A DDP model typically consists of dozens of instances of each of the three concepts (Objectives, Risks and Mitigations), and hundreds of linkages among them (Impact links between Objectives and Risks, and Effect links between Risks and Mitigations). Figure 1 shows the "topology" of the information in one such DDP model: the 50 small blue circles in the top row denote the Objectives, the 31 small red circles in the middle row denote the Risks, and the 58 small green circles in the bottom row denote the Mitigations. The red lines in the upper half indicate the Impact links; the variously colored lines in the lower half indicate the Effect links.

This quantity of information, its convoluted nature, and its origins in the different discipline areas of spacecraft development, combine to make decision-making challenging. Especially problematic is the selection of Mitigations. In almost all applications the total cost of all the identified Mitigations far exceeds the resources available, necessitating the careful selection of which of them to perform. We find that while spacecraft experts' intuitive selections (guided by their skill and past experience) are generally good, there is often some problematic area that use of the DDP model reveals. This is manifested as "unbalanced" treatment of risks: excessive resources are expended to reduce some risks to tiny levels, while other significant risks remain relatively unaddressed. DDP reveals this to the experts through a combination of calculation and visualization: the DDP software computes, for each Risk, the

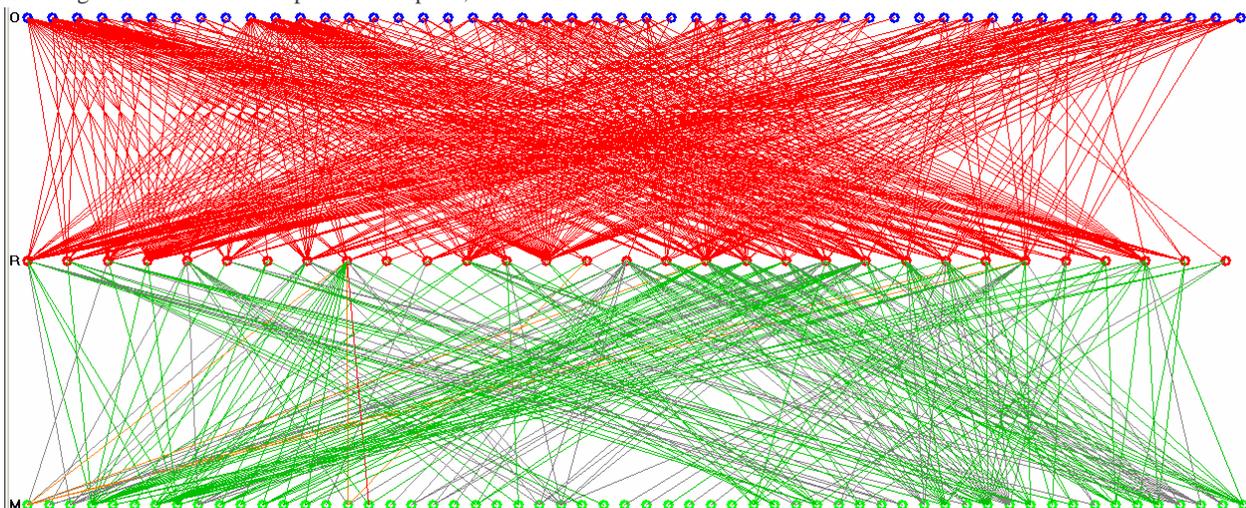


Figure 1. Topology of a typical DDP model

sum total impact is has on the Objectives, taking into account the weights of those objectives and the beneficial effects of the currently selected Mitigations (recall that Mitigations reduce either the likelihood or impact of the Risks to which they are connected by Effect links). Figure 2 shows DDP's bar chart visualization of Risks' status: each bar represents a Risk (the number beneath a bar references a tree-structured listing of the Risks, not shown here). The height of the bar indicates the sum total impact the risk causes – the height of the green portion indicates its sum total impact were no Mitigations to be selected, while the height of the red portion indicates its sum total impact taking into account the effects of the current selection of Mitigations. The DDP tool allows users to change the selection of Mitigations, following which it automatically recomputes Risks' status, and redraws the bar chart. For typically sized DDP models, recomputation and redrawing takes less than a second (on a 1GHz laptop). This makes is easy for users to try “what if” scenarios of alternate Mitigation selections.

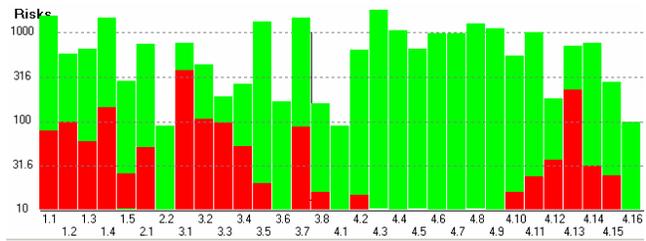


Figure 2. Bar chart of Risks' status

The bar chart shows that some of the Risks remain at relatively high levels, while some others have been reduced to relatively low levels (note that the vertical scale is logarithmic, so the disparities are quite pronounced). DDP can also sort the risks in order of their current status, making these disparities more evident, as seen in Figure 3.

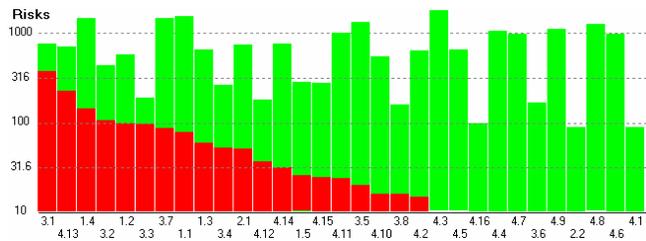


Figure 3. Bar chart of Risks' status, sorted

DDP can highlight the change in Risks' status between alternate selections of Mitigations. Figure 4 shows the changes relative to Figure 3 when an additional Mitigation is selected, effecting (reducing) several of the risks: the yellow portions indicate where Risk levels have decreased.

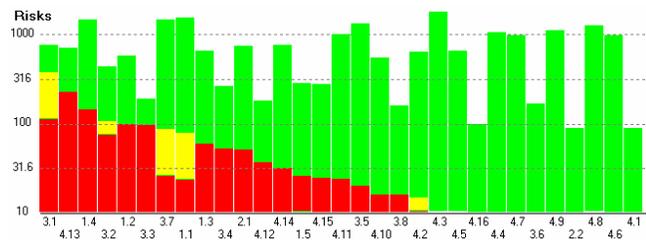


Figure 4. Change (decrease) in Risks' status

If the change between alternate Mitigation selections involves both selections and deselections of Mitigations, then typically some Risks will have increased, while others decreased. Figure 5 shows an example of such, as changes relative to Figure 3: the yellow portions indicate where Risk levels have decreased; the black portions indicate where Risk levels have increased.

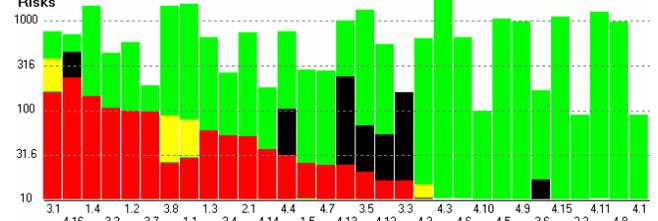


Figure 5. Decreases (yellow) and increases (black) in Risks

Additional features of DDP's bar chart displays include “thumbnail” views of a large number of risks, the ability to color-code risks based on user-defined categorizations, grouping of risks into categories, and filtering to hide from view certain risks. Two of these are seen in Figure 6, whose data is taken from a larger study in which there are almost 70 risks. Of these, the 30 or so highest risks are visible as usual, while the thumbnail at the bottom of the chart shows the entire bar chart in miniature, with the currently visible portion highlighted in the black-bordered rectangle. This rectangle serves as a mouse-sensitive slider bar, allowing the user to change the portion in view. The risks have been color-coded (red, purple, light blue and dark blue) to indicate which of four user-defined categories each belongs. Modest dynamic features are also provided (e.g., the name of an item is displayed as the mouse cursor passes over the item) to assist users.

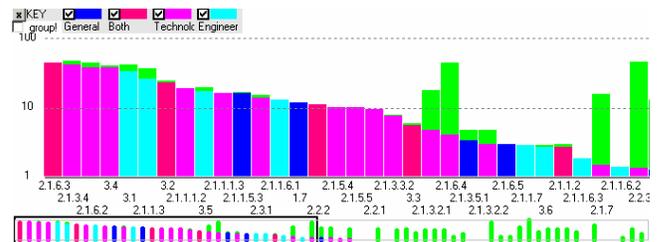


Figure 6. Thumbnail and color coding on Risk bar chart

We have shown here some instances of bar charts displaying the status of Risks from our Objectives-Risks-Mitigations model. We also use bar charts to display the status of the Objectives themselves – specifically, the extent to which Risks are detracting from their attainment, and the status of Mitigations – specifically, the extent to which individual Mitigations contribute (or, if not currently selected, could contribute) to risk reduction. Overall we find bar charts to be a suitable mechanism to present quantitative information on an item-by-item basis, and to present comparisons between pairs of alternate solutions. This allows decision makers to understand the makeup of a given solution. For example, they can easily recognize unbalanced treatment of Risks, compare the Risk “profiles” of alternate solutions, identify Objectives highly threatened by the current Risks, etc.

Discussion: In practice we usually deal with dozens of such items. A few of our studies have involved several hundreds of

items; at these numbers bar charts still seem to be appropriate, but the responsiveness of our implementation begins to degrade.

4. UNDERSTANDING THE SOLUTION SPACE

Finding a desirable selection of Mitigations can be challenging, because of the number of Mitigations from which to select, and the convoluted nature of the way that Mitigations connect to Risks, and Risks connect to Objectives (recall Figure 1). If there are n mitigations, then there are in principle 2^n possible selections from among them. To solve this we implemented simulated annealing [16] within DDP, and use it to locate near-optimal solutions. We have also explored other forms of heuristic search: genetic algorithms and machine learning – for a discussion of these, see [6].

We use visualization to convey the results of such searches, as seen in Figure 7. This plots the result of an amalgam of searches, revealing the overall cost-benefit tradespace. Each of the approximately 300,000 individual points in the black “cloud” corresponds to a distinct selection of Mitigations. The DDP model has been used to calculate the cost and benefit of each such selection, and draw a small black point corresponding to the solution: cost determines horizontal position; benefit vertical position. The upper-left frontier of the cloud is thus the “optimal” boundary, also referred to as the “Pareto front” [20]. Note that while the simulated annealing search is designed to concentrate towards this optimal boundary, we plot a point for *every* selection investigated by the search, not just the “near-optimal” points on that boundary.

For this paper we have annotated the plot with the ellipses to indicate distinct regions on the Pareto front (the points within the interior are all inferior to more optimal solutions, of course). If the budget is too low, the best selections fall within the region where small amounts of additional funding can lead to radical improvements (better attainment of Objectives). Conversely, if the budget is too high, selections fall within the region where a “law of diminishing returns” comes into play. The ideal is to be somewhere in the “sweet spot” region. If the budget is too small to allow this, such a plot can motivate either a request for a budget increase, or serious consideration of descoping (reducing expectations) to be more in line with the budget that is available.

Significant improvement possible Sweet spot Region of diminishing returns

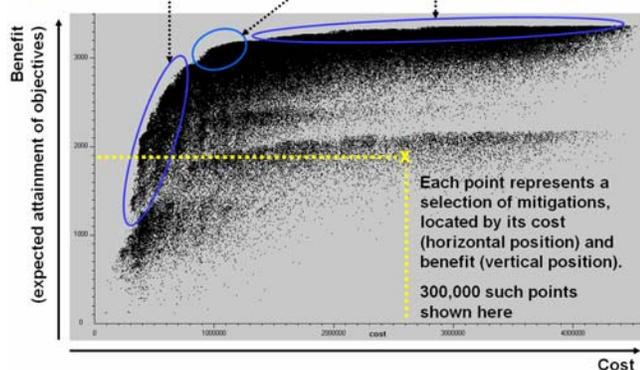


Figure 7. Cost-benefit tradespace chart

Sometimes the cost-benefit tradespace has a much more granular structure, representing different regimes of solutions at different expenditure levels – see Figure 8 for an example taken from another one of our studies.

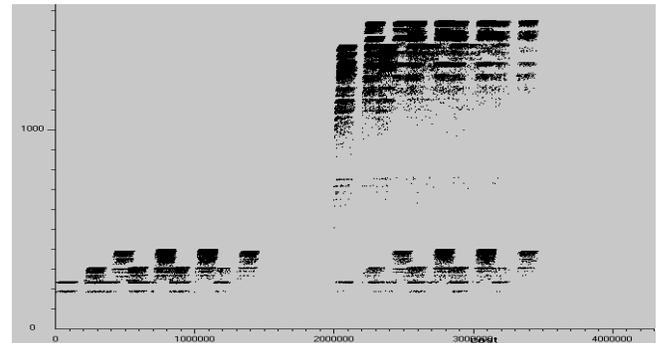


Figure 8. Cost-benefit tradespace chart, another example

Discussion: Overall we find this kind of plot cogently reveals the overall cost-benefit tradespace, information that supports managerial decision making (e.g., can we afford this development? is the funding level appropriate to the problem?).

In practice the main issue we face is the time it takes to compute the cost and benefit values that determine each point’s location – on our typically sized models, this takes several hours to adequately explore the search space using simulated annealing, resulting in hundreds of thousands of points. We store the solutions in a file, from which it is possible to redisplay these plots in a few seconds.

5. EXPLORING THE SOLUTION SPACE

The nature of early lifecycle design is a plethora of options. We find this to be the case when we scrutinize the results of our use of heuristic search to reveal the cost-benefit tradespace. Even when the users narrow their attention to a relatively small area in the tradespace there can be thousands of alternative solutions. This is illustrated with reference to the cost-benefit tradespace shown in Figure 7. Suppose we focus on a neighborhood of interest within the “sweet spot” characterized by solutions costing no more than \$1,000,000, and by attaining at least 95% of the maximum benefit attainable within that region – see Figure 9.

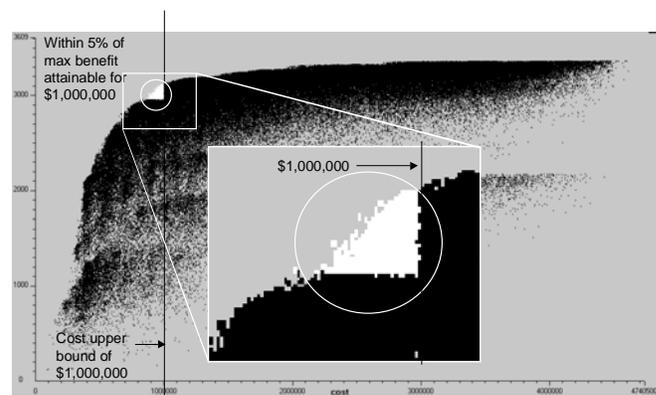


Figure 9. Neighborhood of interest

Within the dataset that gave rise to this picture, there are over 3,000 solutions (i.e., distinct selections of Mitigations). Many of these will be similar – from one to the next, they may differ by only one or two low-cost, low-benefit Mitigations. However there may be some radically different solutions present within that same region. We have experimented with several techniques to explore such regions. Again, we use a mix of computational power to automate the exploration, and appropriate use of visualization to reveal interesting implications. We summarize our experiments in the following subsections.

5.1 Determining Key Decisions

A desirable thing to know is which decisions are *key* (i.e., make a significant difference to the outcome). In our context, an individual decision is represented as selection of a Mitigation, or the *avoidance* of the selection of a Mitigation – also a decision. Recall that each of Mitigation represents a design or development option, so these decisions translate into design or development choices.

Methods for identifying key decisions have been the focus of one of this paper’s authors (Menziez) for several years – for a description of this approach see [17]. In collaboration we have studied their use within our risk-centric framework [11]. Briefly, the study involved several iterations between the DDP tool, and the treatment learning tool, each of which revealed some additional key decisions. In each iteration, the DDP tool was run thousands of times to evaluate its risk model using random selections of Mitigations. These thousands of results were then passed to the treatment learning tool, along with an indication of the preferences among those solutions (in our case, the preference favors solutions that have a high benefit/cost ratio). The treatment learning tool identified several key Mitigation selections and avoidances (Mitigations to *not* select) that help steer towards the neighborhood of interest. These selections and avoidances were then imposed as constraints on the next iteration, meaning that the DDP tool’s random explorations leave those key Mitigations selected (or not) appropriately, randomly selecting from among the remainder. Each iteration thus exposed further key Mitigations. In our study, the process terminated when decisions about one-third of the Mitigations had been identified, the result of which was convergence on a small area within the cost-benefit tradespace. The remaining two-thirds of the Mitigations turned out to be such small contributors to variation that the treatment learning approach could not discern any particularly key remaining decisions among them.

We use visualization to convey the overall consequences of this, as seen in **Error! Reference source not found.**

The convergence towards the compact (red colored) zone of high benefit solutions is strikingly apparent. Note that this study made use of a *different* DDP to that used in the other cost benefit tradespace figures. The net result is knowledge of how to get to a desired area in the search space by having identified the subset of the key decisions and how to make them. The visualization serves to convince the viewers of the efficacy of that key decision set.

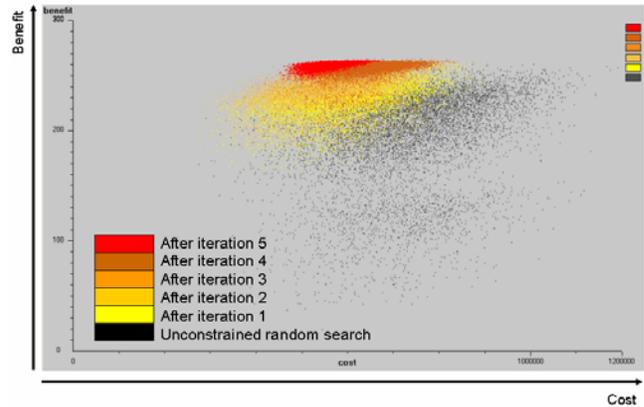


Figure 10. Convergence as iterations identify key decisions

5.2 Understanding Individual Decisions’ Contributions

We have also experimented with a purely visualization-based approach to understanding the contribution of individual decisions [8].

Given the set of points that constitute a cost-benefit tradespace, for a Mitigation of interest, we color each point one color (black, say) if that point corresponds to a solution not involving use of that Mitigation, and another color (yellow, say) if it does involve use of that Mitigation. An example is in Figure 11.

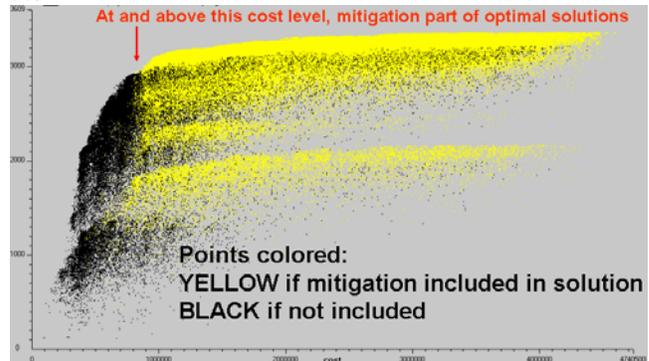


Figure 11. Contribution of an individual mitigation

The broad swathe of yellow points that dominate a large fraction of the Pareto front indicate that the chosen mitigation is key to nearly all optimal solutions above a certain cost level.

Repeating this for each Mitigation we can gather a snapshot of their contributions, seen in Figure 12. (The four groupings correspond to the four major user-defined categories into which these 58 Mitigations were organized.)

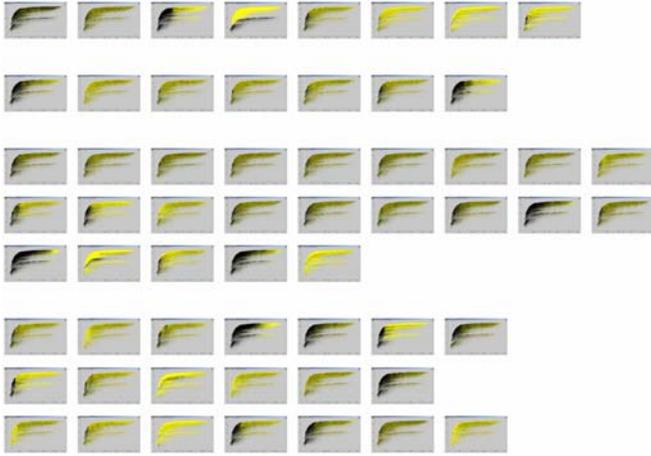


Figure 12. Snapshots of Mitigations' contributions

We have since learned of a visualization tool [21] which uses a combination of techniques, of which color is just one, to show multiple dimensions at once of the attributes of a design. Our approach is a subset of its capabilities. We are currently investigating its use on our datasets, and the preliminary results so far are very promising.

5.3 Identifying Interesting Alternatives

We have also exploring means to distill information from the many solutions that lie within a neighborhood of interest. Our approach to this has been based on a definition of a metric of “similarity” between two solutions (i.e., selections of Mitigations). This metric is user-defined in terms *other* than overall cost and/or benefit of a solution (since all the solutions are within a small neighborhood of similar costs and similar benefits). One useful metric might be based on the cost *profile* – how the costs fall into major categories (e.g., in a hardware-centric study we did, there were categories of design, fabrication, assembly and test; two solutions that had the same overall cost but allocated that cost very differently between those categories would be very dissimilar).

Using such metrics we have explored the use of two techniques: (1) Search for “maximally dispersed” solutions within the neighborhood of interest. The idea of this is to yield a small number of interestingly distinct solutions [10]. (2) Search for clusters of similar solutions. The idea of this is to yield a small number of interestingly distinct clusters, within each of which all the solutions are relatively similar to one another [15]. For both of these, we again turn to visualization to present the results.

The visualization of a modest number of dispersed solutions is seen in **Figure 13**. The union of Mitigations involved in one or more of those solutions form the rows. The grid at the left is used to indicate the distinct solutions, one per column. A black (white) cell indicates that the Mitigation of that row is included (not included) in the solution of that column. Mitigations that are common to all solutions have been filtered out of this table (they are listed separately, not shown here), so what remains is a portrayal of the differences among solutions. In this case the Mitigations are sorted in descending order of their cost. From this it is easy to see that 8 out of 10 of the solutions include the

use of a \$200,000 Mitigation, while 2 of them avoid its use. This is a significant difference given that the sum total cost of each solution is capped at \$1 million.

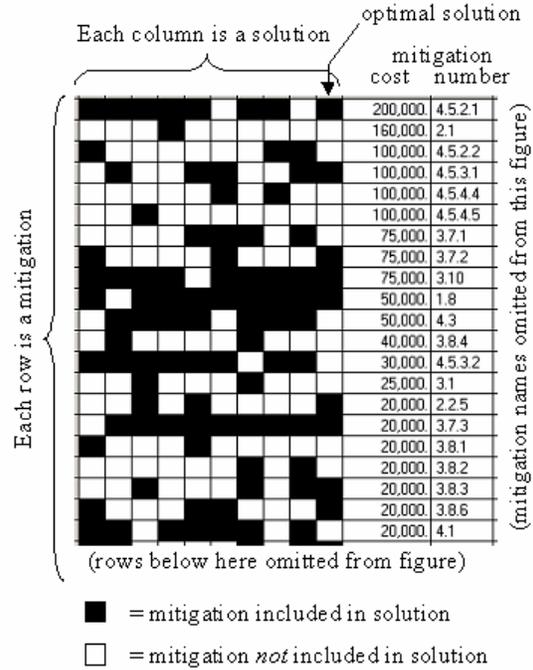


Figure 13. Visualization of 10 dispersed solutions



Figure 14. Visualization of clusters of solutions

The visualization of clusters is similar, seen in **Error! Reference source not found.** Rows correspond to Mitigations, while in this portrayal columns correspond to *clusters*. Since a cluster itself comprises multiple solutions, a given Mitigation may be involved

in none, some or all of the solutions in a cluster. This is indicated by shading the square – white means not involved in any of the solutions within that cluster; black means involved in all the solutions within that cluster, and intermediate shades of grey denote intermediate levels of involvement.

Discussion: Our approaches to exploring the solution space are founded on a combination of computation and visualization. Computation is used to reduce the number of items (key decisions, dispersed solutions or solution clusters). Visualization is used to inform decision-makers of the ramifications of those decisions and to scrutinize the makeup of a modest number (e.g., 10) of solutions.

This combination of computation *and* visualization appears to us a fruitful area worthy of further investigation. As mentioned in the previous section, search-based generation of the cost-benefit tradespace is a time-consuming activity, so we would be particularly interested in approaches that could begin to yield insights as the search is progressing, rather than have to wait until it has concluded to begin to scrutinize its implications.

6. INVESTIGATIONS OF RESEARCH AND TECHNOLOGY PORTFOLIOS

In addition to applications of DDP to assist the infusion of *individual* technologies, in some cases there have been DDP applications that have focused on entire *portfolios* of technologies, the aim being to make a selection of a set of technologies to pursue. The first of these was conducted by JPLer David Tralli, who used DDP to assist activity selection across an entire program of NASA Earth Science Missions [22].

We have since used this same approach in a pilot study of the connections between the needs of software IV&V practitioners, and a related software assurance research program containing multiple research efforts [12].

The purpose was to gauge how well the research program matched practitioner needs. In our pilot study of this approach, based on partial set of data, we again used DDP’s topology visualization (as was seen in Figure 1) to present the data – the result is seen in Figure 15.

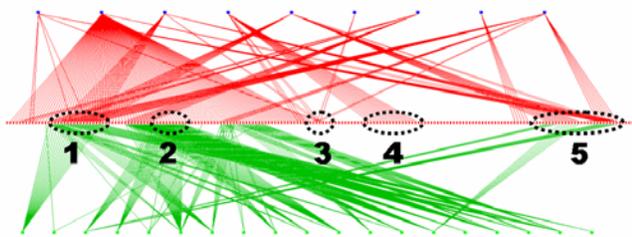


Figure 15. Topology of Needs-Areas-Researchers

The top row represents 9 IV&V practitioners. They were asked to express their needs in terms of the 198 leaf nodes in the “software” area of the ACM Computing Classification System [1]. These 198 leaf nodes form the middle row, and red lines connect practitioners with their expressed needs. The bottom row represents 19 researchers who were asked to express their research in terms of the same 198 leaf nodes in the “software”

area. The green lines connect researchers to the areas they work in.

We have annotated the DDP-generated chart with 5 ellipses, each exemplifying a different phenomenon:

1. Very good overlap between areas of need shared by multiple and those same areas included in multiple researchers’ activities.
2. An area of potentially over-addressed needs. Only one practitioner has expressed needs in these areas, yet multiple researchers have activities in these areas.
3. Unaddressed needs shared by several practitioners.
4. Unaddressed needs of a single practitioner.
5. Good overlap – areas shared by several practitioners and covered by several researchers.

Furthermore, our data included quantitative estimates of how the practitioners’ needs were distributed across the areas they identified (some areas were more important to than others), and quantitative estimates of how the researchers’ activities were distributed across those same areas (some areas were more the focus of the research than others). This quantitative data allowed us to utilize DDP’s risk calculations, but in this model instead of “Objectives” impacted by “Risks” effected by “Mitigations”, we had “Practitioners” with needs for improvements in “Areas of Computer Science” which in turn were areas to which “Researchers” would (if successful) contribute.

To show the quantitative aspects, we couple the topology visualization with the bar chart visualization. This is seen in Figure 16.

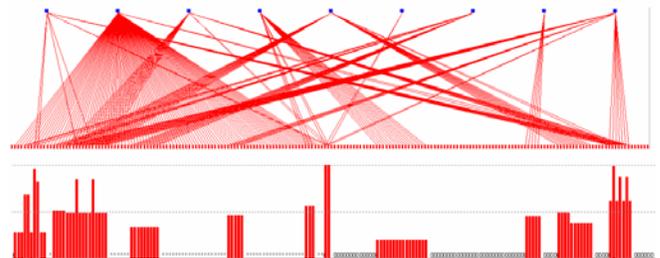


Figure 16. Topology coupled with bar chart

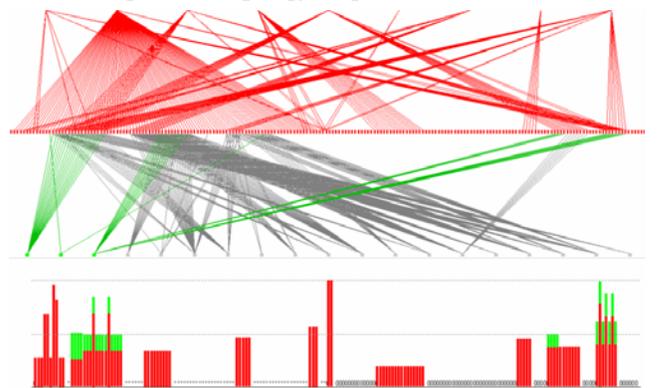


Figure 17. Fulfillment of needs from several research efforts

The red bars indicate sum total need in each of the areas (the higher the bar, the more need; again, bar heights are with respect to a log scale, so the differences are quite pronounced).

The extent to which researchers' activities contribute to the practitioners needs can be calculated and displayed, as seen in **Error! Reference source not found.** This figure shows the status when the left three research activities are selected (identifiable by their connecting lines colored green). DDP has calculated their combined effect at meeting practitioners' needs. The bar chart colors of red and green indicate, respectively, unfulfilled need, and need fulfilled by the currently selected researchers.

Discussion: This topology visualization appears very well-suited to revealing these phenomena in this particular model, particularly when coupled with the bar chart display. This is in contrast to the model that gave rise to topology shown in Figure 1, where there were so many connections it was hard to discern any significant patterns. Most of our applications to individual technologies yield equally inscrutable topologies. There may be something different about the nature of portfolio planning that makes this approach more amenable – at present we have not yet performed enough such portfolios studies to know.

7. CONCLUSIONS AND RELATED WORK

The purpose of this paper has been to suggest that there is scope for the application of software visualization to the earliest phases of the development lifecycle.

We have illustrated this with examples taken from our studies of spacecraft technologies and systems (our studies span software, hardware and combinations of both). We have incorporated into our risk-centric modeling software visualization capabilities specifically for the purpose of informing decision-makers. All figures in this paper are generated by this software, and all are based on actual models constructed in the course of our work.

The visualizations themselves are relatively commonplace (bar charts, 2-dimensional scatter plots, connection graphs and tabular formats). The novelty of this work, if any, lies in its application to early phase decision making. Underlying our approach is the quantitative model attributable to Steve Cornford [5] (see <http://ddptool.jpl.nasa.gov>). Our use of a combination of computation and visualization rests on top of this. We would be happy to learn of visualization results that could help our work - especially algorithms, etc., that would offer improved performance and/or scaling to larger datasets, and other visualizations that would yield additional insights of value to decision makers. For example, one of our colleagues pointed us to the "TreeMap" work [2]; since our information is often organized into hierarchies, we have implemented one of the "Ordered TreeMap" algorithms and are in the process of investigating how best to make use of it within our framework.

The pioneering work on requirements prioritization reported in [14] looked into the challenges of selecting the set of requirements to go into the next iteration of development or release of a product. The approach is based on gathering for each requirement estimates of its cost and benefit. Visualization in the form of a 2-dimensional chart presents the cost vs. benefit position of individual requirements, thus allowing users to select accordingly. Similar approaches are seen in the Win Win project [3] which supports multiple stakeholders to identify conflicts between their respective evaluations of requirements, and to

locate feasible solutions that are mutually satisfactory combinations of requirements. Again, visualizations in the form of 2-dimensional charts are used, provided by the automated aids they have built to support this approach [13]. The use of bar and pie charts for similar purposes is seen in [19]. Our section 4 describes closely related work to this, the key difference being that we do not feel able to ascribe directly to a requirement its cost and benefit – rather, we use our three-layer "Objectives", "Risks" and "Mitigations" model to capture the more intertwined dependencies that we find arise in many of our studies. Also we deal with larger numbers of items – many dozens, sometimes hundreds – for which there is need for additional (still relatively straightforward) mechanisms to sort, elide, filter etc.

Our display of the Pareto front to visualize the cost-benefit tradespace is commonplace in the (typically *non*-software design) optimization world [20]. As we start to delve into the makeup of the cost-benefit tradespace, however, we may be in less explored territory, although we are aware of, e.g., [21] which uses visualization techniques to show multiple dimensions at once of the attributes of a design.

The use of heuristic search techniques "as a tool of optimization of software engineering problems" is discussed in [4]. That article surveys past applications of heuristic search in areas of test data generation, module clustering and cost/effort prediction, and considers potential applications in additional areas. (Interestingly, one of the areas they consider is the aforementioned requirements prioritization problem.) Their focus, however, is on matching software engineering problems to heuristic search methods in order to be able to apply those methods. They do not continue to the point where the search results must be presented to users, and so are not motivated to consider issues of visualization in support of this.

Finally, we mention some other work ongoing at JPL that, like ours, aims to inform decision makers early in the lifecycle: Strategic Assessment of Risk and Technology (see <http://start1.jpl.nasa.gov>). Applications to technology portfolio selection are given in [23]. Their approach rests on construction of decision theory models somewhat more detailed than ours (which take correspondingly more time and effort to construct). They too utilize fairly straightforward visualizations to present the implications of those models to decision-makers.

8. ACKNOWLEDGMENTS

The research described in this paper was done 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. We thank Steve Cornford, who leads the DDP effort at JPL, Ken Hicks and Ken Johnson for their facilitation of technology studies using DDP, Chet Borden, Stephen Prusha and Andrew Shapiro for their insights and guidance, and numerous of our colleagues who have contributed to elements of this work.

9. REFERENCES

- [1] ACM Computing Classification System [1998 Version] <http://www.acm.org/class/1998/>
- [2] Bederson, B.B., Shneiderman, B. and Wattenberg, M. Ordered and Quantum Treemaps: Making Effective Use of 2D Space to Display Hierarchies. *ACM Transactions on Graphics (TOG)*, 21 (4) (October 2002), 833-854.
- [3] Boehm, B., Bose, P., Horowitz, E. and Lee, M. Software Requirements as Negotiated Win Conditions. In *Proceedings 1st International Conference on Requirements Engineering*, (Colorado Springs, Colorado), IEEE Computer Society 1994, 74-83.
- [4] Clarke, J.; Dolado, J.J.; Harman, M.; Hierons, R.; Jones, B.; Lumkin, M.; Mitchell, B.; Mancoridis, S.; Rees, K.; Roper, M.; Shepperd, M. Reformulating software engineering as a search problem. *IEE Proceedings - Software Engineering* (June 2003) Volume 150, Issue 03. 161- 175.
- [5] Cornford, S.L. Managing Risk as a Resource using the Defect Detection and Prevention process. In *4th International Conference on Probabilistic Safety Assessment and Management* (New York City, NY, September 13-18. 1998) International Association for Probabilistic Safety Assessment and Management.
- [6] Cornford, S.L., Feather, M.S., Dunphy, J.R., Salcedo, J. and Menzies, T. Optimizing Spacecraft Design – Optimization Engine Development: Progress and Plans. In *Proceedings of the 2003 IEEE Aerospace Conference* (Big Sky, Montana, March 2003): 7-3361 – 7-3368.
- [7] Diehl, S. and Stasko, J.T. *Proceedings of the 2003 ACM Symposium on Software Visualization* (San Diego, CA, June 11-13, 2003). ACM Press.
- [8] Feather, M.S. Towards Cost-Effective Reliability through Visualization of the Reliability Option Space. In *Proceedings of the 2004 Annual Reliability and Maintainability Symposium (RAMS)* (Los Angeles CA, January 2004) 546-552.
- [9] Feather, M.S. and Cornford, S.L. Quantitative Risk-Based Requirements Reasoning. *Requirements Engineering* (2003) 8: 248-263.
- [10] Feather, M.S., Kiper, J. and Kalafat, S. Combining Heuristic Search, Visualization and Data Mining for Exploration of System Design Spaces. In *14th Annual International Symposium Proceedings of INCOSE 2004* (Toulouse, France, June 20-24 2004).
- [11] Feather, M.S. and Menzies, T. Converging on the Optimal Attainment of Requirements. In *Proceedings of the IEEE Joint International Conference on Requirements Engineering* (Essen, Germany, September 9-13, 2002), IEEE Computer Society, 2002, 263-270.
- [12] Feather, M.S., Menzies, T. and Connely, J.R. Matching Software Practitioner Needs to Researcher Activities. In *Proceedings of the 10th Asia Pacific Software Engineering Conference (APSEC 2003)* (Chiang Mai, Thailand, December 10-12, 2003). IEEE Computer Society, 2003, 6-16.
- [13] In, H., Boehm, B., Rodgers T. and Deutsch, M. Applying WinWin to Quality Requirements: A Case Study. In *Proceedings 23rd International Conference on Software Engineering*, (Toronto, Canada May 2001) IEEE Computer Society, 2001, 555-564.
- [14] Karlsson, J. and Ryan, K. A Cost-Value Approach for Prioritizing Requirements. *IEEE Software*, (Sept./Oct. 1997), 67-74.
- [15] Kiper, J.D. and Feather, M.S. Mining Complex Requirements Specifications to Mitigate Risk via Clustering. In *Proceedings, Workshop on Intelligent Technologies for Software Engineering (WITSE'04)* (Linz, Austria, September 20-25 2004) Austrian Computer Society.
- [16] Kirkpatrick, S., Gelatt Jr., C.D. and Vecchi, M.P. Optimization by simulated annealing. *Science*, (13 May 1983), Volume 220, Number 4598: 671–680.
- [17] Menzies, T. and Hu, Y. Data Mining for Very Busy People. *IEEE Computer* 36, 11 (November 2003), 22-29.
- [18] de Pauw, W., Reiss, S.P. and Stasko, J.T. ICSE Workshop on Software Visualization, In *Proceedings of the 23rd International Conference on Software Engineering* (Toronto, Canada, May 2001) IEEE Computer Society, 2001, 758-759.
- [19] Regnell, B., Host, M., Nach och Dag, J., Beremark, P. & Hjelm, T. An Industrial Case Study on Distributed Prioritisation in Market-Driven Requirements Engineering for Packaged Software. *Requirements Engineering* (2001) 6: 51-62.
- [20] Sen, P. and Yang, J-B. *Multiple Criteria Decision Support in Engineering Design*. Springer-Verlag, 1998.
- [21] Stump, G. M., Yukish, M., Simpson, T. W., and Bennett, L. Multidimensional Visualization and Its Application to a Design by Shopping Paradigm. In *Proceedings 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*. (Atlanta, GA, 2002), AIAA, AIAA-2002-5622.
- [22] Tralli, D.M. Programmatic Risk Balancing. In *Proceedings of the 2003 IEEE Aerospace Conference* (Big Sky MT, March 2003).
- [23] Weisbin, C.R., Rodriguez, G., Elfes, A. and Smith, J.H. Toward a Systematic Approach for Selection of NASA Technology Portfolios. *Systems Engineering* 7 (4) (September-October 2004) 285-302.