First Contract: Better, Earlier, Decisions for Software Projects

Martin S. Feather
Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109, USA
Martin.S.Feather@jpl.nasa.gov

Hoh In
Dept. of Comp. Sci.
Texas A&M University
College Station
TX 77843-3112, USA
hohin@cs.tamu.edu

James D. Kiper
Dept. of Comp. Sci.
and Systems Analysis
Miami University
Oxford, OH 45056, USA
kiperjd@muohio.edu

Tim Kurtz
Science Applications
International Corporation,
NASA Glenn Research Ctr.
4031 Colonel Glenn Hwy.
Dayton, OH 45431, USA
Tim.Kurtz@grc.nasa.gov

Tim Menzies
Dept. Elec. & Comp. Eng.
Univ. British Columbia,
2356 Main Mall,
Vancouver, B.C.
Canada, V6T 1Z4
tim@menzies.com

Abstract

Decisions made in the earliest phases of software development have the greatest effect on the likelihood of project success. These decisions are used to establish the requirements of the product under development, determine budget and schedule, plan the allocation of resources to the development, etc. This decision making is very challenging, given the impediments of incomplete information, absence of a shared vision, hard-to-discern ramifications of choices, etc.

This paper describes an ongoing effort to address these challenges by means of a synergistic collaboration of research tools, techniques and best-practice knowledge. The goals of this effort are to give the end users more soundly based estimates, enhanced abilities to make trades amongst requirements, risks, and development, and better optimized development plans. To achieve these goals, our effort combines research in the areas of estimation, planning, visualization, elicitation, negotiation, machine learning, abduction, and knowledge representation.

A hypothetical scenario is used to introduce and illustrate our approach. We describe the key contributions that each of our research efforts provides, and go on to consider the synergistic benefits of their combination. Realization of this combination is well underway; status and future plans are described. Finally, this collaborative work is related to existing research and practice, to show how our approach furthers the ability to make critical decisions in the early phases of software development projects.

KEYWORDS: requirements elicitation, estimation, project planning, negotiation, tradeoffs, IV&V, risk management, stakeholders, optimization, visualization, collaboration, abduction, induction

1. Introduction

Even seasoned programmers can be surprised and alarmed at the planning effort that precedes implementation. For example, consider the case of Kay. After many years as a valued C++ programmer, Kay's human shield against management work is suddenly promoted to headquarters. Since no one else in her group has her years of experience, Kay is made a manager and told to lead a new project: the software for the Near Earth Dangerous Asteroid Detection and Destruction project (NEDAD2). Her supervisor, a haggard projects manager (HPM), informs Kay that NEDAD2 has a schedule of three years and a $15,000,000 budget. HPM asks Kay for an SDP and a schedule of design reviews by next week.

"Huh?" says Kay.

HPM realizes that Kay knows a lot about memory management in C++ and very little about project management. He provides Kay with a reading list. Kay learns many things including that SDP is a Software Development Plan and that decisions made in the earliest phases of software development have the greatest effect on the likelihood of project success.

Trade-offs lie at the core of all early life cycle decision making. Such trade-offs are trivial if they are between clear alternatives, one of which was superior in all respects to the other. This is rarely so. The norm is for each alternative to be superior along one set of dimensions, but inferior along another, thus necessitating reasoning involving trade-offs. For example:

- Allocation of limited time, money and personnel among the various development and assurance tasks;
- Selection of which requirements to fulfill, what priority to assign to them, and how much risk to accept;
- Trading schedule risk for product performance.

However, decision making in these early stages is hard to do wisely. To help managers like Kay, NASA is funding the development of an integrated tool suite to support better early life cycle trade-off:

- Ask Pete is a *software knowledge tool* which acts as a smart librarian to helps software engineers access a large library of established software engineering knowledge [4]
- ARRT/DDP is an *exploration tool* allowing groups of stakeholders to refine and extend the knowledge from the librarian, and effectively explore the sum total of this knowledge [2].
- TARZAN is an *options reduction tool* which automatically explores and rejects the non-critical issues [9]. This is useful for working with the project knowledge gathered by the other tools, to know which areas of uncertainty need refinement, and to identify the most effective options and tradeoffs.

- VCR/DCPT is a *detailed trade-off tool* which lets stakeholders resolve their conflicts of the critical issues [11].

These tools already exist but only as isolated utilities. The rest of this article argues that their sum is greater than the parts. By connecting them all, each tool can augment and extend the capabilities of the others. In fact, without any one of these tools, a critical function in early life cycle trade-off cannot be performed:

- Without the software knowledge tool, managers must manually and laboriously explore the voluminous software engineering literature to find the knowledge relevant to their project.
- Without the exploration tool, competing stakeholders must manually explore and record their requirements, the risks to their requirements, and the actions that can reduce those risks.
- Without the options reduction tool, managers and stakeholders will be overwhelmed by the sheer volume of what-ifs and unknowns within their project.
- Without the detailed trade-off tool, managers and stakeholders will not be able to understand the interactions and implications of the critical issues within their projects.

The rest of this article describes these tools, their interaction, our progress towards their integration, and how managers like Kay can use them to make better early life cycle trade-offs.

2. A Hypothetical Scenario

Assuming the year is 2002, then Kay could use our tool set and proceed as follows.

Kay connects to the web and downloads the software tool suite from http://tkurtz.grc.nasa.gov/pete [4]. The knowledge tool guides Kay through a set of questions. When Kay can't understand the question, extensive help text guides her through the details. The software knowledge tools uses it's built-in effort estimation capability to estimate that the project will cost too much and take too long; i.e. $77,072,000 to build and 89 months to complete. Further, the software knowledge tool advises that this will project will require "critical levels" of control; i.e. elaborate management supervision. Kay is not surprised that the supervision level is so high but knows she must reduce the cost and schedule.

First, Kay runs some alternatives through the software knowledge tool. These reveal that if Kay gets the most experienced team possible, and uses the best tools available, then:

- The required level of supervision reduces from "critical"' to "high".
- The cost and time estimates reduce to $14,790,000 and 53 months respectively.

Since this is still too long, Kay tries something else. She asks the software knowledge tool to postulate that the project is split into two teams, plus an integration team to combine the outputs of the other teams. The software knowledge tool estimates that each subproject in this new structure will cost $7 million and take 42 months to complete.

Kay realizes she is on to something here. The next alternative she tries is to split the development team into three (plus the integration team). The three teams could work on the space platform, the acquisition and control system, and the tracking and firing system respectively. The result - each system could be built in about 35 months and would cost about $4 million apiece or $12 million for all three systems. The schedule would be close but she would have a $3 million cushion. She would need about 129 engineers or 43 per team.

Happy with the results, Kay then asks the software knowledge tool to generate the initial SDP (Software Development Plan). The software knowledge tool outputs detailed templates of report documents and proposes a schedule when those documents should be delivered. This is good – programmer/manager Kay hates thinking about all that documentation stuff and likes the idea that it's all laid out in a near fill-in-the-box format.

So, after a mere 24 hours, Kay thinks she has a good grasp on the project. She still had six days to put together the project schedules and complete the SDP. Time for lunch.

Over lunch, Kay gets talking to some of the hardware engineers sitting at the same table. For the first time, she hears of "safe-mode": to avoid the adverse effects of solar flares, satellites shut down all non-essential systems and normal operations cease. What if an asteroid sneaks in during the safe mode?

Lunch is followed by indigestion and doubt. How well could the computer hardware operate under such conditions if it had to? To what extent could software be used to mask the hardware problems? Kay is certain of one thing – there's a lot about spacecraft and their software that she doesn't know! She needs more input from experienced engineers.

In order to structure such discussions, she turns to the exploration tool. This second tool supports meetings where groups gather to combine expertise from several areas. To initiate those meetings, Kay exports the SDP from the software knowledge tool into the exploration tool. In discussion with her haggard program manager, they identify domain experts with relevant skills:

- A mission design specialist who worked on near-earth orbit spacecraft in JPL, California;
- A Pentagon scientist in Washington D.C. who understood the atomic weapons to be used in NEDAD2.

- A local NASA expert in San Francisco who specializes in software for real-time systems;
- One of the Software Quality Assurance staff who seemed to know a lot of the things to watch out for.

After some phone calls, three teleconferences are scheduled - one a day for the next three days. The SQA person is available that afternoon, so the two of them make a start at laying out the major categories of project-specific requirements, risks, and risk mitigation activities.

Wednesday morning the whole group convenes. After some initial training with the exploration tool, they review what inputs they have from the software knowledge tool. Working together, they add the detailed project requirements into the categories that Kay and the SQA person had established. Early afternoon they are able to make a good start at linking the requirements to the risks, before having to depart for other meetings. At all times, two rules guided their work:

- When stakeholders disagreed on some point, that point was elaborated. Usually, on elaboration, it was discovered that the "disagreement" was really a confusion of two or more ideas - one of which had not been previously considered by the opposing party. When such a "hidden" point was explicated, the feuding experts often said, "oh, I see what you mean, I had not thought of that".
- When stakeholders truly did not know the details, a *range marker* was added. Such a range marker says that "we don't know, but we think that the possibilities lie within this range".

Kay's last act of the day is to hook the options reduction tool into the exploration tool. The options reduction tool explores all the range markers looking for critical values where the cost and chances of achieving some requirement changes radically. Via extensive what-if queries, the options tool learns what range points are critical and what range points don't change the overall conclusions of the debate. This tool's overnight run makes use of all the spare CPUs lying around on desktops after their owners have gone home.

Next morning, Kay presents the options reduction results to the group. Guided by these, the group revisits the exploration tool's visualizations of the risk landscape, and its relationship to requirements, enabling them to identify several areas of requirements that are particularly problematic. Of the 107 range markers added in yesterday, 10 key range values were identified that have a critical impact on the overall satisfiability of these requirements. As the day begins, Kay cautions her team that while they could discuss all 107 "don't knows", it is probably most productive to explore just those 10 key values. The team agrees and subdivides into several teams to work on these issues for the rest of the morning.

For the critical options, the teams use the detailed trade-off tool to score the relative benefit (importance) and cost of all the relevant requirements. Based on their evaluation data, the trade-offs tool visualizes the data. It turns out that this tool has seen debates over this kind of software before. So the tool uploads historical, similar project data to visually compare the currently agreed evaluation to the historical data. For the current debate, areas of strong consensus and disagreement are highlighted. Further, the trade-off tool automatically identifies and clusters several groups that are conceptually very close. The differences between these groups are analyzed. As a result, the teams effectively and efficiently reach agreement with assistance of the analysis summary provided by the detailed trade-off tool.

After lunch the teams meet again to report some breakthroughs in mutual understanding. The team processes the new insights, allowing them to make some trades among the requirements, and emerge with what they hope will allow a feasible development effort. They enter the new data, just as before, into the exploration tool. Again, at the end of the day, Kay activates the options reduction tool before going home.

The next day sees Kay at work bright and early, studying the overnight run of the options reduction tool on the amalgamated data. It looked promising, but still leaves her with uncomfortably small margins in both cost and risk. When the group has once again convened, they focus on the choice of risk mitigations. As before, the options reduction tool's results point to those options that have the most leverage. They are able to refine effectiveness values for some of the activities within the standard CMM categories. In the course of this, Kay is surprised to learn that "formal inspections" and "formal methods" are not related. She doesn't quibble, especially when it becomes apparent that local expertise in both of these means that they can effectively address several of the risk areas that she had been worrying about the last couple of days. The net result is deemed acceptable to all, much to Kay's relief. Together, they have emerged with a plan that is not only within cost and schedule limits, but also shows how it addresses all the risks that this entire team of people, plus the background data sets, could throw at the problem!

The final icing on Kay's cake is the automatic report generation. The software knowledge tool takes the results from the exploration tool, and generates for her (among other things), the entire SDP her haggard project manager had asked for by next Monday.

Kay felt she had had a great first week. Her first task accomplished, and her notebook filling up with the expansions of the many three-letter acronyms that these NASA folks used incessantly. This job could be fun. She spends her weekend shoveling snow from her driveway - to her, it feels better than those 20-hour, 7-day a week programming marathons! Yes indeed, she muses as she breaks into song in the snow, these tools were just what she NEDAD2 use.

3. Benefits of the Collaboration

This section describes the benefits that stem from our collaboration. Section 3.1 presents an elementwise discussion of the unique benefit that element X provides to the collaboration as a whole, for each of our efforts. Subsection 3.2 presents a pairwise discussion of the synergies gained from combining X with Y, for each pairing among our efforts.

3.1. Unique Contribution of Each Component

The five components of our effort provide their unique strengths to the collaboration as a whole. This subsection considers each of them in turn.

3.1.1 Unique Contribution of the Software Knowledge Tool, **Ask Pete:** The Ask Pete tool integrates cost and schedule estimation, software project tailoring and IV&V criteria into an interview and reporting tool which combines to provide the following capabilities to the collaboration:

- **Characterization**: analysis of various aspects of the project yields a model of the project based on risk, complexity, resources and size,
- **Tailoring**: classifying the project into different levels of effort based on the characterization,
- **Planning**: combining the results of tailoring the project into detailed plans and estimates for development and quality assurance
- **Tradeoff analysis**: adjusting various project, organization and resource factors to reduce risk, and minimize resource and control requirements.

The tool is designed for use by project managers, software quality and IV&V personnel for planning and comparison and negotiating the full range of development activities, and to adjust plans as project parameters change or evolve. It prompts the user for inputs via a series of multiple choice questions and True/False questions preset to gather the information needed for its behind-the-scenes calculations. These include use of COCOMO to compute cost and schedule estimates, and of standardized tables to classify the project into one of a range of "control levels" and IV&V candidacy. Ask Pete combines the results of these computations with its built-in knowledge of center-specific development practices and policies to yield recommendations of software development, quality assurance and IV&V plans.

The collaboration is dependent on Ask Pete for these capabilities in the areas of estimation and planning. Their lack would force the laborious consideration of these same issues, but starting from first principles. In practice, this would likely mean inferior results, with slipshod guesswork replacing well-grounded estimation, and important factors overlooked by mistake or oversight.

3.1.2 Unique Contribution of the Deliberation Tool, ARRT/DDP: This tool, and the process it embodies, melds requirements, risks and mitigation reasoning with sophisticated interactive support. The tool is designed for real-time use in a group setting, where a small set of experts (10 - 15 has been typical in its applications to date) pool their knowledge pertinent to the task at hand. The premise is that the tool can significantly support, but not supplant, their activities. ARRT is a specialization of DDP [1] to the software domain. ARRT/DDP provides the following key capabilities to the collaboration:

- **Elicitation**: facilitating the gathering of project-specific information from human experts representing a range of disciplines. Familiar visual metaphors ease the activities of on-the-fly data entry and organization, while accommodating capture of both qualitative and quantitative information.
- **Understanding**: visual presentation of the sum total of information (computed and derived) to facilitate those same experts' understanding and exploration of options and tradeoffs. A coordinated mix of visual presentations allows rapid alternation of views. Dynamically adjustable summarizations, aggregations and hierarchical presentations facilitate navigation through a large space of information.

Their lack would severely diminish our capacity to use human experts' knowledge and insights while gathering information crucial to the planning process, and subsequently conducting that tradeoff-based planning. This lack would be sorely felt in our setting of early-phase planning of the development of mission-critical software.

3.1.3 Unique Contribution of Knowledge & Knowledge Representation In the context of this paper, the term knowledge representation refers to the addition of information about potential software risk and useful mitigating activities culled from reputable sources, and expert judgements of the impact of various mitigations on specific risk types. This knowledge adds the following capabilities to this collaboration.

- **Experience**: providing knowledge culled from best practice sources about potential risks [13] and associated mitigating activities [12] is a valuable resource for managers of software projects. In addition, the expert judgements captured in this information about the strength of influence of mitigating activities on specific risk gives these same managers the ability to chose effective combinations of mitigations.
- **Expressive power**: the use of an expressive set of logical interrelationships among and between the data elements extends our ability to more accurately model complex relationships among software risk and mitigating activities (e.g., logical fault trees; mitigations that have overlapping effectiveness); the

capability to specify ranges and distributions allows capture of our confidence in, and known bounds on, the knowledge.

Without this knowledge of potential software risks and best mitigating practices, managers would be left to their own resources to determine what risks their software project was likely to encounter, what possible mitigating activities would be effective, and what the strength of these mitigations would be on the predicted risks. A lack of expressive power in this collaboration would severely limit its capability for accurately modeling the interactions between risk and mitigations, thereby compromising a manager's ability to chose effectively among possible activities

3.1.4 Unique Contribution of the Options Reduction Tool, TARZAN. This tool exploits the average shape of the argument space to find the key issues that control the rest of the space. Theoretically, there are an intractable number of such key issues. 20 binary choices implies 2^{20}=1,000,000 options. Fortunately, empirical and theoretical evidence suggests that many of those options are inter-dependent and the key options are a very small subset of all options. By resolving the key options, the other options are forced to follow.

TARZAN works via abduction and induction. Abduction is the logic of argument and can be informally defined as follows: find the assumptions that lead to desired goals without violating constraint rules (for a more formal definition, see [10]). Tacit in this definition is the idea of multiple worlds of beliefs. When assumptions contradict other assumptions, abduction will generate multiple solutions, each of which represents a single consistent set of assumptions. The number of worlds may be huge but, in practice, is surprisingly small [7] and the theoretical explanation for why this is so in [6]). Consequently, if we use machine learning to induce a summary of many abductive runs, we often find that only a small number of the unknowns are critical to the overall behavior of the system.

In practice, TARZAN can never endorse options. Rather, it identifies which options can be safely ignored. Experience with the tool strongly suggests that many options can be ignored and only a few options are truly key to a system.

3.1.5 Unique Contribution of Detailed Tradeoff Tool, VCR/DCPT: Planning of complex software development efforts involves combining inputs and accepting guidance from experts from different development disciplines (e.g., end-users, developers, SQA and IV&V personnel) as well as different aspects of software (e.g., real-time, reliability, resource usage). Wherever their areas of expertise overlap, there is potential for disagreement due to conflicting goals, differences of opinion, subjective bias etc. VCR/DCPT [11] provides the following capabilities to enable users to identify understand and resolve these disagreements:

- **Powerful analysis**: clustering analysis to identify stakeholder subgroups having different opinions, cause-effect analysis to extract the structure of disagreement by analyzing stakeholder profile and group portfolio, calculation of degree of disagreement, etc.
- **Visualization**: intuitive graphical presentations of the nature and degree of disagreement, both among multiple experts' risk assessment data, and against historical data from similar projects.
- **Decision evolution support for rapid reassessment**: via keeping track of all decision rationales in stakeholder profiles, and retrieving (and comparing) previous assessment results via visualized navigation aids.
- **Effective and efficient communication for risk assessment among stakeholders**: supporting groupware capability [3] to allow (even geographically-distributed) stakeholders to present their assessment in standard ways (agreed multi-criteria, units, voting mechanism), as well as in flexible ways by attaching their rationales, analysis tool results, and documents into each voting.

The collaboration is dependent on VCR/DCPT for these capabilities in the areas of analysis, visualization, negotiation and communication support. Their lack would leave the entire planning process open to inefficient conflict resolution, bias, inadvertent subjectivity, and inequitable decisions.

3.2. Pairwise Synergies

Our collaboration comprises the five efforts described in section 3.1. There are thus twelve possible pairings among these. The table below is an index from these pairings to the discussions that follow.

	KT	ET	KR	ORT
Knowledge **T**ool Ask Pete				
Exploration **T**ool ARRT/DDP	3.2.1			
Knowledge & **R**epresentation	3.2.2	3.2.3		
Options **R**eduction **T**ool TARZAN	3.2.4	3.2.5	3.2.6	
Detailed **T**rade-**O**ff **T**ool VCR/DCPT	3.2.7	3.2.8	3.2.9	3.2.10

3.2.1. [ET+KR] Exploration Tool + Knowledge Tool. Our Knowledge Tool, Ask Pete, with its focus on estimation and planning, meshes well with our Exploration Tool, ARRT/DDP, with its strengths in the areas of elicitation and visualization. In their joint use, Ask Pete is run first to gather project characteristics, employs COCOMO and other models to generate cost, schedule and risk criticality estimates, and makes an initial recommendation of risk mitigating activities. This is passed over as a plausible and detailed starting point for

ARRT/DDP. Once within ARRT/DDP, users can customize and tailor the information according to their case at hand and their expert understanding. At this stage, human navigation of the "risk landscape" takes place so as to emerge with a cost-effective and balanced risk mitigation plan. The conclusions of this second phase are passed back to Ask Pete for incorporation into development and quality plans that will indicate the cost, schedule and nature of assurance-related activities over the future life of the project. In conjunction, therefore, the two tools provide complementary capabilities, and couple software factory like process knowledge with the ability to customize, tailor and scrutinize information.

3.2.2. [KR+KT] Knowledge & Representation + Knowledge Tool. These two efforts are complementary in that each brings established knowledge from reputable sources to the task of software project management one for risk management, the other for project size and cost estimate and planning. Ask Pete characterizes the project and provides the initial set of suggested development activities. Each of these has some effect on one or more of the possible risks to the project. These effects are quantified in ARRT/DDP and form the basis for risk balancing. This information is used in conjunction with the software risk information and linkages to mitigations to tailor the activities to eliminate or replace ones that will have a negligible affect on the development risks and to determine additional mitigations that may be necessary to reduce risks to acceptable levels. This information can then be fed back to the estimation and planning tool to determine the scheduling and cost impact of additional mitigations and for incorporation into the development plan.

3.2.3. [KR+ET] Knowledge & Representation + Exploration Tool. The knowledge culled from best-practice sources (CMM activities, SEI taxonomies of risks), and augmented appropriately (cross-linked to indicate which, and how much, each activity mitigates each risk) is used to pre-populate the ARRT/DDP tool. This has the obvious advantages of time savings, serving as checklist-like reminders, etc. Meanwhile, the visualization and associated manipulation capabilities of the ARRT/DDP tool facilitate the navigation of these non-trivial datasets (e.g., we have on the order of one thousand non-zero effectiveness links between these activities and risks). Navigation through the space of software risks, mitigations, and linkages allows managers to experiment with various combinations of mitigations (with varying costs) to reduce risk to an acceptable level. Future work on extending the knowledge representation will refine the (currently rather simplistic) model that ARRT/DDP assumes of requirement, risk and mitigation information. The influences of mitigations on software risks and requirements are complex and inter-related. A language with logical connectives is necessary to represent these interactions. This language should also support ranges or distributions of confidence values, rather than a simplistic numeric rating of the strength of these relationships.

3.2.4 [ORT+KT] Options Reduction Tool + Knowledge Tool. Our Knowledge Tool, Ask Pete, provides the set of possible characteristics and activities of through which TARZAN's machine learning can churn through to create trees. Eventually trimming them to determine the best possible combinations to promote project success.

3.2.5 [ORT+ET] Options Reduction Tool + Exploration Tool. This combination aims to allow us to scale to larger projects and more complex interrelationships among their elements, while retaining human input, insight and guidance. The volume of data that we have to gather increases for larger projects - they will have more requirements to juggle, more risks to be concerned with, etc. More complex interrelationships amongst our data elements (e.g., risk mitigation options that are somewhat overlapping) also add to the complexity with which we must deal. It is inevitable that we will have to gather a significant amount of project-specific information, so elicitation and visualization of that data remains a necessity. However, use of machine learning and abduction allows us to perform sensitivity analysis, so that we can know which subsets of the data are the most critical. This can direct our attention to these subsets, leading us to expand them to finer levels of detail, scrutinize them more carefully, provide confidence range information together with the data itself, etc. Another use of machine learning and abduction is to find near-optimal solutions (i.e., sets of risk mitigations), which can serve as suggestions to the human experts as they formulate their development plans.

3.2.6 [ORT+KR] Options Reduction Tool + Knowledge & Representation Because of the number and complexity possible software risks, mitigations to these risk, and interactions among them, it is necessary to have an automated way of analyzing this information and synthesizing it so that stakeholders can make choices. We suspect that the number of interactions is exponentially related to the number of risks and mitigations. Machine learning provides a technique for allowing a computer to explore this search space and effectively learn which factors are the most significant.

3.2.7 [DTOT+KT] Detailed Trade-Off Tool + Knowledge Tool. When stakeholders (experts) have different estimation results of project characteristics (cost, schedule, risk criticality, benefits of mitigating activities), VCR/DCPT calculates the degree of their disagreement,

visualizes the result with statistical analysis support via eclipses, lines, and dots, and suggest insights for potential resolution of these agreement based on stakeholder and group profile analysis. With VCR/DCPT, it is not necessary for all stakeholders to meet together at the same location or even at the same time in order to estimate and plan the project in Ask Pete. With VCR/DCPT, Ask Pete stakeholders can eventually arrive at agreement on the better effective and efficient way.

3.2.8 [DTOT+ET] Detailed Trade-Off Tool + Exploration Tool. Our Exploration Tool, ARRT/DDP, assumes that input data input can be combined into the same one pool, within which stakeholder identity is not a factor. Our Detailed Trade-Off Tool, VCR/DCPT, makes no such limiting assumption. Indeed, it recognizes that stakeholder identities are critical to identifying mismatches of assumption areas in need of negotiation, etc. The synergy between these two components comes from the ability to switch back and forth between their respective views. These two tools focus on different aspects of the decision making process. Hence, they each bring to bear their own, very different, set of graphical displays and underlying reasoning capabilities.

3.2.9 [DTOT+KR] Detailed Trade-Off Tool + Knowledge & Representation. The knowledge about software risks, mitigations to those risks, and the rich linkages among risks and mitigations is complex and extensive. Decisions about which software risks are significant for a particular project and choices of which mitigations are most appropriate for those risks is a function of the person or group of people making these decisions and choices. Individual stakeholders bring their own perspective and expertise to these tasks. The validity of these choices and decisions is increased when representation from the various classes of stakeholders is broadened. Visualization has been demonstrated to be an effective way for multiple stakeholders to come to a consensus about complex questions and choices.

3.2.10 [DTOT+ORT] Detailed Trade-Off Tool + Options Reduction Tool. Both the options reduction tool and the detailed trade-off tool explore the impacts of unknowns on the requirements. However, the tools are different in terms of the number of options they can test. The detailed trade-off tool excels when users are discussing the details of a small number of options. In the case of vast amount of options, the users could be overwhelmed with screens. The options reduction tool takes millions of options and culls the irrelevant and unimportant ones. This leaves a small pool of options which the reduction tool cannot distinguish between. The detailed trade-off tool then activates and users can explore the details of these remaining options.

4. Status

The collaboration among the previously discussed tools and techniques is an ongoing research effort. The status of this is that our Knowledge Tool, Ask Pete, and our exploration tool, ARRT/DDP, are the nuclei around which the collaborative components are coalescing. Both Ask Pete and ARRT/DDP operate as stand-alone applications. They have been used to support early-phase project planning, each concentrating on the aspects for which it is designed. The first major step of the collaboration has been to make these two tools operate together. Their combination is described in [5]. Also completed is the population of ARRT/DDP with knowledge drawn from SEI sources, augmented as needed (notably to provide quantitative estimates of how much various activities mitigate risks) [2]. Pilot studies are now underway in which this extant combination is being applied to assist in planning of IV&V.

Our Options Reduction Tool, TARZAN's technique of machine learning & abduction has been successfully applied to the COCOMO model [8, 9] independent of this collaboration. Its tight integration with the Ask Pete and ARRT/DDP components is work in progress.

We have built simple data exchange mechanisms to allow us to transfer information between components (e.g., to pass ARRT/DDP activities to VCR/DCPT). These have allowed us to conduct preliminary explorations of machine learning & abduction, and of stakeholder based conflict resolution & negotiation, on real Ask Pete and ARRT/DDP datasets.

The most far-reaching changes will stem from the elaboration of the knowledge representation. This will give us the capability to capture a wide range of relationships between and among requirements, risks and mitigations. However, it will pose challenges to make the activities of elicitation, visualization and reasoning scale to the complexity and volume of information, while retaining their convenience and intuitive appeal.

In the immediate future the primary goal for our collaboration is to complete and enhance the integration of the capabilities described in this paper. We also see important opportunities for capture and reuse of the knowledge and reasoning that takes place over multiple projects' planning and decision making. This would facilitate measurement and use of an organization's institutional strengths, and permit transfer of knowledge between similar projects.

5. Related Work, Conclusions

As part of his Requirements Engineering Research Perspective at ICSE 2000 [14] van Lamsweerde focused on "... modeling as a common denominator to all RE [Requirements Engineering] processes...". The research work that he surveys favors high-quality models that comprise detailed and rigorously expressed product requirements. That work employs tools to conduct

intricate formal reasoning on these representations, and results in a detailed understanding of what the product should be. In contrast, our approach allows users to work with lower quality and less-detailed requirements, that span both product and process (how the product will be developed). We employ a mix of tools that can operate effectively with partial, uncertain, rapidly gathered, information, and nevertheless yield useful results. The kind of reasoning that our tool suite conducts is more "shallow", but must operate in a much larger space of possibilities. Our results encompass planning the development activities, and permit exploration of tradeoffs in the unified arena of both product and process.

The main message of this report is that the synergistic effect of this collaboration has increased the effectiveness of the set of tools that are being developed. Each of these research efforts began life with important goals. As the individual efforts proceed, these goals are being realized. Furthermore, the integration of these tools has expanded their overall utility in significant ways.

- With the software knowledge tool, managers have access to expert knowledge about risks and mitigations from respected sources that is relevant to their project.
- With the exploration tool, requirements, risks, and mitigating actions are captured and summarized in automated ways that make shared understanding and consensus building possible.
- With the options reduction tool, managers and stakeholders can explore the complex space of possibilities within their project in structured ways.
- With the detailed trade-off tool, managers and stakeholders can understand interactions and implications of critical issues within their projects despite the complexity of information in its raw form.

6. Acknowledgements

The research described in this paper was carried out by Science Applications International Corporation (SAIC), the Jet Propulsion Laboratory, California Institute of Technology, he NASA Glenn Research Center, Texas A&M University, Miami University, and University of British Columbia under contracts with the National Aeronautics and Space Administration. Funding has been provided through the NASA Office of safety and Mission Assurance under the NASA Software Program lead by the NASA Software IV&V Facility, UPN 323-08-5P. 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, the Jet Propulsion Laboratory, California Institute of Technology, NASA Glenn Research Center, or SAIC. The authors gratefully acknowledge the significant contributions, guidance and assistance of Barry Boehm, Steve Cornford, Marcus Fisher, Michael Greenfield, Frank Huy, Kenneth McGill, Jungwon Park, Dan Port, Siddhartha Roy, and Siamak Yassini, with special appreciation to John Kelly and Martha Wetherholt.

7. References

[1] S. Cornford, “Managing Risk as a Resource using the Defect Detection and Prevention process”. Int. Conf. on Probabilistic Safety Assessment and Management, Sept. 1998.

[2] S.L. Cornford, M.S. Feather, J.C. Kelly, T.W. Larson, B. Sigal and J.D. Kiper, “Design and Development Assessment”, Proc. of the 10th Int. Workshop on Software Specification and Design, IEEE Comp. Society, Nov. 2000, San Diego, CA. pp. 105-112.

[3] In, H., Boehm, B., Rodgers, T., and Deutsch, M., “Applying WinWin to Quality Requirements: A Case Study”, IEEE ICSE, 2001, Toronto, Canada, May (to appear).

[4] T. Kurtz, "AskPete Web site" http://tkurtz.grc.nasa.gov/pete

[5] T. Kurtz and M.S. Feather, “Putting it All Together: Software Planning, Estimating and Assessment for a Successful Project”, in Proc. of 4th Int. Software Quality & Internet Quality Conf., Brussels, Belgium, Nov. 2000.

[6] T. Menzies and B. Cukic, “When to Test Less”, IEEE Software, 17, 5, pp. 107-112, 2000

[7] T.J. Menzies, S. Easterbrook, B. Nuseibeh and S. Waugh, “An Empirical Investigation of Multiple Viewpoint Reasoning in Requirements Engineering”, IEEE Int. Symposium on Requirements Engineering, 1999, pp. 100-109.

[8] T. Menzies and E. Sinsel and T. Kurtz, “Learning to Reduce Risks with COCOMO-II”, Workshop on Intelligent Software Engineering, an ICSE 2000 workshop, and NASA/WVU Software Research Lab, Fairmont, WV, Tech report # NASA-IVV-99-027, 1999.

[9] Menzies, T. and Sinsel, E., “Practical Large Scale What-if Queries: Case Studies with Software Risk Assessment”, Proc. 15th Int. Conf. on Automated Software Engineering, 2000, pp. 165-173, Available from http://tim.menzies.com/pdf/00ase.pdf.

[10] T.J. Menzies, R.F. Cohen, S. Waugh and S. Goss, “Applications of Abduction: Testing Very Long Qualitative Simulations”, IEEE Transactions of Data and Knowledge Engineering (to appear), Available from http://tim.menzies.com/pdf/97iedge.pdf, 2001

[11] J. Park, D. Port, B. Boehm. and H. In, “Supporting Distributed Collaborative Prioritization for WinWin Requirements Capture and Negotiations”, Proc. of the Int. 3rd World Multiconference on Systemics, Cybernetics and Informatics (SCI'99), Vol. 2, pp.578-584, IIIS, July 1999

[12] M.C. Paulk, B. Curtiss, M.B. Chrissis, C.V. Weber. “Capability Maturity Model for Software, Version 1.1”. Technical Report CMU/SEI-93-TR-024, SEI, Carnegie Mellon University, February 1993.

[13]F. Sisti and J. Sujoe, “Software Risk Evaluation Method Version 1.0”. Technical Report CMU/SEI-94-TR-019, SEI, Carnegie Mellon University, 1994.

[14] A. van Lamsweerde, “Requirements Engineering in the Year 00: A Research Perspective”, 22nd ICSE, 2000, Limerick, Ireland, ACM Press, pp. 5-19.