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

Trade-offs lie at the core of all early life cycle decision making in software development projects. For example, allocation of time, money and personnel among the competing development and assurance tasks; selection of which requirements to fulfill, and how much risk to accept.

Making wise decisions on these tradeoffs is hard to do, for the following reasons:

Best-practice knowledge on software development exists, but is dispersed throughout the software engineering literature. It is hard to find the knowledge relevant to the project at hand, and hard to tailor this knowledge with project specific information.

Different "stakeholders" in the development effort will have different opinions on potentially all aspects of development, for example, requirements and their prioritization, tolerances for risks, costs and benefits of various activities. Reaching agreement on a unified vision is far from trivial.

The sheer volume of information is overwhelming.

Early in the life cycle, much remains uncertain or unknown.

To support managers to make these early life cycle trade-off decisions, NASA is funding the development of an integrated tool suite:

- Ask Pete, a *software knowledge tool*, acts as a smart librarian to helps software engineers access a large library of established software engineering knowledge [4]. The lack of the equivalent capability would force managers must manually and laboriously explore the voluminous software engineering literature to find the knowledge relevant to their project.
- ARRT/DDP, an *exploration tool,* allows groups of stakeholders to refine and extend the knowledge from the librarian, and effectively explore the sum total of this knowledge [2]. The lack of the equivalent capability would leave managers to manually explore and record their requirements, the risks to their requirements, and the actions that can reduce those risks.
- TARZAN, an *options reduction tool,* 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. The lack of the

equivalent capability would leave managers and stakeholders overwhelmed by the sheer volume of "what-ifs" and unknowns within their project

- VCR/DCPT, a *detailed trade-off tool,* lets stakeholders resolve their conflicts of the critical issues [11]. The lack of the equivalent capability would inhibit stakeholders' and managers' understanding of the interactions and implications of the critical issues within their projects.

These tools already exist and function as isolated utilities, and their integration is partially complete (discussed in section 4). This article presents the strengths of each tool, and describes how each augments and extends the capabilities of the others.

2. A Visionary Hypothetical Scenario

In this section we describe our vision for how this integrated tool suite will support decision-making. The description takes the form of a hypothetical scenario, where we highlight the advantages we believe will stem from the integration we are pursuing. **Boldface** is used to indicate the contributions of our tools.

Imagine the year is 2002, and software development manager Kay has just been assigned lead a new project, the "Near Earth Dangerous Asteroid Detection and Destruction" project. Kay has been allocated a schedule of three years and budget of $15,000,000, and has been tasked to prepare a Software Development Plan.

Kay downloads the software tool suite from http://tkurtz.grc.nasa.gov/pete [4]. This 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 $77,072,000 to build and 89 months to complete (i.e., over cost and schedule). 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.

Even given the optimistic staffing assumptions, this is still too long. Kay tries something else. She asks the software knowledge tool to **postulate that the project is split**. After several iterations, she emerges with the following solution: separate development of the space platform, acquisition & control system, and tracking & firing system, and an integration team to combine them. The result - each subsystem 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.

Happy with the results, Kay then asks the software knowledge tool to **generate the initial Software Development Plan**. The software knowledge tool **outputs detailed templates of report documents and proposes a schedule when those documents should be delivered.**

Kay now has a good grasp on the overall project, and is ready to focus on areas of specific concern. For this, she needs to enlist engineers experienced in all the disciplines concerned. She identifies:

- A mission design specialist who has worked on near-earth orbit spacecraft.
- A Pentagon scientist who understands the atomic weapons to be used.
- A local NASA expert who specializes in software for real-time systems.
- One of the Software Quality Assurance staff who knows the overall range of things to watch out for.

Teleconferences are scheduled - one a day for the next three days.

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 data from the software knowledge tool into the exploration tool.** 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, using the exploration tool to record and structure their information.**

When the teleconference starts, the group reviews what inputs they have from the software knowledge tool. Working together, they **add on-the-fly detailed project requirements** into the categories that Kay and the SQA person had established. During the afternoon they **link the requirements to the risks (i.e., decide which risks impact which requirements, and by how much).** At all times, two rules guide their work:

- When stakeholders disagree on some point, that point is elaborated. Usually, on elaboration, it is discovered that the "disagreement" is really a confusion of two or more ideas - one of which had not been previously considered by the opposing party.

- When stakeholders truly do not know the details, a *range marker* is added. Such a range marker says that "we don't know, but we think that the possibilities lie within this range".

The tools come pre-loaded with risk mitigation information. This takes the form of links between mitigation activities (a union of software best-practices drawn from, e.g., CMM [12], and NASA center principles) and known risks (an SEI risk taxonomy [13]) inherent to large software development efforts. These links indicate which mitigations help reduce which risks, and by how much.

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 spare CPUs cycles on desktops.

Next morning, Kay presents the options reduction results to the group. Guided by these, they revisit 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 requirements. The team agrees to focus on these.

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..

By the end of the day, the team has emerged with what they hope will allow a feasible development effort. Kay activates **the options reduction tool on this revised dataset** before going home.

The overnight run of the options reduction tool on the amalgamated data looks promising, but still leaves uncomfortably small margins in both cost and risk. When convened, the group focuses on choice of risk mitigations. As before, **the options reduction tool's results point to those options that have the most leverage**. This leads them to scrutiny of "formal inspections" and "formal methods". Experts in both of these areas are available to work on the project. Given this knowledge, they **refine effectiveness values for some of the mitigation activities within the standard CMM categories**. After several such iterations, the net result is a plan that is not only within cost and schedule limits, but also shows how it addresses all the risks.

The final step is **automatic report generation. The software knowledge tool takes the results from the exploration tool**, and generates for Kay (among other things) the entire Software Development Plan she had been tasked to produce.

3. Benefits of the Collaboration

This section describes the benefits that stem from our collaboration. Subsection 3.1 discusses the unique benefit that each of our elements provides, while subsection 3.2 discusses the synergies gained from their combination.

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 Contribution of the Software Knowledge Tool, Ask Pete

The Ask Pete tool integrates cost and schedule estimation, software project tailoring and criteria used to determine the extent of independent verification and validation (IV&V) 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 preset questions (multiple choice and True/False) 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 results of these computations with built-in knowledge of NASA-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 has often meant inferior results, with slipshod guesswork replacing well-grounded estimation, and important factors overlooked by mistake or oversight.

3.1.2 Contribution of the Exploration Tool, ARRT/DDP

The ARRT/DDP 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 facilitator guides 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 Contribution of Knowledge & Knowledge Representation

In the context of this paper, the term knowledge representation refers to use 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, we have augmented this knowledge with expert judgments about the strength of influence of mitigating activities on specific risks. This is key to guiding managers to be able to choose 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 Contribution of the Options Reduction Tool, TARZAN.

The TARZAN 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. By resolving the key options, the other options are forced to follow.

TARZAN works via abduction and induction. Abduction, the logic of argument, 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 worlds, each of which represents a single consistent set of assumptions. The number of worlds may in principle be huge but, in practice, is surprisingly small [7] (theoretical explanation for why:

[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 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] allows stakeholders (co-located or otherwise) 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. Synergies of combination

Our collaboration comprises the five efforts described in section 3.1. This section explores the key additional synergies that arise from their combination.

3.2.1. 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 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.

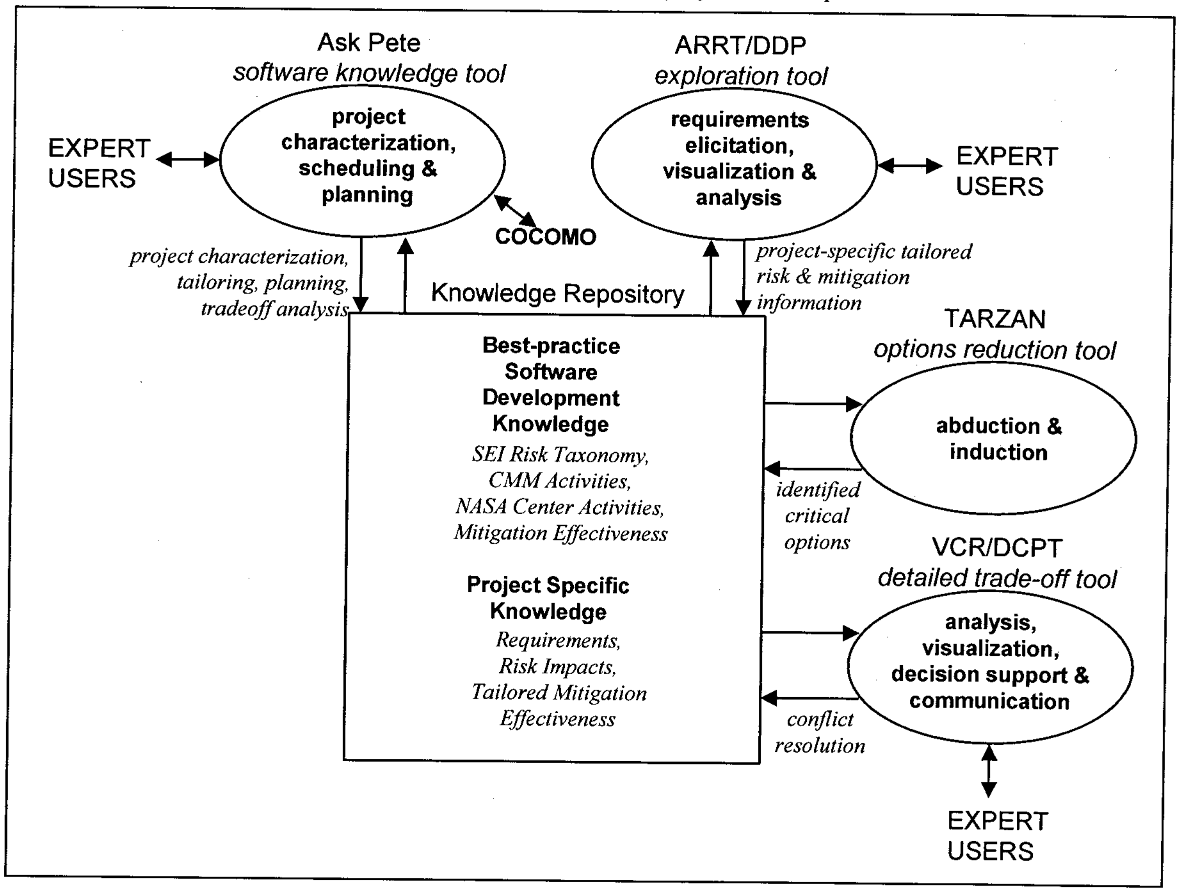
3.2.2. Knowledge & Representation + Exploration Tool. The knowledge culled from best-practice sources (CMM activities, SEI taxonomies of risks), which we have augmented appropriately (cross-linked to indicate which activities mitigate which risks, and by how much) pre-populates the ARRT/DDP tool. This has the obvious advantages of timesavings, 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.

3.2.3 Options Reduction Tool + Knowledge & Representation. The volume of data that we have to manipulate increases for larger projects - they have more requirements to juggle, more risks to address, etc. 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.

3.2.4 Detailed Trade-Off Tool + Knowledge & Representation.

Individual stakeholders bring their own perspective and expertise to project planning. The validity of these choices and decisions is increased when representation from the various classes of stakeholders is broadened, but gives rise to numerous differences of opinion(e.g., on requirements; relative value, costs of attainment, severity risks). VCR/DCPT uses statistical, stakeholder and group profile analysis to calculate the degree and nature of disagreement, and uses visualizations of this information that encourage insights for potential resolution. Ultimately, both the Ask Pete and ARRT/DDP tools assume that all input data input can be combined into the same one pool, within which stakeholder identity is not a factor. 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. Application of VCR/DCPT to the underlying knowledge base permits the resolution of these differences of opinion.

The options reduction tool contributes synergistically here too. 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. 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 further resolve, but which are amenable to scrutiny by human experts via VCR/DCPT.


Ask Pete
software knowledge tool
project characterization, scheduling & planning
EXPERT USERS
ARRT/DDP
exploration tool
requirements elicitation, visualization & analysis
EXPERT USERS
COCOMO
project characterization, tailoring, planning, tradeoff analysis
Knowledge Repository
project-specific tailored risk & mitigation information
TARZAN
options reduction tool
abduction & induction
Best-practice Software Development Knowledge
SEI Risk Taxonomy, CMM Activities, NASA Center Activities, Mitigation Effectiveness
Project Specific Knowledge
Requirements, Risk Impacts, Tailored Mitigation Effectiveness
identified critical options
VCR/DCPT
detailed trade-off tool
analysis, visualization, decision support & communication
conflict resolution
EXPERT USERS

4. Integration Status

This section discusses the status of our integration of the previously described tools and techniques.

Our Knowledge Tool, Ask Pete, and our Exploration Tool, ARRT/DDP, (both of which can be operated as stand-alone applications) are the nuclei around which the collaborative components are coalescing. We have completed first major step, to make these two operate together [5]. Very briefly: information is transferred back and forth between the two tools by means of shared database using an agreed upon data schema. Ask Pete yields an initial suggestion of risk mitigation activities, which ARRT/DDP refines. We are in the process of transferring more of the information gathered from the user's answers to Ask Pete's questions, for example, to set the severity of various risks.

Also completed [2] 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). Pilot studies are 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]. This was done independent of this collaboration, but gives a promising indication of both viability and utility. Tight integration with the Ask Pete and ARRT/DDP components is work in progress. To date we have performed initial experiments on extracting the relevant information from ARRT/DDP in a form suitable for input into TARZAN. Similar explorations into information exchange between ARRT/DDP and VCR/DCPT are also underway. These have permitted 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, less-detailed requirements, that span both product and development process (e.g., schedule, budget). 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.

The main message of this paper is a vision of enhanced early lifecycle software development planning, and the substantial progress we have made towards its attainment. Our vision makes use of best-practice knowledge and tailoring to the project at hand, using a set of sophisticated tools to support this reasoning. We have described how each of our tools offers key benefits towards attainment of this vision. Furthermore, their combination yields synergistic benefits that in concert will allow users to make optimal use of large amounts of knowledge in their key decision making activities.

6. Acknowledgements

The research described in this paper was carried out at Science Applications International Corporation (SAIC), the Jet Propulsion Laboratory, California Institute of Technology, the 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.

[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.