Using Stakeholder Preferences to Make Better Architecture Decisions

Neil Ernst, John Klein
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA, USA
{nernst, jklein}@sei.cmu.edu

George Mathew, Tim Menzies
Computer Science
NC State University
Raleigh, NC, USA
{george.meg91,tim.menzies}@gmail.com

***Abstract*—A roadmap to modernize the architecture of an existing system must satisfy many strongly-positioned stakeholders and satisfy the constraints of continuing business operations as the plan is implemented. Our previous work reported on a method to engage with stakeholders to model architecture options for a modernization roadmap. These models have proven to be too large to analyze all options manually: *Ad hoc* approaches must be employed to prune the space of possible solutions, which risks dropping optimal solutions. We report here on a method that efficiently collects stakeholder preferences about architecture options and uses an automated search-based optimization approach over the full solution space to identify the most important architecture decisions, i.e., the decisions that have the most influence on stakeholder satisfaction.**

Keywords—goal model, search-based software engineering, architecture decision support

I. Introduction

IT system modernization projects present special challenges for architecture decision making. Labeling a project as "modernization" implies that there is an existing system that provides sufficient value so that it is worth investing to maintain or improve that capability. While all systems have stakeholders who should be respected, in a modernization project the stakeholders use the existing system to deliver essential services to the enterprise, giving them a bigger voice in architecture decisions.

In this context, the architect must develop a roadmap defining a sequence of architecture changes [5]. Stakeholder cooperation is needed to successfully execute the roadmap; e.g., they provide necessary domain knowledge, provide funding for operation of the new system, and will perform acceptance testing of the new system. The architecture decision-making process must include this large and diverse group, who may have conflicting preferences from differing goals and evaluation criteria. Modernization changes technology, and also processes and organizational structure, and our experience is that decisions can have broad impact and trigger strong stakeholder reactions.

This paper describes an architecture decision-making process which addresses the specific challenges of modernization projects, and can be applied in other less challenging contexts. The process is scalable and transparent, and allows direct stakeholder participation in either collaborative or asynchronous modes. The process has four steps:

1. Capture decision alternatives identified by stakeholders in a goal model [10], along with cost and benefit values;
2. Collect stakeholder preferences about the options using the analytic hierarchy process (AHP);
3. Apply a search-based optimizer that performs heuristic sampling of the entire decision space represented by the softgoal model;
4. Rank solutions on cost, benefit, and stakeholder satisfaction, and identify the architecture decisions that have the highest impact.

The contribution of this paper is a stakeholder preference collection step in this process, and use of stakeholder preferences in architecture decision-making.

II. Eliciting Decision Spaces with Goal Models

A goal model expresses the relations among *softgoals*, which are subjective; *hard goals*, which can be objectively satisfied; and *tasks or services*, which reflect activities necessary to satisfy a goal. These elements are refined with and/or relations [10]. Fig. 1 shows a typical goal model used to specify modernization roadmap architecture options.

Reasoning on a softgoal model labels each hard goal and task/service element as *do* or *do not*, in accordance with the semantic constraints of the inter-element relations. We call such a labeling a *solution candidate*. A solution candidate embodies a set of semantically acceptable architecture decisions. A solution candidate can be evaluated by summing the cost, risk, stakeholder satisfaction, or other metrics from all elements labeled *do*, and by evaluating the benefit produced by the root soft goals satisficed by the labeling and the root hard goals satisfied by the labeling [7].

We used a goal model to capture the decision space for our modernization planning in a process we described in [5]. The challenge is to obtain, from a set of technical stakeholders, their assessment of costs and benefits for a range of architectural options. The input to this process is a set of architecture risks identified in some architecture analysis; in our case, using the Architecture Tradeoff Analysis Method (ATAM) [3]. The workshop facilitators prioritize the risks, and then create scenarios that 'test' the risks. In a 2-day workshop, technical stakeholders (e.g., chief architects, development managers, team leads) are asked to suggest technical options–decisions–for mitigating the risks. For example, if a *risk* is that data access relies on a direct database connection, leading to modularity

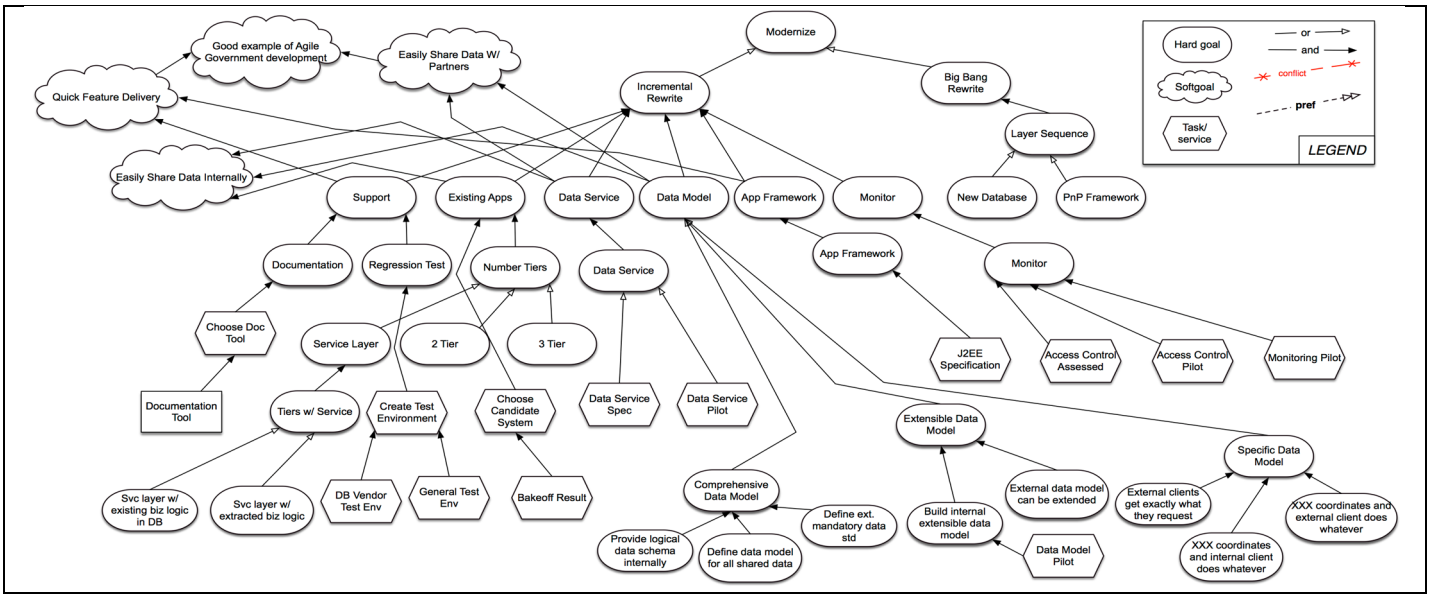

Fig. 1. Modernization roadmap options model example (from [5])

violations, a potential *option* is to introduce a service layer as a mediator.

The collected set of options is analyzed by the facilitator team to identify dependencies and collapse similar options, and the result is then presented back to the stakeholders in another session. A lightweight cost/benefit analysis elaborates the pros and cons (technical benefits to selecting the option) and then asks for ordinal, relative costs for implementing that option (e.g., High/Medium/Low relative to other options). The result of this is a set of options that the facilitators can structure into the goal model shown in Fig. 1. The modeling step is currently based on the skill and judgment of the facilitators at identifying similar options (decision points) and dependencies. The next phase of the approach is a member-checking exercise in which the structure model is evaluated with the stakeholders a final time.

III. Prioritization with the AHP

The AHP is a method for making a decision, described in detail by Saaty [13]. Briefly, the AHP represents the decision as a hierarchy of criteria and options. The decision maker first makes pairwise comparisons among all of the criteria, expressing his judgment about the relative importance between each pair of criteria. Then, for each criterion, the he judges the relative ability between a pair of options to satisfy the criterion. These are subjective, personal, and relative judgments about what will satisfy the stakeholder.

The AHP assigns numeric values to the judgments and then weights the option judgments by the criteria judgments to rank the options. We then combine the individual stakeholder judgments in our search-based optimizer.

We applied the AHP to collect preferences about the decisions in model shown in Fig. 1. That model contains six decisions–there are six goals refined using exclusive *or* relations. AHP treats each decision independently, with its own criteria. However, this set of decisions is coherent (i.e. decisions about the same system, made at the same time), and so we used a single set of criteria for all, since the architecture qualities represented by the criteria are system-wide properties and their relative importance should not change from decision to decision. This system was developed by a government agency, so we selected the six most relevant criteria from a set used to assess government IT projects[1]: (1) Schedule; (2) Life-cycle costs; (3) Dependencies and interoperability; (4) Risk of failure; (5) Risk of not achieving business goals; and (6) Security. The resulting AHP hierarchy for one decision—"What type of data model should we develop?"—is shown in Fig. 2.

Stakeholder preferences were collected using a spreadsheet-based form. In our pilot experiment, this took stakeholders less than 30 minutes to make the 75 pairwise judgments required by this model. In cases where the number of pairwise comparisons becomes too large, there are approaches that sample comparisons across a group of deciders, to reduce the workload on each individual to an acceptable level [6].

We used an online AHP data analysis package (https://github.com/gluc/ahp). The analysis produced a ranking

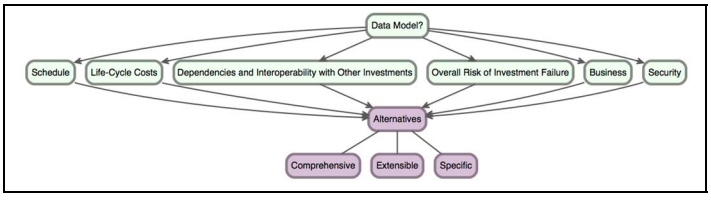

Fig. 2. AHP hierarchy example

[1] http://csrc.nist.gov/roi/wksps0603-notes/Risk-Handout.pdf

example table for one stakeholder for the decision hierarchy from Fig. 2. The first column shows how this stakeholder weighted the criteria, with the pairwise judgments transformed into a weighted ranking: Business risk was the most important (35.3%), followed closely by Schedule risk (30.6%). The others were all ranked relatively low. The first row of the table shows the final result of the AHP calculations, which weight the stakeholder's pairwise judgments among the options by the stakeholder's criteria weights. In this case, the stakeholder prefers the "Specific" option for a Data Model, with a weight of 63.7%. The "Extensible" option has a weight of only 26.6%, while the "Comprehensive" option has a weight of only 9.7%.

	Weight	Specific	Extensible	Comprehensive	Consistency
Data Model?	100.0%	63.7%	26.6%	9.7%	! 37.0%
Business	35.3%	26.4%	4.7%	4.2%	1.1%
Schedule	30.6%	19.8%	8.5%	2.2%	5.6%
Overall Risk of Investment Failure	12.6%	5.7%	5.7%	1.1%	0.0%
Dependencies and Interoperability with Other Investments	9.9%	4.6%	4.6%	0.7%	0.0%
Life-Cycle Costs	8.8%	6.2%	2.1%	0.5%	! 25.4%
Security	2.8%	0.9%	0.9%	0.9%	0.0%

Fig. 3. AHP Example

The rightmost column shows the measure of the transitive consistency of the pairwise judgments: If A is judged as better than B, and B is judged as better than C, then we would expect that A would also be judged better than C. Saaty discusses why the judgments should be reviewed when the consistency metric is greater than 10% [13]. (Although the AHP literature labels this metric "consistency", note that the value increases as the judgments become more inconsistent.) The data analysis package that we used calculates the aggregated consistency metric (37.0%), however we must look to the metrics for each criteria to make an assessment. For the "Life-Cycle Costs" criterion, the metric was significantly above the threshold, at 25.4%. We accepted this inconsistency, since the stakeholder ranked this criterion as unimportant and the inconsistency was likely due to a lack of any strong preferences about the options. However, a similar metric for one a highly-ranked criteria would warrant review. Possible causes for inconsistency by a single stakeholder include data entry error on the form, misunderstanding of the alternatives, or even intentional misrepresentation to undermine the project. If many stakeholders show inconsistency on a particular judgment, this might indicate that there is little real differentiation among the alternatives.

IV. A SHORTER METHOD

The output of the modeling process, and the subsequent AHP prioritization, serve as inputs to our decision ranking tool, SHORT [11]. Fig. 4 shows the methodology. Inputs are the softgoal model from Section II, including cost/benefit values from the stakeholders, and the preferences elicited from AHP. A search-based algorithm evaluates potential combinations of decisions according to a) decision cost; b) decision benefit and preference satisfaction; c) softgoal satisfaction. This results in a set of optimal (non-dominated) solutions.

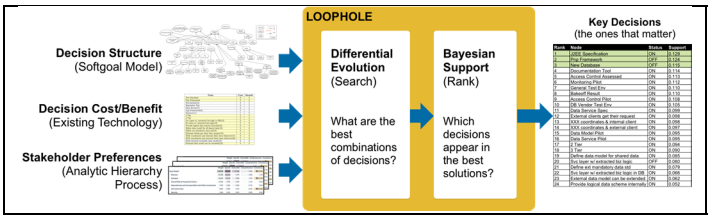

Fig. 4. End-to-end method

This heuristic search-based approach scales better than complete approaches, particularly when there are cross-tree relationships in the goal model, as discussed in [11].

A Bayesian ranker produces a list of key decisions (Fig. 5). Conceptually, the ranking is based on the probability that a decision appears in a particular solution, given that the solution is optimal. Details are provided in [11]. These are the decisions which account for the majority of the variance in the decision value space, i.e., making the top 3 decisions below accounts for the majority of the cost/benefit/satisfaction value. Note that each decision includes the decision state (*do* or *do not*, as discussed above in Section II).

V. DYNAMIC ASPECTS OF DECISIONS

One of the characteristics of the work we have described in this paper is that it focuses on a relatively static picture of the world. Decisions are abstract representations of an ideal software system. In more recent work, we have been using the notion of ongoing, dynamic analysis of software-in-use to inform our assessment of the design options. This is particularly important in the emerging 'intelligent, connected' software systems [15]. There, the challenge will be to understand how the decisions an organization makes affect, and are affected by, external components. Our approach investigates dynamic configuration mechanisms such as dependency injection/inversion of control to understand how these impact architectural decisions. Consider the modules in Fig. 4. We may have created links between modules A and B, and C and A, and be able to understand these with the SHORT approach. However, run-time analysis informs us of the dynamic dependency between B and C, which introduces a cycle, possibly reducing modularity. We are working to bring this type of run-time analysis into our decision-making approach.

Rank	Node	Label	Support
1	J2EE Specification	ON	0.129
2	Pnp Framework	OFF	0.124
3	New Database	OFF	0.115
4	Documentation Tool	ON	0.114
5	Access Control Assessed	ON	0.113
6	Monitoring Pilot	ON	0.112
7	General Test Env	ON	0.110
8	Bakeoff Result	ON	0.110
9	Access Control Pilot	ON	0.108
10	DB Vendor Test Env	ON	0.105
11	Data Service Spec	ON	0.099
12	External clients get their request	ON	0.098
13	XXX coordinates & internal client	ON	0.098
14	XXX coordinates & external client	ON	0.097
15	Data Model Pilot	ON	0.095
16	Data Service Pilot	ON	0.095
17	2 Tier	ON	0.094
18	3 Tier	ON	0.090
19	Define data model for shared data	ON	0.085
20	Svc layer w/ extracted biz logic	OFF	0.080
21	Define ext mandatory data std	ON	0.079
22	Svc layer w/ extracted biz logic in DB	ON	0.066
23	External data model can be extended	ON	0.062
24	Provide logical data scheme internally	ON	0.052

Fig. 5. Ranked decisions

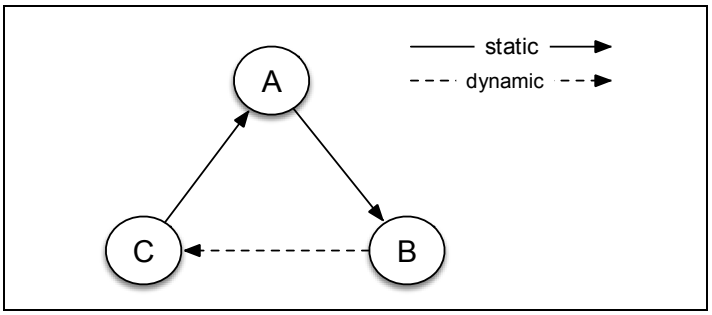

Fig. 6. Static, compile-time vs dynamic, run-time dependencies

VI. Related Work

Tofan et al. reported on a tool to increase consensus in group architecture decision making [14]. Key differences compared to that method are our support of asynchronous judgments (no meeting required) and allowing preferences to be reported anonymously or attributed, to match the culture and policy of the organization.

Research in requirements prioritization covers some of the same territory. Karlsson et al. [8] analyzed six prioritization approaches and concluded AHP was most useful. Similarly, Achimugu et al. [1] also found AHP most used, but suffering from scalability challenges. Our notion of "key" decisions can alleviate this. Finally, like us, Pitangueira et al. [12] analyzed search-based approaches to prioritization, including work on the Next Release Problem [2]. In particular, they noted there was relatively little work applying search-based techniques to industry challenges, which we do here.

VII. Conclusions and Future Work

We described a process for making software design decisions using goal models and search-based decision ranking. Our approach leverages light-weight stakeholder input to create a simple model of the potential solutions, and uses AHP to determine stakeholder preferences. The SHORT process then finds the key decisions, which we use to greatly simplify the design decisions that need consideration by the stakeholders in the revised AHP process. As future work, we are expanding on our work to include the dynamic decisions we identify from run-time design analysis, and planning systematic validation of the end-to-end method.

Acknowledgment

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center. [Distribution Statement A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution. ATAM® is registered in the U.S. Patent and Trademark Office by Carnegie Mellon University. DM-0004507

References

[1] P. Achimugu, A. Selamat, R. Ibrahim, et al., "A systematic literature review of software requirements prioritization research," *Information and Software Technology,* vol. 56, no. 6, pp. 568 - 585, 2014, doi: 10.1016/j.infsof.2014.02.001.

[2] A.J. Bagnall, V.J. Rayward-Smith, I.M. Whittley, "The next release problem", *Inform. Softw. Technol.,* 43 (14) (2001), pp. 883–890

[3] P. Clements, R. Kazman, and M. Klein, *Evaluating Software Architectures: Methods and Case Studies.* Addison-Wesley, 2002.

[4] N. A. Ernst, A. Borgida, J. Mylopoulos, et al., "Agile Requirements Evolution via Paraconsistent Reasoning," in *Proc. 24th Int. Conf. on Advanced Inf. Systems Eng. (CAiSE 2012)*, Gdansk, Poland, 2012, pp. 382-397. doi: 10.1007/978-3-642-31095-9_25.

[5] N. A. Ernst, M. Popeck, F. Bachmann, et al., "Creating Software Modernization Roadmaps: The Architecture Options Workshop," in *Proc. 13th Working IEEE/IFIP Conf. on Software Architecture (WICSA/CompArch 2016)*, Venice, Italy, 2016. doi: 10.1109/WICSA.2016.39.

[6] P. T. Harker, "Incomplete pairwise comparisons in the analytic hierarchy process," *Mathematical Modelling,* vol. 9, no. 11, pp. 837 - 848, 1987, doi: 10.1016/0270-0255(87)90503-3

[7] I. J. Jureta, A. Borgida, N. A. Ernst, J. Mylopoulos, "Techne: Towards a New Generation of Requirements Modeling Languages with Goals, Preferences, and Inconsistency Handling," in *IEEE Int. Requirements Eng. Conf. (RE)*, Sydney, NSW, Australia, 2010, pp. 115-124. doi: 10.1109/RE.2010.24.

[8] J. Karlsson, C. Wohlin, and B. Regnell, "An evaluation of methods for prioritizing software requirements," *Information and Software Technology,* vol. 39, no. 14, pp. 939 - 947, 1998, doi: 10.1016/S0950-5849(97)00053-0.

[9] M. Kassab and N. Kilicay-Ergin, "Applying analytical hierarchy process to system quality requirements prioritization," *Innovations in Systems and Software Engineering,* vol. 11, no. 4, pp. 303-312, December 2015, doi: 10.1007/s11334-015-0260-8.

[10] A. van Lamsweerde. "Goal-oriented requirements engineering: a guided tour". *International Conference on Requirements Engineering*, Toronto, ON, pages 249–263, 2001.

[11] G. Mathew, T. Menzies, N. A. Ernst, et al., "SHORTER Reasoning About Larger Requirements Models," *(In Review)*, 2017, https://arxiv.org/abs/1702.05568

[12] A. M. Pitangueira, R. S. P. Maciel, and M. Barros, "Software requirements selection and prioritization using SBSE approaches: A systematic review and mapping of the literature," *Journal of Systems and Software,* vol. 103, pp. 267 - 280, 2015, doi: 10.1016/j.jss.2014.09.038.

[13] R. W. Saaty, "The analytic hierarchy process---what it is and how it is used," *Mathematical Modelling,* vol. 9, no. 3--5, pp. 161 - 176, 1987, doi: 10.1016/0270-0255(87)90473-8.

[14] D. Tofan, M. Galster, I. Lytra, et al., "Empirical evaluation of a process to increase consensus in group architectural decision making," *Information and Software Technology,* vol. 72, pp. 31-47, 2016, doi: 10.1016/j.infsof.2015.12.002.

[15] E. Woods. "Software architecture in a changing world". *IEEE Software,* 33(6):94–97, Nov 2016.