

Title: Understanding Organizational Risk in Architectural Design

Submitted by Len Bass and Mark Klein
in Collaboration with : Jim Herbsleb (School of Computer Science, CMU), and Matt Bass
(Siemens Corporate Research)

Date: July 22, 2005

Purpose and goals of the proposed research

It is well-known that architectures (both software and system) become deeply “embedded” in the organizations that design and build software intensive systems (e.g., [2]). Architectures determine key characteristics of organizations, such as work assignments, and have implications on communication patterns, habitual ways that people select and filter information, and organizational problem-solving strategies. Changing the architecture, even in seemingly simple ways, can cause serious mismatches between the architecture and the organization, leading in some cases to complete failure of the firm [3]. Organizations are notoriously difficult to change, and we do not yet have any tools for understanding the kinds of changes we are imposing on them when we perform architectural design. We don’t know how to assess the risks, nor do we know how to address the risks once they are identified.

Suppose, for example, that the designer of an architecture who was considering two alternative designs knew that design 1 requires the developers of a particular component to access expertise from a separate organization requiring extensive long distance communication and design 2 requires the same developers to access expertise from a co-located site. Everything else being equal, design 2 is preferable since it will reduce organizational risk by lowering the communication overhead required to solve any problem that arises. Organization structure, existing communication patterns, location of people and resources, shared work history, and potentially other factors not yet identified, will influence the risk, time required, component quality, and development efficiency of the design and construction of software components. Organizational factors should be considered during the making of design decisions, just as performance or reliability considerations are considered.

It is also well-known that a system’s software architecture is a central artifact in the development of software intensive systems. It is a primary determinant of the system’s quality attributes. It is the bridge between business goals served by a system and its implementation. The SEI has invested considerable effort in understanding the principles of design and analysis of software architecture. However, people must effectively collaborate to realize the benefits of software architecture. Conversely organizational inhibitors to collaboration can be inimical to realizing the potential benefits of software architecture. Therefore understanding the relationship between types of organizations and types of architecture is vital to realizing the benefits of software architecture.

The purpose of this IR&D is to investigate the relation between organizational characteristics and software architecture. If two people are working together on the same component, they must

communicate about all aspects of the component. If they are working on different components, they must communicate about the ways in which their respective components interact. The two situations may require different types and bandwidths of communication. Depending on the fashion in which two components interact, there may be a requirement for high or low bandwidth communication between the people. The communication bandwidth among people is affected by many different elements such as co-location, organizational structure, technological support, and knowledge of where different types of expertise reside in an organization. In the modern world, most large systems are constructed across multiple sites, usually involving multiple distinct organizations. Understanding the organizational characteristics that are essential to effectively building a system with a particular architecture will provide fundamental knowledge that can be exploited to more effectively manage and design large software systems.

If the relation between organizational characteristics and software architecture were well understood, it would have application in architecture evaluation, architecture design, and in assessing the readiness of an organization for architecture centric development. During an architecture evaluation, discovering a mismatch between architectural decisions and organizational structure is discovering a risk to the development effort. During an architecture design, discovering a mismatch between proposed architectural decisions and proposed organizational structure allows for the correction of the mismatch or at least for allowing for it in the project management. There might be a tradeoff, for example, between optimizing the performance of a system and optimizing it for the organizational structure of the developing organizations. One question that the SAT initiative is frequently asked is “how ready is organization X for architecture based development?” Being able to analyze the structure of organization X with respect to common architecture usages is one aspect of being able to answer this question. A fundamental understanding of the relation between organizational characteristics and software architectures could allow us to go beyond assessing readiness in general, and determine the range of architectures an existing organization could build.

A complicating element of the understanding of organizational characteristics is that they evolve over time, and desirable characteristics may change, depending on the current stage of the project development. During the initial stages of a project, one might see, for example, particular patterns of communication among the groups working on the infrastructure. During later stages one might see communication between application developers and infrastructure developers.

The study we propose with this IR&D extends and applies work that Professor James Herbsleb in CMU’s School of Computer Science (a collaborator in the proposal) has undertaken during the last several years. It combines his work on collaboration and coordination with existing SEI work on the dependencies among different components of a software architecture. Professor Herbsleb has developed methods for collecting relevant organizational information and for the last year has been applying and refining those methods in conjunction with Siemens Corporate Research (the other collaborator in the proposal). Especially promising for the proposed research is an adaptation of social network methods, which use communication patterns and related information to construct graph representations of the ways in which information flows through organizations [7]. A questionnaire collecting such information was developed and pilot tested on a large, multi-site project at a single point in time. The SEI has categorized different types of architectural dependencies and studied mechanisms for breaking the propagation of changes

occurring as a result of these dependencies [1, Chapter 5]. These two lines of work provide powerful tools with which to begin the proposed research.

We propose in this study to combine and extend these two pieces of work to examine several real, diverse projects over a period of time in order to better understand the relation between the organizational characteristics and the software architecture decisions being made. If this work is successful it will strongly impact the future of the SAT initiative in opening up an organizational dimension to our work that does not currently exist.

The team in this proposal is ideally suited for the work. Jim Herbsleb has created this line of inquiry. Mark Klein and Len Bass are world experts in software architecture design and analysis. Len Bass is partially responsible for the existing SEI work in dependency. The other collaborator (Matt Bass) has been working with Professor Herbsleb on the problems of distributed development. He has identified a development effort within Siemens that Professor Herbsleb has used for his initial investigations and will identify development efforts for us to use in the IR&D. The SEI has leadership in software architecture and successful completion of this study will enable us to extend that leadership into a new and emerging area of great interest to both DoD (with multiple contractors involved in every project) and industry (with the distributed development of projects an economic necessity).

DoD challenge problems addressed:

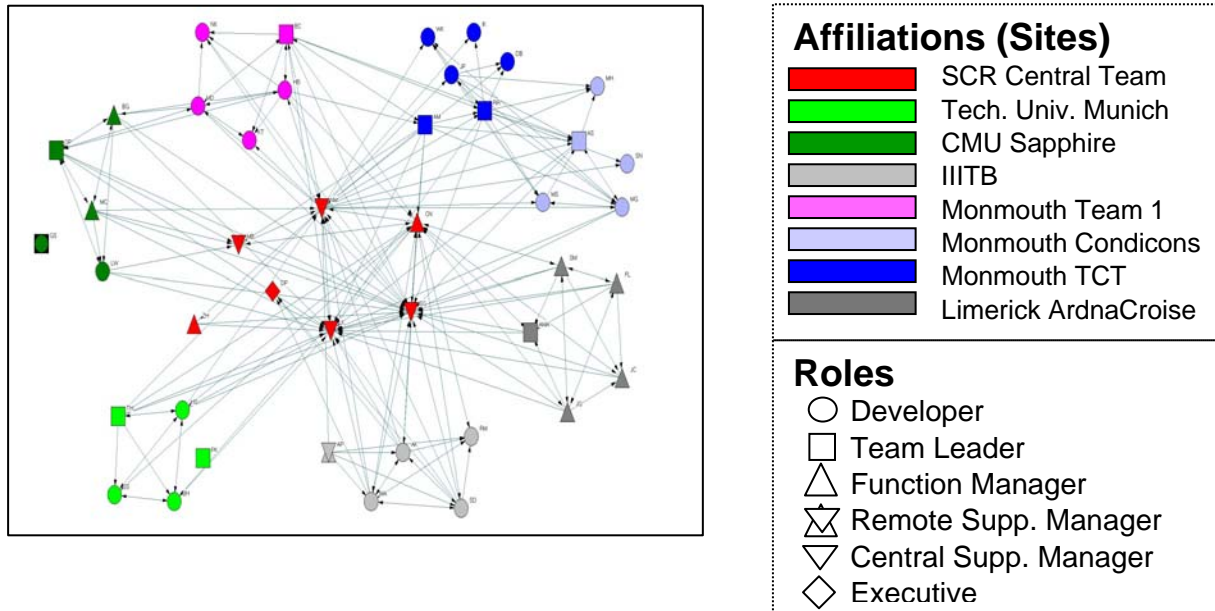
- Thrust Area: Software Technology R&D-. Develop improved, enhanced or new processes, principles, methods, and tools for determining expected properties of software systems before they are built and for confirming their as-built properties. Determine how to incentivize use of these technologies by DoD acquisition programs and their contractors.
- Thrust Area: Software Metrics for Acquisition Management - Develop a unified suite of value-added strategic metrics for assessing incremental benefits of products, artifacts, and processes spanning a system's life.

Background

When a development project is distributed across multiple sites a number of communication and coordination problems are multiplied from single site development (see, e.g., [4-6]). Some of the problems are:

- Lack of unplanned (e.g. hallway) contact
- Lack of knowledge of whom to contact about what
- The difficulty of initiating contact
- Ability to communicate effectively
- Lack of trust or willingness to communicate openly

To assess the types of communication that exist within a particular project, Professor Herbsleb asked members of the development team for an experimental Siemens development¹ to fill out a short questionnaire and give information about who they communicated with and the topic about which they communicated with respect to several different factors mentioned above. Figure 1 is a sample of the data collected that shows communication patterns with respect to technical information.

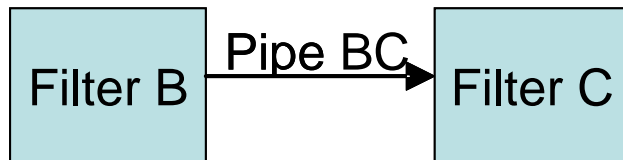


The graph visualization is interesting in itself, and can provide relevant information at a glance. One can see here, for example, that some of the teams had contact only with the central team, while others had considerable contact with each other. These patterns fairly accurately reflect technical dependencies, moderated by such things as geographic distance and the existence of collaboration technology, which limit the ability and influence the cost of communication. They only reflect one data point, however. Our research will enable an understanding of how these patterns change over time, among other things.

While the visualizations are useful, the real power of social network theory comes from the wide variety of measures one can compute for such graphs, including clustering coefficient (the tendency to form highly interlinked clusters of nodes only loosely linked to other clusters), the centralization of the graph, average distance between nodes, and many, many more (see, e.g., [8]). These measures allow one to precisely measure a variety of subtle characteristics of such networks, and have been shown to predict the effectiveness and efficiency of organizations in powerful ways. We would expect that organizations that can effectively build systems in one architectural style to look very different from organizations designed to build systems in a different architectural style.

¹ Siemens has set up a test bed for large multi-site projects which involves teams of graduate students at five universities (including CMU) working on a large, year-long project. The questionnaire was piloted in this environment, and the results were enthusiastically received by Siemens. Preparations are under way to administer a version of this questionnaire to several large Siemens development projects.

Consider the following hypothetical example of a software architecture. Figure 2 is a sample of two components that interact via filter B sending data to filter C.



Filter C depends on the operation of Filter B in both a syntactic and semantic sense, so they have a syntactic data dependency and a semantic data dependency, as well as a sequential dependency. What we would like to understand is *what do we need to know about this dependency in order to predict the sorts of communication and coordination that will be required for two teams to build these filters efficiently and effectively?* Must they be co-located? Must they have a history of working together? Will they need sophisticated collaboration technology? Will they need daily synch-up technical meetings? We would like to be able to look at an architectural description and answer such questions.

The example above assumes the components have been partially designed. At least a decision has been made that these two components are going to be constructed as filters with a pipe between them. But earlier in the life cycle, the developers of Filter B had only a work package and the developers of Filter C had another work package. How was the decision reached that they would both be developed as filters? Was this a decision of the individual teams or was it a decision made by a global architecture team? In either case, we would expect some pattern of communication that corresponds to the making and promulgation of that decision.

The data gathered about the communication patterns in the early stages of the life cycle will necessarily be different from the data gathered about the communication patterns in the later stages of the life cycle. In both cases, however, we hypothesize

- there are definite, measurable organizational patterns,
- these patterns can be understood by analyzing graph structures representing communication, collaboration, and awareness in the organization, and
- different software architectures will require different organizational patterns in order to effectively coordinate building of the software.

The work being proposed is fundamentally empirical in nature. We propose to collect communication patterns from several different types of development, identify the stages in the life cycle for which the data is being gathered, and collect information about the types of architectural decisions that are being considered. We then propose to analyze the data that is gathered and generate specific hypotheses about the relations between the organizational patterns, the software architecture, and any problems that have arisen. We will test these hypotheses on data sets that were used to generate the hypotheses..

Overall Approach

The overall approach is to collect data from several projects over time. We will collect organizational data of several kinds that have been shown to be relevant to understanding organizational communication patterns and problem-solving ability. Our goal is to develop data collection techniques and metrics that will allow us to predict its ability to develop products with various kinds of architectures. We will begin with a social network questionnaire that has been piloted in a software context, and gathers information about communication patterns, awareness of expertise, criticality of interactions, types of information sought, communication media used, as well as the developers' judgments about how well the project is going, and the adequacy of communication. We will administer this questionnaire to the entire project at about two week periods, which is feasible since pilot studies show it takes less than 10 minutes to complete. These frequent observations will allow us to observe how patterns evolve, and how they related to the particular work being done in each period. We will supplement the survey with a limited number of in-depth interviews to make sure we understand the full context of the project and the patterns we are observing.

We will also collect data about architectural dependencies as the architecture evolves. We believe it is the pattern of dependencies that will largely determine the kinds of coordination needed in order to successfully build the product. We will start with the set of dependencies identified in [1], and generate a monthly snapshot of all the dependencies among work packages. We will also keep track of which work package is assigned to which person, as well as the team structure and geographic distribution of people. Each snapshot will be used to construct a graph of the dependencies.

Finally, we will experiment with various ways of computing the fit between the organizational description and the architectural description. We will look for places where the development is proceeding well, and where it is experiencing coordination problems, and we will look for ways of understanding these differences. We will make use of the many techniques for comparing graphs in order to see which kinds of relationships between organizational and architectural graphs are most informative about coordination problems in projects. We will try to identify mismatches that precede coordination problems to see if they are generally predictive of looming issues.

Collaborators

There are two different sets of collaborators involved in this proposal – CMU's School of Computer Science as represented by Jim Herbsleb and Siemens Corporate Research as represented by Matt Bass. SCS and Siemens have been working together for a year on the problem of distributed development and some results have been reported at the International Conference on Software Engineering [6]. This IR&D proposal will join the elements of architecture and architectural analysis to the organizational studies that have been carried out thus far.

Funding for the time of Jim Herbsleb and a portion of a post doctoral researcher will be provided from this IR&D proposal.

Funding for the time of Matt Bass and the Siemens personnel who will participate in the studies will be provided by Siemens.

Success Criteria

1. Ability to identify mismatches associated with identified coordination problems.
2. Ability to find patterns that predict coordination problems.
3. Conduct a post mortem for the systems monitored. To what extent were there problems reported that can be traced to a mismatch between organizational structure and architecture and to what extent are the problems in organizational structure mirrored in the communication patterns.

Cost and Schedule

Cost:

Len Bass – 33%

Mark Klein – 15%

Jim Hersleb & 30% post-doc \$70K

Travel to perform interviews and to present results at conferences \$15K

Schedule:

Oct 1 – start

Dec 1 – systems to be monitored determined, type of monitoring determined, questionnaires and interview questions finalized.

April 1 – initial data gathering completed

June 1 – preliminary analysis completed and hypotheses created

Sept 30 – hypothesis tested with new data, follow-up interviews

Sept 30 – final report written

- [5] Herbsleb, J.D. and Mockus, A., An Empirical Study of Speed and Communication in Globally-Distributed Software Development. *IEEE Transactions on Software Engineering*, 29, 3 (2003), p. 1-14.
- [6] Herbsleb, J.D., Paulish, D.J., and Bass, M. Global Software Development at Siemens: Experience from Nine Projects. in *International Conference on Software Engineering*. 2005. St. Louis, MO.
- [7] Scott, J.P., *Social Network Analysis: A Handbook*. 2000, Thousand Oaks, CA: Sage Publications.
- [8] Wasserman, S. and Faust, K., *Social Network Analysis: Methods and Applications*. 1994, Cambridge, UK: Cambridge University Press.