SUPPORTING COORDINATION OF INTERDEPENDENT WORK:

A CONSTRAINT NETWORK REPRESENTATION OF DEPENDENCIES IN DESIGN AND ENGINEERING TEAMS

KEYWORDS: COORDINATION, INTERDEPENDENCE, CONSTRAINTS, ENGINEERING, DESIGN

ABSTRACT

Moving beyond the individual and designing technology that effectively supports the work of multiple people is a central goal in the field of HCI. This is a particularly challenging goal given the emergence of new organizational forms that defy traditional theories of groups and organizations. Variously called "postbureaucratic organizations," "adhocracies," "network organizations," or "heterarchies," these new forms of organizing are based on flows of information, experience rapid change, cross organizational boundaries, emphasize horizontal collaborations, and require adaptive capacity. To support collaboration in these new kinds of organizations, we need a better understanding of interdependent work and how interdependencies can be successfully managed. Current theories of coordination do not provide enough detail about the underlying connections between work tasks, and how these dependencies influence collaboration. Neither do they give sufficient insight into how to coordinate work in highly volatile environments typical of new organizational forms.

In this paper, we propose a theory of coordination that represents work tasks as a network of linked actions and constraints. The properties of this *constraint network* determine the impact that the actions of one worker completing a task can have on the tasks of other workers. These impacts must be accommodated appropriately so that all tasks can be completed successfully. The need for this specific accommodation, as described by the constraint network, is what we call the "coordination requirements" for the work. Coordination activities, in our theory, are actions that workers perform specifically to manage, address, and satisfy the coordination requirements. We use observational data from an engineering project to illustrate several properties of constraint networks that we have observed to impact the effectiveness of coordination activities. We discuss how the concept of constraint networks is general enough to represent many kinds of work, extending our understanding of coordination and advancing the field of HCI.

INTRODUCTION

System design in the fields of Information Systems and HCI is moving beyond the individual, increasingly considering "aspects of interaction that extend beyond a single user", and "large and small groups of people's interaction with technology and with each other through technology".[1] Effectively supporting collaboration – particularly over distance – continues to be a central challenge in designing systems for groups and organizations (see, e.g., Olson, Malone, & Smith, 2001). We argue that in order to advance the field of HCI beyond the individual, and more fully integrate it with the field of Information Systems to achieve the goal of supporting group and team-based collaboration with technology, we need a better understanding of interdependent work. Current theories are increasingly inadequate for understanding and addressing the most difficult issues in supporting coordination, particularly in new organizational forms.

Many writers have described organizational forms, enabled by information technology, in which hierarchical control plays a much less dominant role than in organizations of the past. Variously called "postbureaucratic organizations" (Kellogg, Orlikowski, & Yates, 2006), "adhocracies" (Mintzberg & McHugh, 1985), "network organizations" (Powell, 1990), or "heterarchies" (Powell, 1996), these new forms of organizing are based on flows of information, experience rapid change, cross organizational boundaries, emphasize horizontal collaborations, and require adaptive capacity. Existing views of coordination, formulated to describe and explain more traditional and bureaucratic organizations, are not well-suited to these highly volatile new forms. These characteristics demand a view of coordination that describes and explains a rapidly changing landscape of task interdependencies.

In this paper, we make four contributions. First, we inform theory on coordination by extending our prior work on coordination requirements in software engineering (see, e.g., Cataldo, Wagstrom, Herbsleb, and Carley, 2006). We introduce theory that describes tasks as consisting of actions embedded in networks of constraints. The constraints bind tasks together, and the properties of these networks give rise to differing coordination needs.

Second, we introduce an initial coding scheme we have developed that allows us to code and analyze observational data. It is based on prior work for coding design meetings (Olson et al,

[1]CHI 2009 subcommittee, "Interaction Beyond the Individual," http://www.chi2009.org/Authors/CallForPapers/Subcommittees.html.

1992), but adjusted to be appropriate for the new theory. While the coding scheme still requires tuning, and is specific to the setting we are studying, the broad outlines can be readily adapted to analysis of observational data in a wide variety of settings where coordination might be studied.

Third, we present example results from an ad hoc engineering project. The results shed light on several properties of constraint networks that we have observed to influence the nature and extent of required coordination among individuals and teams.

Fourth, we present our plans for future analysis of constraint network data. We anticipate that standard network analysis techniques, especially such measurements as centrality, average closeness, and clustering coefficient, will provide insight into coordination requirements and their evolution over time.

Theories of Coordination

The literature on coordination is extensive and complex. We will attempt only to provide illustrative examples to sketch the background for our own work.

In describing theories of coordination, it will often be convenient to split the discussion of coordination into two parts: coordination requirements and coordination activity (Gerwin, 2004). *Coordination requirements*, as we use the term, are the dependencies among tasks that can exist at all levels of an organization, and can arise from many sources, including technical dependencies among parts of a product, workflow, and shared resources. For example, a coordination requirement arises when one group produces a part that is used by another to make a larger assembly. A different kind of coordination requirement arises when two teams designing interacting components must agree on the interfaces by means of which their components will interact.

Coordination activities, on the other hand, are actions that workers perform specifically to manage, address, and satisfy the coordination requirements. Coordination activity, as pointed out in the organizational behavior literature, can take several forms, including communication, planning, and working from shared representations (March & Simon, 1958). The effectiveness and efficiency of coordination activity is a product not only of how skillfully the activity is carried out, but also how well-suited the activity is to the underlying coordination requirements it

attempts to address, a view that fits broadly within the contingency theory perspective. Contingency theory holds that performance of an organization is a function of the "fit" or congruence between attributes of the organization's situation and its characteristics (Donaldson, 2001).

Traditional theories of coordination requirements generally describe and explain patterns of task dependency that are highly stylized, relatively large scale, and stable. For example, work flows can reflect sequential dependencies, where one process consumes the output of another, or reciprocal, where two processes exchange outputs (Thompson, 1967). These characterizations are quite useful for organizational design, and research has shown that different coordination mechanisms tend to be adopted for different patterns of dependency (Van de Ven, Delbecq, & Keoenig, 1976). These contingencies between dependency types and coordination mechanisms are important, but they assume stable and highly stylized patterns of dependency which can be addressed by deliberate implementation of an enduring coordination mechanism.

Alternatively, one can characterize coordination problems as one of several standard types, such as sharing or flow, and match these standard kinds of coordination issues with one of several mechanisms that are known to be possible solutions to these specific problem types (Malone, Crowston, & Herman, 2003). Again, while highly useful for designing relatively durable business processes, this approach is of limited utility for understanding new organizational forms, where interactions are continually morphing, and behaviors tend to be adaptive and improvisational.

A third example of a theory that addresses contingencies between dependencies that need to be coordinated and the activities appropriate for coordination is media richness theory (Daft & Lengel, 1991). This theory basically holds that media differ in their richness, meaning their ability to mimic face to face interaction. Richer media, such as video and audio, convey subtleties of tone of voice, gesture, and eye contact, making them much richer than, e.g., text-only e-mails. Richer media are required for more uncertain and ambiguous tasks. This is a step in the direction we take here – one could view our theoretical work as attempting to give a more detailed and predictive description of what makes tasks uncertain and ambiguous.

We note that there is a very rich literature on what we have called coordination activities. They are sometimes summarized as falling under three categories, which could be roughly called

coordination and mutual adjustment, shared representations, and coordination by "programming" (March & Simon, 1958). Distinctions between explicit modes of coordination and implicit modes are also often made (Wittenbaum, Vaughan, & Stasser, 1998). Communication has received particular attention, as in the work of Clark and Brennan on common ground in communication (Clark & Brennan, 1991), which is highly relevant to the design of communication technology. Others have distinguished several perspectives on coordination activities, such as the information processing, the social, and the political (e.g., Kellogg, Orlikowski, & Yates, 2006). These examples are typical of the research on coordination activities, in that it provides a rich and nuanced view of how coordination happens, but only vague notions of what sorts of activities are needed for coordinating particular kinds or patterns of task dependencies.

Coordination requirements in non-routine tasks such as software development are highly variable (Cataldo et al, 2006). In product development organizations, coordination requirements tend to be driven by the technical dependencies among the parts of the product being created (Cataldo, Wagstrom, Herbsleb, & Carley, 2006; Henderson & Clark, 1990; Sosa, Eppinger, & Rowles, 2004) – experts are brought on when required, designs are created and handed off, and ad hoc teams are created and disband once a specific technical requirement has been satisfied.

Most current theoretical views have two shortcomings: First, as we have pointed out, they tend to focus on relatively stable, enduring patterns of dependencies in organizations. This is helpful for purposes such as organizational design in traditional, bureaucratic organizations, but is less helpful in understanding how to support non-routine intellectual work, in which coordination requirements are highly volatile.

Second, theories of coordination tend to focus much more heavily on coordination *activities* than on coordination *requirements*. Much HCI research has focused, for example, on various communication media and their affordances, common grounding, co-presence, and awareness. As a community, we have accumulated a broad knowledge of tools and activities that we know are useful for communicating about and coordinating work tasks, and we have carefully considered the limitations of current and foreseeable future tools in supporting these activities (e.g., Olson & Olson, 2000). In contrast to this, we know relatively little about what creates the

requirement to coordinate tasks, how to measure or predict the required coordination, or whether particular tools or sets of tools will be adequate to coordinate given collaborations.

Because of these two limitations of current theory, i.e., the focus on enduring patterns of coordination and the focus on coordination activities as opposed to coordination requirements – current theories tell us relatively little about how to support non-routine work. In order to design technology that supports coordination of non-routine work, it is important to understand 1) how to predict and measure coordination requirements and 2) what particular coordination activities and tool support are needed for any given set of coordination requirements. We see our theory and results taking a step toward addressing these concerns.

We begin by describing how technical dependencies give rise to coordination needs. Later, we generalize the theory to encompass other kinds of work.

THEORY DEVELOPMENT: CONSTRAINT NETWORKS

We focus here on engineering tasks, in which the primary activity is making engineering decisions. Our data come from an engineering environment, so we begin building our theory in this domain.

In order to capture the coupling of work tasks, we adapt the idea of constraints and constraint networks (for an overview of constraint satisfaction problems and algorithms, see Yokoo, 2001). In constraint satisfaction, a problem is represented as a set of decisions with constraints operating over those decisions such that the way in which one decision is made constrains how linked decisions can be made. As a simple example, in designing a floor plan for the first floor of a house, deciding on the size and shape of the living room constrains how the other rooms can be sized and laid out. Many problems can be represented as constraint satisfaction problems, and there are many algorithms for solving problems formulated in this way. In general, this framework provides a way to describe how decisions are linked to other decisions, and to talk about the nature of the linkages and networks that bind decisions together.

This view is captured in the following proposition:

P1: Artifact design is a process of making decisions, and these decisions are joined by constraints in a potentially large and complex network (which we will simply call the constraint network).

Our view of coordination arises naturally from this view of the work. When people are making decisions that constrain each other, they need to coordinate to ensure that no constraints are violated:

P2: The need for coordination among individuals and teams arises from the constraints on the decisions they are making.

Different projects, different parts of projects, or even the same parts of a project over time will have constraint networks that vary widely in many ways, including such standard characteristics as density, average closeness, and clustering coefficient. The decisions are also assigned to different individuals or teams. The structure of the constraint network and the assignment of decisions to individuals determine the degree and nature of coupling between tasks the individuals are performing:

P3: Task coupling between individuals and between teams is the result of the properties of the decision constraint network and the assignment of decisions to people.

Together, these three propositions constitute our proposed constraint theory of coordination. In the next two sections, we describe our research setting and a coding scheme that can be used to analyze observational data for empirical studies of design activity based on our theory.

DATA COLLECTION

Site and project. We studied coordination in a joint engineering venture assembled to compete in the Google Lunar X PRIZE challenge to send a mobile robot to the moon. At the time of writing, we had observed the project for approximately 7 months. It was a complex engineering effort, with very tight coupling among tasks performed by four organizations. While the project will eventually include more sites and become much larger, during the period of observation most of the work occurred at one site, and involved a (mostly) collocated group of about 30 engineers from various disciplines including mechanical, electrical, structural, software, and systems engineering. Teams were generally organized around these disciplines, with a technical lead for each team. The work style tended to be much more collaborative than hierarchical, with fluid responsibilities and constant lateral communication.

Most team members performed their engineering work in a large "project room" which was an open space with desks, chairs, computers, and telephones. A few team members were located in the "high bay," a much larger space nearby used to assemble and test robots. A few other team members had individual offices in the same building. Team members attended a weekly "all-hands" meeting, which involved short status update presentations from each sub group on the team. At these meetings everyone was collocated except for a systems engineer at a remote location who participated via telephone. There were also occasional “technical” teleconference calls with members of other organizations involved in the project focused on getting remote members’ advice on technical issues.

Data collected. We recorded both types of meetings, all-hands and technical, taking detailed notes, audio recording the occasional technical meetings, and video and audio recording the all-hands meetings. We also conducted at least one semi-structured interview with every project member in order to understand their role, get a detailed description of the work they had done in the preceding week, and learn how they interacted with the other engineers. Detailed notes were taken of the interviews, and audio was recorded.

ANALYSIS METHODS

We developed a coding scheme for analyzing the observational data, focusing on coupling in the ongoing technical work. Our development of a coding scheme was influenced by an analogous scheme used to map out argument structure in design discussions, creating networks of decisions, alternatives, and criteria (Olson et al., 1992). However, since our research questions center on how design decisions are linked together, rather than focusing on arguments constructed in support of decisions, we looked for frameworks that would be more appropriate for describing this sort of connectivity.

As mentioned above, we found such a framework in *constraint satisfaction.* In constraint satisfaction (Yokoo, 2001), a problem is represented as a set of decisions with constraints operating over those decisions. The way one decision is made constrains how linked decisions can be made.

In principle, if we had perfect knowledge of all the decisions that needed to be made on a project, knowledge of every possible choice for each decision, and a perfect knowledge of all constraints among these choices, the design process could be fully represented and solved by a constraint

satisfaction algorithm. This is not possible in practice, of course, because information, possible alternatives, and the consequences of decisions only become known over time, as a project evolves. Nevertheless, thinking in terms of decisions and constraints is a good way to represent the connections behind work at a particular point in time. In the design process we observed, these connections drove coordination activity. The engineers were primarily engaged in making technical decisions, and considering how well these decisions satisfied the many constraints imposed by the contest rules, physical conditions in space and on the moon, and decisions that had already been made.

In order to capture the constraint satisfaction character of the design process we observed, we created a simplified notation for constraint networks. An "observed constraint network" (see Figures 1-3) has just two kinds of nodes and one type of link. Engineering decisions and their alternatives are represented in *decision nodes*. These represent the decisions and possible alternatives referred to in a design discussion. We use a list of component types to label the decision nodes to indicate what type of component or components the decision is primarily concerned with (see Appendix A).

Constraints that are mentioned or clearly implied are represented as *constraint nodes*, labeled with the general terms the engineers use to describe the constraint. (The current list of constraint terms is in Appendix A.) A link from a decision node to a constraint node indicates that the decision is subject to that particular constraint. We used both meeting and interview data to construct observed constraint networks. In this paper, we will show a few illustrative examples of constraint networks. These examples show how important properties of constraints and components, described in the next section, bind decisions together in highly variable ways.

CONSTRAINT AND COMPONENT PROPERTIES.

We have observed that constraint networks have at least three distinct properties that affect the nature of coupling among tasks: *diffusion, violation detection,* and *diversity*. We present examples of observed constraint networks that are derived from design discussions to demonstrate how these three constraint network properties influence work.

Constraint Diffusion.

Constraint diffusion refers to how widely a constraint's direct influence extends across the set of decisions. Highly diffuse constraints affect a great many decisions, while constraints that are low in diffusion affect only a few. For this reason, decisions subject to highly diffuse constraints may have consequences that propagate throughout many parts of the overall design.

Constraint Violation Detection.

For some constraints, it is relatively easy to detect whether they are violated in the current state of the design. *Constraint violation detection*, then, is important since one would like to remedy the violation as soon as possible. Since decisions are embedded in a network of constraints, if discovered late in the process a constraint violation can force the reconsideration of many related decisions.

Decision Constraint Diversity

Decision constraint diversity is the level of variety in the types of constraints that affect a particular decision. Some decisions are bound to other decisions via only one or a few different types of constraints, while for others, many different types of constraints must be considered. When *decision constraint diversity* is high, it is more difficult to trace through all the potential consequences of a decision.

OBSERVED CONSTRAINT NETWORKS

We illustrate the influences of these different constraint network properties using example networks and design discussions drawn from our observations.

INSERT TABLE 1 ABOUT HERE

Total Mass Constraint

An example of a constraint with very high diffusion is *total mass*. The launch vehicle has a strict limit on the mass it can carry into orbit, so the total payload must not exceed this limit. Every physical component has mass, and contributes to the total. Total mass is highly diffuse, since all

component-related decisions are constrained by mass. Figure 1 provides an abstracted constraint network showing how the total mass constraint is linked to the component choice decision for every component in the system.

INSERT FIGURE 1 ABOUT HERE

The diffuse nature of the total mass constraint influences the nature of coordination around decisions affecting total mass. As we write this paper, the current design is about 14 kg overweight, so a decision must be made about where to reduce mass. Discussions about the mass issue are very wide-ranging, potentially touching any component of the system, or even resulting in the removal of entire subsystems. The team has considered completely removing the radar range-finding component of the system to reduce mass. Such a change would drastically alter how the system computes distances from the lunar surface during landing. The mass constraint violation could lead to similarly large changes in most any subsystem. The result is that very difficult optimization decisions – which of the many different systems is more important for overall success – potentially requiring expertise from anywhere in the team.

Total mass is an example of a constraint where violation detection is relatively simple. The project maintains a list of all the components that comprise the system (called the Master Equipment List, or MEL), each of which has known mass. A running sum of the total mass is maintained in a spreadsheet, and if this sum exceeds the launch vehicle boost capacity, the total mass constraint is violated.

Since detection of a total mass constraint violation involves simply summing the known masses of all components, there is very little direct coordination required to make this determination, other than inputting and verifying the actual mass of components in the MEL.

Component Temperature Range Constraint

Each of the components in the system is designed to operate within a certain temperature range. The temperature experienced by each component is dependent on a myriad of factors including

the material properties of the component, the physical layout of the component relative to the rest of the system, the temperature of nearby components, etc. Determining whether the temperature range constraint of a component is violated is extremely difficult in practice.

For example, we observed a discussion of the violation of the thermal range constraint for the sidearm of the rover, a long assembly mounted lengthwise that connects the wheels on each side of the rover to the chassis. The team discovered, using complex thermal modeling techniques, that the sidearm would have thermal problems because it was situated on the radiator of the rover. If it got heated by the sun, it would stay extremely hot, because it was blocked from above and could not radiate the heat outward. The conclusion of this analysis was that the sidearm assembly could not stay in or out of the sun for too long. The team discussed how to handle this issue, and considered two options: changing the mission operations plan to frequently turn the rover so the sidearm would go in and out of the sun every so often, or coating the sidearm with a thermal insulation.

Because temperature is dependent on so many factors, many of which are dynamic and change over time, it is nearly impossible for team members to independently assess the thermal implications of a particular decision. The team must use advanced thermal modeling software to assess whether thermal equilibrium is violated by any of the decisions they have made. This also means the team must evaluate the thermal limits of any component based on the system as a whole and the current mission plans. Using modeling techniques to analyze complex constraints, like thermal range, allows the expression and automatic aggregation of a large number of disparate decisions. This aggregation also means that the thermal analysis must be redone when any changes are made to the design, making it extremely difficult and time-consuming to assess the thermal implications of any one decision on the project.

INSERT FIGURE 2 ABOUT HERE

Any decision with a thermal constraint – heat produced, sun exposure, coatings, proximity, and conductivity – can have implications for design decisions of other engineers. The difficulty detecting thermal constraint violations means that considerable time and effort must be invested

to discover these implications. The modeling tool plays a key role in facilitating coordination around thermal range, in much the same way that the MEL does for mass. However, detection of a thermal constraint violation requires far more work and modeling skill than does the summing of mass in the MEL.

Cable Choice and Constraint Diversity

The rover has an antenna mounted on top of a shaft that also supports several cameras. Selection of a coaxial cable to connect the antenna to the equipment generating the signal turned out to be influenced by surprisingly many types of constraints. The thickness of the cable determined its mass, and also the degree of attenuation in the signal. Cable thickness and sheathing also impacted how much electrical interference it would produce in the other cables going through the shaft, potentially compromising signal quality. The stiffness of the cable influenced the shaft's ability to rotate so the cameras could orient toward desired video targets. The thickness of the sheathing on the antenna influenced the likelihood that failure would occur if rotation resulted in cable abrasion. Finally, the cable had power implications, since a thinner cable would require more power to be applied in order to operate the antenna. A seemingly straightforward choice involved substantial coordination among engineers of many disciplines.

INSERT FIGURE 3 ABOUT HERE

In addition to observing links associated with individual components and constraints, we have also piloted our coding scheme to construct dependency networks from transcriptions of video data from weekly review meetings. These meetings have a consistent format – each team lead presents the accomplishments of the past week and plans for the upcoming week. The presentations are typically highly interactive, as other participants ask questions, make suggestions, and tease out the relevant consequences of project decisions, and consult on problems and issues.

Our approach is as follows: First, we segment the meeting into *decision discussions*. Most of the discussion in these meetings concerns design decisions. (For the moment, we are not coding the remaining discussion, which tends to cover topics such as project and meeting management,

digressions, etc., see, e.g., Olson, et al, 1992). The decisions are often not actually made at these meetings, but the discussion reveals the consequences of recent and upcoming decisions. The discussions tend to be fairly free-flowing, and the discussion of one decision leads naturally to a discussion of related decisions. We segment the transcript by identifying which decision is under consideration, and begin a new segment when a different decision is discussed. Returning to a previously discussed decision also marks a segment boundary. A two-hour meeting typically has well over a hundred decision discussion segments.

For each segment, we code the constraint types and components that are considered, based on a standard set of top-level components identified by the project, and a standard set of constraint types we are developing inductively. We then construct dependency networks based on constraints and components that are mentioned in the same segment. For example, if a segment focuses on selection of a lens for the HD video camera, and the discussion considers mass of the lens, then we would record a link between the constraint *mass* and the component *optical*. If the discussion also considered the mission requirement to capture video with a particular resolution, we further recorded a constraint-constraint link between *mass* and *project requirements*. If the discussion also mentioned a thermal control component, we recorded a component-component link between *optical* and *thermal control*. This approach is conceptually similar to the one we adopted in Cataldo, et al (2006), where we found that measuring things that were changed together to be a highly useful way of measuring dependencies. Components that are frequently mentioned together in the context of a single decision are highly likely to be interdependent at that point in time.

INSERT FIGURE 4 ABOUT HERE

Figure 4 shows an example of a component network constructed from observation of a weekly project meeting. (To reduce clutter, we show only links with more than 5 instances of co-occurrence, and removed isolates.) The diagram makes clear that some components such as *optics* and *software* are densely bound up with dependencies, while other components such as *solar panels* and *navigation and attitude determination* have only a single link. This appearance is confirmed by eigenvector centrality, which takes into account how connected a node is to other nodes, and how central are neighboring nodes. Eigenvector centrality here shows very

high values for *optics* (1.0) and *software* (.89), and very low values for *solar panels* (0.0) and *thermal control* (0.03). It was computed here over the component-component network.

INSERT FIGURE 5 ABOUT HERE

Figure 5 shows the component-constraint network (a bipartite or association network of two node types and one link type) from the same meeting. It shows the constraints (blue rings around the dots) that bind components together. One can see, for example, that *optics* is central in the component network by virtue of its links to eight distinct constraints. *Software*, which was also had high centrality in the component network has links to only 4 constraints, suggesting that while it was involved in many decision segments, it does not participate in as many ways in those decisions.

Figure 5 also shows a grouping of nodes by color, reflecting Girvan-Newman groups. This measure partitions the nodes into "communities" or groups based on the degree of interconnectedness among members of the partitioned sets. This grouping hints that components tend to cluster into three groups that can be thought of as "virtual modules" – i.e., highly interdependent internally, and independent across groupings. Further, these groups differ dramatically in the nature of the constraints that bind them together. The red group tends to be bound together by *business-related* concerns (e.g., cost and timely availability) and *manufacturability*. The green group is almost entirely bound together by *optics*, and the fact that decisions about optics are constrained in many different ways. The blue group appears to be bound together by what the robot must do on the surface of the moon, reflected in constraints of *mission requirements*, *con ops* (the detailed plan of what the robot will actually do and in what order), and the moon *environment*, which sets the conditions under which the movement must occur.

PLANNED ANALYSIS

The examples we have shown illustrate how local properties of constraint networks influence the nature of coordination that is required among engineers and between engineering teams. As we code more data, however, we plan several additional kinds of analysis to deepen our understanding of constraint networks, their evolution, and their effects.

The key to these planned analyses are the aggregation and filtering strategy for our constraint networks. We are coding the constraint networks at the smallest possible granularity – that of the individual decision. As long as a single decision is under discussion, we record all the constraints that are mentioned with respect to that decision, and create a small network with one decision node and one or more constraint nodes. As soon as another decision is mentioned, we start a new network, again with one decision node and one or more constraint nodes. Typically, the successor decision is brought up because of some constraint that bore on the first decision which led to discussion of the second. In our coding, this constraint is linked to each decision in the two networks. This scheme lets us aggregate in several different ways to reveal properties of the constraint networks. The two main approaches are to aggregate by node type, and to aggregate over time periods.

In all of our analyses of aggregated networks, we will begin by using standard network, dyad, and node measures to analyze the networks.

Network Analysis

We expect that a number of additional network measures, applied to our constraint networks, will have a substantial impact on the nature of the coordination requirements. For example, networks with a high clustering coefficient will likely represent a highly modular design, and should be effective and preventing coordination requirements from propagating across large numbers of people and teams. Decisions with high betweeness are likely to be particularly critical, and perhaps contentious, since they are linked to otherwise relatively independent clusters of decisions. Decision nodes with high centrality are likely to represent difficult decisions that will require much discussion and may be very difficult to change, given the number of other constraints and (indirectly) decisions they are linked to. Networks with high average closeness may be particularly difficult, since the effects of decisions will tend to propagate broadly through the network in few hops.

Aggregation of Networks

As we described earlier in our results, it appears that different kinds of constraints may be associated with different network properties. To examine this possibility, we will aggregate networks over node types. For example, we can investigate the network properties of thermal constraints by aggregating all atomic networks that contain a thermal constraint. We will

produce a weighted graph that will show us the strength of links between thermal constraints and various decisions, and will show us how strongly thermal constraints are linked – through decisions – to other constraints.

We will also aggregate networks over time, for example, producing one network for each meeting we analyze. This will help us to understand how decisions and constraints are actually discussed, and how teams draw a boundary around the things to be considered. Since every decision is likely to be linked to every other decision in some way in the overall graph, this sort of analysis will help us to see what subgraphs discussion focuses on, and perhaps to understand how the team copes with bounded rationality.

We will also examine the evolution of the networks over time. We expect the networks to change over time. Once a decision is made, it becomes a constraint in future networks. Once a particular cable is chosen, for example, this choice becomes a constraint for future decisions. This evolution will change the nature of the network over time.

Outcomes

We will be able to observe outcomes in the project, which we can then relate to both the constraint network and the observed coordination activities to understand the impact of particular patterns of coordination requirements and to understand the contingencies between coordination requirements and coordination activities.

Outcomes are observable in several ways. One is to note decisions that are made that have to be reversed. Another is to note how long it takes to make a decision. We can also see outcomes as a result of the extensive testing activities, as when prototypes are built and tested on surfaces and missions approximating actual conditions. We will use our interviews to ensure that we understand decisions that are made, as well as if, when, and why they are changed.

DISCUSSION

In this paper we have: (1) developed theory that explains coordination requirements in terms of constraint networks, (2) provided an analysis scheme for capturing and representing coupling among decisions observed in meeting data, (3) identified some important properties and illustrative examples of constraint networks that influence the ways in which tasks are bound

together, and (4) described our plans for applying network analysis techniques to the corpus of network data we are collecting.

Generalizing the Theory

In our theory, coordination requirements are the result of constraints among design decisions about an engineered artifact. The decisions and the constraints form a constraint network whose properties determine the coordination activities that need to happen.

Our data analysis so far has looked only a few examples of properties of constraint networks and nodes that appear to influence coordination requirements in major ways. We expect, however, that many network properties such as network density, clustering, closeness, and so on will impact coordination requirements in systematic ways.

Since it is properties of the constraint network that determine appropriate modes of coordination – rather than the specific character of the constraints as engineering dependencies – we expect that the theory and our results can be generalized to any work in which small work units affect each other in ways that can reasonably be characterized as constraints. For example, in a collaborative strategic planning task, the computations, assumptions, and goals of individuals will often constrain each other. We posit that network properties are key in understanding and predicting coordination requirements. Engineering projects and a strategic planning projects may, in some aspects of the work, share similar underlying constraint network structures. In these cases, we hypothesize that coordination requirements are also likely to be similar, and plan to test this notion in future work.

This approach lets us address coordination in a way that is largely *independent* of organizational form. We take the view that task dependencies are inherent in the nature of the work, and that these dependencies need to be coordinated in some way. This allows us to begin thinking about how well various patterns of coordination requirements – as expressed in constraint networks – can be satisfied with different kinds of coordination activities and tool support. The coordination requirements for a given unit of work are seen as fixed, and serve to express the coordination needs that people functioning within any organizational form will need to address in order to successfully undertake that work.

The results of this research will inform the development of smarter collaboration tools. By understanding and representing the coupling in the ongoing work, tool designers can intelligently

provide features that support different coordination activities, such as configuring awareness and communication channels in an automated fashion dependent on who needs to be aware of different work tasks or involved in communication about a decision. Because our notion of constraint networks captures changes that happen in work over time, configuration of tools, and involvement of different collaborating parties can be done in a timely fashion as work evolves and coordination requirements change.

Supporting the interaction of multiple users and collaboration at the group level is an increasingly central goal both in the field of HCI and in the field of Information Systems. Our work directly addresses that goal. It is important that we base interactive system development on representative and sophisticated techniques for analyzing user needs and requirements in a group setting. We believe the constraint network representation may provide a generally useful method for modeling tasks and the activities needed to manage those tasks. It may be a first very preliminary step towards analysis techniques that would support system development for interdependent and dynamic group work tasks (in the same way techniques such as GOMS or cognitive task analysis support user modeling and design of systems for individual-level tasks (Card, Moran & Newell, 1983)). This research will bridge the gap between IS and HCI, taking HCI beyond the individual and bringing IS to the group level, by directly contributing to our understanding of interdependent work, informing the design of tools for supporting collaboration, and sowing the seeds for new design methodologies of the future.

REFERENCES

Card, S., Moran, T., and Newell, A. (1983). *The Psychology of Human-Computer Interaction,* Lawrence Erlbaum Associates, Hillsdale, NJ.

Cataldo, M., Wagstrom, P.A., Herbsleb, J.D. and Carley, K.M. (2006). Identification of coordination requirements: implications for the Design of collaboration and awareness tools In *Proceedings of the 2006 20th anniversary conference on Computer supported cooperative work*, ACM Press, Banff, Alberta, Canada.

Clark, H.H. and Brennan, S.A. (1991). Grounding in Communication. in Resnick, L.B., Levine, J.M. and Teasley, S.D. eds. *Perspectives on Socially Shared Cognition,* APA Books, Washington, DC, 1991, 127-149.

Daft, R.L. and Lengel, R.H. (1986). Organizational Information Requirements, Media Richness, and Structural Design. *Management Science, 32 (5).* 554-571.

Donaldson, L. (2001). *The Contingency Theory of Organizations*. Sage, Thousand Oaks, CA.

Gerwin, D. (2004). Coordinating New Product Development in Strategic Alliances. *Academy of Management Review*, 29(2): 241-257.

Kellogg, K.C., Orlikowski, W.J. and Yates, J. (2006). Life in the Trading Zone: Structuring Coordination Across Boundaries in Postbureaucratic Organizations. *Organization Science, 17 (1).* 22-44.

Malone, T.W., Crowston, K. and Herman, G.A. (2003). Organizing Business Knowledge: The MIT Process Handbook. MIT Press, Cambridge, MA.

March, J.G. and Simon, H. (1958). *Organizations.* John Wiley and Sons, New York.

Mintzberg, H. and McHugh, A. (1985). Strategy Formation in an Adhocracy. *Administrative Science Quarterly,* 30 (2). 160-197.

Olson, G.M., Malone, T. and Smith, J. (Eds.). (2001). *Coordination theory and collaboration technology*. Lawrence Erlbaum Associates, Hillsdale, NJ.

Olson, G.M. and Olson, J.S. (2001). Distance Matters. *Human-Computer Interaction, 15*. 139-178.

Olson, G.M., Olson, J.S., Carter, M.R. and Storrosten, M. (1992). Small group design meetings: An analysis of collaboration. *Human-Computer Interaction.*

Powell, W.W. (1996). Interorganizational collaboration in the biotechnology industry. *Journal of Institutional Theoretical Economics, 120*. 197-215.

Powell, W.W. (1990). Neither Market nor Hierarchy: Network Forms of Organization. *Research in Organizational Behavior, 12*. 295-336.

Thompson, J.D. (1967). *Organizations in Action: Social Science Bases of Administrative Theory*. McGraw-Hill, New York.

Van De Ven, A.H., Delbecq, A.L. and Keoenig, R. (1976). Determinants of Coordination Modes within Organizations. *American Sociological Review, 41 (2)*. 322-338.

Wittenbaum, G.M., Vaughan, S.I. and Stasser, G. (1998). Coordination in Task-Performing Groups. in Tindale, R.S., Heath, L., Edwards, J., Posavac, J.J., Bryant, F.B., Suarez-Baleazar, Y., Henderson-King, E. and Myers, J. eds. *Theory and Research on Small Groups* (pp. 177-204), Plenum Press, New York, NY.

Yokoo, M. (2001). *Distributed Constraint Satisfaction: Foundations of Cooperation in Multi-agent Systems*. Springer, New York.

LIST OF TABLES AND FIGURES

TABLE 1.

CONSTRAINT NETWORK EXAMPLES AND THEIR PROPERTIES.

	Constraint diffusion	Violation detection	Constraint diversity
Total mass	High	Easy	Low
Temperature range	Medium	Difficult	Low
Cable selection	Low	Medium	High

FIGURE 1.

DIFFUSE TOTAL MASS CONSTRAINT NETWORK

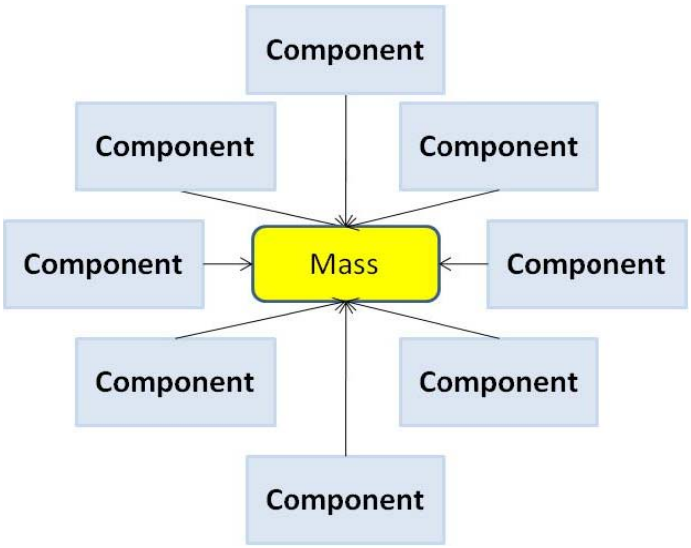


FIGURE 2.

CONSTRAINTS RELATING DECISIONS TO SIDEARM DESIGN

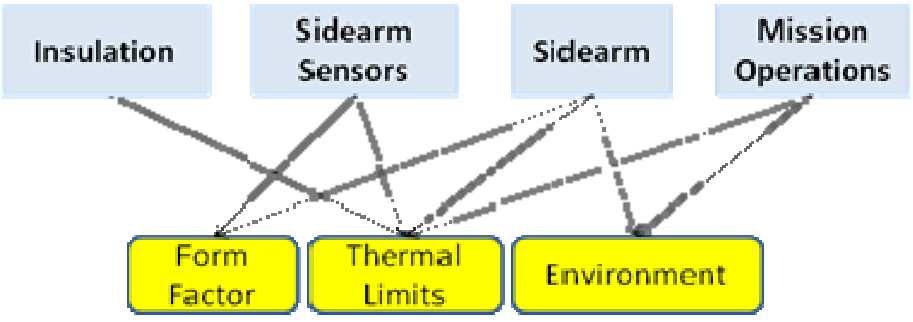


FIGURE 3.

DIVERSE CONSTRAINTS ON CABLE THICKNESS

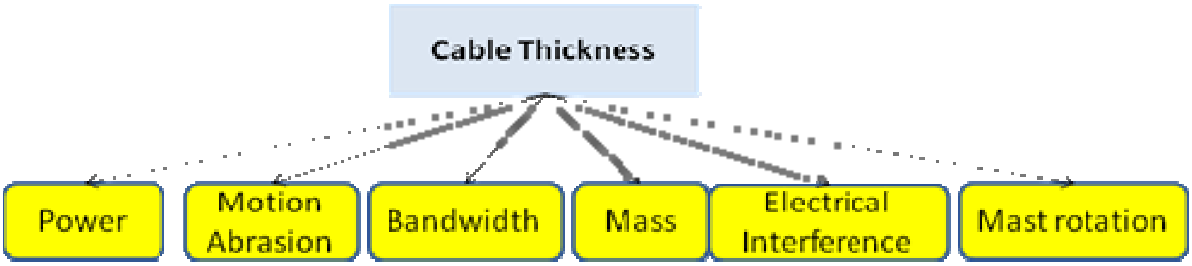


FIGURE 4

OBSERVED CONSTRAINT LINKAGES AMONG COMPONENTS

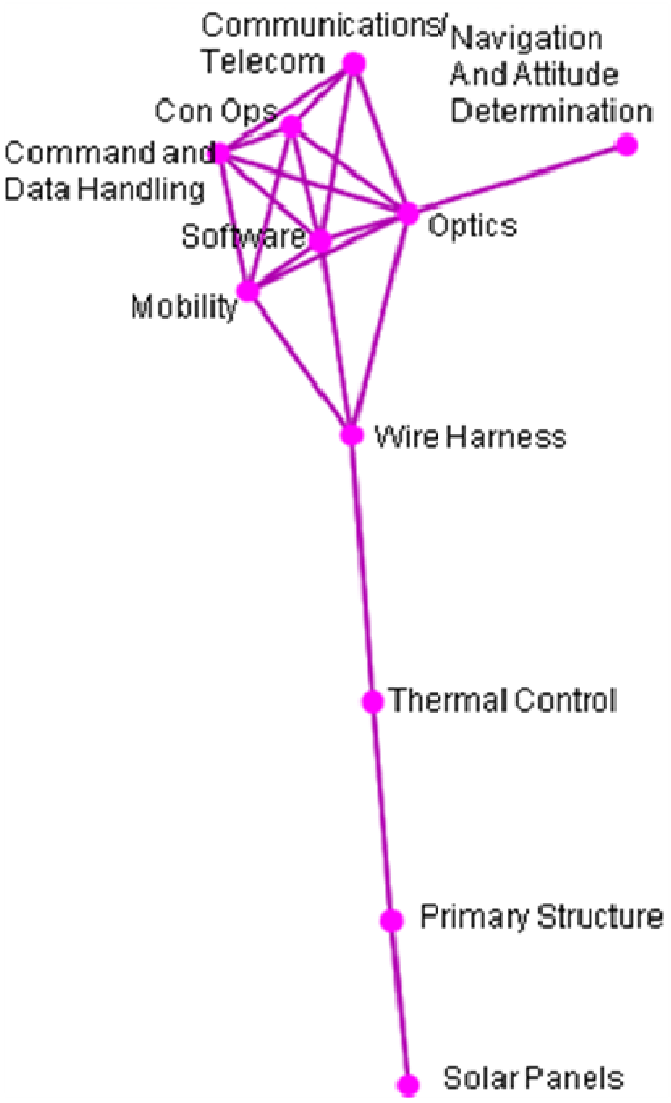
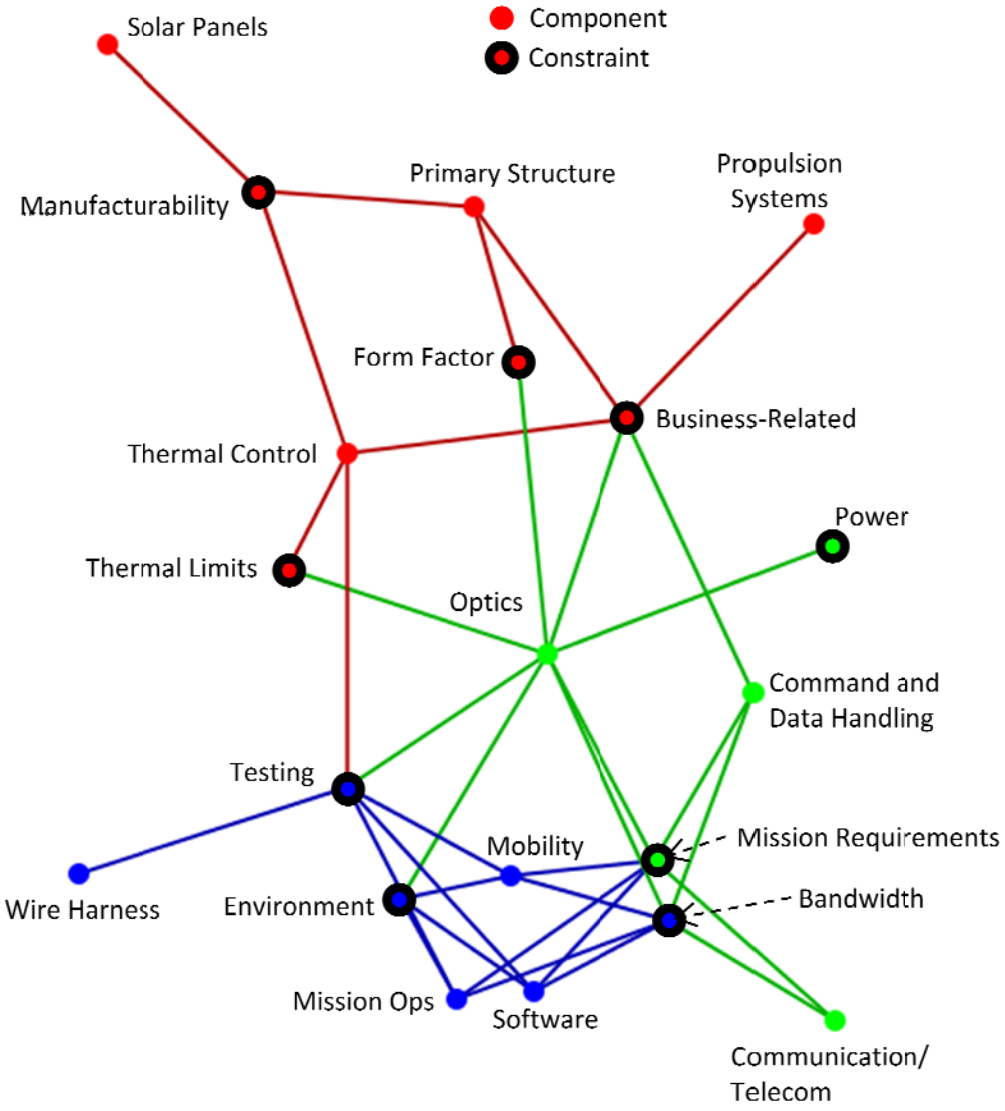


FIGURE 5

BIPARTITE CONSTRAINT NETWORK



APPENDIX A:
CODING SCHEME DESCRIPTION FOR ENGINEERING TEAM MEETINGS

The verbal content of meetings is segmented by decisions that are discussed. Each time a new decision is discussed, a new "atomic" graph is created. Each atomic graph represents discussion of a single decision and the components and constraints involved in the decision. Most decisions involve a single component and various ways it can be designed or configured, such as determining whether the lander attachment structure will be either a frangibolt or a shape memory alloy. Generally, one or more constriants are explicitly mentioned or clearly implied. For example, frangibolts require more power than shape memory alloys, but have less mass. Some decisions may concern more than one component, such as a discussion of whether to place a thermal sensor in the rover chassis.

Segmenting discussion. A new graph is drawn for each decision that is discussed. Whenever the discussion focus changes from one decision to another, a new graph is drawn, even if the second decision is a subdecision, closely-related decision, or a decision that bears on some or all of the same constraints.

Components. Component names (examples above are in red) are drawn from a high-level list of components. Mission operations (the sequence of actions the launch vehicle, lander, and rover will undertake) is also considered a component. (We may eventually want to break this down into several components.)

Constraints. Constraint names (examples above are in blue) are drawn from a list of general constraint types.

Links. For atomic decisions, a link is drawn from a component to a constraint when that constraint influences choices made about the component. For decisions that concern single components, the graph will consist of one component node and one or more constraint nodes,

with a link from the component node to each constraint node. For decisions concerning more than one component, a link will generally be drawn from each component node to each constraint node.

System Components List

1. Navigation and Attitude Determination
2. Mobility
3. Optics – Camera system
4. Communication / Telecom
5. Thermal Control
6. Command and Data Handling
7. Power
8. Lander structure
9. Wire harness
10. Propulsion systems
11. Mission operations
12. Software

Constraint Types

1. Thermal limits
2. Structural integrity
3. Form factor (size, shape, ...)
4. Power (storage, efficiency, use over time,)
5. Mass
6. Mission Requirements
7. Environment (physical environment during mission)
8. Business Related (cost, availability, etc)
9. Manufacturability (can actually make it)
10. Bandwidth