The Geography of Coordination: Dealing with Distance in R&D Work

Rebecca E. Grinter, James D. Herbsleb
Bell Labs, Lucent Technologies
263 Shuman Blvd
Naperville, IL 60566

Dewayne E. Perry
Bell Labs, Lucent Technologies
600 Mountain View Drive
Murray Hill, NJ 07974

{beki, herbsleb, dep}@research.bell-labs.com

ABSTRACT
Geographically distributed development creates new questions about how to coordinate multi-site work. In this paper, we present four methods product development organizations used to coordinate their work: functional areas of expertise, product structure, process steps, and customization. We describe the benefits and difficulties with each model. Finally, we discuss two difficulties that occur irrespective of the model used: consequences of unequal distribution of project mass, and finding expertise.

Keywords
Geographically distributed development, coordination mechanisms, collaborative work.

DISTANCE AND COORDINATION IN R&D WORK

The last ten years have seen a dramatic rise in the studies of collaborative work. Many of these studies focus on the moment-by-moment aspects of collaborative work such as how people handle exceptions, errors, and unexpected contingencies [10]. In order to examine these subtleties of work the majority of these studies have examined co-located work settings.

At the same time, many corporations have increased the amount of distributed work that they do. Mergers, acquisitions, alliances, and market demands are some of the reasons why companies have distributed development across geographically separated sites [5]. This is particularly true for Research and Development (R&D) work, which we are especially interested in.

R&D work differs interestingly in some ways from other types of work, in that it has a distinctive life cycle, often from months to years, with typical stages through which the work passes. A typical life cycle begins with requirements, proceeds through design, construction, several stages of testing, and delivery to the customer. The pressure to deliver according to a pre-arranged schedule is often intense, and affects all of the work.

In addition to this life cycle for specific deliveries, the products that we studied have a longer life cycle in which they are reworked repeatedly as new features and enhancements are made. In addition to the time scale and typical stages, R&D work generally involves both tightly-coupled work, as when one or more teams work on a particular part of a product, and loosely-coupled work, as when different teams work on relatively independent parts of the product.

R&D work requires significant coordination. The different parts of the life cycle must proceed in a coordinated way across all parts of the product, and each step of the process must hand off seamlessly to the next. The scale of operations can involve hundreds or even thousands of people. A given product may also have quite a long life; one of the products we describe is in its second decade of production. For these reasons, people working on a product need to coordinate their actions.

This research set out to examine how R&D projects coordinate their efforts across multiple sites. An investigation of coordinating work across distance is particularly timely since the proliferation of electronic networks and tools sometimes lead to the claim that distance is becoming irrelevant [4]. If true, this would be good news indeed, since there are many economic forces acting to split projects across sites (see, e.g., [5]).

Distance and Disruption of Ad Hoc Communication

There is good reason, however, to be skeptical about claims that distance makes no difference to R&D work. One reason is provided by the findings from studies of work that show that informal, unplanned, ad hoc communication is extremely important in supporting collaboration [7, 12, 14, 19].

For example, in their study of telecommunications engineering, Perry, Staudenmayer and Votta [19] found that developers spent over 15% of their time communicating with their colleagues informally. A study of several projects by Curtis, Krasner and Iscoe [7] found that breakdowns in informal communications could lead to misunderstandings in design conventions and rationale. Further, they suggest that separation in time and organizational distance can lead to these kinds of breakdowns. As Kraut and Streeter [14] observed, in their survey of 65 projects in one large development company, as the size and complexity of the software increases, the need for supporting informal communications increases dramatically.

GROUP 99 Phoenix Arizona USA

These findings suggest that informal communication plays a critical role in coordinating R&D work. Therefore, one of the central problems of distributed development is generated by the fact that distance profoundly reduces the amount of such communication [2, 11-14]. The primary reasons for this reduction appear to be fewer opportunities and higher cost.

Opportunities are reduced in several ways. I may determine that I need to talk to X, and at the next opportunity, i.e., the next time I see that X is available by, e.g., walking by his office, I will stop in for an exchange. If I never spontaneously see X, because he is resident elsewhere, this is much less likely to happen. It may also be the case that during the conversation "at the water cooler," important work related exchanges are triggered although that was not the original intent.

In addition to reducing communication opportunities, distance increases the "cost" (i.e., the difficulty, inconvenience, frustration, and so on) of communicating. Depending on the time zones involved, there may be many fewer (or no) overlapping business hours. It is very difficult to tell if someone is available for a conversation, without cues such as open doors, knowledge of local schedules, vacations, holidays, and so on. Moreover, voice mail and e-mail are less likely to be answered by someone you do not know well. With additional difficulties of language and culture, the awkwardness increases further.

Many kinds of tools, especially video, have been designed and deployed with the intent of increasing informal communication. Examples include Portholes [8], Cruiser [9], and a video wall [1]. Despite some successes with these technologies, they are no longer in use even in the organizations where they were initially deployed. So, informal communication across sites remains much less frequent than same-site interactions.

Four Different Bases for Coordinating Project Work

Frequent ad hoc communication, however, is only one way that individuals can coordinate their work. Other approaches focus on designing the organization and assigning the work so as to reduce the amount of informal communication required. Although this will not eliminate the need for informal communication, the goal is to reduce it to a more manageable level. In this section, we introduce four models that organizations may adopt for this purpose.

In 1987, Thomas Allen and Oscar Hauptman [3] described two models used for organizing R&D work. The first comes from the observation that a particular function is generally carried out by people with similar expertise. A functional organization creates organizational units based on these functional areas of expertise. A corporation following this model would have units organized around functional areas like system engineering, human interface design, testing, and development. Allen and Hauptman's second model of organizing focuses on projects, placing all work necessary to produce a product release in one organizational unit.

Allen and Hauptman have provided several criteria for selecting one model or the other. The faster the pace of technical change in the functional areas of expertise, the more likely that the functional model is preferable. The functional model facilitates dissemination of technical knowledge by placing experts in the same organizational units. The more difficult the dissemination problem is, the more likely this model is superior. Similarly, the longer that tasks typically take, the more the functional model is to be preferred. If projects typically last several years, for example, an expert who is out of touch, for that period, with colleagues in the same discipline is likely to become outdated. If projects are short, on the other hand, this risk is much smaller. Finally, if the interdependence among tasks is high, the project organization is likely to be better, since much ad hoc communication among project team members will be required to manage the interdependencies.

Allen and Hauptman's analysis can be applied in a straightforward way to geographically distributed organizations. Presumably, the same forces that would make it desirable to place either an entire project or a functional area in one organizational unit to support communication, would also make it desirable to co-locate individuals on similar bases. Thus, we will refer to the "functional area" model and the "project" model for geographically distributed organizations. By this, we mean that functional areas of expertise or projects are co-located (whether or not they correspond to official organization-chart units of the organization).

In addition to these models, at least two other bases have been suggested for designing R&D organizations: product structure and development processes. As early as the 1970's David Parnas observed that the decomposition of the design problem into units — known as modules in software development — corresponded to a division of labor [17]. In other words, early decisions about a product's structure — the architecture of the system — would determine who would need to coordinate their work with whom.

In their study of automotive engineering, Olson and Teasley [16] observe that the architecture of the system influenced the communications required among project members. They found that when the system architecture has independent components these can be built by geographically separated teams because less communication is required to align the pieces. Olson and Teasley's study is particularly interesting because they studied a case of distributed development. Originally, the distribution of work did not follow the architectural demands, and a remote team attempted to work jointly on a piece of the system. Over time, the remote team slowly relinquished control of that part of the development effort, and focused on supplying a piece that was more independent. This, they argue, came about because of decreased communications opportunities for communication among the local and remote sites.

In a study of distributed integration work, Herbsleb and Grinter [12] found that the difficulty of communicating

across sites greatly impeded the ability to assemble the product. Again, the authors point to the difficulties as focusing on the architecture of the product itself. This tight relationship between the architecture of the product and the communication structure was observed over 30 years ago by Melvin Conway [6]. Conway's Law — as it is known today — says that the structure of the product will mirror the organizational structure of the people who designed it.

The implication of this line of work is that co-locating development of a particular product component is another plausible model for organizing geographically distributed work. The tasks associated with designing and constructing the internal workings of a particular component are generally highly interrelated, so co-locating everyone performing those tasks should facilitate the needed communication. The tasks associated with designing and constructing different components, on the other hand, are likely to be much more loosely coupled, and require less communication.

The last area to which we turn in examining possible foundations for models of geographically distributed R&D organizations is the development process. Products typically pass through several stages of work, from requirements, through system engineering, design, and construction, through several stages of testing. At each stage, a particular mix of expertise and particular kinds of resources (e.g., access to the customer, detailed understanding of the local infrastructure, complex test laboratories) are required. For these reasons, it often makes sense to co-locate the activities from a given process stage (to share a test lab, for example, or to locate design facilities near a customer). Analogously with the case of product structure, the coupling of tasks within a process step is likely to be much tighter than the coupling between process steps. Thus, it will often make sense to co-locate a process step, while distributing different steps across sites.

Thus, we have identified four basic models that might be used by geographically distributed R&D organizations. They are formed along some of the basic dimensions along which work must be coordinated [12]. The essential premise, shared implicitly by Conway, Parnas, and explicitly by Allen and Hauptman, is that the best design will identify the most difficult dimension of coordination, and facilitate coordination along this dimension by means of co-location. As discussed in the previous section, co-location facilitates ad hoc communication, which is certainly one effective means of supporting coordination.

Strategic use of co-location can ease coordination problems along one dimension, but unfortunately, R&D organizations must coordinate along *all* of the dimensions simultaneously. Co-locating development work by component, for example, may help solve the problem of intertask dependencies for each component. However, that does not address other problems of distributed organizations, such as integrating the components, optimizing use of scarce resources across projects, or disseminating state of the art technical information among technical experts.

In contrast to the "co-location with lots of communication" approach to coordination, one might instead adopt various "coordination mechanisms" [20-22]. Coordination mechanisms provide a common view or map of how the various pieces of work fit together. So, for example, component interface specifications [18] play the well-known role of helping to coordinate the work between developers of different components. If the designers of two components agree on the interface, then design of the internals of each component can go forward relatively independently. Designers of component A need not know much about the design decisions made about component B, so long as both sides honor their well-specified commitments about how the two will hook together. Similarly, if development agrees to deliver unit tested code that satisfies a particular specification to testers on a certain date, then as long as both sides agree on the specification, on what "unit-tested" means, and on the delivery date, the testers do not need to concern themselves with precisely how and when the development tasks are carried out.

It was with these insights that we began our examination of how R&D product groups coordinated work in practice. In the rest of the paper we present four models that these projects used to organize work across multiple sites, how they operated, as well as the benefits and problems associated with these approaches and the coordination mechanisms they relied on. First, however, we briefly outline the organizations studied and methods we used to gather and analyze the data.

SITES AND METHODS

Our data are drawn from several development projects within Lucent Technologies, a telecommunications systems company. Specifically, we gathered data from six different organizations throughout the corporation. Each organization is briefly described here:

Alpha, divided across numerous sites, builds a very large telecommunications product. In this study, we conducted interviews at three sites.

Beta began in United Kingdom as a product for that market. Initially, developers were co-located, but later two versions of the product were created, and developers were located close to each of two separate markets. A third site, acquired later, took their old product and integrated it into Beta's product.

Gamma has numerous sites. The product is software, generally for monitoring and managing networks. It must interoperate with software and hardware built by a number of vendors. The basic product is built in the USA, and additional work for deliveries to particular customers is performed in Europe.

Delta produces a series of smaller products that are marketed together. Collectively, the components provide a network product, including a manager component that ensures that all the others work together.

In each organization, we conducted five or six interviews with a variety of different people involved in the project, including developers, and middle and upper management. We prepared different interview guides for developers and for managers, in order to reflect their different skills and knowledge. These interview guides provided some structure for the interview but were designed to allow the individuals to speak at length.

We conducted 27 interviews in these organizations. Interviews provided us with information about the project evolution and the current state of distribution of work. However, to supplement our understanding of their products we also used technical project documentation, organizational charts and development timelines.

After having all the interviews transcribed and collecting the supporting project materials, we began analyzing the data for discussions about the distribution of work among different sites. Specifically we tried to isolate the experiences of cross-site development, some background about how decisions about where to locate development got made, and the different ways of distributing work across sites. We also captured data about some of the difficulties and benefits of working across site.

Theta began at two sites simultaneously. One site was completely new, while the other site was acquired. Theta's product evolved from a one that existed at the acquired site. Other sites provide substrates.

At Theta, we did not use the same interview guides as we did elsewhere. We have been working with Theta to support their cross-site collaboration needs for over a year now. We have studied parts of the development work in detail as well as looking at their overall division of labor. We have gathered considerable data from Theta from which we extracted an understanding of how they organize their work.

After analyzing each organization individually, we compared our findings across organizations. We looked for cases where different organizations shared common characteristics, to help us build a more general picture of that method of coordinating work. However, we also sought differences among the organizations studied, because these contrasts helped us refine our understandings of the implications each model has for coordination.

From our analysis of data gathered from all six organizations, we were able to clearly distinguish four patterns or models of geographical distribution. Most organizations had a single model that dictated much of the large-scale distribution of work across sites, but every organization also showed evidence of more than one model. For example, an organization might assign work to several sites based on different process steps, and further split the work of the "coding" step across several sites based on the product structure.

In the next four sections, we will present four different models of organizing R&D work that we found: functional area, product structure, development process, and customization.[1] For each model, we describe its essence, the benefits and drawbacks of the approach, and the coordination mechanisms it relies on.

FUNCTIONAL AREAS OF EXPERTISE

What Is It?

Expertise for a specific functional area involved in development of the product is located at a single site. A site may serve as the location for more than one functional area of expertise, but each area of expertise is located at a single site.

What Are The Benefits?

Larger Pool of Experts

One of the major reasons given for using this arrangement, in contrast to co-locating projects, is that several advantages flow from having all the experts in a given functional area together in a single "pool" at one site, as opposed to a larger number of smaller pools.

Better load balancing. If a number of projects are staffed entirely by smaller, distributed, local pools experts, it is very difficult to ensure that the staffing needs of current projects precisely match the locally available expertise. It may often be the case that experts in some areas are idle while the needs for other types of expertise cannot be met. Unless projects are very consistent in the mix of expertise they require, this resource constraint can seriously slow development. Where experts in each functional area are kept in a single, large pool, however, the chances are much better that the overall demand for expertise across all projects will average out, and there will be fewer critical shortages.

Developing and enhancing expertise. Our respondents also report that it is much easier to make some expert time available when there is a single, large pool. This is critical for mentoring novices, especially when much time is required for a productive level of expertise to be acquired. Similarly, it is easier to provide time to keep up with technical developments, e.g., by attending conferences, participating in standards bodies, and so on, if the experts are in a single, large pool. A few people can be assigned to keep up, to develop new project-spanning assets such as tools or embedded technologies.

What Are The Problems?

Managing Projects That Span Sites

When an organization co-locates functional areas of expertise, then many projects are likely to involve multiple sites. Designing, constructing, and testing a specific release of the product will have to be accomplished across distances.

The organizations we observed using this model had at least two advantages that allowed them to manage projects successfully across sites. First, the staff was generally very experienced, and the product had been around for a long time. This made it relatively easy to predict, for any given release of the product, where the work had to be done, how

[1] Interestingly we did not find any examples of organizing R&D work around fully co-located projects.

much effort it would require, and what the interdependencies across sites were likely to be.

Our respondents regarded the assignment of responsibility as the most critical factor in cross-site project management. A single overall project manager owned responsibility for the entire product release. In general, the project was assigned to a manager who resided at a site where the bulk of the work would be done, if there was such a site. In addition, product releases consisted of a set of features. A feature manager was assigned for each feature. A feature team varied from one or two developers up to about 10, and generally took on the order of three months to complete their work on the feature. Feature development also often spanned sites, creating, in effect, small distributed "projects" within the larger project.

While these arrangements in general produced very good results, several informants mentioned that unexpected events tended to be very disruptive. One informant mentioned that when an unusual project creates some confusion about how and where some of the work should be done, serious difficulties can arise. Where these things are clear, being distributed across sites does not have a large effect. When it is unclear, it gets more difficult, "because you can't just wander down the corridor and sit on someone's desk and say, 'look, what do you think?'"

In these more difficult ambiguous situations, personal cross-site networks become extremely important. As one respondent mentioned, these things can be resolved over the phone "if you have a good relationship with someone." When these sorts of contacts cannot be made because an adequate network doesn't exist, or when these discussions do not resolve an issue, it is escalated to a manager, quite often the project manager. They play a major role in repairing communication breakdowns. Travel and face-to-face meetings are often required when this happens. They seem to be the only way of getting "unstuck" when there is a problem.

Coordination Mechanisms

Co-location helps solve the problem of dissemination of state-of-the-art technical information. This model relies on accurate project plans and a consistent and detailed development process to coordinate releases and features across sites.

PRODUCT STRUCTURE

What is it?

The product structure model for organizing R&D work splits the organization across sites along lines suggested by the product architecture. Specifically, each component is developed at a single site, but the development of different components is potentially distributed among different sites. This model takes advantage of any clean separations of pieces into isolated parts that can be developed alone.

What Are The Benefits?

Independent Operating Environments

One of the most significant advantages Delta gained by arranging work this way was that the individual components did not need to know or follow the same processes. Furthermore, each component could also use different tools for their development environment. In other words, each component had process and tool independence from the other components and the component manager, as long as they all adhered to the same interfaces among the components. By sharing a common, internationally recognized telecommunications standard they had some help with defining and remaining consistent to an interface that united the components.

Independence in processes and tools allowed the different sites to use procedures and technologies that they were familiar with. This was advantageous for several reasons. First, the components provided various kinds of functionalities. Some of the components have highly graphical elements, while others are almost exclusively application specific integrated circuitry. The processes and tools that the developers of the different components need differ radically. Arranging Delta by product structure allowed the individual components to be able to select their own processes and tools that allowed them to work effectively.

Second, Lucent Technologies acquired several of the components through mergers and acquisitions. In these cases Lucent Technologies, and specifically Delta's management, received much more than just a component, they inherited people, processes and a development infrastructure. The people working on that component with the necessary skills and expertise would have spent considerable time retraining, and consequently not releasing new versions of their component, if they had been pressured to abandon their entire development environment for the sake of unifying tools and processes.

Third, many of Delta's sites are internationally located. While several of them have access to similar tools and people with education that is comparable to that found in the United States, it is not always identical. Access to technologies depends on whether the vendors of the equipment used have good sales and support channels in the countries where Delta has locations. Further, the design of processes depends on the education of those who will develop them. The ability to get processes used also depends in part on the ease with which they are understood and make sense to their intended audience.

Like all large companies, Delta's processes had to conform to international standards like ISO 9000 but to the extent that they were independent, the procedures could also be designed to meet the needs of the local developers. By allowing the use of different technologies and processes, Delta's management avoided having to find a common set among the different sites working on Delta components.

What Are The Problems?

Testing and Changing the Component Manager

Due to the separation of the components that together comprise Delta's network product, testing of each piece gets done locally on-site. Analogous to the separation of other parts of the development the local testing can proceed smoothly because all the experts are local, so bugs can be quickly resolved. However, this is not true for the

component manager. The component manager can be tested locally for internal consistency, but its very role, as the device that manages the other products means that eventually it has to be tested in concert with them.

Delta originally decided that the solution to this testing problem was to have each component test itself for internal consistency and then test the interactions between the product and the manager. The problem with that arrangement was that when the interactions between component and manager produced spurious behavior there was no one locally who had expertise with the manager since all the experts were at the distant site, where the manager was built.

As soon as the expertise for resolving the problem was not local the length of time to provide a solution increased. Sometimes the problem was compounded by an inability to diagnose it correctly, based on exactly the same kinds of issues. Sometimes the problem that was seen was not actually the cause, and so the first attempt did not work.

Eventually Delta abandoned this arrangement of testing. The manager is now tested at one site. Experts representing the individual components come to that site if the component manager testers need their expertise.

One Feature, Many Components

Delta's strength of organizing around separate components that together provide its network product, is in one respect its biggest challenge. Part of creating the network means building features and applications that run over the network in a consistent fashion. In other words, the trade off for separating components, is having geographically distributed features. While the component manager is one source for unifying the behavior of all the individual components, features still rely on all the substrate parts providing information and behaving in predictable and reliable ways.

In the current project structure, each component evolves independently, which makes it hard to align features across them until integration time. Recently, Delta has begun experimenting with feature coordination teams. These teams span the components and the manager. Their purpose is to focus on what features need to be provided and how each component will provide any data and functionality required. Collectively, these teams will begin coordinating the functions that span multiple components in the requirements stage rather than reverse engineering it out of the final shipped releases.

Coordination Mechanisms

Co-location helps in coordinating the dependencies that arise within the design and development of each component. Standards and clear interface specifications are used to coordinate across components. While the sites do not have, nor seem to need, a shared process, the points at which components become available for integration and testing are well specified, and this is relied upon.

PROCESS STEPS

What Is It?

Work is broken up into process steps such as systems engineering and testing. These steps are then used as handoffs among various locations.

What Are The Benefits?

Closer to the Customer

Locating parts of a product destined for multiple markets close to the potential customers allows the development team to take advantage of local knowledge. In telecommunications, this can often mean an understanding of the protocols and standards that govern that country's telephone network. Alpha, a large project selling its product in many markets took advantage of this local knowledge by locating their requirements gathering processes near to the customers.

Specifically, during the course of the interviews we did with members of Alpha, the management was in the process of deploying systems engineers in countries with large markets for their product. Alpha's management took the front end of the development process and separated systems engineering in ways that allow the requirements for revisions, extensions and enhancements to be written at remote sites, and then pulled together.

Better Use of Scarce Resources

Telecommunications products like other systems require rigorous testing to ensure that they behave as expected. However, unlike some products, telecommunications systems can demand extensive resources for testing. Theta' product is no exception to this rule. It needs to interact with many other products, built by the corporation as well as other vendors, and some types of tests require setting up a fully working network with all these other products and testing Theta's product against it.

What complicates the testing in a network setting is that the other elements are also being developed simultaneously. The testing labs where Theta's product is tested are vital, but complex, resources that do not belong to the project. Specifically, the labs maintain a network of elements, evolve those elements over time to remain current, and then test the products against each other to see that everything works as intended.

Theta like many other organizations used a process model to hand off this testing to this special testing resource. It was well beyond Theta's own resource budget to maintain a lab of this level.

What Are The Problems?

Temporal Dependencies

In the usual case of process handoffs that we observed, the site receiving the handoff can do little until it receives the work product from the previous stage. This sometimes has adverse effects, as we observed in Gamma. One effect is that the split in process ownership limits opportunities to compensate for delays. For example, if one group owns both coding and testing, and if a difficulty with a particular code module arises, then some testing of other modules and combinations of modules can generally go forward.

However, if different groups own the two steps and the process calls for a handoff only when all modules are ready, there is less flexibility to minimize the effects of such delays.

Another effect occurs when there are different priorities among sites. In Gamma, a relatively large site often develops several versions of the product that vary by the revenue they will generate. Other, remote sites use the results from this larger site as input for their own development work. If a more profitable version experiences difficulty in development, the large site prioritizes based on revenue, and pulls resources off the other versions. When a remote site places a priority on a lower-revenue version of the product, then these changes at the large site will cause delays that cannot be overcome, and will force the smaller site to deliver late even when that site could have completed its process steps on time. When this occurred, it was a major source of frustration.

Handoff Points

The handoff point between two process steps owned by different groups that reside at different sites can be quite difficult unless there is a very clear agreement on what is to be handed off, how, when, and in precisely what condition. Gamma, for example, reported significant project management problems until a very clear common view was achieved of such phrases as "system verification is ready" and "100% of the tests are complete and 95% of the tests were successful."

The geographic separation of the sites involved in handoffs, and the attendant reduction in ad hoc communication, can also lead to troublesome misunderstandings. One site concluded that handoffs of any significant piece of work, design or code, do not go well unless a person goes with it to demo, explain, show what it is. Otherwise; for example, testers may claim that they can't even install the software, or something similar. This site found that a demo of the code gives both sides some comfort that there is something there. Things then go much better.

Coordination Mechanisms

Co-location helps with coordinating and efficiently using scarce resources such as test labs, with maintaining the right mix of expertise for each process step, and for coordinating within-step dependencies. Cross-site work relies on clear, agreed-to specifications of handoff points, a stable plan that keeps everyone informed about the dates these points will be achieved, and a shared understanding of the overall development process.

CUSTOMIZATION

What Is It?

The customization model has one geographical site that owns the core code for the product. Other sites involved in the project make changes to the code base such as adding features and enhancements for a specific customer base. The core site handles changes that require non-market specific alterations. We did not see this model in the literature; but it is a hybrid composition, as we will point out, of component separation and process steps.

What Are The Benefits?

Closer to the Customer

We found three reasons to locate the customization effort close to its market. First, the telecommunications domain is replete with both international and country or region specific standards. Locating a development site within a region that has local variants or standards in place allows the corporation to find people with that expertise available for development.

Second, it gives the customers a local site to submit requests for fixes and enhancements. By locating a site at the country of sales, customers avoid having to manage significant time-zone differences and the distances that come with them. It makes the corporation locally accessible to them.

Finally, having a site near the customer, with access to the same data and telecommunication infrastructure, is a distinct advantage. The customization site is familiar with the infrastructure, and can often connect directly to the customer's equipment much more easily for purposes of installation, testing, and acceptance of the product.

Good Division of Labor for Code Ownership

In order to build a product that has a core of common functionality as well as custom features, the common and variable parts should be cleanly separated in the product architecture so that the core can be reused, with little or no change, for each custom version. Isolating the customizable parts is just good design.

This type of design also is appropriate for supporting the strong division of labor that is required for geographically separated work. The core site "owns" the relatively stable code of the unchanging core, while the customization sites "own" the custom code. Gamma, for example, characterized the relationship as the core site developing a toolkit, and the custom site as using the toolkit to generate a final product. Minor changes in the "toolkit" were often required for specific deliveries, but there was rarely any confusion about which site was responsible for what part of the development. In this respect, customization incorporates a version of the "component" model.

Moreover, customization as we saw it implemented required a process handoff between the core and customization site. It was important that both sites agreed on the state the product would be in at the handoff point, how much testing had been performed, and so on. It was also important for the core site to deliver on the date promised so the customization site could plan for the availability of resources. In this respect, it shares a similarity to the process handoff model.

What Are The Problems?

Trust

We found several trust issues with this model that probably emerged in part from the distance between the sites. One of the problems emerges when the core site questions whether the remote sites can handle the work assigned to them. In the case of Beta, the core site worried that the newer remote location could not make the required

changes to the product. In part, this arose because the remote developers had less background with Beta than those at the core location. This, and a lack of documentation, which we discuss later in this section, led to the remote developers calling people at the core with lots of questions. This may have exacerbated any sense that the lack of experience with Beta was creating problems.

Compatible Infrastructure Issues

Since the sites work on the same code, although the remote locations do not alter the core product, it is vital that all tools are commonly shared. Specifically, all sites need to share exactly the same tools in order to ensure that the software behaves in similar ways across the multiple locations. Differences in versions of compilers, or build tools, may prevent the code from running, or produce spurious behavior.

People on Beta talked to us about preventing this kind of incompatibility by bringing people from the remote site to the core site and training them about the tools used in developing their product. Visits from the remote sites to the core site on Beta introduced new developers to Beta so that they could carry on developing the product when they returned home. However, simultaneously, it introduced these developers to the development infrastructure used to construct Beta, which they could then replicate remotely. This was particularly true of the testing environment, which Beta management had decided to replicate at both sites. Both the core and remote testing teams were going to great lengths to ensure that they replicated the environment precisely to avoid producing different behaviors at the different sites.

Coordinating Processes

Following on from the compatible infrastructure since the different locations work on the same code it helps that they follow the same processes at each site. In Beta, one of the remote sites was not using the same processes as the core location. In addition to creating difficulties transitioning code from one site to another, it also led to feelings that the remote site was rejecting experience gained by core site developers.

Specifically, these developers had designed their processes because of many years of experience working on Beta. Developers at the core site wondered why the remote site was not interested in these empirically tried processes for supporting the development work. This difference in processes led to difficulties for the managers of Beta as well. The different ways of working give people opportunities to enforce their own ideas, works as long as nothing has to pass between the different locations. In this case, these became difficult, and usually required department heads to make a decision about how best to proceed.

A Lack of Documentation

When the first remote site joined Beta, its developers were new to this kind of product. In addition to being unused to the area of development they were now a part of, they were also unable to find much information about the product itself, due to a lack of relevant and up to date documentation. This left the remote developers in the position of relying on the core site.

Developers at the core site were aware of this problem. Fighting tight development schedules, their own priorities focused on product releases rather than rewriting and updating existing Beta documentation. They were the ones who told us that when new developers at the central site join, these employees have questions, but due to the proximity can spread their queries out among the people at that site.

Their remote colleagues did not have this option. As the developers at the core site told us, people at the remote site tended to find one person at the core site and then ask all the questions that they had. This had the effect of slowing down some of the core developers seriously, because they were in continuous demand from people at the remote location.

Coordination Mechanisms

Co-location in the customization model supports coordinating with customers, and helps manage the core development interdependencies, as well as any interdependencies introduced specifically in the requirements for a given customer. Cross-site coordination requires a clean split in the product structure between the core and the custom parts, as well as a common understanding of the handoff between core and customization process steps, and a reliable plan specifying when the handoff will occur. Good documentation of the core code would be enormously helpful to the customizers.

PROBLEMS IRRESPECTIVE OF DOMINANT FORM OF ORGANIZING

During the course of our data analysis, we found a number of difficulties coordinating multiple site work. Some of these difficulties are well-known features of working internationally such as language and cultural differences, and managing multiple time zones. However, we also found hurdles to multiple site development work that did not seem to be any easier to resolve whatever model was used to organize the work. In this section, we briefly describe these issues: distribution of project "mass" and locating expertise.

Distribution of Project "Mass"

Independent of these models, we saw a number of consequences of a very unequal distribution of an organization between a large hub, or center, and one or more smaller satellites. This unequal distribution seems to have certain consequences, regardless of how the satellites relate to the hub.

We're constantly surprised

The central site was in all cases where the weight of decision-making authority arose. There are inevitably side conversations, hall talk, "meetings over the water cooler," and so on, where early notice of current thinking on technical questions or management decisions is disseminated. Before formal decisions are made, most people are aware of what is likely to happen, or at least what is being considered. They have an opportunity to prepare long before anything is formalized. They have

opportunities to make their opinions heard through a variety of informal channels.

For satellite sites, on the other hand, it is difficult not to be constantly surprised. Not having access to the corridor conversations, people at remote sites may have no clue about what is happening until a decision has formally been made.

Potentially serious problems flow from this. For one, decisions that seem relatively unimportant to the central site may affect the satellite in significant ways, simply because the issues are not obvious to the "center." Even when there is no single "killer" consequence of a decision, the cumulative effect of many surprises can be substantial. As one manager of a satellite site remarked, it is as if you are "fighting upstream instead of going with the flow."

The only solution that seemed effective was simply to invest more time and effort in travel, communication, and information gathering. The belief at remote sites was that the burden was almost entirely on them to keep in touch — the central site doesn't really understand the problem.

Out of Sight, Out of Mind

The general belief is that those at other sites are much less responsive to coworkers who reside at a different site. They are much less likely to provide information quickly, to follow up on requests, and, generally, to consider concerns of the other site. Meeting face to face seems to help; several respondents noticed a significant change for the better after they had actually met the person at the other site. Nevertheless, our respondents believed that they still did not get the same level of attention they received from co-located coworkers.

Finding experts

The challenge of locating experts is a well-known problem. It is true within R&D work as studies have discovered [2, 15]. We found that expertise was always missing in multiple site projects, because some knowledge was always remote. Further, we noticed that it seemed particularly acute during times when new people were introduced to the projects although it did not disappear as time went on.

However, what we also noticed was that the kinds of expertise missing changed as the dominant form of organizing varied. To summarize what drives the search for experts at remote sites:

- Functional Area: lack of expertise of distant systems areas,
- Product Structure: lack of understanding of the internals of components built at remote sites,
- Process: lack of knowledge about what happens during other processes,
- Customization: lack of knowledge of core or a lack of knowledge about how the core is customized depending on which site is involved.

Again, the most common workaround was to use one's own personal networks, generally acquired and nurtured by travel and face to face time. Escalating to a project manager was also a common, and reasonably effective practice. However, we want to stress how non trivial this problem can be for R&D work, when it may involve multiple countries, many people, and substantially slow down product release schedules.

One particularly interesting example of the lengths to which developers sometimes needed to go to find and use expertise at other sites is provided by an episode in which problems arose with a large, complex component installed in a network in Mexico. The problems were referred to a development site in the UK, but it was not clear at first where the problem was, so experts at several sites began a daily conference call to check progress. The first hour was concerned with checking status, about this particular problem and in general, and the second hour was devoted to reviewing what had been tried and what should be done next to find the problem. There was considerable urgency, since the problem was affecting a customer.

After the first few days, the conference call began to stretch out to about 6 hours a day, as people stayed on the line while actually running various tests and pooling their expertise to interpret the results. As theories about the source of the problem evolved, different sites were brought into the call. At various times, people from the US, Belgium, and Poland became involved, as well as representatives from other vendors and the customer. Documents were exchanged via the web, and e-mail attachments. The activity on the calls modulated between foreground conversations about the problem to listening in the background to relatively sparse comments as various members of the distributed team worked on their part of the problem relatively independently.

This way of working was successful in the end, but rather awkward. It is also worth noting that respondents mentioned that the call was made possible by personal relationships that were developed earlier in face to face meetings. The UK staff knew whom to call, and they believed that the remote participants were much more receptive because of the face to face history.

CONCLUSIONS

In this paper, we introduced four models for coordinating R&D work across multiple sites: functional areas of expertise, product structure, process steps, and customization. For each model, we described the benefits and difficulties we identified during our studies of groups using these ways of organizing work. We also described two difficulties that seem to reoccur regardless of the model used: consequences of unequal distribution of project mass, and finding expertise.

Choosing the right model of distributing R&D work across multiple sites depends on what coordination problem is being solved. If circumstances allow — i.e., those responsible have a choice about how to arrange their work — then selection of the appropriate division of labor should be driven by the hardest coordination problem in the project. Clearly, as we have argued, no model completely resolves all the coordination work needed on a project. However, coordination mechanisms such as

interface specifications, development processes, reliable plans, standards and protocols for product design, can be used, in lieu of some informal communication, to reduce cross-site dependencies.

In this paper, we have used real projects to illustrate the benefits and problems of the models we have proposed. Large projects may use more than one model, especially if you consider large subsystems of the product as separate entities, with their own multiple site coordination issues. In future work we hope to better understand how these models and the underlying coordination mechanisms can be composed to support coordination of very large product developments.

REFERENCES

1. Abel, M.J., Experiences in an Exploratory Distributed Organization, in *Intellectual Teamwork: Social Foundations of Cooperative Work*, J. Galegher, R.E. Kraut, and C. Egido, Editors. 1990, Lawrence Erlbaum Associates: Hillsdale, New Jersey. p. 489-510.
2. Allen, T.J., Managing the Flow of Technology. 1977, Cambridge, MA: MIT Press.
3. Allen, T.J. and O. Hauptman. The Influence of Communication Technologies on Organisational Structure: A Conceptual Model for Future Research. *Communication Research* 14, 5, 1987, 575-587.
4. Cairncross, F., The Death of Distance: How the Communications Revolution Will Change Our Lives. 1997, Cambridge, MA: Harvard Business School.
5. Carmel, E., Global Software Teams. 1999, Upper Saddle River, NJ: Prentice-Hall.
6. Conway, M.E. How Do Committees Invent? *Datamation* 14, 4, 1968, 28-31.
7. Curtis, B., H. Krasner, and N. Iscoe. A Field Study of the Software Design Process for Large Systems. *Communications of the ACM* 31, 11, 1988, 1268-1287.
8. Dourish, P. and S. Bly. Portholes: Supporting Awareness in a Distributed Work Group. in *ACM CHI'92 Conference on Human Factors in Computing Systems*. 1992. Monterey, CA.: ACM Press 541-547.
9. Fish, R.S. Cruiser: A multi-media system for social browsing. *The ACM SIGGRAPH Video Review Supplement to Computer Graphics* 45, 6, 1989.
10. Grinter, R.E. From Workplace to Development: What Have We Learned So Far And Where Do We Go? in *International ACM SIGGROUP Conference on Supporting Group Work GROUP '97*. 1997. Phoenix, AZ: ACM Press 231-240.
11. Herbsleb, J.D. and R.E. Grinter. Conceptual Simplicity Meets Organizational Complexity: Case Study of a Corporate Metrics Program. in *20th International Conference on Software Engineering (ICSE '98)*. 1998. Kyoto, Japan: IEEE Press 271-280.
12. Herbsleb, J.D. and R.E. Grinter. Splitting the Organization and Integrating the Code: Conway's Law Revisited. in *21st International Conference on Software Engineering*. 1999. Los Angeles, CA: 85-95.
13. Kraut, R.E., C. Egido, and J. Galegher, Patterns of contact and communication in scientific research collaborations, in *Intellectual Teamwork: Social Foundations of Cooperative Work*, J. Galegher, R.E. Kraut, and C. Egido, Editors. 1990, Lawrence Erlbaum Associates: Hillsdale, New Jersey. p. 149-172.
14. Kraut, R.E. and L.A. Streeter. Coordination in Software Development. *Communications of the ACM* 38, 3, 1995, 69-81.
15. McDonald, D.W. and M.S. Ackerman. Just Talk to Me: A Field Study of Expertise Location. in *ACM Conference on Computer Supported Cooperative Work (CSCW '98)*. 1998. Seattle, Washington: ACM Press 315-324.
16. Olson, J.S. and S. Teasley. Groupware in the Wild: Lessons Learned from a Year of Virtual Collocation. in *ACM Conference on Computer Supported Cooperative Work CSCW '96*. 1996. Cambridge, MA: New York, N.Y.: ACM Press 419-427.
17. Parnas, D.L. On the Criteria to be Used in Decomposing Systems into Modules. *Communications of the ACM* 15, 12, 1972, 1053-1058.
18. Perry, D.E. The Inscape Environment. in *11th International Conference on Software Engineering*. 1989. Pittsburgh, PA: IEEE Press: Los Alamitos, CA 2-12.
19. Perry, D.E., N.A. Staudenmayer, and L.G. Votta. People, Organizations, and Process Improvement. *IEEE Software* 11, 4, 1994, 36-45.
20. Schmidt, K. Of Maps and Scripts: The Status of Formal Constructs in Cooperative Work. in *International ACM SIGGROUP Conference on Supporting Group Work GROUP '97*. 1997. Phoenix, AZ: ACM Press 138-147.
21. Schmidt, K. and L. Bannon. Taking CSCW Seriously: Supporting Articulation Work. *Computer Supported Cooperative Work (CSCW): An International Journal* 1, 1-2, 1992, 7-40.
22. Suchman, L., Plans and Situated Actions: The Problem of Human-Machine Communication. 1987, Cambridge, UK: Cambridge University Press.