Global Software Development at Siemens: Experience from Nine Projects

James D. Herbsleb
Carnegie Mellon University
School of Computer Science
Pittsburgh, PA, USA 15213
+1 412 268 8933
jdh@cs.cmu.edu

Daniel J. Paulish, Matthew Bass
Siemens Corporate Research
755 College Road East
Princeton, NJ, USA 08540
+1 609 734 6500
daniel.paulish,Matthew.Bass@siemens.com

ABSTRACT

We report on the experiences of Siemens Corporation in nine globally-distributed software development projects. These projects represent a range of collaboration models, from co-development to outsourcing of components to outsourcing the software for an entire project. We report experience and lessons in issues of project management, division of labor, ongoing coordination of technical work, and communication. We include lessons learned, and conclude the paper with suggestions about important open research issues in this area.

Categories and Subject Descriptors

D.2.9 [**Management**]: Life cycle, Productivity, Programming teams.

General Terms

Management, Design, Economics.

Keywords

Geographically distributed development, global development, multi-site development, outsourcing.

1. INTRODUCTION

Geographically-distributed development presents enormous promise and enormous challenges. Numerous pressures have converged to vastly increase the extent of multi-site and outsourced software development projects [1]. These pressures include cost factors, increased capacity, and tailoring of products for particular geographically-defined markets. Geographic and organizational decisions about software development are typically made in the context of supply chain management [4].

ICSE'05, May 15–21, 2005, St. Louis, Missouri, USA.

Previous research has identified a number of important problems that must be overcome in multi-site and outsourced software development. The most pervasive problem seems to be the greatly reduced communication in multi-site projects as compared to single-site projects [10]. The technical work needed for a given project does not change just because the work is done by individuals separated by distance. Rather, distance reveals, by its absence and the resulting disruption, the subtle role that frequent planned and unplanned communication play in coordinating the work of software projects [6, 12].

In this paper, we present an experience report capturing the results of a multiple-case study of 9 software development projects in a large, geographically-distributed corporation. We examine both multi-site and outsourced projects, and include some that were judged successful and others that were not. Our purpose is to provide a deeper understanding of the ways in which software projects can coordinate their work. We conclude with lessons learned, and also identify research questions we hope to see addressed in future work.

2. EMPIRICAL METHODS

Our primary data collection technique was semi-structured interviews. In the remainder of this section, we describe the selection of interviewees, the interview process, the projects, and our analysis techniques.

2.1 Interviewees

We conducted 18 interviews at three Siemens sites. In order to ensure that we had several relevant perspectives represented among the interviewees, we included eleven people who were assigned project management or technical leadership responsibilities, five who had various middle management positions, and two with executive or senior management roles. Included among the interviewees was one who had recently transferred from a European site, and whose experience of a multi-site project was primarily from this perspective. We also interviewed one project manager who was nearing the end of a stay of several months at an Indian site.

2.2 Interview Process

All interviews but one were one hour in length, and conducted individually, face to face. They were semi-structured, meaning that the interviewer had a list of topics to be covered, but did not use a verbatim script. The questions were open-ended, giving the interviewees an opportunity to report what they had observed and experienced. The one exception to this face to face procedure was an interviewee who was located in India. Because of technical difficulties, after a brief phone conversation, the interview was conducted via e-mail. All other interviews were tape recorded, and the interviewer took detailed notes during the interview itself.

Interview topics included:

- role and responsibilities, project descriptions
- how the development work was divided among sites;
- how the work was managed;
- how the sites were kept in synch;
- processes and tools;
- cross-site relationships;
- communication practices; and
- problems, issues, and best practices.

2.3 Projects

The interviewees were able to report their direct observations on nine projects. Most of the interviewees had experience on two or three projects, so for all but two of these projects, we had at least two sources of information. In the remainder of this section, we provide a brief description of each project. In order to preserve confidentiality, we do not use the real project names. Our descriptions are necessarily brief, both because of space constraints and confidentiality issues. Our main objective is to try to be as clear as we can about the extent and variety of experiences on which this report is based.

Project **alpha** was a co-development effort between two sites, one in the US and one in Japan, involving two different companies. Neither site had complete control over the development; functionality was determined in a process of negotiation. The project developed basic, "foundational" embedded software for a new hardware product built by Siemens. The collaboration was undertaken because both companies marketed the Siemens product, and neither had the development capacity to produce the needed software in the desired time frame. While the companies would eventually become competitors, to some extent, and wanted eventually to build software features that would distinguish their version of the product, this project focused on basic software that was required to make the product function. The Siemens site focused on developing the code that was "closest" to the hardware, since they had greater expertise in this area. The Japanese site focused on building functionality on top of this. Siemens staff peaked at about 55 people, the Japanese site peaked at about 12.

Project **beta** created case management software for a particular type of medical case. It included a number of different "workspaces," including one for physician planning, one for therapy planning, and one that provided simulations based on patient data. An Indian firm was engaged to actually write the software because of their specific expertise with the Siemens infrastructure software, acquired while working with other Siemens divisions. Program management was done at Siemens, who also arranged for all subject matter expertise, and did the final build.

Gamma was a project to design and build a new graphical user interface (GUI) for a medical application. Siemens did the concept, most of the architecture work, high level design, and graphical design. Detailed design and coding of components was outsourced to an organization in India. In parallel with this implementation, Siemens designed UI screens that used the components. The US site had a maximum of 9 people, while the Indian site peaked at 5 staff.

The **delta** project created a development environment for use by Siemens product groups and Siemens customers. It supported the writing of scripts for new analyses of medical data, and provided a framework that applications could be plugged into. The environment was a customization of a commercially-acquired development environment. The project also included providing a run-time component for the applications developed in the environment.

The project included six Siemens staff, and three developers in an Indian organization. The Siemens staff made changes to the run-time component and the environment. The Indian staff wrote a wrapper for the run-time component so it could be integrated into the development environment.

Epsilon was an integration project, in which one specialized type of building management system developed by one Siemens division was integrated into a suite of building management software built by another Siemens division. The work focused primarily on a foreign system interface (FSI) that was used by the application suite to interact with the specialized system. The "suite" site developed a protocol stack that would communicate through the FSI, while the "specialized system" site had four staff augmenting the FSI in order to support all of the desired functionality.

Project **zeta** was a viewer and editor that provided about a dozen applications for a variety of views of building systems, and which allowed the user to alter configurations.

The US site built a platform on which the applications would run, as well as building some applications that were primarily targeted at the US market. A group in Switzerland built applications (to run on the US-built platform) that were targeted primarily at European markets. The project involved about 30 US staff, and 17 Swiss staff.

Eta was a project in which GUIs for three existing products, originally written in Visual Basic, were rewritten in Java. The plan was for all of the software to be written entirely by an Indian organization, with the US organization providing the Visual Basic code, documentation, training, and expertise as needed to answer questions, etc. There was a project manager and technical lead at the US site, and five developers and a project manager at the Indian site.

The **theta** and **iota** projects are parts of one very large, multi-year project involving about 8 sites around the world. Theta and iota are relatively separable parts of this very large project, and are largely unrelated to each other. They were designed and managed fairly independently, except for a high-level architecture. For these reasons, we consider them as separate projects.

Theta is a project to build a network component that collects data and assists with maintenance of many functions of the network. The US site has one project manager, and all design and development work is done by another organization in Eastern Europe. The US project manager also performs some technical oversight, e.g., by participating in all reviews, and building analysis models. There are 9-10 staff assigned to the project at the eastern European site.

The **iota** project designed and built tools to help with system layout and ordering, helping the user go from system requirements to preliminary layout to ordering components. It had substantial interfaces with other systems, e.g., for design and inventory. The US site has 5 people involved in technical leadership and project management roles, and was responsible for the initial architecture and specifications, as well as oversight responsibilities. The architecture and specifications were handed off to an Indian site, which did about 80% of the coding. The US site retained a few modules where they had more expertise. The Indian site had a project manager, technical lead, and eleven developers.

2.4 Data Analysis

Data analysis followed standard protocols for the handling of qualitative data (e.g., [13]). Interview preparation involved examination of previous literature, and started with protocols that had been used in previous research on globally-distributed development [9]. We also examined available artifacts, such as product materials and design documents. After the interviews were complete, the interview notes were examined for common themes, and tentative conclusions were cross-checked with other interview material. Additional questions were sent by e-mail to the interviewees, who provided clarification and additional information. The interview findings were organized with respect to the emerging themes.

3. RESULTS

3.1 Management and Control

3.1.1 Business and personal incentives

Among the early, key decisions in creating a multi-site, multi-organizational arrangement are those that affect the incentives of the participants. At the business level, collaborating organizations may also be competitors in some areas. This happened in several ways that we observed, for example, when integrating a product into a suite, the suite was viewed as something of a competitor by the unit that developed the standalone product. It was not always clear that the revenue consequences would be favorable for this organization or the people involved in the development. In other cases, a cooperative development may gradually become more competitive over time when the organizations are selling products independently. In such cases, it may not be clear to all parties that cooperation is in their best interests.

Similar tensions were perhaps even more troublesome where developers were concerned that multi-site arrangements are a prelude to job cuts. Cooperation and open communication in such cases may be seen as making oneself more easily replaced.

Misalignment of interests can also divide along lines of the product structure. We observed several cases where there was considerable ambiguity about where particular functionality should reside. In one case, developers of different components both wanted to lay claim to the functionality because it was particularly critical or "glamorous." In another case, developers wanted to avoid it because it was potentially troublesome. Decision-making occasionally had a negative impact on the product structure, since decisions were made for personal or political reasons, rather than technical considerations.

3.1.2 Project planning and tracking

We saw several cases in which differences in project planning and tracking discipline caused substantial problems. In some cases, it seemed that project management inexperience on the part of the service provider resulted in a plan that was late, lacking in detail, and ultimately unrealistic. While this could be a problem in any context, it seems much harder to address across organizational and geographic boundaries. There is little opportunity to interact informally, to express doubts, to assist with revision, and generate a better plan. The arrangements are confined to more formal forms of communication, typically comments on documents. This is

a very impoverished and slow-paced form of communication. One contracting organization place a person at the service provider site for an extended time, just in the hope that they could gain insight into project status by talking to the staff.

In addition, in a fixed-price contract, telling a service provider that a plan is unrealistic amounts to telling them they should charge more. This sort of communication may be optimal in an established, ongoing relationship, but it is likely to seem inappropriate before such a relationship is established.

A related issue arose in two projects where the contracting organization was unhappy with the skill and experience level of the developers working on the project. Unlike developers in one's own organization, one often has little control over who gets assigned to the project, other than requiring certain skills. And even if one has some choice, as one manager pointed out, it's very difficult to judge the quality and skill of technical staff at another site.

Inability to gain visibility into project status was a major source of frustration on two of the projects. Project leads and managers were very concerned that things would not be delivered in a timely way, and in fact the concerns were justified. In both cases, the projects represented initial collaborations with the distant sites.

Tracking information was particularly important in projects where code was actually being developed at both sites and needed to be integrated on a phased schedule. Late deliveries, especially when unanticipated, resulted in developers at the other site with little to do, and caused considerable frustration. Again, since information has relatively few pathways to cross sites, without explicit communication it is difficult to know what to expect. On the other hand, the lack of detailed tracking data was not always a problem, especially on small projects where there was frequent communication and good relationships.

3.1.3 Process compatibility

Surprisingly, process compatibility did not seem to be a major issue in any of the projects we looked at, in spite of the fact that in many cases the projects were the first instance of two sites working together. As one manager said, they didn't need any "big process synch-up." In many cases, the service providing organization adjusted their process by adopting the templates of the contracting organization, particularly for medical systems where there were often certification concerns.

The projects were not completely free of concerns about process compatibility, however. One complaint voiced by one manager was that time was wasted translating information into the forms required by two different quality control organizations, and there could have been significant savings of effort had processes been shared. There were also occasional comments from several projects about some initial difficulties in communication that turned out to have their origin in different processes with different terminology.

There was also occasional confusion about roles, such as project manager, that had very different meaning in different organizations. In some organizations, for example, project managers had hands-on responsibility for keeping the project on track, making decisions about resources, and exerting substantial control over the project. In one of the service provider organizations, project managers were assigned several projects, and had primarily an information-gathering and reporting function, with no substantial line management responsibilities. Such differences sometimes caused confusion and frustration until project managers at the contracting organization figured out they needed to talk to technical leads, not project managers, at the service provider organization.

In spite of these issues with process compatibility, it was not considered by any interviewee to be among the major problems. These issues were overcome fairly quickly, as people learned a bit about how things operate "over there." This was facilitated in many cases by frequent cross-site visits, as we discuss below.

Related to process incompatibility is what could be called issues of engineering culture or style. There was a glaring difference, on several projects, between US and European engineers on how they approached engineering problems. The tendency in the European organizations was to do much more up front design work, and begin implementation only when the design work had reached a level of completion acceptable to them. The US teams, in contrast, wanted to begin implementation much sooner. Several interviewees reported that this difference created some frustrations, with the Europeans viewing the American teams as leaping ahead injudiciously, and with the Americans feeling the Europeans would never get around to implementation. As one interviewee put it, however, it was "not really a big deal, but a definite difference." This was pretty typical of the interviewee comments, everyone reporting that after some initial frustration and confusion, both sides were able to understand and adjust to the differences.

3.1.4 Process maturity

The experiences of the interviewees with respect to process maturity were fairly complex. One contracting organization boasted a very high process maturity level, but the experiences with teams from that organization were not uniformly satisfactory. In fact, some teams seemed to have extremely immature processes, to the surprise and disappointment of those in the contracting organization.

Process maturity around the planning and tracking of projects created serious problems in projects that were not

going particularly well. Where the service provider was meeting deadlines, had reasonable levels of quality, and informed the contracting organization of problems early, the issue of project management visibility did not arise. It is not clear what level of detail could or would have been provided by the service provider if asked. On the other hand, where deliveries were late, or of poor quality (these tended to co-occur in our sample of projects), the contracting organization was generally unable to get any detailed information about project status, and was quite frustrated about this. The interviewees had the strong impression that this information was not being withheld, rather it simply seemed not to exist.

One interviewee summarized fairly well the impressions of several interviewees on questions of process, lauding a service provider for being "process-oriented but flexible." He appreciated that he could always understand what the service provider was doing, and where the project stood, yet the organization was willing to make minor exceptions as needed to streamline the process. In a similar vein, one interviewee stressed the good results of a policy of allowing technical staff to make decisions, in direct consultation with technical staff in the service provider organization, on all matters not affecting schedule or functionality. Pushing decision-making down to the lowest appropriate level appeared to save considerable time and effort.

3.2 Development Environment and Tools

3.2.1 Development environment

In those projects where actual code development was being done at multiple sites, most used commercial tools that provided some support for a distributed code base and change management system. One project initially set up separate code branches at each site, but concluded this was a serious mistake when integration became extremely difficult. They quickly moved over to a single branch for both sites. Most projects also had build capabilities at both sites, and the ones that didn't regarded this as something they definitely wanted in the future.

One significant problem that one project encountered stemmed from the fact that one site did not have a complete hardware configuration, including hardware, firmware, connectivity, etc., that would allow them to duplicate errors discovered at the other site. They had been reluctant to spend the money, and eventually concluded this was a false economy.

In answer to a question about the most important lesson learned in distributed development environments, one project manager who was located in India responded, "Connection speed, connection speed, connection speed!" and went on to discuss how this slowed the work. None of the European or US interviewees mentioned this, and, interestingly, none seemed to have any idea it was such a big problem for their Indian colleagues.

3.2.2 Collaboration technology

All teams made use of basic collaboration technology such as telephone and e-mail, and many made use of application sharing to share documents and presentation slides, and even to show a service provider team live demonstrations of their code running on laboratory machines in the contracting organization's location. One team also used a change management system as an asynchronous communication medium, to pose questions that others could answer. This seemed to work particularly well to make it easier for people to ask questions who otherwise seemed reluctant to do so, and the perception was that it drastically reduced the amount of e-mail.

While it may not be a conventional form of collaboration technology, two projects set up a photo gallery to help people get a sense of those they were collaborating with. In one project, these photo galleries were printed and put up in almost every cubicle, while in the other a photo-annotated organization chart was created and posted. Both projects thought that having these photos continuously made a substantial difference in creating a sense that "it really is a person over there."

3.3 Clarity About Who Does What Where

Communicating clearly to a service provider organization exactly what is desired was a difficult problem, resolved with varying degrees of success, and through various means, in the projects we observed.

3.3.1 Communicating what is desired.

Different projects used different artifacts and different communication regimes in order to convey what was desired. In the Eta project, the idea of which was to rewrite a user interface in a different language, the contracting organization gave the service provider the original source code, the operator's manual, a configuration manual, and training on the product. In addition, they had three technical leads available for answering questions both at the outset and as the work progressed. The service provider organization sent a project manager, two engineers, and a quality control manager to the contracting organization's site for several weeks. The original idea was to have the project manager stay at the site for the duration of the project, but this did not work out for reasons largely unrelated to the project. Beyond this, there was no formal communication regimen established beyond deliveries and reports in the contract. There was, however, significant informal communication, largely through e-mail.

This project was clearly the least successful of those we studied, and problems in communicating what was desired was identified as a primary reason for this. The code, documentation, and training were not sufficient. The

developers were new to the domain, and relatively inexperienced generally, and had difficulty grasping the functionality well enough to reproduce the UI. None of the documentation was specifically designed to convey this information.

Fairly early on, the parties agreed that a functional specification, which did not exist, would be very helpful in providing needed guidance. The initial plan was for the service provider organization to write the functional specification, but after several late, incomplete, and defect-ridden attempts, the contracting organization took over this task, and assigned six engineers full time for three weeks to create the document. The resources were pulled from development work on the next release, at considerable cost to that project.

The communication regimen presented another problem. The service provider organization wanted to manage communication very tightly, funneling it all through their project manager. In order to ask a technical question or to get detailed status information, all communication had to go through this bottleneck, which introduced much delay, and proved very frustrating for the contracting organization.

In stark contrast to this experience, the Theta project used UML analysis models -- built precisely to serve this purpose -- as the primary artifact in communicating with the contracting organization. The communication regime included weekly status calls, and technical consultation about every other day. Someone, usually the US-based project manager, traveled to the other site approximately every other month. There have been relatively few difficulties in conveying what was to be built, and the deliveries have been of good quality. This project had many advantages, however, including service provider domain knowledge and small project size. One interviewee attributed the success primarily to the use of the analysis models, which eliminated much of the ambiguity that would have been unavoidable in natural language.

The Gamma project made use of an architecture and high-level design, graphical designs as the primary artifacts for communicating a new graphical interface to be built by the service provider. In addition, they required that it be built as extensions to a particular set of foundation classes. The communication regime included approximately weekly reviews as the service provider created the detailed design. These reviews proved fairly difficult, and much revision was necessary. The US-based technical lead reported that in retrospect, he should have specified a bit more detail, down to indicating which methods were to be used in the implementation. Overall, however, the project was considered successful.

Two projects, Epsilon and Zeta, used evolving interface specifications as a primary artifact. Epsilon used a foreign system interface that was being augmented to support additional functionality as the project progressed, while Zeta adopted a layered architecture in which one site designed and built a platform that provided services to applications designed and built at another site. Both of these projects proved very difficult, and understanding exactly what was to be built was a major difficulty. In each case, there were issues about where functionality should reside, what form the interface should take, and about the timing of the development so that testing could be done in a timely way. The communication regime in Epsilon included weekly status meetings and some informal communication, although one site complained bitterly about unresponsiveness on the part of the other. In Zeta, communication included weekly status meetings and occasional lateral communication among developers. One of the two projects was eventually successful, while the other was not (for a variety of reasons). But in both cases, communication about what to build, and how the pieces at the different sites would work together, was considered to be very difficult.

3.3.2 Communication about requirements

Other important issues also emerged from our data. One was that in the context of a service provider arrangement, particularly with service provider organizations that have relatively little domain expertise, there can be frequent disagreements about whether a particular communication about the requirements is correctly characterized as a clarification (hence, describes work covered by the original contract) or a change (requiring renegotiation of dates and remuneration). There is always the opportunity for some gamesmanship on this question, but there is also considerable room for misunderstanding.

If requirements are not clearly conveyed, issues may not begin to surface until integration, as the sites realized that their interpretations differed. In Alpha, for example, the Japanese and US markets differed considerably in the details of how doctors interact with patients, and these differences led to substantial differences in interpreting requirements about what information should be available to users. It did not occur to them until fairly late in the game that their domain expertise was leading them to different interpretations of the product requirements.

The beta project also did not recognize significant mismatches until relatively late in the game. The service provider had considerable expertise in an important infrastructure technology, but, as the US-based project manger said, "We missed that it had to live in a bigger environment." Since this had not been adequately communicated, it was difficult to integrate the application into real settings because it had not been built with this in mind.

3.3.3 Architecture design

Finally, we note that architecture design across sites seemed to have very different characteristics than the co-development and service provider relationships that were the focus of most of our data collection. In our interview with the technical lead and overall architect of a very large project, there were relatively few artifacts that could be usefully shared across sites. The architect summarized his experience: "e-mail and documents are no good for this kind of work."

Rather than trying to work in more distributed fashion, the multi-site architecture work occurred mostly in a series of week-long workshops, held about once every other month, at a single site. Each was focused on a particular topic: the first was on business goals, and other topics included sunny day scenarios, error handling, a key protocol, and visualization. A modeling tool was used extensively to capture the content developed at the workshop. Each workshop was structured so that all the important decisions were made by the last day of the workshop, with all the key players physically present. This was regarded as critical, since reaching a decision using e-mail or teleconferencing is notoriously difficult.

3.4 Communication

3.4.1 Face to face communication

One of the most consistent comments made spontaneously by almost every interviewee concerned the importance of meeting people face to face and spending some time with them. A typical remark from one technical lead: "I can't emphasize too much how important this is -- something about understanding each other's worlds -- it makes things much easier later on." Another typical comment is, "don't rely on e-mail, communicate face to face as much as possible."

One of the views consistently expressed is that face to face contact was essential in avoiding and overcoming cross-cultural misunderstandings, and for developing relationships and trust with people at other sites. Two interviewees emphasized that travel was essential, even when there wasn't a specific immediate need. As one project manager put it, "drink a few beers together -- it makes a big difference." Relationships built in this way were seen as a very valuable resource, especially when problems arose.

Face to face communication across widely-separated sites is extremely expensive, of course, in both time and money. Nevertheless, every project decided to have at least one person travel at least every other month, and most funded much more travel than this. When issues arose, travel increased dramatically, sometimes involving either a member of the contracting or service provider organization relocating for extended periods of week or months. One manager, who had just had an unsuccessful experience, said that in the future, for all multi-site projects he would have someone from his own organization, who understands the business and the requirements, relocate to the other site "for the duration of the project."

From the other side of the relationship, a project manager who relocated from a contracting organization in the US to a service provider in India for an extended period, said in an e-mail to the authors, "I have an uphill task of creating an understanding and appreciation for the unique conditions under which the supplier operates." He pointed out that the contracting organization realizes that new team members in their own organization need considerable support, but often don't realize the need is perhaps even greater when bringing service provider employees at a remote site into a project.

Another insight about face to face contact is that it provides by far the fastest "pace" of interaction, compared with other modes of communication. During final integration on one project, for example, the contracting organization decided that issues requiring the service provider's expertise were arising so often that they had to have someone on site to respond immediately, rather than waiting potentially a day for each one. This issue of the pace of different forms of interaction arose frequently, and a slow pace was often a source of frustration. Face to face was clearly superior in this regard, especially when work hours at different sites have little overlap.

3.4.2 Communication and culture

Subtle cultural differences often complicate communication, and can lead to frustration and misunderstandings. "Culture" has many meanings, of course, and it was often difficult for our interviewees to determine if the differences they pointed out were due to corporate, technical, or national culture, but all three seemed to play a role.

An example of corporate culture was seen in a co-development project between two US sites, one of which was recently acquired from another company. Staff at the new site tended not to respond quickly, or at all, to e-mails from their colleagues at the other site. This was seen as rude, and even obstructionist by two interviewees. One said that he realized they were understaffed, but "still, it violated etiquette." Another example of conflicting corporate cultures was reported by an interviewee who regarded his own organization as having very open communication, but was working with a service provider that apparently had a culture in which people were extremely reluctant to admit problems. The interviewee was very frustrated by his inability to get accurate information.

Differences in technical culture affected communication as well. In the larger project of which the Theta and Iota projects are parts, several companies were participating in technology development. The technical lead responsible

for designing the architecture noted that the different companies had different "philosophies" about moving material, but that these differences did not show up in the early discussions. The participants assumed their own interpretations for the vocabulary used in the high level technical discussions, but when they actually got down to the details, "everything exploded." They finally realized, as the technical lead put it, that "this isn't how we do things!" At the least, the difference in technical cultures postponed the discovery of differing assumptions, and at worst caused people to think they had been mislead.

National culture also appeared to pose something of a barrier to communication. In particular, many interviewees reported a difference in Asian culture, as compared to European and American culture, with respect to expressing agreement and disagreement, and asking questions. Acknowledgement was sometimes mistaken for agreement, and the Europeans and Americans thought they had a commitment, when their Asian colleagues were merely trying to be polite. One American interviewee also reported that he had worked out ways of trying to ensure that his Asian colleagues actually understood him, because he felt that they were reluctant to ask questions, at least as compared to his American colleagues. One interviewee stressed the importance of a culturally diverse team at each site "so that people are accustomed to being among and communicating with people who are different."

3.4.3 Time zones

Working across a large number of time zones was an enormous issue, of course. This makes it very difficult to schedule meetings, as every time is inconvenient for some one. In general, it seems that time zones were a particular problem when there was a need for fast-paced interactions. One interviewee noted that multiple time zones caused a particular problem for projects with lots of reviews, which in that organization were all performed synchronously. Other interviewees noted that multiple time zones made it particularly difficult to get information in order to fix bugs, both during integration and during post-release technical support. In both cases, it is frequently necessary to get information about how the code written at the other site works, and it was very difficult and time-consuming to acquire this information with e-mails and phone calls.

3.5 Strategic Considerations

Many of the experiences of our interviewees suggest that there are a number of important considerations that that span particular projects. All the interviewees who addressed this issue stressed the importance of establishing and maintaining long-term relationships with service provider organizations.

3.5.1 Domain knowledge

As one interviewee noted, it is important for the service provider to get to know the contracting organization's products, projects, and technology. We observed several projects in which the lack of such knowledge seemed to be a major cause of problems. Developing this sort of domain knowledge takes time. In two of the projects we observed, the contracting organization had delivered training of the sort that would be given new employees to members of the service provider organization, and found that this level of instruction was insufficient.

In project beta, on the other hand, domain knowledge about a particular proprietary infrastructure of the contracting organization was the primary reason for engaging the service provider, and key to their success. As this example suggests, selecting a service provider organization is a strategic decision, because the payoff may increase over time as the learning curve is overcome. There are clearly other important strategic considerations -- such as determining which technologies embody core competencies that should be kept in-house -- that are beyond the scope of this paper.

Several technical leads and managers mentioned the issue of maintaining the product or the part of the product developed by a service provider organization. Particularly where the service provider does all of the development work, the contracting organization may not have the technical expertise in the product to maintain it effectively. One manager suggested that at least one technical staff from the contracting organization should be involved in the details of the work, to form a core of technical expertise for in-house maintenance.

3.5.2 Establishing trust

Another important characteristic that takes time to establish is trust, which several interviewees identified as important. The contracting organization needed to trust that deliveries would be on time and up to quality standards, and that any problems would be communicated quickly. The service provider needed to trust the contracting organization to "be reasonable" about problems if they were communicated honestly.

Finally, establishing trust over time was judged by one executive as an important key to success because of the extreme difficulty of judging talent at a distance. When someone is on-site, with frequent interactions, one can much more quickly judge the person's capabilities and expertise in various areas.

4. DISCUSSION

4.1 Lessons Learned

While it is always difficult to boil the rich and varied experiences of many projects to a few key ideas, we attempt here to draw out the lessons that seem most general and

compelling. Several of these lessons, particularly about communication and environments, are similar to lessons found in validation [7] and maintenance [2] activities.

4.1.1 Communicating the work to be done

Communicating what is desired of the service provider is likely to be more difficult, often *much* more difficult, than anticipated. The contracting organization is so immersed in its own products and technology that it is hard to realize how much knowledge they take for granted, and how many subtleties will need to be made explicit and communicated.

Several things can be done to improve the situation. One is to do a better job of anticipating the extent of the problem if the service provider has not been engaged before. Does the service provider have domain expertise? How experienced are the developers in general? In order to avoid inflated claims, examine the resumes of developers who will be assigned to the project. Domain knowledge and experience will make it much easier to convey what is desired.

If there is relatively little domain knowledge, then be prepared to provide extremely detailed direction. Successful projects we observed provide such things as UML analysis models, or guidance about what foundation classes to use, and even what methods in those classes to modify. Documentation developed for other purposes is generally inadequate.

Finally, develop ways of testing the service provider's understanding. Especially when schedules and plans seem optimistic, dig into the details, make sure the service provider really understands what is required.

4.1.2 Project management

Be prepared to be intimately involved in project management of staff at the service provider organization. Either from the outset, or at the first sign of trouble, manage them as you manage your own staff, being aware of what they are doing on a day-to-day basis. Do not assume that the project management will be adequate until the service provider has a track record. Agree on the "management infrastructure" of detailed milestones, tracking metrics, and reporting up front.

4.1.3 Communication

It is very important that technical staff at all sites interact directly. They need to know, or be able to find out, who to contact about what, and there must be some shared expectations that attempts to communicate will be responded to quickly. Do not permit communication bottlenecks to develop, such as funneling all communication through a project manager. While a central contact may be useful until the technical staff learn "who's who" at the other sites, if it becomes a bottleneck, it will slow the work, perhaps dramatically.

Collaboration tools like chat and IM have proved useful in a variety of settings in software development [8] and elsewhere [3, 5, 11]. Such tools have properties very different from e-mail, including, importantly, a potentially faster pace of interaction as well as support for asynchronous multi-party conversations. Given the importance of these characteristics in the projects we examined, we think such tools hold considerable promise, and we particularly encourage projects to try out tools that support asynchronous collaboration (e.g., wikis, discussion boards) to help overcome time zone differences.

4.1.4 Travel

Spend the travel budget early. Once people have met, and worked together for a few days or weeks, everything else works better. Projects typically underestimate the need for travel, and the value of face-to-face contact. In order to start to overcome cultural differences, develop trust, and enhance all other means of communication, it is hard to overstate the importance of spending time at other sites.

4.1.5 Development environment

Try to set up a single "virtual site" to the extent it is possible. If code is being developed at multiple sites, use tools that allow you to maintain a single branch. Share a change management system, so both sides can get a detailed understanding of the number and the nature of outstanding problems. Have build capability at both sites, and do frequent, preferably daily, builds so that problems can be identified and solved quickly.

4.1.6 Communities of practice

Bringing a new employee "up to speed" is a difficult process [14] that even with considerable training seems to require much interaction with more seasoned employees in order to "learn the ropes" (see, e.g., [15]). The struggles of the technical staff of service providers to understand the business, applications, processes, and culture of the contracting organization are reminiscent of those that new employees experience, and overcome only through rich interactions with a community of practice, i.e., experienced co-workers who know how to deal with the messy reality of the job, including handling exceptions, applying rules to real cases, and interpreting the environment. Finding ways to establish distributed communities of practice should be a top priority. Where does a service provider employee go to get questions answered?

4.2 Dimensions of Coordination

It seems clear there is no single "right" model of successful collaboration across sites. We find it useful to think about the possibilities in terms of "dimensions" of coordination. There are many ways to coordinate work, including shared processes; shared, detailed, project management; modular product structure; shared past experience; background knowledge, e.g., of the domain; and finally, planned

communication regimes, including status meetings and technical reviews.

To the extent that these are effective, a project can avoid being overwhelmed by the need for unplanned communication. If unanticipated problems arise, the consequence is always a sudden need for unplanned communication in order to resolve issues and renegotiate the current arrangements. Since unplanned communication is the "coordination technique of last resort," one could think of the volume of unplanned communication as a measure of how well the other means of coordination are working. We would like to see research on how these "dimensions" of coordination play together, the extent they can be traded off against each other, and principled ways of understanding, for any given project, how to use them to best advantage.

5. ACKNOWLEDGEMENTS

This work was supported by the Siemens Software Initiative, which encourages best practice sharing among Siemens software engineers. We also gratefully acknowledge the support of the Software Industry Center at Carnegie Mellon University and its sponsors, especially the Alfred P. Sloan Foundation.

6. REFERENCES

1. Arora, A. and A. Gambardella, *The Globalization of the Software Industry: Perspectives and Opportunities for Developed and Developing Countries*. 2004, National Bureau of Economic Research: Washington, DC.
2. Bianchi, A., et al. *An Empirical Study of Distributed Software Maintenance*. in *International Conference on Software Maintenance*. 2002. Montreal, Canada.
3. Bradner, E., W.A. Kellogg, and T. Erickson. *The adoption and use of "Babble": A field studyof chat in the workplace*. in *European Conference on Computer-Supported Cooperative Work*. 1999. Copenhagen, Denmark: Kluwer.
4. Brereton, P., *The Software Customer/Supplier Relationship.* Communications of the ACM, 2004. **47**(2): p. 77-81.
5. Churchill, E.F. and S. Bly. *It's all in the words: Supporting work activities with lightweight tools*. in *GROUP '99*. 1999. Phoenix, AZ.
6. Damian, D.E. and D. Zowghi, *Requirements Engineering challenges in multi-site software development organizations.* Requirements Engineering Journal, 2003. **8**: p. 149-160.
7. Ebert, C., et al. *Improving validation activities in a global software development*. in *International Conference on Software Engineering*. 2001. Toronto, Canada.
8. Handel, M. and J.D. Herbsleb. *What is Chat Doing in the Workplace?* in *Conference on Computer-Supported Cooperative Work*. 2002. New Orleans, LA.
9. 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 (ICSE 99)*. 1999. Los Angeles, CA: ACM Press.
10. Herbsleb, J.D. and A. Mockus, *An Empirical Study of Speed and Communication in Globally-Distributed Software Development.* IEEE Transactions on Software Engineering, 2003. **29**(3): p. 1-14.
11. Isaacs, E., et al. *The character, functions, and styles of instant messaging in the workplace*. in *ACM conference on Computer supported cooperative work*. 2002. New Orleans, LA.
12. Kraut, R.E. and L.A. Streeter, *Coordination in Software Development.* Communications of the ACM, 1995. **38**(3): p. 69-81.
13. Miles, M.B. and A.M. Huberman, *Qualitative Data Analysis: An Expanded Sourcebook*. 2nd ed. 1994, Thousand Oaks, California: Sage Publications, Inc.
14. Sim, S.E. and R.C. Holt. *The ramp-up problem in software projects: a case study of how software immigrants naturalize*. in *International Conference on Software Engineering*. 1998. Kyoto, Japan.
15. Wenger, E., *Communities of practice: Learning, meaning, and identity*. 1998, London, UK: Cambridge University Press.