Please cite as: Boland D & Fitzgerald B (2004) Transitioning from a co-located to a globally-distributed software development Team: A Case Study at Analog Devices Inc, 3rd. International Workshop on Global Software Development, Edinburgh, Scotland, pp.4-7

Transitioning from a Co-Located to a Globally-Distributed Software Development Team : A Case Study at Analog Devices Inc.

David Boland
Analog Devices
david.boland@analog.com

Brian Fitzgerald
University of Limerick
bf@ul.ie

Abstract

Global software development has become an extremely important issue for organizations at present in the climate of increasing tendency towards globalization and global outsourcing. A number of studies have been conducted which have identified a set of problematic areas which are common across projects, including language and cultural differences, trust factors, communication across temporal and spatial distances, lack of shared contextual awareness. This study of global software development at Analog Devices Inc. (ADI) is especially noteworthy for a number of reasons. Firstly, the project has recently moved from a co-located to a globally-distributed one, and thus the team had already had experience of being co-located, a factor that has not typically been the case in the studies published to date where teams are being established who have not previously been co-located. Also, as language and cultural factors were not an issue, the study was able to focus on the problems of communication over temporal and spatial distances. The study discusses how ADI attempted to address these problems and identifies the initiatives that worked well, and, more importantly, those that did not work as well. Among the findings was the fact that trust, which had been very solidly established among team members during co-location, was significantly eroded as the project team was reconstituted on a distributed basis.

1. Introduction

There have been several documented studies on globally distributed software development teams [e.g. 1,2,3,5,7, 9,10]. A common feature in most of these studies, however, has been that the teams at the various development sites have had little or no previous experience with each other. Also, many of the case studies have involved very large development teams and substantial geographical and temporal distances (i.e. greater than 8 hours). This particular case study, however, was able to observe a very small development team (less than 20 developers), that had worked together for four years and were being redistributed into a global development team across two development sites; one in the United States and the other in Ireland. Many common global development problems including language and culture were not an issue and this allowed us to concentrate on how communication and temporal problems affected the group and how they attempted to overcome them.

The paper is structured as follows: The next section provides some background on the case study company, Analog Devices Inc. Following this, the procedures and processes that were established in the move from co-location to a distributed team are identified. The next section discusses the success of these procedures and processes, and also identified the problematic areas where these did not work as well. Finally, the conclusions and implications of the study are addressed.

2. The Company

Analog Devices Inc. (ADI) is a world-leading semiconductor company specializing in high-performance analog, mixed-signal and digital signal processing (DSP) integrated circuits (ICs). ADI currently has a worldwide workforce of approximately 8,600 employees, including 3,100 engineers. There are development/manufacturing facilities in the United States, Ireland, United Kingdom and the Philippines.

Analog Devices is one of the few semiconductor companies that have an internal division that provides automatic test equipment (ATE) for the ICs the company produces. Analog Devices' ATE division is called the Component Test Systems (CTS) division. The latest ATE platform at CTS has been in development since 1999 and for all that time the entire development team, both hardware and software engineers, have been co-located.

In 2003, it was decided to distribute some of the team members to the development facility at Limerick, Ireland. The primary purpose for the relocation was to ensure that CTS was better represented at the remote site. This would provide better support to the local customers and their concerns/issues would be more accurately relayed to CTS.

3. Creating a Globally Distributed Development Team

There are many problems to be addressed when establishing a globally distributed development team, including, for example, language and cultural differences, trust factors, communication across temporal and spatial distances, lack of shared contextual awareness [2, 4, 6, 8, 9]. CTS, however, believed that the creation of their team would be successful as some of these problems would not be an issue. The problems included:

1. **Language**. All the members of the team spoke English and used a common vocabulary for identifying specific hardware or software components. Therefore, the team should have no difficulty understanding each other.
2. **Culture**. Although not all members of the team were from the same geographical region, they had been working together for four years at the time of the move to a distributed team, and thus had developed their own 'CTS' culture. Unintentional rudeness, hostility or other communications issues should not be a problem.
3. **Trust**. The developers had established strong levels of trust between each other as a result of working together for a long time.

Therefore, CTS was able to concentrate on addressing the remaining global development problems of communication across temporal and spatial distance, and shared contextual awareness. The following procedures/processes were enacted to address these issues.

Single Software Manager

Due to the size of the development team, it was decided to continue with one software manager for all developers across all sites. The software manager is responsible for assigning tasks that will reduce cross-site dependencies especially with regard to expert dependencies (i.e. assign tasks to the particular subsystem expert directly or have experts and developers co-located).

Weekly Task Report

To facilitate the work of the software manager each developer was required to submit a task report at the beginning of each week. The report includes a list of their specific goals for the week and a summary of their progress for the previous week. The report also indicates if the developer intends to make any deliveries during the week (i.e. check their work into the main source tree). This reporting process enables the software manager to be aware of work progressing across all the development sites and provides the necessary information to coordinate tasks among the developers.

Delivery Report

A new check-in procedure was introduced to ensure each developer was kept aware of all the work progressing at each development site. At check-in the developer must submit a report outlining a description of the changes/features they are checking into the main source tree. This description includes the specific files (source code, documentation, etc) that have been changed or added. The report also includes the primary purpose behind the delivery and how to test the changes/new features.

New Communication Tools

CTS developers rely heavily on informal communication to design, implement and debug their systems. To help facilitate informal communication across the development sites, developers were encouraged to use AOL's Instant Messenger (IM). Microsoft's Net Meeting was also made available to all developers.

Quarterly meetings

Once a quarter all the developers are gathered together to meet face-to-face for one week. This business trip is called a 'sync up' trip. Development goals and future projects are discussed but the primary purpose for the trip is to increase the team's morale and to maintain the camaraderie between the developers.

4. Results

The globally-distributed development team has been operational for four months. In general, the group is performing well but communication and temporal problems have resulted in reduced productivity, trust and morale levels. The following are the procedures and processes which have been initiated and seem to be working well:

Software manager and weekly task reports – Reduced inter-site dependencies

The software manager was able to make good use of the weekly task reports and has been successful at

assigning the majority of tasks between the sites appropriately.

Delivery reports - Maintained awareness and trust levels

The delivery report has been successful at maintaining group awareness and has made it easier for each developer to know who is working on what, who are the experts on particular subsystems, the problems being addressed and the problems outstanding. This procedure, if combined with the absence of other communication problems, was perceived to be sufficient at maintaining trust levels between the developers. Other communication problems, however, did become evident and thus eroded this procedure's effectiveness in this area.

Quarterly sync-up meetings – Maintained morale and motivation levels

These trips have proven to be very successful and developers have commented on feeling 'energized' and highly motivated after meeting with all the team members.

Friendship – An important contributor to awareness

Some of the developers had become good friends during the period they were co-located. These friendships proved invaluable to maintaining informal communication channels between the development sites. When these developers needed to discuss an issue, through either synchronous or asynchronous communication channels, they invariably discussed other, unrelated, issues. Discussions of this nature are a critical component of software development [10]. At CTS these discussions gave each developer greater insight into the particular operations at each site and resulted in greater overall awareness.

However, it was also the case that in some areas, the procedures and processes initiated did not work as well as anticipated:

Communication Tools - Not as effective as hoped

All the developers took the opportunity to use IM but they found that the tool was only adequate for transmitting yes or no style questions. Net Meeting was never used by the developers due to the effort it required to setup and use. Both, IM and Net Meeting are primarily synchronous communication tools and developers indicated that they prefer to use the telephone to converse if an opportunity for synchronous communication is available. This suggests that some of the technology for synchronous communication which is commonly provided does not afford developers sufficient richness as a communication to be perceived as useful.

Communication levels - Did not match co-located levels

Overall, the communication bandwidth was not adequate to compensate for the richness of informal communication between co-located developers. As a consequence, minor issues, usually discussed through informal communication channels [10], were not discussed between the development sites. This resulted in the introduction of bugs into the system. Also feedback on successful deliveries, an important contributor to morale, was completely lacking. Feedback on successful deliveries had in the past been usually done at CTS through informal channels via chance meetings with end users or other developers. Due to the lack of informal communication, however, many developers have stopped getting this feedback and thus their morale has been adversely affected.

Remote experts - Led to productivity and trust problems

When a developer is working with unfamiliar code and the subsystem expert is co-located, the developer would seek their advice on the change/feature they intended to make. The time taken to converse with the expert is usually only a few minutes but if the expert is remote this time can become several hours or even days. Thus in the interests of rapid development most minor changes are made to the subsystem without consulting the expert [10]. These changes, however, may have overlooked subtle design considerations within the subsystem and thus have introduced bugs or other problems.

When these problems become evident (either through expert analysis of the delivery report or errant runtime behavior) significant time is wasted at each development site to address the issue. The level of trust between the expert and the particular developer is also reduced. Merely raising the newly discovered problem with the group can also adversely impact morale, especially if the expert is not very 'diplomatic' at pointing out the problem. Thus, the previous high level of morale and trust that had been built up over the years between developers was possibly eroded somewhat.

Time zone differences - Led to productivity loss

Time zone differences are fundamental source of difficulties for a globally distributed development team [1, 6, 7, 10] and this was no different at CTS. Each day developers arrive to work with an inbox full of questions and other issues from the remote site. To resolve these issues takes significant time for the developer and thus their productivity is affected. Developers indicated that the majority of these issues could actually be resolved quickly, if synchronous communication was available. Also, even when synchronous communication would be possible, the extra effort to try accomplish a rich and

detailed interaction through a narrow communication channel such as IM would also affect productivity.

5. Conclusion

The new global development team at CTS is performing at acceptable levels. It is interesting, however, that given the ability to concentrate on communication and temporal problems the team could not retain the level of productivity it enjoyed when all the members were co-located. Most of the loss in productivity was a result of inadequate processes that were established to address the geographical and temporal distances. There were also several unexpected problems, including the effort required to maintain a globally-distributed development team. This has resulted in an increased workload for some of the developers and thus resulted in a drain on their productivity.

Today, some developers are occasionally failing to follow all of the processes due to project deadlines, workload or other issues. Thus productivity, awareness, trust and other areas will begin to be adversely impacted unless the process can be improved.

Clearly a zero cost, synchronous communication channel that can work around time zones would drastically improve GSD – so we need either a Star Trek transportation device or a time machine! In the somewhat unlikely event of either appearing in the foreseeable future, we will continue to work on the problems and issues identified here.

6. References

[1] Kiel, L., "Experiences in Distributed Development: A Case Study", *ICSE Workshop on Global Software Development*, May 2003.

[2] Pyysiainen, J., "Building Trust in Global Inter-Organizational Software Development Projects: Problems and Practices", *ICSE Workshop on Global Software Development*, May 2003.

[3] Passivaara, M., "Communication Needs, Practices and Supporting Structures in Global Inter-Organization Software Development Projects", *ICSE Workshop on Global Software Development*, May 2003.

[4] Damian, D., Chisan, P.A., and Corrie B., "Awareness meets requirements management: awareness needs in global software development", *ICSE Workshop on Global Software Development*, May 2003.

[5] Oppenheimer. H..L., "Project Management Issues in Globally Distributed Development", *ICSE Workshop on Global Software Development*, May 2002.

[6] Carmel, E., and Agarwal, R., "Tactical Approaches for Alleviating Distance in Global Software Development", *IEEE Software*, March/April 2001.

[7] Battin, R.D., Crocker, R., Kreidler, J., and Subramanian, K., "Leveraging Resources in Global Software Development", *IEEE Software*, March/April 2001.

[8] Herbsleb, J., and Mockus, A., "Challenges of Global Software Development", *7th IEEE International Software Metrics Symposium*, April 2001.

[9] Herbsleb, J., Mockus, A., Finhost, T.A., and Grinter, R.E., "Distances, Dependencies, and Delay in a Global Collaboration", *ACM Conference on Computer-Supported Cooperative Work*, December 2000.

[10] Herbsleb, J., and Grinter, R.E., "Splitting the Organization and Integrating the Code: Conway's Law Revisited", *ICSE Workshop on Global Software Development*, May 1999.