Lessons Learned from an eXtremely Distributed Project

Ben Hogan
ThoughtWorks, Level 7, 16 O’Connell Street
Sydney, New South Wales, Australia 2000
ben.hogan@thoughtworks.com

Abstract

Distributed development projects are highly complex and introduce new challenges for organizations in areas such as communication, collaboration and knowledge transfer.

On a recent 30-person, mixed skill team distributed across Australia, India and the UK we attempted to overcome these challenges using a number of techniques such as daily stand-ups, exchanging ambassadors, small autonomous teams and a dedicated build team. We also used a number of tools such as instant messaging to assist with distributed communication.

This experience report describes each tool and technique we used and the lessons learned in their application.

1. Introduction

Agile teams rely heavily on high quality communication between all team members and any impediment to this can have a big impact on success.

Distributing an agile project across countries and time zones can significantly exacerbate the communication challenges involved.

This report gives our experiences and lessons learned from an extreme example of distributed development, with client and development teams spread across three countries, time zones and cultures.

Explaining how we attempted to overcome these communication challenges and how we modified our approach over time using our experience.

Our project was an urgent short run assignment staffed in the initial two month phase by a 30-person mixed skill team distributed across Sydney, Bangalore and the UK.

The end-client was based in the UK, and provided an online platform for use by advisers and consumers in the UK financial services market. The main development center was based in Sydney, and we had 20 additional staff in India.

In the second phase of the project we split the team into two smaller ten-person teams distributed evenly across Sydney and Bangalore as a result of feedback from our retrospectives.

Meeting the delivery deadline was the primary business focus, with financial benefits for delivering on the agreed date. The client’s primary reason for choosing to use distributed development was the flexibility to very rapidly scale up the team size to meet the targeted deadline and to take advantage of potential 24 hour development between teams.

We used a weekly iteration cycle for planning, tracking progress and reporting. We also scheduled end of project retrospectives and daily stand-up meetings.

2. Initial approach

In the first phase of the project we tried to apply a number of techniques starting with daily stand-ups that lead us to establish bridging relationships and then exchanging ambassadors.

2.1. Daily stand-up

One of the first issues we faced with the distribution of team members was setting up and maintaining regular communication mechanisms. We had used daily stand-up meetings [1] in previous projects to...
improve communication, collaboration and co-ordination so we tried to adapt them for our distributed team using a teleconference call.

We held stand-ups for the local team at 10:00am which took between 10 and 15 minutes. We held the daily stand-up with Bangalore at 3:30pm and this lasted between 15 and 30 minutes. Each of us quickly discussed what we had been working on and our plans for the next 24 hours. We had about 3-6 people locally and 8-10 people remotely participating in these stand-ups.

We anticipated that distributed stand-ups would be more difficult than a collocated stand-up, but were surprised by how poorly the communication worked at these meetings.

We experienced very low voice quality and intermittent connections that caused difficulties in hearing and understanding each other, this was especially true in large groups over a conference phone. Subsequently we tried using voice over IP software as an alternative to landlines. We found it provided significantly improved sound quality and the connection was surprisingly good. We could use our laptops' built-in microphone and speakers as a conference phone and we found this largely solved the connection quality problems.

It took us a few months to become familiar with each other’s accents, and we frequently needed to request clarification during calls. We had to be patient with each other as these difficulties made communication much slower than in our usual stand-ups. Outside of the stand-ups we were able to use instant messaging as an alternative to voice calls to eliminate misinterpretation due to differences in accent.

We also noticed that larger groups didn't work well via conference phone. We were later able to limit the number of participants to 8 people or less.

We found that our daily stand-ups did not provide a suitable place for building critical relationships and trust between the teams. We needed to try other techniques to do this, in particular creating implicit bridging relationships between individuals at both sites.

2.2. Bridging relationships

Under our tight deadlines and with a large team we had no time to build one-on-one relationships with every member of the team.

In an attempt to address this we setup a one-on-one peer relationship between the managers in Sydney and Bangalore.

Each manager represented their local team, collecting issues, requests for help and questions and passing them on to his remote counterpart.

These bridging relationships improved communication between the teams, allowing the managers to co-ordinate the activities of their local teams. It also allowed a deeper trust to form between the two managers as they were able to rely on the frequent communication to follow up on any concerns or actions.

We implemented this technique using daily calls (often up to an hour) between the two leads. These calls acted as a funnel of information and context between the teams, and carried much more complex communication than the stand-up was able to address.

2.3. Ambassador exchange

It was very difficult for us to meet face to face at first. This made it difficult for the Bangalore team to establish business context, as they were two steps removed from the UK and only received a trickle of information. No technique could replace the ability of face-to-face communication to provide visual and contextual information.

We found exchanging ambassadors between our sites to be one of the most effective techniques for improving cross-team communication. It allowed us to build personal relationships and provided a mechanism to build trust and transfer knowledge. The ambassadors were able to communicate lessons learned as well as set future direction for the project.

We sent a number of ambassadors, both developers and managers, between Bangalore and Sydney. We often rotated ambassadors, sending a replacement out as the preceding ambassador returned home. While we would have liked every team member to participate as an ambassador this was not possible. Not all team members wanted to travel and many of those that were willing could not, due to the significant costs for flights and the need to arrange visas for visitors.

When the ambassadors arrived at their destination we provided a basic orientation covering facilities, and more importantly, we provided an introduction to every staff member on site. We walked each ambassador around the workspace and pointed out relevant people and departments and described their roles and responsibilities.

The ambassadors spent up to a month with the team, giving them the opportunity to see multiple iterations of delivery. They participated in the stand-ups, daily meetings and retrospectives of the visited team. We felt there were diminishing returns in additional experience after the ambassador had spent 3 weeks with the visited
team. Keeping the ambassadors longer also delayed the hand-over of their experience to their home team.

The ambassadors also spent time socially with the team at after work event. In Sydney the client hosted occasional Friday afternoon barbeques and in Bangalore there were frequent team dinners. This social interaction was significant in building ties between team members that lasted after the ambassador returned home.

Their active participation allowed the ambassadors to act as champions for the needs of the remote team, raising their home team’s pain points in daily work as well as in retrospectives. They also used their physical proximity to ensure the appropriate resources such as over-night support and network services were available, and that issues were addressed quickly and effectively.

We observed that the ambassadors would naturally keep in very close contact with their friends from back home. This network of friendships was an important communication channel. The ambassadors’ relationships, knowledge of personalities and culture acted as a communication short-cut between the teams allowing for complex and subtle communication to flow in both directions.

The ambassadors would bring their knowledge back to their home office on their return. They were able to provide much of the missing contextual information to their colleagues. During their stay they had absorbed a significant amount of tacit knowledge and this knowledge was disseminated to their team members as they worked closely with each other.

The ambassadors did not have any formal debriefing session on their return, but communicated their experience informally within their team. In retrospect I believe that there would be value in holding a facilitated debriefing session both when the ambassadors arrive at their destination and again when they return home.

2.4. Unfiltered communication

Our Indian ambassadors discovered an interesting effect when they arrived: the Sydney team had tried to protect the Bangalore team from the complexities and politics of the local site, "simplifying" the communication with them. This caused problems of frustration and low morale in the Bangalore team due to missing information such as not understanding the underlying reasons for project decisions.

Luckily the ambassadors were able to pick up on the political and historical forces operating on the client site and could take this to the remote team.

On reflection, we felt that we had made a fundamental mistake in trying to filter information that was communicated with Bangalore.

In hindsight, I would recommend discussing the history leading up to the current project including the relationships and goals of the sponsors and stakeholders with any new team at project initiation.

I believe that communicating this kind of contextual information is critical to building trust between teams and that trust is critical for effective distributed development.

3. Adapted approach

At the end of our first phase of delivery we held retrospectives at both sites and ambassadors took the results of these to the other site for discussion.

As a result of our experiences we changed our structure to smaller and more autonomous teams for the next phase of our delivery. We tried “mirroring” between teams and setting up a dedicated infrastructure team.

3.1. Small teams

Our initial large team had caused us a number of difficulties. The large number of participants in meetings introduced pressure not to talk and made the meetings much longer. This lead to less effective communication and precluded a deep understanding of each others' work.

We noticed these effects, and split from one 30 person team into two smaller 10 person teams distributed evenly across Sydney and Bangalore.

Using two smaller teams improved our ability to talk daily on the phone and made building relationships with each member of our team achievable. The smaller teams reduced the communication overhead considerably and allowed the teams to work much more closely with each other across the sites. This also helped the teams align with each other and improved our morale through increased trust in each other and a greater feeling of camaraderie.

While this technique is applicable to all projects, we felt its effect was amplified on our distributed project as the effort of communicating within a large distributed team was much larger than we had experienced in equivalently large collocated teams.

3.2. Multidisciplinary teams

In the initial phase of delivery most of the user-interface development was implemented by the Bangalore team and most of the back-end development
was implemented by the Sydney team (often referred to as a horizontal split of work). The client had implemented this work allocation in order to avoid "unnecessary" communication between the local and remote teams with a desire to improve the overall project efficiency.

In our second phase, we split our work vertically, with each team working across an entire feature including user-interface and back-end components. We found that this vertical work allocation encouraged communication and improved co-ordination. We felt this technique produced much larger productivity and efficiency gains than could be saved by avoiding some communication through horizontal or component based work allocation.

Not only did each team work on entire features, but each site had a full complement of project managers, team leads, business analysts, testers and developers.

We found these multidisciplinary teams to be enormously helpful, giving the teams autonomy and improving efficiency. This also helped with the time-zone differences as the developers did not have to wait for analysts to return to work the next day to get questions answered.

### 3.3. Mirroring

The newly structured teams were much smaller with only 3 developers at each site. This allowed us to extend the bridging relationships between individuals using a technique we called mirroring. This involved each role on the team (developer, BAs, testers) working closely with their remote counterpart on the same tasks, doing 24 hour work across time-zones and handing off the work to each other.

The Sydney developers would often hand over development in their afternoon and spend the last two hours of their day collaborating with their peer on the remote team. We used tools like voice-over-IP and Remote Desktops to pair on code and to explain progress and issues.

This resulted in each team having the required context to maintain the team’s autonomy. This kind of mirroring facilitated rich communication between the teams by making it part of day to day task completion.

### 3.4. Dedicated Infrastructure Team

In the initial phase of the project, we had daily network and infrastructure outages for the first two weeks. This amounted to over 50 developer days of lost time. We measured our output in story points at the end of each iteration. The impact of the infrastructure outages can been seen in the low iteration 1 velocity of 18 points compared to later iterations delivering around 70 points each. The impact on iteration velocity is shown below in Figure 2:

![Figure 2. Iteration velocity chart](image)

These outages not only caused delays but also produced high levels of frustration for the remote team, lowering morale and having a negative impact on productivity even when the infrastructure was working.

Our project needed reliable distributed infrastructure to allow the teams to work on a common code base and to use continuous integration.

In phase 2, we tried to improve our infrastructure stability by introducing a dedicated infrastructure and build team. The team was responsible for providing staff working development environments to enable distributed development including:

- Fast, reliable and open internet access at both sites;
- Reliable virtual private network;
- Fast reliable access to a shared version control server;
- Stable and up-to-date development environments including databases, proprietary systems, external third-party services and configuration.

The infrastructure team monitored networks and environments and provided late night support to the Bangalore team for when outages caused significant problems.

### 4. Tools

Over the life of the project we used a number of communication tools to assist our collaboration with our remote team members; these included instant messaging, voice-over-IP, video documentation and remote desktop software.

#### 4.1. Instant messaging
Most distributed teams probably use some kind of instant messaging to communicate. As we had expected this was a vital tool to facilitate communication for our project.

We used Yahoo! Instant Messenger and found the following configuration was very helpful:

- Everyone involved with the project had the software installed and knew how to use it including customers, testers, business analysts and developers.
- We added the whole team to the address book of the tool such that everyone had the contact details of every other team member.
- We used contact groups for each geographic location.

This configuration allowed us to see when team members who were online at each site, and made it cheap and easy to pass on a message or ask a question of the appropriate person.

The team established some protocols for using instant messaging; We endeavored to be signed-in from when we first arrived till we had finished for the day. This allowed us to see who was at their desk and who could be contacted. We also used the status message to provide information about when we were not interruptible (for example when we were busy, or pairing).

We also used the conference chat feature to allow group discussion. This worked well for focused topics and to introduce local and remote developers who were working on similar things.

4.2. Voice over IP

Along with Yahoo! for instant messaging we also used the free voice-over-IP tool Skype [3] with headsets for computer based voice calls.

Initially we used Skype for stand-up calls. We later extended our use of Skype throughout the organization as a telephone replacement, particularly for calls with the UK, and we used Skype-out for internet-to-phone calls where necessary. We found the address book to be a convenient storage location for quick access to important contacts.

The free Skype-to-Skype calls also provided a small financial benefit as they avoided high international call costs.

One of the disadvantages of using voice over IP was the additional load on the network it produced. This was particularly disruptive in the initial phase of the project where we had limited internet bandwidth available.

4.3. Video documentation

Distributed development projects need to use every communication channel available. One such channel that is often overlooked is video documentation.

We created some very short video documents during the project, and found these worked well as a communication and training tool. We used the video documents to very quickly and cheaply demonstrate how to configure and operate the client's proprietary workflow system using nothing more than a modern digital camera.

The project manager looked for opportunities for making the video documents such as recording a short overview of the system architecture and a video on how to use a complex tool. The documents were placed on a shared network drive where the remote team could access them.

We set the camera's resolution and quality settings fairly low, so the video file could be copied directly to a file share for use by the remote team, with no need for re-encoding or re-sizing. This made creating a short video document almost as simple as taking a photo of a whiteboard, with no time spent post-processing the video.

Here are some tips for creating good video documents that worked well for us:

- Create very short video documents, around 3-5 minutes on a single topic. Your camera may already automatically create a new file every time you start and stop the video, so use this often to chop up the video.
- Automate the transfer of new video clips when your camera is connected to your PC. We used the Canon driver and Windows XP to auto-detect the connection of the camera and automatically transfer all video files to the disk. This saved time and reduced the overhead of using the technique.
- Use expressive file naming to identify the topics for easy reference.

We discovered that a very quick and easy way to do a low-fidelity screen capture was to set a low screen resolution (800 x 600 worked well for us), and hold the digital camera close to the screen. At this resolution it is possible to see enough screen detail to follow the speaker explaining an on-screen tool or topic. This also has the advantage of allowing quick transitions between the face of the speaker and the screen as appropriate.

Next time we would try to encourage both the local and remote teams to produce video documents to take advantage of two-way communication improvements.
4.4. Remote desktop pairing

When local and remote team members wanted to discuss or review some code together we used remote desktop software. We found Real VNC [3] was a free and effective remote desktop implementation.

This worked well, and combined with voice-over-IP allowed very rapid transfer of code-base knowledge. The ability to gesture with the mouse was helpful for communicating focus and emphasis during code discussions. We used this successfully to do remote code reviews. In the first phase of the project we used this infrequently as a review tool after the fact. In the second phase with smaller teams we were able to do this whenever the mirroring developers needed, often as part of their daily hand-over process.

5. Conclusion

Looking back at the techniques we tried on our project we feel a few could have been improved, some worked well, some very well and a few were extremely effective.

Of all the techniques we tried, the most disappointing result was from attempting daily stand-up meetings via teleconference. A potential future improvement to distributed stand-ups maybe to combine all the forms electronic communication at once, for example, using video conferencing while demonstrating on a remote desktop and using instant messaging to clarify discussions.

We also could have done better in ensuring we did not filter out important information when talking with the remote team. The importance of establishing a shared understanding of context including details of history, politics and relationships suggests that in future projects we could benefit from making this kind of communication an explicit step of project initiation.

We felt the use of bridging relationships and multidisciplinary teams were both effective structures and would use them again in a similar context.

The movement to smaller teams in the second phase worked very well for us. Although larger teams can reduce the risk of isolated silos of information and can encourage shared ownership, they can also impose a heavy communications burden which is amplified if the team is distributed. The advantages of a smaller team should be balanced with these demands and teams should be willing to shift their approach as needed.

The most effective technique we used in our project for improving communication, providing tacit knowledge transfer and building relationships and trust was exchanging ambassadors between sites. In any distributed project the importance of face-to-face communication should not be underestimated.

The additional investment required to maintain communication between sites over and above what would be needed for a collocated may come as a surprise. The economic impact on a project must be considered carefully, as Gartner reports:

"Viewing labor costs alone does not provide a realistic picture of the total expense of an offshore deal. Many hidden costs — including expenses associated with infrastructure, due diligence, communications, governance, overseas travel and cultural training — will offset the cost advantage of wage differentiatation." [4]

Agile processes call for frequent deployment for continuous testing and feedback. This introduces a need for established development environments and supporting networks and infrastructure. While establishing a dedicated build team helped us address these issues retrospectively we would recommend that they be considered before starting any distributed project. Distributed teams are extremely reliant on good infrastructure setup for remote development and need reliable network and telecommunications support for communication.

Distributed projects need to be continuously looking for techniques and opportunities to improve communication. Distributed teams should look out for every tool and communication technique available to them and, if possible, consider inventing some of their own.

6. References


7. Acknowledgements

I would like thanks the following people for their contributions to this report: Suzie Newton, Chirag Doshi, Sudhir Tiwari, Gianny Damour, and Zac Zavos.