Applying Distributed Development Patterns Stories and Pattern Sequences

Lise B. Hvatum

The experience reflected in these patterns comes from years of working for an international technology company with product development centers in Europe, North America and Asia, and through interaction with people from other companies practicing distributed development. The patterns were initially built on observing internally established practices, and later influenced by learning through reading, conference participation, and discussion.

Despite the effort of capturing over thirty patterns to create a solid foundation of knowledge, it is not enough to make the material really useful to an organization. The manager or team member of a distributed team faced with all these knowledge pieces must feel like a kid opening a large box of Lego just to find that the instruction manual is missing. There are lots of parts, but no beginning or end. It may look overwhelming, and if he or she needs all the parts depends on the complexity of the system they are building.

This paper follows the stories of three projects as a way to illustrate the use of the patterns in the context of a small, medium and large project. Alternatives and possibilities are explored, while making it clear what needs to be prioritized, or even practiced to the level of "must-do".

Background

A distributed team is not a new invention. People have collaborated on research and product development projects internationally for decades if not more. Several years ago I worked as the manager for projects split between Norway and England, without considering that to be a problem. Traveling was easy and inexpensive. The project members would visit on a regular basis, and communicate frequently by phone and e-mail.

Then a few years ago I got involved in projects running between the US and Asia. What had been small, manageable issues now escalated to become major obstacles. The Asian location hired several young developers with little knowledge of the business domain. In-depth business and technology expertise was mainly with people in the US locations. Cultural differences as well as language problems were hurting communication. The time zones allowed no overlapping work time, which meant full day delays in exchanging e-mails, and caused very early/very late presence for video conferences and phone conversations. Travel was time consuming and expensive.

Luckily the people involved were enthusiastic and wanted to make it work. Although stumbling from one problem to the next, the teams eventually found ways to collaborate and be productive. As managers and key engineers moved on to other activities, their experience was captured to benefit future teams.

Introducing the Patterns

The patterns themselves are not included in this paper as it would make it far too extensive. Most of the patterns are presented in papers at PLoP conferences [EuroPLoP 2004, PLoP 2004, EuroPLoP 2005], and they are all represented by thumbnails in the appendix. Here is an overview to aid the readability of the paper. After reading the following two pages, the reader should have enough knowledge about the patterns to be able to appreciate the stories and pattern sequences following this introduction.

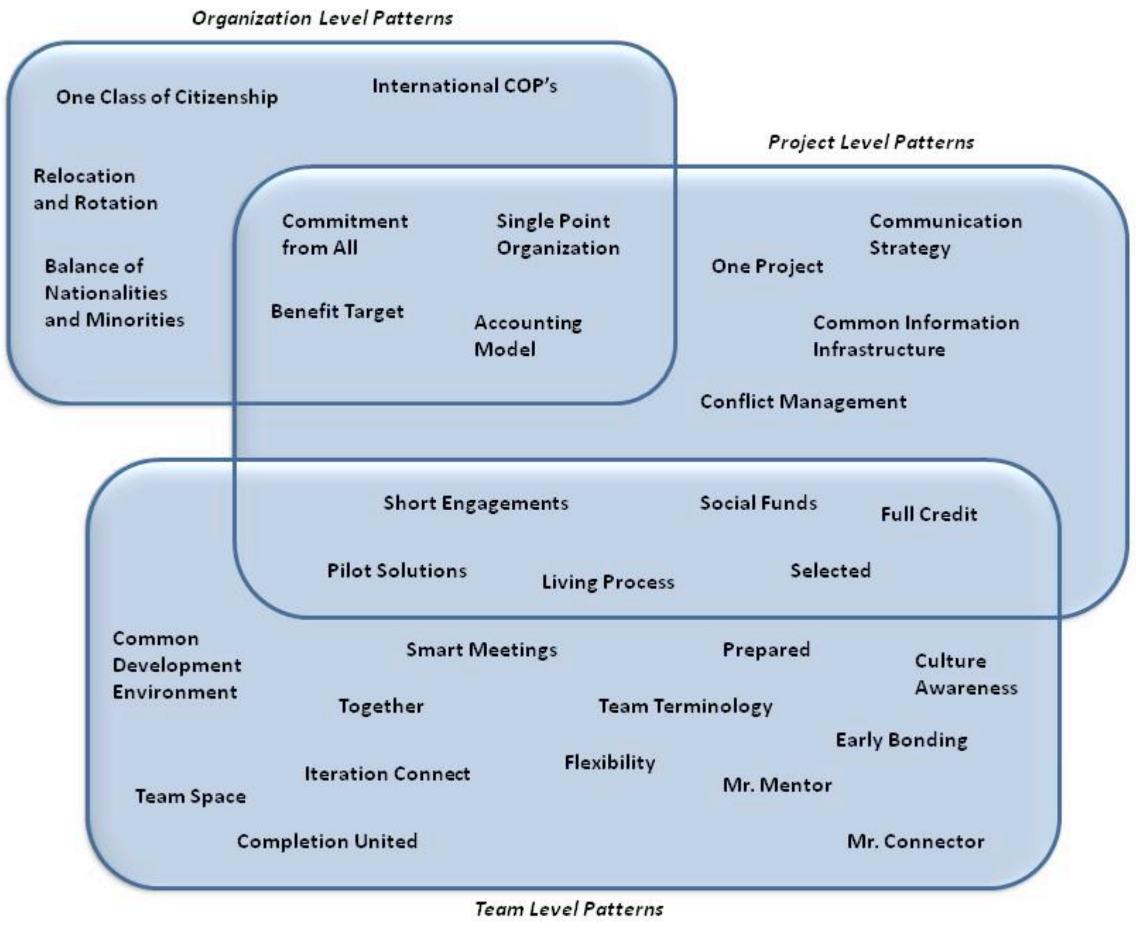


Figure 1: Distributed Teams Patterns Overview

The patterns are sorted into three overlapping levels. The first set of patterns applies for the whole organization. Some are independent of the project: **One Class of Citizenship**, **Relocation and Rotation**, **Balance of Nationalities and Minorities**, and **International COP's.** These patterns focus on creating an organizational culture that will enable the organization to succeed with distributed teams.

Other top-level patterns are needed to create an organizational structure that will support a distributed project: **Commitment from All**, **Benefit Target**, **Single Point Organization**, and **Accounting Model**. The purpose of each of these patterns is given in figure 2.

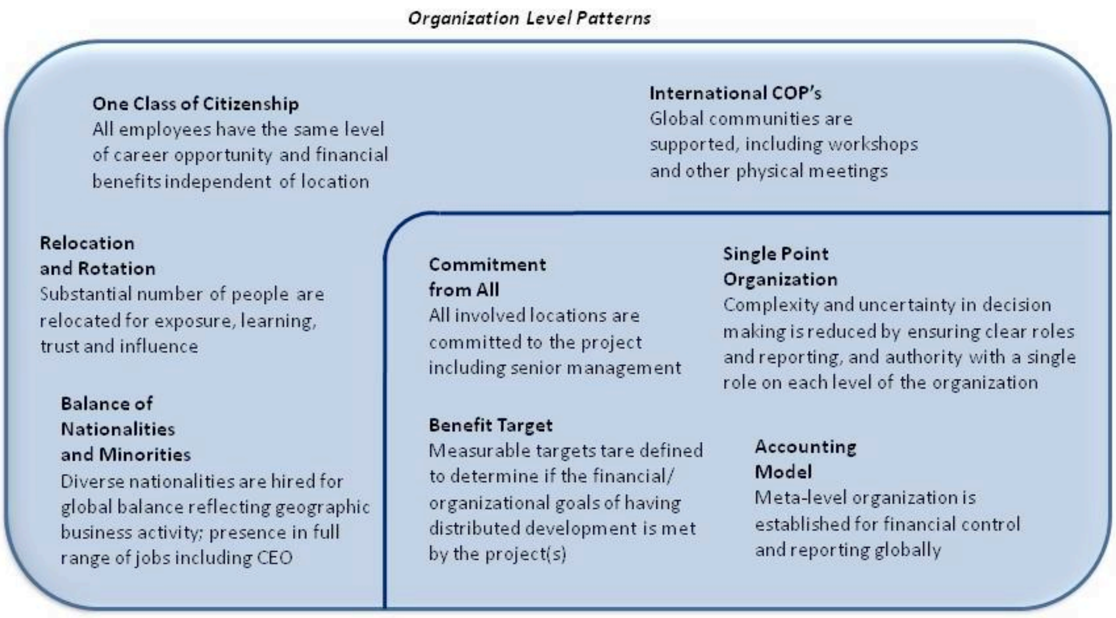

Organization Level Patterns
One Class of Citizenship
All employees have the same level of career opportunity and financial benefits independent of location
International COP's
Global communities are supported, including workshops and other physical meetings
Relocation and Rotation
Substantial number of people are relocated for exposure, learning, trust and influence
Commitment from All
All involved locations are committed to the project including senior management
Single Point Organization
Complexity and uncertainty in decision making is reduced by ensuring clear roles and reporting, and authority with a single role on each level of the organization
Balance of Nationalities and Minorities
Diverse nationalities are hired for global balance reflecting geographic business activity; presence in full range of jobs including CEO
Benefit Target
Measurable targets tare defined to determine if the financial/ organizational goals of having distributed development is met by the project(s)
Accounting Model
Meta-level organization is established for financial control and reporting globally

Figure 2 Patterns applied for the overall organization

Additional patterns are applied on the particular distributed project: **One Project**, **Communication Strategy**, **Common Information Infrastructure**, and **Conflict Management** to set up the project properly for a distributed setting. These patterns are typically applied by the Project Manager.

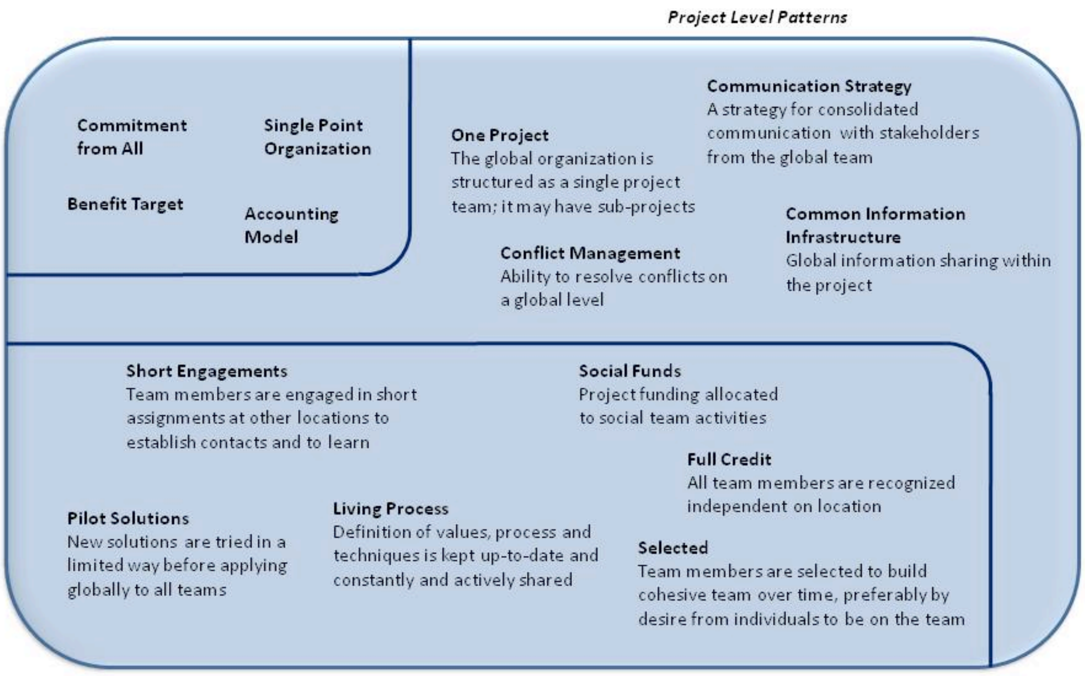
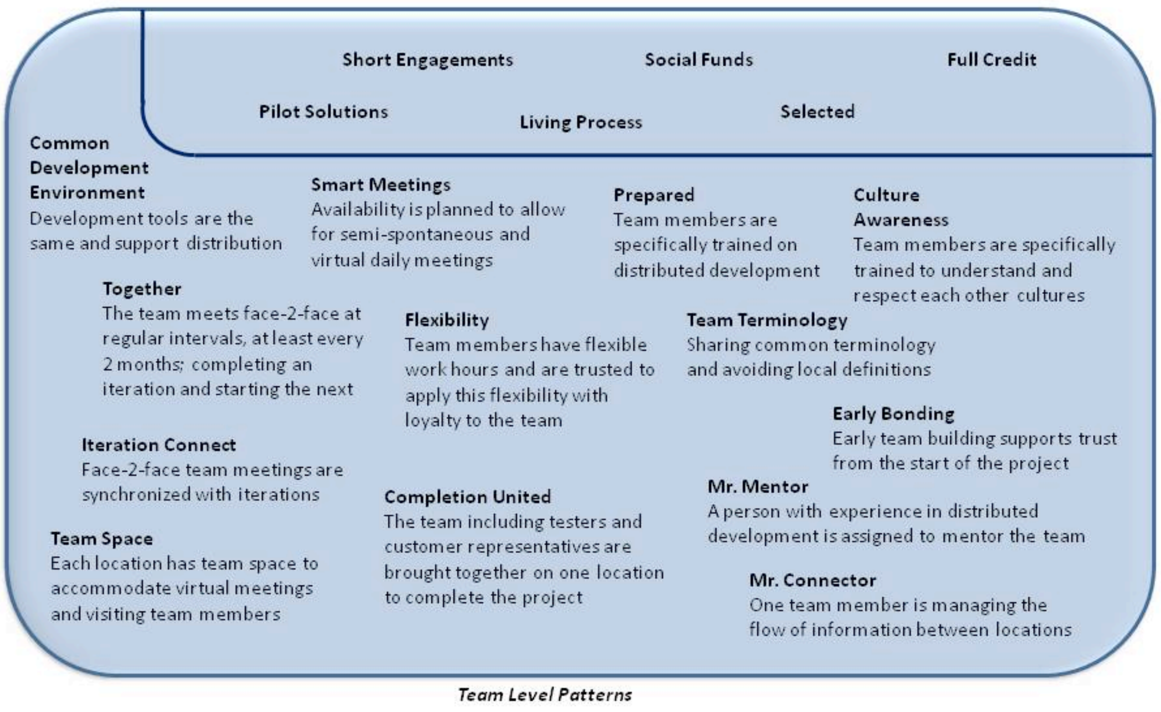

Project Level Patterns
Commitment from All
Single Point Organization
Benefit Target
Accounting Model
One Project
The global organization is structured as a single project team; it may have sub-projects
Communication Strategy
A strategy for consolidated communication with stakeholders from the global team
Common Information Infrastructure
Global information sharing within the project
Conflict Management
Ability to resolve conflicts on a global level
Short Engagements
Team members are engaged in short assignments at other locations to establish contacts and to learn
Social Funds
Project funding allocated to social team activities
Full Credit
All team members are recognized independent on location
Pilot Solutions
New solutions are tried in a limited way before applying globally to all teams
Living Process
Definition of values, process and techniques is kept up-to-date and constantly and actively shared
Selected
Team members are selected to build cohesive team over time, preferably by desire from individuals to be on the team

Figure 3 Patterns applied for the project

Finally there are a large number of patterns that are applied by the team to enable the team for distributed development. The more control the team is taking in adapting and implementing these practices the better is the chance of successful implementation. The patterns on this level are practices that apply on a team level only, although some of them require financial and organizational acceptance from the organization.



Short Engagements
Social Funds
Full Credit
Pilot Solutions
Living Process
Selected
Common Development Environment
Development tools are the same and support distribution
Smart Meetings
Availability is planned to allow for semi-spontaneous and virtual daily meetings
Prepared
Team members are specifically trained on distributed development
Culture Awareness
Team members are specifically trained to understand and respect each other cultures
Together
The team meets face-2-face at regular intervals, at least every 2 months; completing an iteration and starting the next
Flexibility
Team members have flexible work hours and are trusted to apply this flexibility with loyalty to the team
Team Terminology
Sharing common terminology and avoiding local definitions
Early Bonding
Early team building supports trust from the start of the project
Iteration Connect
Face-2-face team meetings are synchronized with iterations
Completion United
The team including testers and customer representatives are brought together on one location to complete the project
Mr. Mentor
A person with experience in distributed development is assigned to mentor the team
Team Space
Each location has team space to accommodate virtual meetings and visiting team members
Mr. Connector
One team member is managing the flow of information between locations
Team Level Patterns

Figure 3 Patterns applied internally by the team

Introducing the Projects

In the following sections, we will look at projects requiring an increasing level of formality and organizational structure. The project stories build on real experience and reflect real projects. For confidentiality reasons actual names and project details have been somewhat modified, although in a way that should not affect the understanding of why and how a pattern is applied.

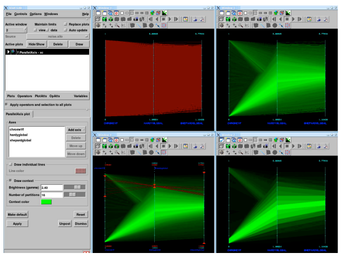
The first project is a small activity running for a few months, and the focus of the story is team communication and collaboration. The second team is larger and requires some support from the organization. The third project is large and involves multiple locations resulting in the need to establish practices and workflows far outside the project itself. The essential characteristics for each project in relation to distributed teams are summarized in the table below:

Project	Visualization Tool	Asset Management System	Control System
Size	Small	Medium	Large
Budget (USD)	0.3M	5.9M	32M
Effort	1.3 my/7 people	27 my/27 people	120 my/55 people
# Locations	3 2 in China, 1 in the US	3 US, France, India	6 2 in US, UK, France, Italy, India
Time Difference	China: Same time zone China – US: 13 hours	US – France: 7 hours US – India: 11 hours	US – UK: 6 hours US – France/Italy: 7 hours US – India: 11 hours
Complexity	Low	Medium	High
Innovation	No	No	Yes (new)

Throughout the project stories the following formatting and structure is used:

1) Pattern names are written in **bold**.
2) Throughout the stories text in *italics* headed *Exploration* explain and suggest alternative implementations to better understand the use of the patterns.
3) Side effects and negative results of applying the patterns are included in the exploratory text under *Issues*.
4) A few external patterns are mentioned in the exploratory text. These are represented with underscored font. They are listed with references on page 23.

At the end of the paper there is a short discussion of other patterns and pattern languages that complement the patterns in this paper. The reader looking for more detail will benefit from reading the actual patterns (see references).



Picture Source: Google Images
(sample picture for illustration purposes)

Project #1 – The Data Visualization Tool

Company A sells measurement data to customers. The data by itself is meaningless unless you know how to represent it graphically in a way that makes sense to analysts in the client company. So it is customary to provide a viewer tool with the data. The behavior of the viewer tool depends on the physics of the measurement which is already well known. It is developed on a software framework which is mature and stable. The developers have made several similar viewers for other measurements before. This is a low-risk project estimated to take about 3 months (15 man-months of effort) including testing, with a USD 250k budget.

The resources assigned to the project are a manager (Mark) and three developers in China (Yuan, Chen, and Feng), and a domain expert (Alfredo) and a physicist (Linda) in the US. There is also a qualification engineer in China (Li), but in a different location. They are all experienced and have worked together before.

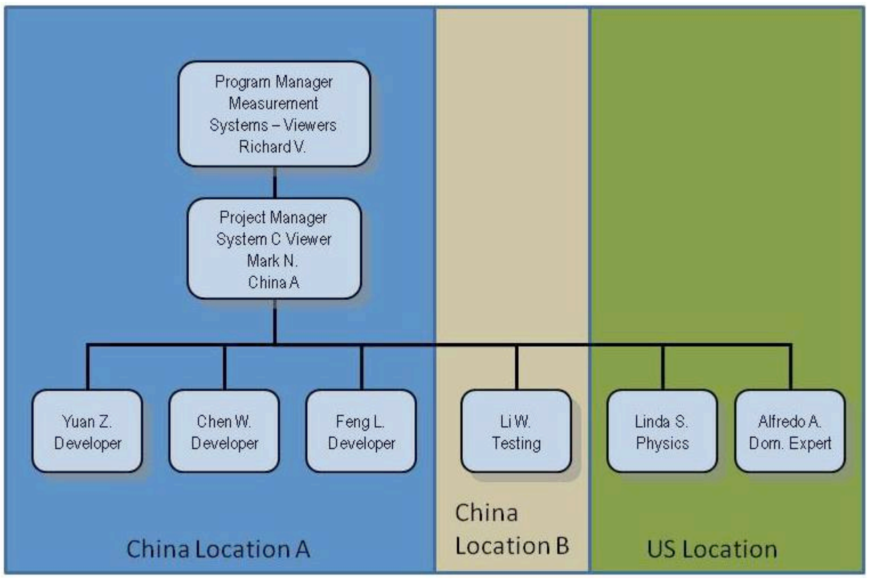


Figure 4: Team organization for the Data Visualization Tool team

The company has learned to always establish a **Single Point Organization** which establishes well-defined roles and reporting on each level of the organization to ensure clear decision making and responsibilities. It creates the new project as a part of the Chinese project portfolio under the program manager (Richard) who is in charge of a family of viewers for similar products. This is natural as this is the location of the project manager. Mark makes sure to run the development as **One Project** with shared team objectives, and to get **Commitment from All** especially for the external resources. Alfredo is assigned full time to the project for its duration, but Linda has another project to deal with as well.

Exploration: It is important to ensure that there is a clear, communicated and simple reporting structure for the project. In such a ***Single Point Organization****, decision at each level and in each entity in the organization is sitting with a single role to remove confusion. Roles and their assigned authority are clearly defined.*

Team focus is also made clear through applying the ***One Project****, having only one manager and team world-wide with common objectives and goals. The project manager is the only interface to the customers and upper management in the company when it comes to defining the scope of the product, change management throughout development, and all other major decisions. The stakeholders can directly collaborate with team members in discussing the product (requirements discussions, demos, acceptance testing), but no decisions are made without the project manager in charge.*

The up-front ***Commitment from All*** *involved locations supports the project with the resources needed throughout the development. The resources are definitely people with particular skill sets and knowledge, but may include other support like facilities, access to test systems etc.*

Issues: There is an assumption in the above exploration that the organization is structured in projects. That may not be the case for your company, and so the implementation of the above will need to take another shape. What is important to achieve is the clear structure and responsibilities for decision making in respect to the developed project, but also for the utilization of resources. In particular it is necessary that an individual engineer assigned has enough time available and that priorities (and possible priority conflicts) are taken care of by management and not left to the individual to deal with.

Even though his team is experienced, Mark decides to fly over the two people from the US for one week so they can start the project **Together**. The team starts planning the product and creates the initial project plan. The qualification engineer Li is also coming to the development location in China for the same week so that the whole team is there. Mark enforces the **Early Bonding** by using **Social Funds** for some teambuilding. The team members already have a good level of **Culture Awareness** and are well **Prepared** through earlier experience with distributed teams, but their week together enforces this.

Exploration: The most important time for a team to physically be ***Together*** *is at the start of a project. Even for larger systems, a good foundation both for the system architecture and the team collaboration (trust) can be done in a two-week boot camp. The goal is to establish a common understanding of the product, and of the main principles of the system architecture. Several possibilities should be discussed, and an initial framework agreed on. This is a first achievement for the team, and by tightly sharing the evolution of this framework each individual will feel the ownership and enforce the* ***Early Bonding*** *that builds trust in the team that they will benefit from throughout the development. By including all resources in the event, they are on the same level, share the terminology and the understanding of the product, and can plan for their own tasks in the project early. Having a customer representative present as in this case is very beneficial. Many mistakes and misinterpretations of requirements can be avoided by the customer being heavily involved at this stage.*

One good effect of early collaboration is the ***Early Bonding*** *between team members from the beginning. By observing each other during team interactions, all get a better understanding of the individuals, their communication style and their knowledge.* ***Social Funds*** *should be used to invest in good people relations through social activities. This can include team dinners, an outing to visit a local attraction, an evening at a local go-cart track, etc. Strong team interrelations and trust is invaluable on a project. Although hard to quantify directly, most problems on software projects can be traced back to communication problems within the team or with the stakeholder.*

When talking with developers who have experienced multi-cultural teams, they often point out the importance of understanding the body language and communication style of other nationalities. Unless all team members are familiar with the main cultures represented it may be a good investment to send the team members to training to build up their ***Culture Awareness****. The same goes for training on distributed development to ensure the team members are well* ***Prepared****. An investment in training the team members on issues and solutions for multi-site software development will surely pay off, although it may be hard to find this kind of training offered yet.*

Issues: All of the above patterns require that the company is willing to do the financial investment in travelling and training. If this is not possible in your company you need to emulate the effect of together-time. You still need the effort on architecture and product vision, and a virtual workshop may be a way forward. This means that team members at each location run local workshops, with limited sessions run with the complete team using collaboration tools (Live Meeting, phone, video conferencing etc.). For team members to better know each other you could create a gallery where team members can post pictures, stories and personal statements, or even link up with tools like MySpace. The manager must make sure that people are comfortable with using these methods, and that what is posted is not offensive to any of the team members.

The team agrees on short iterations of one week. They have a **Common Development Environment** that makes their collaboration easy. A full integration test on completed functionality (Use Cases) is done by Li on every integration baseline. The team also has a collaboration space (their **Common Information Infrastructure**) where they post all their documents, but that also includes chat options and live meeting capabilities.

Exploration: The importance of a common development process and a common development environment is often not understood by a company. Most fundamental is the shared code base, especially if the development is truly one product and the developers need to share/modify some of the same files. Iterative development with frequent integration into a common baseline, and automated testing based on full unit test coverage is giving the team the benefits of good control of progress, and early detection of problems. This way of working really means that the team must be using the same development tools.

Common space like a wiki, SharePoint or using a collaborative development tool like VersionOne ensures that each team member always has access to the latest version of important project documents. You may also decide to share the history of your chat sessions since a discussion thread may have information valid for other team members.

Issues: For historic reasons, locations may be using and have experience with different development tools. The organization must be willing to invest in common tools for development and team collaboration and train people on the use of these.

During the development the team has frequent **Smart Meetings**, meetings that emulate a Daily Scrum in a distributed setting, to keep each other up to date on the progress. Attending these meetings is mandatory for all team members, but except for that time there is a lot of **Flexibility** for the individuals in work hours. Mark pays special attention to Linda to make sure her dual assignment is properly managed.

Exploration: For a short development it may not be necessary to have more session where the whole team is on one location, but the best is if the teams can meet face-to-face about every 6 weeks. If traveling budgets are too restrictive, or the team members know each other well from earlier development activities, face-to-face meetings may be less essential. But the ***Smart Meetings*** *emulating the daily Scrum are totally necessary, and all team members participates each time unless there is a very strong reason they cannot make it. In these meetings the team members keep each other up to date with individual status, any changes, and any dependencies they have to each other. Because meeting hours may be at odd hours for some team members (in the Data Visualization Tool project the majority of the team sits in China, and the meetings take place in the morning as they get to the office, meaning the team members in the US need to attend late in the evening local time), work days are kept flexible for the individual as long as they work enough hours and participate in any common events.*

Issues: This loyalty must work both ways between the company and the employee. The focus must be on progress rather than work hours, and respect the ***Flexibility*** *for each person in when they work.*

At the completion of the project Mark decides in agreement with the team to do a partial **Completion United** by bringing in the stakeholders and the qualification engineer Li only. He and Li travel to the US, where they have a weeklong workshop with stakeholders. With Linda and Alfredo as support, Li is teaching the users to use the new viewer, and she is tracking any defects and improvement requests during the day. One developer in China is on call during the US work hours to fix any major defects should they find any, but the other two are working normal hours fixing problems during regular Chinese work hours. They make sure a new baseline is built and working before the start of a new work day in the US. In this way they sort out any problems in the system within a few days. The test users from the key clients are happy with the product, and it is ready for shipment.

The team is confident that the project has fulfilled the requirements and that the new Visualization Tool is completed. The stakeholder has a couple of small requests that are sorted out in two days, and the project is at its end. Mark makes an open report back to his management pointing out the effort of the team members and crediting them with their achievement making sure that all team members get **Full Credit** for their involvement. He and Li celebrate with the US team members before leaving, and again with the China team members when back in China. He buys all a small gift to show his appreciation of their contributions.

Exploration: The completion phase brings the extended team together for a very efficient closing of the project. This activity can be done as a combination of testing and training of the beta user community. Being ***Together*** *for the final phase is also an opportunity to get retrospect and get closure for the team. It is important to make sure that all members of the extended team are recognized for their contributions.*

Issues: It may be hard to get the financial support in your company to bring the team together at the end of the development. Combining it with training users may make it *easier, since it will be cheaper for a small development team to travel to many users than the other way around. Maybe it could even be possible to get the clients to pay for this training, but do manage the customer expectations of quality by making it clear that they are getting trained on a Beta version and they will likely find issues with this system.*

The sketch below shows the patterns sequence from the above story. If patterns are deemed necessary they are marked in **bold** with a thicker line around the shape. Note that each of these patterns can be applied independently, although they mostly benefit by being used with other patterns that they relate to in the sequence diagram (like Early Bonding and Social Funds).

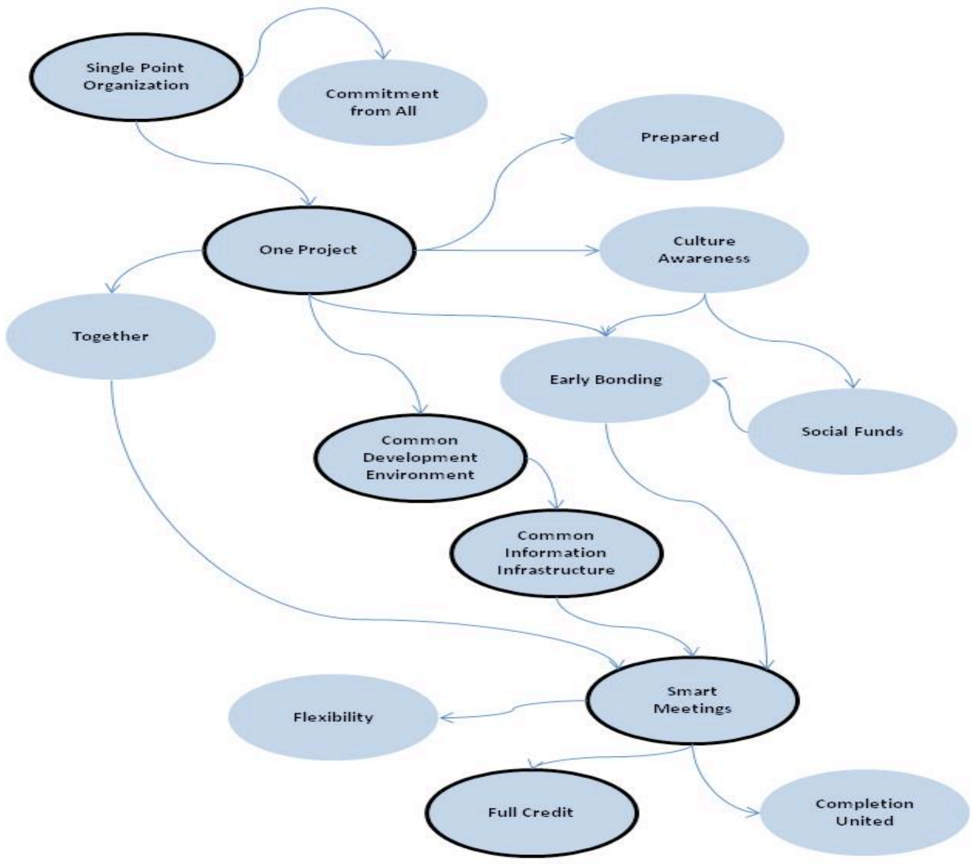
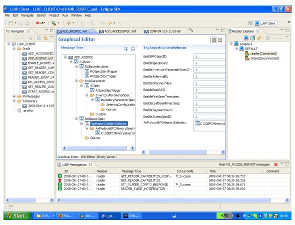


Figure 5: Patterns sequence for the Data Visualization Tool team

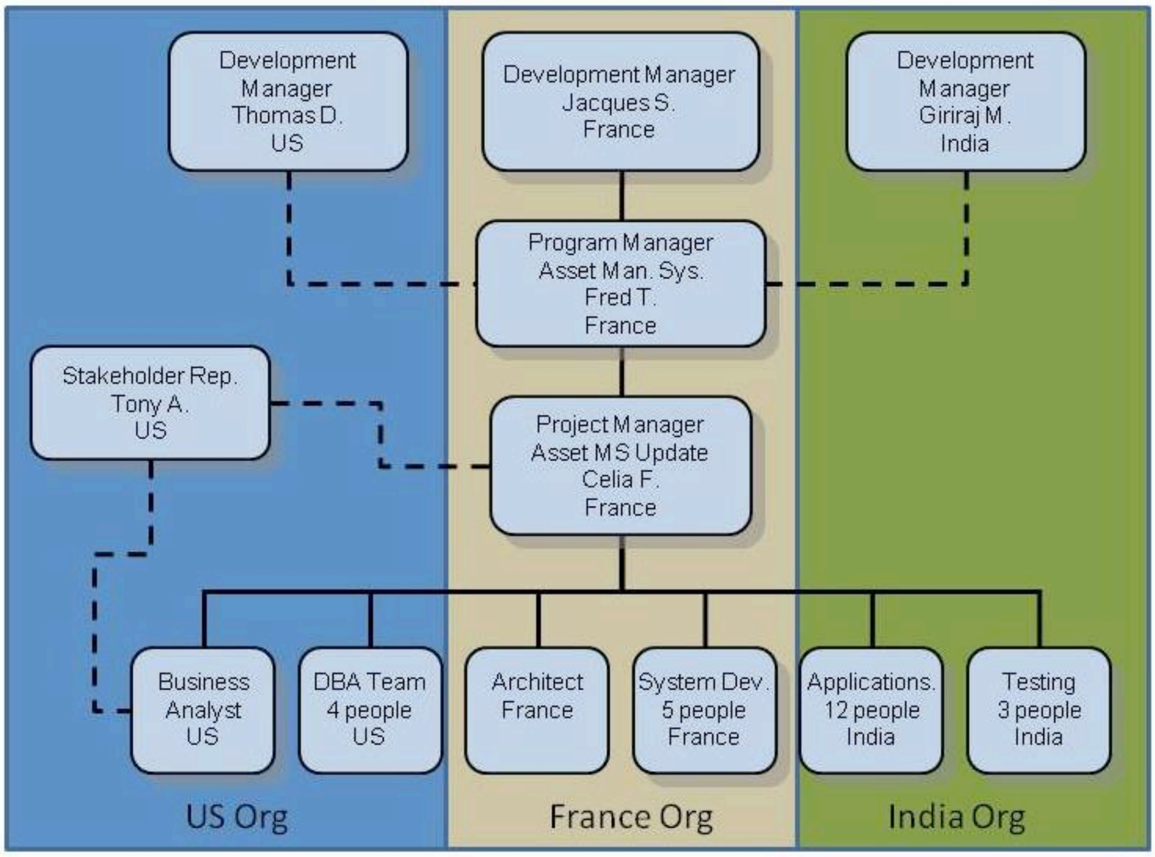


Picture Source: Google Images (sample picture for illustration purposes)

Project #2 – Asset Management System Update

Company B is developing an asset management system for important client. The current system is having some seve problems caused by use of outdated 3rd party technology, a because some of the implemented business logic is unstable. new project is started to re-factor parts of the system a replace an Access database with Oracle. The project development time for this project is one year (27 man-yea with a budget of USD 5.9M.

The existing team is expanded with an Oracle DBA (data base administrator) and several developers. Team members are located in the US, in France and in India. The developers in India have less experience with the technology, and are new to the company. The most experienced people (the architect and the system developers) and the project manager are located in France, while the DBA team sits in the US. The stakeholder representative is also located in the US. A testing team of three people are co-located with the application development team in India. Note that only the team with the project manager is part of the budgeted effort of 27 man-years. The other roles are part of the organization but manage/support several projects and are budgeted otherwise. The team structure looks like this:



Development Manager Thomas D. US
Development Manager Jacques S. France
Development Manager Giriraj M. India
Program Manager Asset Man. Sys. Fred T. France
Stakeholder Rep. Tony A. US
Project Manager Asset MS Update Celia F. France
Business Analyst US
DBA Team 4 people US
Architect France
System Dev. 5 people France
Applications. 12 people India
Testing 3 people India
US Org
France Org
India Org

Figure 6: Team organization for the Asset Management System team

This is clearly an organization that is more challenging than Project #1, with dependencies and need for good collaboration between multiple sites both within the team and within the management levels.

Celia, the project manager, works on two levels when starting up the project. She collaborates with the involved development centers to make sure there is **One Project** organization where all team members share the team objectives with the same priority. With her own management located in France it is natural that this project is added to the French project portfolio. She also makes sure there is a **Single Point Organization** for all decision making needed in the organization to support the project, and she ensures **Commitment from All** for team allocation and local resources at each location. This includes **Team Space** to accommodate visits, and an **Accounting Model** to make sure the project can deal with budgets and financial reporting in a uniform and consolidated fashion. Finally she agrees a **Communication Strategy** with the stakeholders and the center management at the different locations to keep all informed the way they desire.

Note! The exploration text will not repeat what has already been explored in previous project examples, but rather focus on what is introduced in each project.

*Exploration: Creating the **Single Point Organization** is highly important to enable the operation and performance of the team. This will provide a clear direction and faster decision making as all team members are clear on who needs to be involved and how decisions are made. With multiple projects the organization needs to balance the project ownership between entities to even this out in the bigger picture, and ensure credit to all involved at project completions.*

*Accounting needs to include both what part of the budget is allocated to each location, and how local spending gets rolled up into a global project account. How to handle deviations in planned spending needs to be included, usually overspending is taken at the location that has the project in its portfolio (for the Asset Management System this would be the location in France where the Project Manager is located). The project status communicated according to the project's **Communication Strategy** will include spending and resource issues at the various locations, as well as progress on system functionality.*

*A part of the commitment from each location must be how to accommodate visiting team members, and any needs for system testing and acceptance. Local **Team Space** should be designed to accommodate visiting team members and also allow for virtual meetings (no need to find free meeting rooms elsewhere in case of urgent meetings that are not pre-planned).*

***Commitment from All** also comes in the form of supporting the project objectives, and allowing for the project team objectives to take priority over local objectives for the location. For the global team to share objectives is totally necessary for the joint focus on the **One Project**.*

Issues: The first time a large organization is faced with the challenges of a distributed team it will most likely be a difficult and time consuming process to apply the patterns as described in the above paragraphs. The demands on the organization will challenge established and hard to change internal workflows and authority distributions. With multiple locations and organizational entities involved this can be a political challenge as *the supporting centers may feel that they do the work but lose the influence and the credit at delivery.*

Celia focuses on her team from the start. She interviews her assigned team members and decides to substitute two people: one because he has problems with travelling and flexible working hours, the other because she seems to be overly negative about working across locations. This way Celia starts with a team where the members are **Selected** to increase the probability of a well-functioning team. Since distributed development is new to almost all her team members, Celia decides to ensure that her team is well **Prepared** by sending them to an intensive training program on distributed development, as well as an internally developed session on **Culture Awareness** to improve respect and team communication.

Exploration: With the extra challenges a distributed team has, it is important that all the team members are positive to this way of working, and that they are given all the support possible to succeed. A well-known pattern in agile development is <u>Self-selecting Teams</u> where the team shares common interests and addition of new team members is a team and not a management decision. This pattern applies well for distributed development with the high level of commitment and team loyalty needed from every team member.

Issues: Even with all these preparations it is not a given that all team members will function well together, and Celia needs to keep an eye on team chemistry as well.

In the initial phase, the team decides to use SharePoint for their collaboration and to store their documents, serving as their **Common Information Infrastructure**. They also select tools to be part of their **Common Development Environment**: source control system, requirements management tools, IDE, issue tracking system etc. There are some discussions and disagreement about some tools because of previous experience, but Celia makes sure that the team reaches an agreement in a friendly way.

*Exploration: This is another area where the **Commitment From All** must apply. License costs will likely be taken by the project itself, but local resources are needed to ensure that the development tools are installed and operating locally.*

Issues: In addition to the tools, the project team needs to have clear rules for how to manage and use the tools. This will include how requirements are managed (who can add new requirements, how are requirements traced in deliverables, who is signing off that a requirement is fully implemented etc.), how documents are managed (always check out from the repository before modifications, who can modify what), and the process around accepting new code into the main developer baseline. As an alternative to SharePoint the project documents can be stored in the source control system using the same check-out/check-in mechanism, and with for instance a wiki page as portal into the documents for easy access/overview.

Just like Mark, Celia kicks off the project with their first **Together** event, and combines it with team activities for **Early Bonding** spending some of the teams **Social Funds**. Since they spend 10 days together near Paris it is not hard to find social activities, including a trip to Château de Versailles in the weekend. Celia is creating a very special memory for the team by taking them to a beautiful historic location, and of course including a good dinner with lots of conversation. It will be hard not to create friendships during this week for the team.

*Exploration: Some of the time the team is **Together** should be spent on pure social activity as this gives team members time to get to know each other personally, creating bonds* *across locations that will strengthen collaboration for later. With multiple locations and team members visiting locations in some kind of rotational order, there is opportunity for each hosting part of the team to expose visiting team members to some of their local culture.*

Issues: Social activities must be planned with respect to other team members. Bull fighting would be a valid example of an activity that people may find it difficult to enjoy.

The team decides to have 3 week iteration cycles, and get **Together** every 3rd cycle (every 9 weeks) and so to do a 5 day **Iteration Connect** every 3rd iteration. Again the team has **Smart Meetings** during the 8 weeks that they work at their respective locations, and remember to apply **Flexibility** in the work style.

Exploration: Doing the end of an iteration and the planning of the next when the team meets face-to-face has turned out to be a practice that teams find to be very powerful. This is the time of the project where the communication is most intense. The team will run the system together and align their understanding of the desired operation, as well as of how to proceed and what changes may be needed for the underlying architecture. It can be a good idea to walk through all Use Cases (or other requirement representation) for the next iteration together as a team to make sure there are no unclear points and that all team members understand the requirement details the same way.

Issues: Meeting physically with the whole team is usually not possible for each iteration (unless iterations are made long which is generally a bad idea).

A few weeks into the project some tension is starting to build in the team. The architect is upset because he feels that some of the team members in India are ignoring his directions on the architecture. The DBA team in the US seems to be out of sync with the development team in France, and these teams are accusing each other for delays in the last iteration. It becomes clear to Celia that what is going on needs more specific attention than what can be done in the **Smart Meetings**, and it is still more than 5 weeks until the team is due for another **Together** session. As she is frustrated as well, she decides to bring in a facilitator and another senior project manager to discuss with team members and analyze the problems.

After a couple of days interviewing various team members the facilitator and project manager have a pretty good understanding of the problems, which all have their root cause in communications problems. Since it is not possible for this large team to meet more often, they decide for other means to improve communication within the team. Celia manages to transfer an experienced colleague (Tom) to India for the duration of the project who takes on the role of **Mr. Mentor** for the Indian team members. Since Tom also has experience with the technology used for the development he becomes a core asset for the team. Celia also assigns somebody at each location in the role of **Mr. Connector** to reduce the risk of miscommunication between the locations. Finally the team decides to establish the set-up with the external facilitator and second project manager as a permanent **Conflict Management** solution; each time team members are frustrated they know they can contact this "crisis team" for help. The next time the team is together Celia makes sure to spend some of the **Social Funds** for the team to have fun together and get over any personal conflicts.

Exploration: It is unrealistic to think that a team of this size even without being distributed will function without any conflict for several months, especially as the pressure to deliver *increases. Using outside resources to help resolve differences ensures an impartial approach, and having an established procedure will likely mean that conflicts are managed earlier and before they get time to deepen. To have a* ***Mr. Mentor*** *involved who has experience with distributed development and can anticipate problems and who understands the team members frustrations is invaluable, and this role will pay off for the team from the beginning.*

Very often conflicts are caused by misunderstandings and different perceptions of the system to be built. By identifying individuals that serve as the knowledge hubs locally any team member knows where to go with questions. ***Mr. Connector*** *does not necessarily know the answer to every question, but he or she must know how to get to the answer within reasonable time. These roles at the various locations must keep in close contact with each other, and update each other whenever a local question is resolved. It would be beneficial to use a tool to capture these issues, as many questions may be valid for other developers as well (FAQ database, accessible and searchable by all team members).*

Issues: For small teams it may be hard to find the right people for the ***Mr. Mentor*** *or* ***Mr. Connector*** *assignments. It may be possible to get help from other people in the locations, for instance having multiple teams sharing the resource for these roles.*

Finally after 13 months, the full team and the stakeholder representatives get together for a final **Completion United** event scheduled to last 4 weeks. Extensive acceptance testing is done, and although the event has to be extended for additional 2 days, the project is successfully completed and delivered. The final evening, the full team goes out to celebrate; spending what is left of their **Social Funds** and some more. Celia makes sure her team members get **Full Credit** for their work on the project. She extends her thanks to all locations involved in the project to enable local management to take credit for the contribution of the location.

Exploration: Bringing the team and the stakeholders ***Together*** *for the final tuning is usually very efficient for a project. Preferably the testing activity should be done in a team room with continuous communication between testers, developers and stakeholders. The team can use Information Radiators[18] on the walls of the team room to show the list of bugs and needed modifications, plus the status on correcting these. For distributed teams, it is important to avoid the delays that would normally arise if the testing and customer input is at one location and corrections have to be done by developers at other locations. Valuable time will be lost in misinterpreting customer feedback when time zone differences mean that the attempted correction is only checked by the customer the next day.*

As for any other project, to celebrate success (even if the project was slightly delayed) and to give credit to everybody involved is important. For a distributed team of this size there have been many people involved in addition to the core team, and a good project manager will make sure the local management of these people are informed of their contribution (not least to ensure support on future projects). One should remember the support of functions like accounting, personnel, local administrators etc.

Issues: If the team cannot be physically at one location for the final work, they should try to emulate the ***Completion United*** *as much as possible by using collaboration tools and flexible work hours (so they are actually working simultaneously for this period even if it means working nights at some locations).*

Below is a sketch of the patterns sequence from this story. If patterns are deemed necessary they are marked in bold with a thicker line around the shape. Note that each of these patterns can be applied independently, although they mostly benefit by being used with other patterns that they relate to in the sequence diagram (like **Early Bonding** and **Social Funds**).

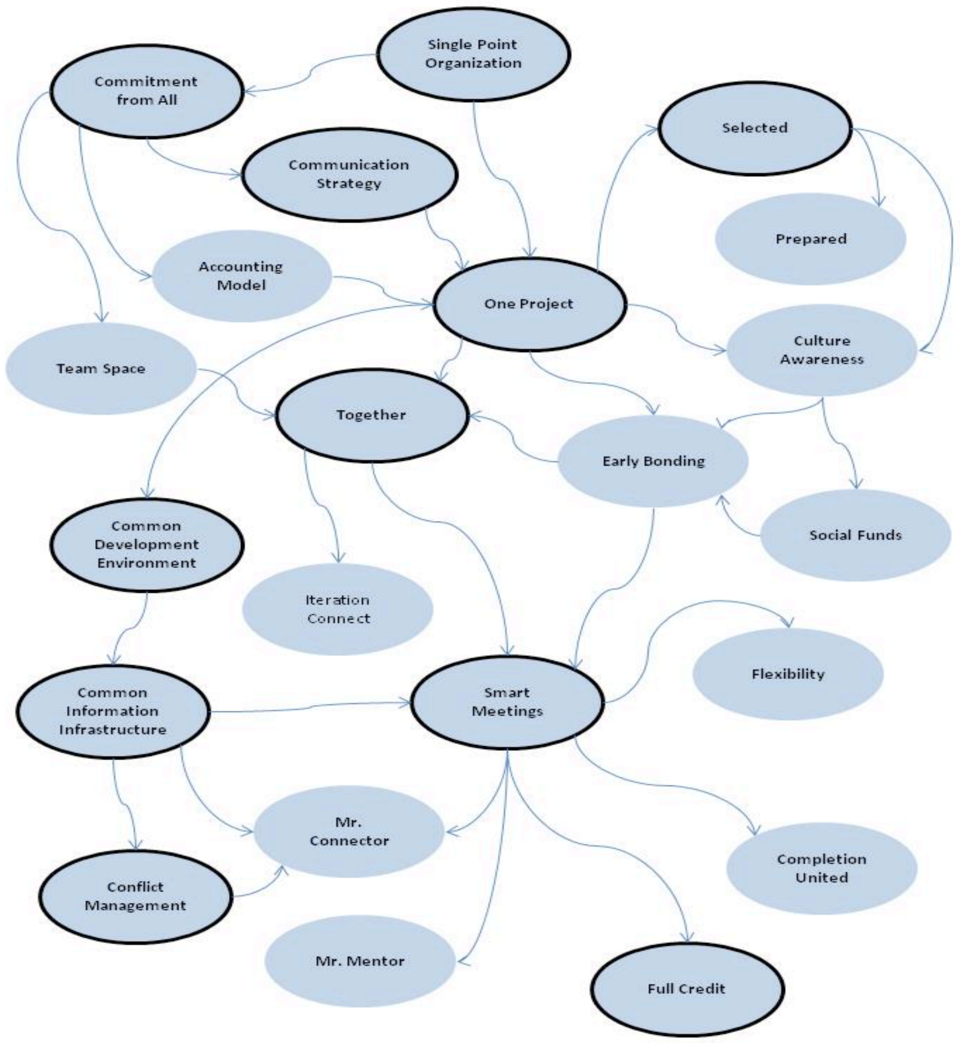


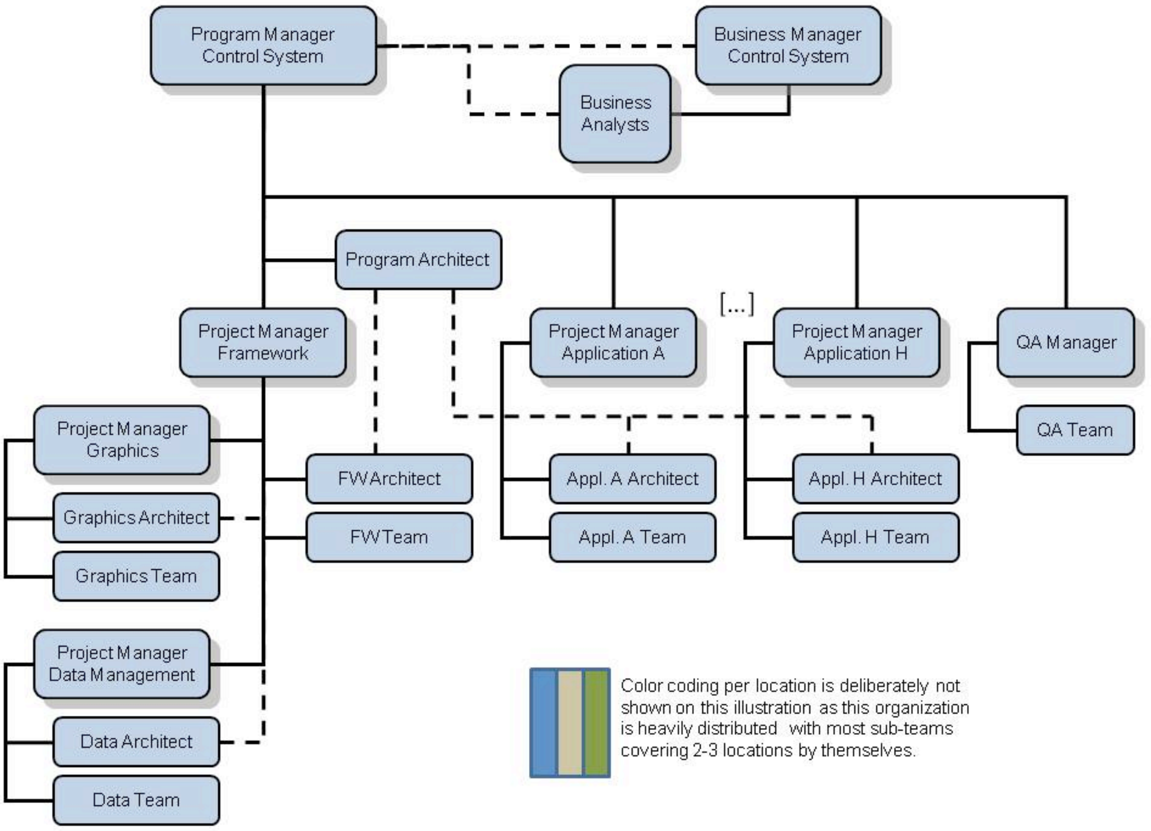
Figure 7: Patterns sequence for the Asset Management System team

Picture Source: Google Images
(sample picture for illustration purposes)

Project #3 – New Control & Monitoring System

Company C is developing a major new control monitoring system for internal use. The project has abo 120 staff years and a budget of USD 32M. It is distribut on 6 locations from the US to India, and is expected to ta 3 – 4 years to complete. It is completely new developme and although there are senior team members and manag with experience from earlier generation systems (i.e. th understand the problem domain well), the technolo chosen is new to most of the team members. This is clea a major undertaking for the company, and the success of t development is core to the survival of the company.

An undertaking of this size is really more of a program than a project, and in this case it is split in a number of sub-projects each with a project manager and an architect. A core team is developing the system framework, another team the graphics, a third team works on the data management part, and several teams are working on control and monitoring applications. The overall organization is shown below:



Program Manager Control System
Business Manager Control System
Business Analysts
Program Architect
Project Manager Framework
Project Manager Application A
[...]
Project Manager Application H
QA Manager
QA Team
Project Manager Graphics
Graphics Architect
Graphics Team
FW Architect
FW Team
Appl. A Architect
Appl. A Team
Appl. H Architect
Appl. H Team
Project Manager Data Management
Data Architect
Data Team
Color coding per location is deliberately not shown on this illustration as this organization is heavily distributed with most sub-teams covering 2-3 locations by themselves.

Figure 8: Team organization for the Control & Monitoring System team

Each of the sub-projects may again be a distributed team similar to the stories in project#1 or #2 (depending on its size) and the patterns sequences from these stories would apply. In this case the focus is on managing the large team, and so the individual team practices are not repeated here.

The development of this large system (about 3M lines of code) is still organized as **One Project** (or program) with high-level objectives shared by all. On the sub-project level this is broken down into more detailed objectives for the individual project manager or developer. In this case the **Commitment From All** is coming down from the CEO level as the future of the company depends on this project succeeding. A multi-level **Single Point Organization** is defined as is seen in the organization chart above. Although there are functional reporting lines as well, there is never more than one direct reporting line for any individual, and the authority is clearly defined along this.

Exploration: For a complex organization like this it is even more important to pay close attention to a clear definition of roles and responsibilities, communication of shared goals, and distribution of authority for decision-making. Some of the patterns mentioned in the comments section at the end of this paper aim at organizing the work with the geographical organization in mind. Conway's Law, Divide and Conquer, Loose Interfaces and Organization Follows Location are all patterns that will further improve the work situation for a distributed team and (most likely) lead to better performance and a higher quality architecture.

This organization decided to go beyond having a defined **Communication Strategy** by introducing dedicated resources assigned to facilitate frequent communication between the different entities (information champions). These resources support the program manager and the various project managers in their effort to keep the development synchronized. As part of this communication there is a published **Team Terminology** to ensure that the problem domain terms are well understood.

Exploration: Recognizing the importance of knowledge and communication within the team from the start, the Program Manager actually decided to invest in having a few individuals with team communication as their primary focus. Information was managed on wiki's, through small newsletters, and by presentations. Unfortunately, this team missed an opportunity in not providing a mechanism for bottom-up information from individuals. They focused too much on the leaders need to inform the troops, and lost some of the knowledge from individuals to managers that could make their decisions even better. These information champions had some collaboration with the ***Mr. Connector*** *roles, but their coverage tends to be more on project status, requirements/functionality status and feedback from testing and stakeholders, while* ***Mr. Connector*** *focus is more on the technical aspects of the system.*

The emphasis on frequent communication through use of information champions and the ***Mr. Connector*** *people was not only needed to keep the development team focused, but it was an important means to manage stakeholder expectations. Major delays and difficulties were communicated on a regular basis to the stakeholders, and this early notification meant that there were few bad surprises at the later part of the project. Delays were agreed when they actually happened, rather than optimistically believing that the development could catch up later. This is not to be interpreted as the stakeholders being happy about delays, as they were not… but at least they got to know early and could plan accordingly.*

A **Living Process** is evolving and discussed frequently, and kept documented in the **Common Information Infrastructure**. By re-visiting/updating the core principles of the development process every iteration, all tem members keep their understanding of the collaboration up to date as well. The fundament of the development is agile on the level of the individual teams, while for the program level it is rather the Unified Process[19] flavored with a strong focus on iterations and frequent integration into a common and verified baseline.

As can be seen from the patterns diagram below, most of the team patterns related to development process are still used on the program level, although on a sub-system level. It is not possible to bring the whole team **Together**, but representatives from each sub-system will meet at major integration points and run a meta-level iteration transition, including demonstrations of completed functionality to the stakeholders.

Exploration: Running an iterative process with major iterations every month on the system level, and additional smaller iterations within the teams turned out to work really well. A major challenge with a large and distributed team is that sub-teams deviate over time from each other, so forcing the system together on a monthly basis avoided major re-work and delays.

Another part of team communication is through the levels of assigned architects. They are supported in their effort by the **Common Information Infrastructure** allowing all team members to keep up to date across locations as the architecture is evolving; and not the least through the **Common Development Environment** where frequent code integration is supported. The system architecture is a constant focus. The initial architecture was used to structure the development assigned to each sub-project team (by applying Conway's Law; each part of the system is assigned to a project team). Interfaces between the system components are well-defined and any changes must be approved by the team of architects.

Exploration: The role of the architects and the system architecture cannot be underestimated in distributed development. An architecture with clear interfaces and few interdependencies lends itself more easily to distributed teams, and the development team assignment should map to clearly bounded system modules, which again reduces the communication need between sub-teams.

Going into details on what happened on this large project would make this paper too long. The core categories of challenges experienced throughout the development were:

- Architecture and Design consistency – keeping the architecture and design quality throughout the long development time, and communicating enough details to all developers for them to understand the framework they were developing within, and finally making sure code was not breaking architecture and design principles. This was an ongoing battle for the architects, but their presence combined with frequent integration and good communication (**Together** and **Smart Meetings**) through the **Common Information Infrastructure** and the **Common Development Environment** contributed to keeping these problems under control.
- Requirements to framework – as application work increased it became harder to prioritize and complete functionality in the framework needed to support the application. Framework changes led to needing to change applications that were believed to be completed. Eventually a formal process was put in place as part of the **Living Process** to manage interdependencies in the development which helped to decrease the frustration and reduce delays caused by rework or waiting for framework functionality.

- Stakeholder Management – not least because of the importance of this project for the company, stakeholder interest and expectations were very high, and the team had to incorporate frequent visits from upper management, project audits, presentations etc. The **Communication Strategy** and the information champions were of great help, but it still meant that the Program Manager and his staff spent a lot of their time managing out, and gave them less time to focus on the project.
- Knowledge Transfer – people coming on during the project who needed training in how to develop on this framework, tasks being redistributed, people leaving for multiple reasons, all meant that there was an ongoing need for internal training and supervision. The **Mr. Connector** and **Mr. Mentor** roles were very valuable in this context, as was the **Common Information Infrastructure** where new team members could find historic/background information as well as easy access to current status and plans and architectural modifications to understand their task in relation to the overall system.

To some surprise very few people issues surfaced on this project. It may be that the diversity and heterogeneity of the team made each team member more open to other cultures and ways of working. It seemed like the bigger challenges of delivering a system of this size contributed to bond the team members very strongly against the "common enemy" of delivering on time.

When this project finally completed after more than 5 years and a budget overrun of 20% it was still seen as a huge success by Company C. Certainly some people had left the company during this time, and a few not voluntarily. There had been difficult periods where everybody had severe doubts if this project would succeed, and the initial product scope had been somewhat reduced. But the clients were positive after the first teething problems, and the company was still solid. In the retrospective the practices around team communication and trust-building stood out as practices never to forget.

A possible patterns sequence for this story is:

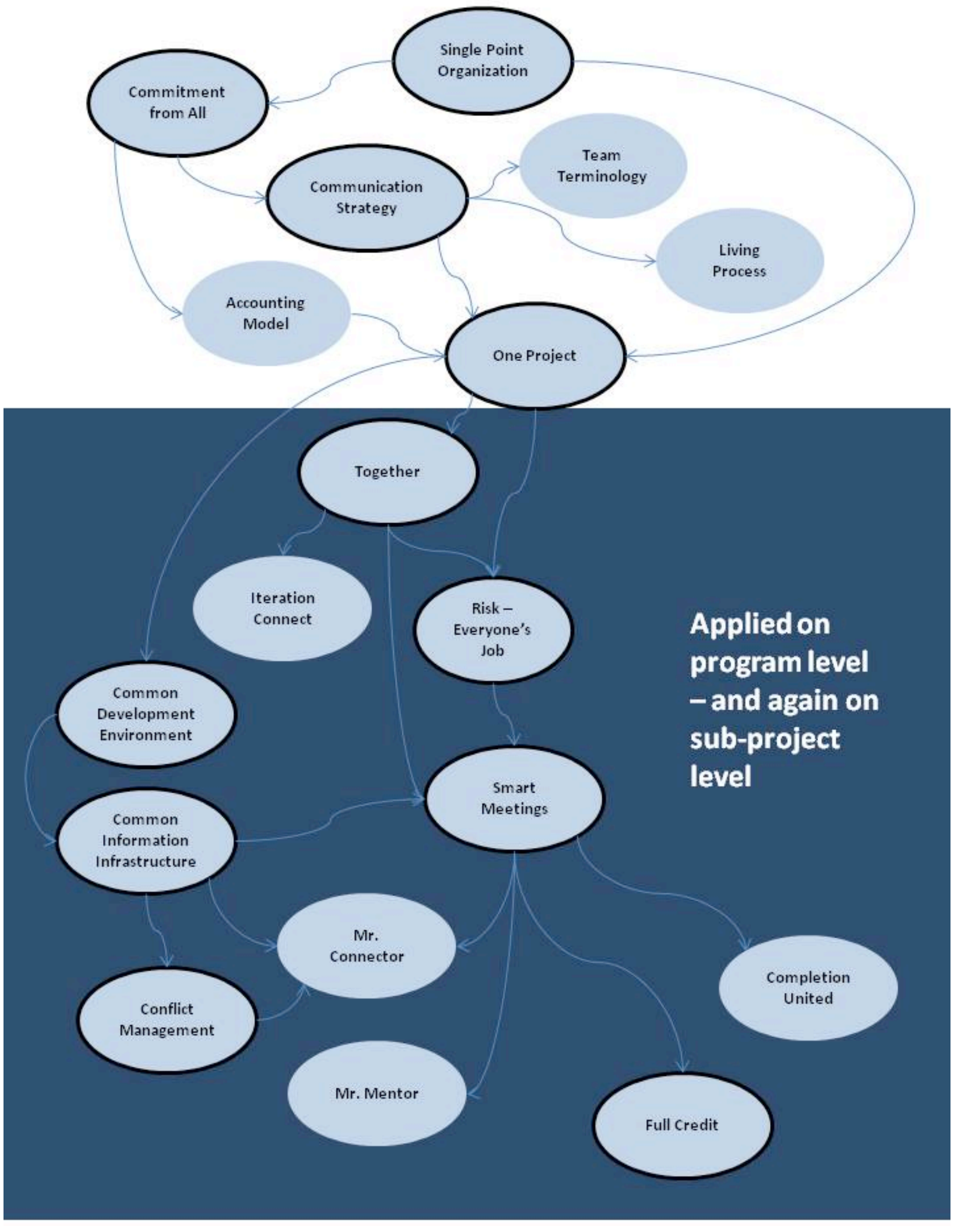


Figure 9: Patterns sequence for the Monitoring & Control System team

Other Patterns in this Collection

There are seven patterns from the thumbnails that are not used in the three stories in this paper. Most of these patterns are patterns that can be applied long-term in an international organization to create a culture that more easily would support distributed development: an international workforce with one class of citizenship independent of nationality, and global communities of practice for sharing of knowledge.

Relationship to Other Patterns

Note that the focus in this paper is on distributed development, and the stories and discussions are not extensively covering other aspects of project management or development.

As mentioned in the introduction, there are other patterns outside this collection that are of great value to distributed teams, too many to explicitly mention here. I have selected a few collections to elaborate on here, and can just suggest to interested reader to explore more pattern collections for useful practices.

The book by James C. Coplien and Neil Harrison "Organizational Patterns for Agile Software Development" has one hundred patterns that will help the performance of a software project team. Below is a short list of patterns that are especially important for distributed teams sorted into organizational focus (how to structure the team and the work) and people focus (communication and mentoring):

- Conway's Law that promotes aligning the organizational structure with the structure of the business domains reflecting the structure of the architecture
- Divide and Conquer to partition a larger organization into parts with mutual interest/focus
- Loose Interfaces to limit the number of explicit, static interfaces to allow for more rapid development
- Organization Follows Location to organize work so that people who are co-located work on the same subjects
- Subsystem by Skill recommends organizing sub-teams based on their skills
- Form Follows Function to create domains of expertise
- Size the Organization to balance good communication and the ability to mentor new people joining (start with about 10 people on the team)
- Producers in the Middle to make sure the producer roles are at the center of communication as they need to have necessary information to ensure the right product is being developed
- Named Stable Bases promotes having stable code bases for people to work against to give more stability for their development; needs to be carefully applied in combination with Conway's Law and Organization Follows Location, as well as the attitude to continuous integration
- Community of Trust that aims to build a foundation of trust and respect to benefit effective communication
- Face to Face Before Working Remotely as emphasized in this paper as well,
- Unity of Purpose to make sure all agree on the purpose of the team from the start
- Day Care where one person is assigned the role of mentoring the novices because this is more effective than distributing the task among all the experienced staff
- Compensate Success is the basis for the Full Credit pattern in this paper, emphasizing on the importance of rewarding any success for the team
- Team Pride which is important for any team but even more when the team is particularly challenged which is usually the case for distribution

- Self-Selecting Teams where the team members share common interests also outside work, and where the selection process is a team effort and not a manager decision

Acknowledgements

A big thank you to Till Schümmer for his very thorough work as a shepherd for EuroPLoP 2009! I also want to thank Amir Raveh for his initial shepherding of this paper for EuroPLoP 2008, for giving me good feedback and surviving my slow progress. I am grateful to shepherds of earlier work that has led up to the patterns in the thumbnails (Jens Coldewey, Mary Lynn Manns, Neil Harrison, Didi Schütz and Joe Bergin), and of course all the workshop participants giving valuable feedback to the patterns.

Related Reading

1. "Organizational Patterns of Agile Software Development" by James O. Coplien and Neil B. Harrison, ISBN 0-13-146740-9, Pearson Prentice Hall 2005
2. "Using an Agile Software Process with Offshore Development" by Martin Fowler, http://www.martinfowler.com/articles/agileOffshore.html
3. "Global Software Development Handbook" by Raghvinder Sangwan, Matthew Bass, Neel Mullick, Daniel J. Paulish, and Juergen Kazmeier, ISBN 0-8493-9384-1, Auerbach Publications, 2007
4. "Can absence make a team grow stronger?" by A. Majchrzak, A Malhutra, J. Stamps and J. Lipnack, Harvard Business Review, May 2004
5. "Interaction Patterns of Agile Development" by Jens Coldewey, 2004
6. "Capable, Productive and Satisfied" by Paul Taylor, PLoP 1998.
7. "201 Principles of Software Development" by A.M. Davis, McGraw-Hill, 1995
8. "Culture Clash – Managing the Global High-Performance Team" by Thomas D. Zweifel, ISBN 1-59079-051-0, Swiss Consulting Group 2003
9. "Global Teams – How Top Multinationals Span Boundaries and Cultures with High-Speed Teamwork" by Michael J. Marquardt and Lisa Horvath, ISBN 0-89106-157-6, Davies-Black Publishing 2001
10. "Mastering Virtual Teams – Strategies, Tools, and Techniques that Succeed" by Deborah L. Duarte and Nancy Tennant Snyder, ISBN 0-7879-5589-2, Jossey-Bass 2001
11. "Trust within Global Virtual Teams" by Olivier Chavaren, ISBN 0-595-27577-X, iUniverse, Inc 2003
12. "Virtual Teams – Reaching across Space, Time, and Organizations with Technology" by Jessica Lipnak and Jeffrey Stamps, ISBN 0-471-16533-0, John Wiley & Sons 1997
13. "The Manager's Pocket Guide to Virtual Teams" by Richard Bellingham, ISBN 0-87425-615-1, HRD Press, Inc 2001
14. "Global Software Development – Managing Virtual teams and Environments" by Dale Walter Karolak, ISBN 0-8186-8701-0, IEEE Computer Society 1998
15. "Managing Virtual Teams – Practical Techniques for High-Technology Project Managers" by Martha Haywood, ISBN 0-89006-913-1, Artech House 1998
16. "Working Virtually – Managing People for Successful Virtual Teams and Organizations" by Trina Hoefling, ISBN 1-57922-032-0, Stylus Publishing 2001
17. The Distance Manager – A Hands-on Guide to Managing Off-Site Employees and Virtual Teams" by Kimball Fisher and Mareen Duncan Fisher, ISBN 0-07-13065-4, McGraw-Hill 2001
18. http://alistair.cockburn.us/Information+radiator
19. http://en.wikipedia.org/wiki/Unified_Process

Appendix: Pattern Thumbnails

The short descriptions below are based on the author's earlier papers at EuroPLoP and PLoP conferences, but include some that are not yet published or submitted (ongoing work). A major revision is planned for some of the earlier work.

1. **Relocation and Rotation**
   You need an international workforce with a strong common identity that collaborate well and that is not having a "them" and "us" mentality.
   Relocate a part of your workforce on a rotational basis to other engineering centers. This will build relationships and trust across locations, and enforce common work methods.
2. **Balance of Nationalities and Minorities**
   You want a workforce that is representative of your business involvement worldwide, and where individuals are equally respected independently of their background.
   Hire with a clear goal that the different nationalities represent the amount of involvement in a country or region. The distribution of nationalities must be reflected in the management organization all the way up to the CEO.
3. **One Class of Citizenship**
   You want a workforce with a strong common identity, where individuals are equally respected and feel equally valuable to the organization.
   Make each location comply with the overall company policies and give all employees the same opportunities and the same benefits within what is possible given local laws and regulations.
4. **International Communities of Practice**
   You need to fully benefit from the knowledge and creativity of your whole workforce to stay competitive. You want the participation of every individual, and that they actively participate and collaborate with other experts within their application domain.
   Grow and support technical communities of practice on the global level. Provide means of networking through blogs and wikis, and enable workshops and conferences where members can physically meet.
5. **Benefit Target**
   The success of distributed teams is heavily discussed. Results are often questionable, and the drawbacks often by far outweigh the achieved benefits.
   To be able to keep the focus on the expected benefits and to determine achieved improvements, define 3 major targets for the expected benefits that are not in conflict with each other. The benefits must be specific to the organization and allow the organization to determine afterwards if each targeted benefit was reached.

6. **Commitment from All**
   Even if the project is distributed over several locations, the management interest and attention may be stronger in some locations. This may lead to lower priority and lack of resources in the other locations.
   Any engineering center involved must have a clear commitment to the project. In the initial phase, a clearly defined involvement must be agreed between the involved centers.
7. **Single Point Organization**
   Running projects in a distributed way often results in confusion and frustration for the involved parties. Conflicting objectives and unclear authority and responsibility distribution are just examples.
   The complexity can be reduced and controlled by a clear and communicated organization. This is more than defining the roles and responsibilities: it is key to ensure that the decision-making authority lies with only one manager at each level in the organization.
8. **One Project**
   The daily life of team members will be influenced by the location. Developers may find conflicts between local objectives and what is expected as product deliveries.
   Create a clear project identity across the centers. All team members share the same team objectives.
9. **Communication Strategy**
   Distributed teams may have a more complex set of management and stakeholders, and how/when/whom to keep informed may be a challenge.
   The team needs to work out a clear communication strategy that lists who to inform, what each are expecting, and the communication means.
10. **Accounting Model**
    A project team depends on efficient accounting "services" in the organization to keep track of spending versus funding. These services are rarely set up to support a team on multiple locations.
    Set up a meta-level model of the accounting that defines how to manage the project financial status, and assign a clear owner within the accounting organization.
11. **Common Information Infrastructure**
    A project team must share information across the complete team, not based on physical location.
    Set up an information structure for the global team that enable consolidated reporting of status and easy sharing of information.
12. **Common Development Environment**
    Distributed teams have the same needs for baseline management as co-located teams. Keeping the code-base in sync requires immediate updates at all locations.
    Carefully identify, select and implement one single common development environment for the project team to use regardless of their location. All components of this common development environment must support multiple locations with different time zones, and come with worldwide 24/7 support.
13. **Culture Awareness**
    All cultures have their unwritten rules, and there are many examples of collaboration problems that stem from a lack of understanding of ethical and behavioral code. Note that the culture aspect includes organizational culture!

Since we are not all born social anthropologists, some specific training may be necessary to learn to "read" team members from a different culture, and to ensure respect and good working relations.

14. **Selected**
    Working on a distributed team is demanding, and requires flexibility in work hours, ability to travel and stay at other locations, and good social skills for the collaboration within a very heterogeneous team.
    Team members should be screened for personality and the team carefully built over time for maximum cohesion.
15. **Prepared**
    Distributed teams have some added challenges, especially related to communication. The negative effect of not being co-located can be made significantly smaller by applying good practices from the start of the project, so make sure all are trained on distributed development from the start.
16. **Social Funds**
    Team members cannot be expected to privately pay for social team events. These events are important for the bonding and thereby for team efficiency.
    Make sure funding is allocated to build personal relations between the team members (team building, celebration of achieved milestones etc.).
17. **Mr. Mentor**
    There is much to learn from colleagues that have managed distributed teams before, but this learning may need some support from the organization. Each individual does not necessarily know who to ask, or feel that they can "bother" colleagues repeatedly for advice.
    By assigning a mentor with experience in distributed development to a new team, ongoing and on-the-job learning is made possible.
18. **Mr. Connector**
    Distributed teams struggle with keeping on the same page between locations.
    Instead of everyone on the team trying to be on top of all ongoing issues, designate a team member at each site to manage the flow of information, and who knows how to get answers to different kinds of questions.
19. **Living Process**
    A shared development process adapted to the team's needs becomes even more important with the increased challenge and need for formality.
    Spell out the fundamental values, methodology and techniques used and keep it up-to-date for all team members (revisit at each iteration).
20. **Early Bonding**
    Trust and personal connections need to be built early and are key to have a functional team.
    Emphasize on building the team from the start of the project, and include social events.
21. **Smart Meetings**
    If co-located, you can do frequent progress meetings, and have spontaneous get-togethers when needed. In a distributed setting meetings need to be organized up front. But your communication needs are still there.
    By allocating common time and making team members available to each other at

agreed times during the week, you can mimic spontaneous contact (with a delay), and you ensure frequent communication in the team (several times per week) to avoid locations drifting apart on issues.

22. **Together**
    Even with Smart Meetings, communication suffer by not being co-located.
    Make sure the team physically meets frequent enough during development. For 1-2 year software projects, we have come to believe a 6-8 week frequency is the best compromise between all effects of traveling and the need to meet, each time staying together for about 10 days.
23. **Iteration Connect**
    The most busy communication phase in a project is at the end of an iteration cycle, plus during the iteration assessment and the start of a new iteration.
    By synchronizing the length of the iteration (or a multiple of iterations) with the frequency of the physical team meetings we found the meeting time was spent most effectively.
24. **Completion United**
    The final phase of a project is again a time where good and fast communication is important. System testers doing the final verification face significant problems when the team is distributed.
    We have found it amazingly effective to bring the development team together with all the testers and stakeholder/user representatives on a single site for completion.
25. **Flexibility**
    The load of communication with other locations that frequently have to happen outside work hours is a big load on employees and on their families. The workday often gets very long, in reality working one normal day plus an extra shift early or late in the day.
    To balance the effort and get closer to a normal workload, redefine the team's work hours and give the team members flexibility outside common team time.
26. **Short Engagements**
    Short-term assignments at the other location will enable team members to get to know other team members better.
27. **Conflict Management**
    Choose and publish a pre-defined way of managing conflicts, possibly involving an independent party outside the core team.
28. **Full Credit**
    It is amazing that this one needs to be written down, but experience tells us to do it: Make sure <u>all</u> team members get credited for success.
29. **Team Space**
    Local team rooms need to accommodate visiting team members, VC equipment need to be easily available etc.
30. **Team Terminology**
    Make sure you do not get local "dialects" and that everyone understand the lingo of the problem domain.
31. **Pilot Solutions**
    Many of the practices we recommend have a certain cost associated, although we firmly believe they pay back multiple times over. But, there is management to convince.

To gain experience, try out a new solution on one team first. Be sure to record the results well.