A Global Teaming Model for Global Software Development Governance: A Case Study

John Noll, Sarah Beecham, Ita Richardson  
Lero, the Irish Software Engineering Research Centre  
University of Limerick, Ireland  
Email: {john.noll,sarah.beecham,ita.richardson}@lero.ie

Clodagh Nic Canna  
Ocuco Ltd., Ireland  
Email: clodagh.niccanna@ocuco.com

Abstract—Global software development (GSD) is increasingly the norm. As firms expand into new markets overseas, acquire companies in different countries, and recruit talented developers in new locations, projects become distributed by necessity. As projects become increasingly distributed, and include external participants such as outsourcing vendors, conventional top-down management oversight and control becomes more difficult. How can organizations ensure that the activities of all parties involved are aligned with the strategic objectives and values of a software development undertaking? The Global Teaming Model is a framework that specifies practices and recommendations for Global Software Development. As such, it serves as a means to organize GSD practices relevant to Software Development Governance.

We conducted a case study of a small team engaged in Global Software Development, identifying governance shortcomings. Then, we used the Global Teaming Model to identify governance practices that would address those shortcomings. We identified several aspects of the team’s interactions with other teams in the company that would be improved by recommendations from the Global Teaming Governance Model. The Global Teaming Model provides a blueprint for Software Development Governance that organizations seeking to globalize their current development projects in a controlled way can use to implement good Software Development Governance.


I. INTRODUCTION

Global Software Development (GSD), involving geographically distributed teams and individuals, is increasingly commonplace. No longer solely an approach to reduce costs by outsourcing development to lower-cost countries, GSD is the inevitable result as organizations expand into new markets, acquire companies in different places, and hire talented developers in remote locations.

Due to reduced communication, remote teams must be more autonomous than their co-located counterparts. Further, while acquisitions might make a company legally part of another, the development teams may retain their independence and identity, at least initially. Finally, companies are forming closer relationships with their user community, as well as open source communities, seeking contributions from individuals who have no formal relationship with the company [1]. As a consequence, Global Software Development efforts evolve toward “virtual organizations” [2] that feature less structure and more autonomy. Working in such an environment requires “a shift from ownership (I have complete control and can dictate what’s best) to stewardship (I have a responsibility to govern, to collaborate for the common good)[3].”

In this climate, Global Software Development efforts need governance to ensure the activities of each team or individual are consistent with the overall strategic goals and organizational norms. Good governance defines clear decision-making authority, roles and their responsibilities, task allocation, and practice alignment with organisational or business goals.

Software Development Governance (SDG) addresses these needs as they apply to software development in particular. However, although there exist models and standards for IT Governance [4, 5], research in Software Development Governance is less mature, and models for Global Software Development governance are missing. As such, it can be difficult for an organization embarking on a Global Software Development effort to create good governance.

Software development is a highly creative activity and therefore when implementing governance procedures, there needs to be a balance between the creative and technical aspects [6]. Also, a high level of formality is not always necessary [7] or even desirable: “Centralization or formal structure negatively influences knowledge sharing . . . in GSD [8].”

The Global Teaming Model (GTM) [9], was created by some of the authors of this paper as a CMMI®-style “process area” to establish goals and practices for Global Software Development, based on empirical studies of GSD. The Global Teaming Model defines 64 recommendations for conducting Global Software Development; these recommendations address organization, communication, decision making, and measurement. In our previous work, the GSD helped describe related research topics such as GSD Architectural Knowledge Management [10], and motivation of the global software engineer [11]. As such, we hypothesize that the Global Teaming Model can be used to define Software Development Governance for a software development effort, by defining decision making structures, processes, and metrics.

To test this hypothesis, we performed a case study of a small global software development team. As part of this study, we identified issues the team was facing. Then, we assessed the degree to which the team, and organization, implemented
the recommendations of the Global Software Development. Finally, we compared these issues to the GTM to see which issues might be addressed by GTM recommendations that the team or organization has not fully implemented.

We found 24 recommendations that were not implemented, or only partially implemented. We also identified 6 issues that could be the result of a lack of, or incomplete, Software Development Governance. Of these, 5 would be addressed by the missing or partially implemented GTM recommendations.

These results suggest that the Global Teaming Model is an effective framework for organizing Software Development Governance recommendations for Global Software Development

The remainder of this paper is organized as follows. In the next section we examine Software Development Governance, and provide an overview of the Global Teaming Model. Following that, Section III presents the method used for this study. Then, in Section IV we discuss the case study, results, and limitations. Finally, Section V offers some conclusions, and directions for future work.

II. BACKGROUND

A. Software Development Governance

In general, governance can be defined as the “the arrangements and practices that an organization puts in place to ensure its activities are adequately and appropriately managed [12].” These arrangements include specification of roles and their decision rights [1, 13, 14], and metrics and procedures for monitoring and assessing decisions [1, 7, 13, 15]. Thus, governance has both a structural aspect, comprising the roles, chains of decision-making, and metrics, and a dynamic aspect, comprising monitoring and control of decisions [13].

Based on this general view of governance, we define Software Development Governance as the specification of:

1) who makes what decisions about a software project (structure),
2) what measurements are used to inform and monitor those decisions (monitoring), and
3) when decisions are made and metrics collected (process).

There is also a push towards including the how of decision making to this list [14], i.e. how will these governance decisions be made and monitored [7] which forms part of the governance process.

Taking a lifecycle process view of governing software development, Dubinsky and colleagues [16] note four main phases [2,5,7]: “(1) setting goals and assigning roles and decision rights;... (2) deploying measurements, policies, and controls...; (3) executing these mechanisms in practice; and (4) assessing the execution of the mechanisms and using these assessments to refine the project goals.” This SDG lifecycle is iterative and aims to align the development process with business goals.

1) Governance vs. Management: What separates software development governance from software project management? The difference is similar to the difference between software requirements and implementation: in the same way that requirements specify what needs to be implemented, and implementation defines how, governance defines what decisions are made by whom, and what metrics are used to monitor those decisions; management is the act of decision making, and the collection of metrics that inform those decisions and measure their outcomes. A third dimension, process, specifies when decisions need to be made, and when measurements are made.

As an example, from a governance perspective, Scrum [17] defines roles for a software development team: Product Owner, Scrum Master, Team Member. Each of these roles has certain decision-making responsibilities: the Product Owner sets priorities for features of the product, and selects the features to be implemented in each sprint, within the constraints of the team’s “velocity”; the latter decision takes place during the sprint planning meeting, and is fixed for the duration of the sprint.

Team members are responsible for estimating the effort required for each feature in terms of “story points” during the backlog refinement meeting. Project progress is measured by the Scrum Master as the number of story points completed; this is used to both monitor the overall progress of the product toward completion, and by the Product Owner to select the features for the next sprint.

In this way, Scrum specifies the governance structure, as defined roles with distinct decision-making responsibilities. It defines the governance metrics, as story points completed (among other things). Finally, it defines governance process, as a set of nested iterations and “ceremonies” that specify when decisions need to be made and when metrics are collected.

These are summarized in Table I

<table>
<thead>
<tr>
<th>Governance dimension</th>
<th>Scrum component</th>
</tr>
</thead>
<tbody>
<tr>
<td>Structure: decision making roles &amp; decisions</td>
<td>Roles: Product Owner – feature priority; Team Member – feature estimates, implementation design; Scrum Master – metrics, resources</td>
</tr>
<tr>
<td>Monitoring</td>
<td>Metrics: Features implemented, story points completed, velocity</td>
</tr>
<tr>
<td>Process: decision events</td>
<td>Sprint planning meeting, backlog grooming, daily scrum</td>
</tr>
</tbody>
</table>

TABLE I

For example of Software Development Governance.
implicit by informal communication: staff can infer the governance structures, metrics, and process simply by observing who makes decisions, what information they base those decisions on, and how the development process facilitates decision making and monitoring.

In a Global Software Development context, explicit governance is essential because of the communication barriers imposed by geographic, temporal, and cultural distance: there are fewer opportunities to see how governance shapes the project due to the lack of informal encounters common in colocated projects. Also, because global software development can include participants who are not part of the “home” organization, decision making structures must respect the autonomy of these participants, as well as the necessity of remote teams to act independently outside of home office working hours. Finally, metrics must account for confidentiality needs of independent participants, as well as local laws and culture. In such an environment, governance acts like “environmental pressure” that shapes decision making rather than controlling it absolutely.

While Software Development Governance (SDG) comes under the banner of the wider, more researched and mature area of IT Governance, it has distinct properties that according to Dubinsky et al. [16] need to be investigated separately. Areas such as how risks and conflicts are managed, the cost benefit of software development practices (such a Scrum) and how software development frameworks such as the CMMI, or in our case the Global Teaming Model, support SDG are still needed, especially in a distributed team setting. As noted by Dubinsky et al [16], more work is needed that considers how different global models affect governance frameworks and tooling.

B. The Global Teaming Model

The Global Teaming Model (GTM) is a model for global software engineering, with particular emphasis on organization and management of globally distributed development teams. The Global Teaming Model follows the hierarchical structure and nomenclature of the CMMI® [20] (see Fig. 1). At the highest level there are two broad goals, “Define Global Project Management” and “Define Management Between Locations.” These goals are decomposed into Specific Practices that define broad categories of practice that lead to the parent goals.

Specific Practices are further elaborated into Sub-practices. Finally, Sub-practices have one or more recommendations that specify detailed actions to be taken. In total, the GTM has five Specific Practices, twenty Sub-practices, and 64 recommendations, that have been validated against a real industrial case [11, 21].

The first GTM goal – “Define Global Project Management” – recognizes that global project management, while encompassing the expected tasks of any project management setting, must also include new tasks related to managing a virtual software engineering team comprising distributed individuals, teams, suppliers, communities, etc. The first goal comprises three Specific Practices: “Global Task Management,” which defines how distributed teams are organized into a virtual organization structure, and how tasks are allocated to teams; “Knowledge and Skills,” which concerns both project knowledge management and staff training; and “Global Project Management,” which defines decision-making roles and responsibilities in the virtual organization.

The second goal, “Define Management Between Locations,” focuses on communication and collaboration among distributed participants. This is achieved through two Specific Practices: “Operating Procedures,” which is concerned with the way communication between teams, and meetings involving more than one team, are conducted; and “Collaboration Between Locations,” which focuses on how distributed teams work together to set objectives, establish responsibilities, and plan work.

1) Global Teaming and Software Development Governance: Ramasubbu and Balan [22] noted the need to adapt normative models (such as the CMMI) for distributed development. The Global Teaming Model is one such model, and recognizes that distributed projects will be governed differently to co-located projects.

The Global Teaming Model defines what needs to be done for a global software development effort to be successful. In total, the GTM organizes 64 recommendations into a hierarchy of practices that define a Global Teaming “Process Area” that can be used as a guide for starting and managing a GSD virtual organization. The Global Teaming Model is supported by tools to automate the selection of GTM practices [23–25].

The Global Teaming Model does not, however, specify how practices and recommendations should be implemented. This is by design: factors such as product domain, geographic target markets, virtual organization composition, culture (organizational and national) of the participants, and technology, together determine how specific recommendations should be implemented. And it is the selection of a set of governance activities that will influence communication and knowledge management between distributed software development teams [26]. As such, the Global Teaming Model can be viewed as a governance model, because it focuses on the what rather than the how of Global Software Development.

In summary, while the literature in SDG is rich in definitions and challenges, we find few recommended practices. As noted by Dubinsky et al [16], there is a need for more SDG empirical studies to be conducted within companies in which development is a core part of the business, this should focus on assessments of members of project and software teams, in addition to assessments at the organizational level.

III. Method

To assess the Global Teaming Model as a framework for Software Development Governance, we used the GTM to assess the governance of a software team engaged in Global Software Development of a product in the healthcare domain. This research was part of an ongoing software process improvement study we are conducting in cooperation with the company’s
engineering management. The study employs a participant-observer approach in which one of us engaged one of the company’s software development teams with “moderate” participation [27]. The observer performed the following activities:

a) Observe “ceremonies”: As the team followed the Scrum [17] development methodology, Scrum ceremonies were observed over a three month period:

- Sprint Planning - during this meeting, the “Product Owner” selects features to be developed during the next two-week development “sprint.” This is done with participation from the rest of the team to ensure that effort is balanced.
- Daily Scrum - this meeting occurs daily for 15 minutes in the morning, during which team members review their accomplishments of the previous day, report any obstacles, and state what they intend to do for the current day.
- Backlog Grooming (Refinement) - the team spends two hours every other week examining open issues (the “Product Backlog”) to understand implementation and other details of each issue, and agree an effort estimate.
- Sprint Retrospective - this meeting occurs at the end of each two-week development sprint; during this meeting the team reviews what went well and what needs improvement, and agrees any process changes that need to be made for the next sprint.

b) Conduct interviews: Each team member was also interviewed individually, either in-person or via video conference. Interviews lasted between one and two hours and covered the development process from the team member’s point of view.

c) Examine artifacts: Where applicable, observations and statements made during interviews were compared to the artifacts involved, namely entries in the issue tracking database. Also, company process documentation was examined, in part to understand terminology, and also to compare the desired process to reality.

d) Document process & feedback: After two months of observations, an “as is” process description of the team’s actual development process was created using the PML process modeling language [28]. The team was assembled at the head office to review and comment on this description; corrections and elaborations were incorporated into a revised description and circulated later for validation.

e) Identify issues: Ceremony observation and interview notes were also examined to identify issues facing the team; the list of issues was discussed with the team during the company’s annual on-site meeting.

f) Assess GTM implementation: The company and team’s software development process, management, and activities were assessed against the 64 Global Teaming Model recommendations. Using observations and interview notes, as well as process documentation, we assigned a value to represent the extent to which each recommendation in the GTM has been
implemented, ranging from fully or mostly implemented, to minimally or not at all implemented.

5) Compare issues to missing GTM practices: In the final step, the issues identified in the previous step were compared to the GTM recommendations that had not been implemented. For each such recommendation, we made a qualitative judgement as to whether the GTM recommendation would have prevented or mitigated the issue in question.

The results of this analysis are presented in the next section.

IV. CASE STUDY

We studied a small team of six developers distributed across Britain and Ireland. This team, which we will refer to as TeamAB, is part of the software development organization of an Irish SME that markets healthcare practice management software. The company, which we will call PracMed for anonymity, is pursuing an aggressive growth by acquisition strategy; over the past three years alone they have acquired three competitors, whose software development teams must be integrated into the company. As a result, PracMed is experiencing growing pains as it tries to incorporate acquired products into both its product offerings and code base.

TeamAB comprises three members who continued after the earlier acquisition of a UK competitor, plus three new members hired after the acquisition. Post acquisition, TeamAB is responsible for maintenance of both the legacy product of the acquisition, and PracMed’s core product. TeamAB is also responsible for new features in the core product that are targeted for small practices in the UK, Ireland, Sweden, France, and Mexico. Separate teams develop features for customers in other parts of North America and Europe, and for large practices who pay for specific customizations.

TeamAB comprises six members who originate from four countries on two continents, and have four different native languages. Two team members work from home in Britain, two work at a company office in Britain, and two at the corporate headquarters in Ireland.

PracMed has a dozen software projects distributed across Europe and North America, involving approximately 50 developers. TeamAB was the first team to adopt Scrum [17] as part of PracMed’s transition to Agile development. At the time of our study, TeamAB had been following the Scrum approach for six months.

A. Issues

TeamAB has been largely successful in their transition to Agile development. However, they are only six months into this transition, and so are still experiencing a number of issues, including several that are a direct result of Global Software Development.

1) QA Staff: The first and most pressing issue TeamAB faces is a shortage of testers. TeamAB comprises three programmers, a Product Owner, a Scrum Master, and one test engineer. However, because PracMed does not employ automated testing, testing often becomes a bottleneck, resulting in features implemented near the end of the sprint sometimes slipping into the next sprint due to lack of testing.

2) PO Dual Roles: PracMed is not able to dedicate a person full-time to be TeamAB’s product owner. As such, this person must also perform customer support duties for about half of his working hours. This has two consequences: the team occasionally has to postpone certain Scrum ceremonies such as sprint planning or retrospectives due to the Product Owner being on support duties.

Lehto and Rautiainen [29] warn against splitting the product owner’s role and responsibilities, but TeamAB’s Product Owner sees this dual role as a positive, since he gains first-hand contact with customers, and also has a wider variety of duties which he considers to be more stimulating.

3) Estimates: A large portion of TeamAB’s activities are maintenance tasks that require investigation of the cause of failures. As such, it is difficult to estimate the time required to implement a fix since the cause is unknown at estimation time. Also, PracMed uses “story points” rather than time as the unit of estimation. A story point is a measure of effort; TeamAB has several reference features with different values that the team uses to establish the effort required to implement a new feature or fix, by comparing it to a reference feature. However, this process seems to be highly variable, and some team members have commented that they would prefer to use time for estimating. The end result is that the sprint backlog is frequently too ambitious and the team fails to meet the target.

4) Motivation: TeamAB has a high degree of team cohesion, and despite the variety of backgrounds and experience, members are highly supportive of each other and the team objectives. However, when interviewed, team members reported neutral motivation (neither low nor high), suggesting there may be some organizational or process issue that is dampening motivation.

5) External Schedule Pressure: A Scrum principle states that the sprint backlog is fixed for the duration of the sprint [17]; new features or issues that arise mid-sprint are placed on a backlog for consideration in the next sprint. TeamAB uses two-week sprints, which means a fix for a critical issue might be delayed two weeks or more. This causes some tension between TeamAB and other teams that are developing customer-specific enhancements, since TeamAB is responsible for fixes to the core functionality for all products. Consequently, customer-specific enhancements that depend on core functionality may be delayed until a fix is scheduled and implemented by TeamAB.

6) Errors Introduced by Other Teams: PracMed’s configuration management follows a Mainline pattern [30] in which all teams commit their changes to a single trunk codebase. Branches are made for releases, which only accept fixes to failures (not new features). Having a single trunk means that each project stays in sync with others. However, because PracMed does not employ automated tests, changes made by one project can potentially have adverse effects on other projects that go undetected. This is due to the fact that test engineers on a given project focus on a project’s functionality, and do not necessarily have the expertise (or time) to test the behaviour of all other products. TeamAB experienced the consequences of
this in several sprints observed during the case study period, including one extreme case where a release was delayed two months by a change made by another team that effectively negated a complicated fix made by TeamAB to a persistent issue.

B. GTM Assessment

After observing TeamAB for three months, and interviewing each team member, we assessed PracMed’s organization and procedures against the Global Teaming Model. Using our observations and interview notes, we assigned a value to represent PracMed’s implementation of each of the 64 recommendations specified by the Global Teaming Model (see Table II and Table III). We rated each practice in Table II or Table III as ‘A’ if, according to our observations, the practice is fully or mostly implemented, ‘B’ if partially implemented, or ‘C’ if not, or only minimally, implemented. Recommendations considered not relevant to TeamAB are shown in strikeout text; if we did not have sufficient information to make an assessment, a ‘?’ is shown in column 1.

Overall, PracMed has partially or fully implemented 39 of 57, or 68%, of practices considered relevant to PracMed’s context (7 were deemed not relevant to PracMed). PracMed exhibited strength in many process areas, particularly those related to Defining Global Project Management (Goal 1, Table II); however, the practices associated more with day to day operations, “Define Management Between Locations” (Goal 2, Table III) were variable. Of particular concern are those practices associated with implementing strategies for conducting meetings between locations (Sub Practice 2.1.4), and defining how conflicts and differences of opinion between locations are addressed.

C. Discussion

As can be seen from Table II and Table III, there are 13 GTM practices that have not been implemented. Following is a discussion of how these practices (should they be implemented in the future) might resolve or mitigate the issues faced by TeamAB.

Three of the six issues – motivation, external schedule pressure, and errors introduced by other teams – are related to governance and can be addressed by GTM practices; the other two are mainly management, rather than governance, issues. We discuss these in more detail below.

1) Governance-related Issues:

a) Motivation: Motivation is partly addressed by the following GTM recommendation under Sub Practice 2.2.1: “The global team is viewed as an entity in its own right, regardless of the location of its team members and its performance should be judged and rewarded accordingly.”

Members of TeamAB view the team’s function as doing the work that “nobody else wants to do,” namely maintenance of the two products for which they are responsible. Maintenance effort is often seen as a burden [31] and a de-motivating activity [32]. Although TeamAB does some new development of features requested by small practices in the markets for which they are responsible, development for large customers is done by dedicated teams, mainly in North America. Because they share a common codebase, these dedicated teams have on occasion committed changes that break features for which TeamAB is responsible, disrupting TeamAB’s release schedule. Some TeamAB members feel that management holds TeamAB responsible for these schedule slips.

The GTM recommendation to judge a team’s performance on its own merits might improve motivation, as team members might feel more in control of their situation.

b) External Schedule Pressure: External schedule pressure results from conflicting priorities between TeamAB and other teams. This could be alleviated by better communication among teams, as suggested by these GTM recommendations:

- Set up a strategy to handle, monitor and anticipate where conflict between remote locations may occur. The strategy should include how conflict will be resolved and how a person responsible for that resolution is selected.
- When defining the global strategy for dealing with conflict, different types of conflict have to be taken into account, for example conflict due to fear as well as cultural differences.
- Plan, facilitate, encourage and monitor communication between teams.
- Ensure that relevant team members are made aware of how and when they will receive inputs to products, need to distribute outputs, and when complete work products are required.

These recommendations would give other teams better insight into how TeamAB sets its priorities; at the very least, this would help them understand why certain issues are not being addressed as quickly as they would like. These recommendations would also give TeamAB a better understanding of other teams’ needs, which would allow TeamAB to consider these needs when selecting issues to fix in a given sprint.

c) Errors Introduced by Other Teams: The GTM has three recommendations that address errors introduced by other teams:

- Retain tasks that require frequent communication between groups within collocated teams.
- Ensure that relevant team members are made aware of how and when they will receive inputs to products, need to distribute outputs, and when complete work products are required.
- Each location should understand how their modifications to the product unit can affect the other locations.

The first recommendation would ensure that changes that might affect other products would be communicated quickly to engineers working on those products. The other recommendations would raise awareness of the impact of changes on other teams and products.

2) Other Issues: The QA staff shortfall issue is not so much a governance issue as a management issue: as might be expected, developers tend to finish implementation later in the sprint, putting pressure on the QA team member to test
these completed features or fixes before the sprint is over. The solution would be to implement automated tests, or add QA staff, or both; the former is a decision that Scrum would leave to the “self-managing” team, while the latter is a staffing decision for more senior management.

The variability in accuracy of estimates appears to be a consequence of the kind of issues TeamAB deals with. Many are bug fixes that require investigation before a fix can be made; the time required to complete the fix can only be accurately estimated when the investigation is complete. As such, it is difficult to estimate the time to complete such issues. This experience appears typical of many projects; studies have found that about two-thirds of projects substantially overrun their estimates [19]. In either case, the governance structure and process for making these decisions is in-place.

D. Validation

We asked members of TeamAB to review and comment on our observations. Their response was largely affirmative, although they did point out that meeting minutes are taken and archived on the project’s internal wiki.

We also asked a senior manager to perform an assessment of PracMed’s implementation of the Global Teaming Model practices, using the same rating scale; these are shown in parentheses Table II and Table III. As can be seen, there is considerable discrepancy between the implementation level assigned by external observers, the level assigned by senior management: in all cases were the external observers assigned ‘C’ (minimally or not at all implemented), management assigne ‘A’ or ‘B’.

There are several possible explanations for this discrepancy. One is that management has greater visibility into the PracMed’s overall organization, and therefore is in a better position to assess implementation of practices than the observers, who have (so far) only observed PracMed from the viewpoint of TeamAB. Another is that TeamAB has not implemented GTM practices to the same degree as the rest of PracMed. Finally, it is possible that management do not have the same insight into the day-to-day activities of TeamAB as the observers. As mentioned above, all but one of the practices rated ‘C’ by the external observers have to do with day-to-day operations rather than strategic issues.

In order to understand these discrepancies, we are conducting a more detailed assessment involving all TeamAB members, as well as members of other teams, and managers at multiple levels.

E. Limitations

As with any empirical study, this research has some limitations, which we can characterize as threats to validity.

1) Threats to Construct Validity: This research is part of a larger study with the purpose of identifying PracMed’s successful practices as well as areas that need improvement, in advance of transitioning the entire development organization to Agile development methods. As such, the larger study is not specifically focused on Software Development Governance. This means the observations and other data collection may not reflect governance as well as if they had been designed specifically for this purpose.

2) Threats to Internal Validity: The primary research method used for this study is participant observation with moderate participation. This means that the very act of observation may affect the people being observed. In this instance, the consequence is that certain behaviour on the part of TeamAB is moderated. As a result, the magnitude of the issues identified may be even greater than what was observed.

Also, the discrepancy between the observer’s assessment and senior management’s assessment of GTM practice implementation indicates there may be some limitations on what the research approach allows us to observe.

3) Threats to External Validity: This study is of a single team in a single company; as such, we must be cautious about generalizing the results, as they may not apply in all or even many cases. We can, however, say that the results support our hypothesis that the Global Teaming Model is an effective framework for creating good Software Development Governance for Global Software Development.

V. Conclusions

Software Development Governance is an emerging research topic that is gaining traction among practitioners due to its potential to improve software project alignment with organizational goals and strategy. This is especially true for Global Software Development, where traditional hierarchical organization and decision-making give way to more loosely structured, fluid organizations with more autonomous members.

We hypothesized that the practices and recommendations comprising the Global Teaming Model facilitate Software Development Governance. We demonstrated this capability through a case study of a distributed team within a company engaged in globally dispersed development across Europe and North America. We showed how recommendations from the Global Teaming Model that have not (yet) been implemented could address some of the outstanding issues the team is facing.

In particular, the Global Teaming Model has many practices to support issues associated with Motivation in a distributed team [11, 33], External Schedule Pressure, and Errors Introduced by Other Teams. Areas such as addressing staff shortfall and estimates were found to be outside the scope of the Global Teaming Model.

The results show how the Global Teaming Model [9] can facilitate good Software Development Governance of Global Software Development efforts by explicitly identifying recommendations that need to be considered when undertaking software development involving widely distributed teams.

A. Future Directions

As part of the larger software process improvement effort of which this study is a part, we plan to implement the missing Global Teaming Model recommendations at PracMed, and
assess the effect on issues faced by TeamAB. We are also observing a second team, which we plan to assess in the same way as presented here.

Acknowledgments

We would like to thank the members of TeamAB – Dan Geare, John Gilbert, Anton Jebbaratningsham, Imogen Neary (née Davies), Simon Paget, and Sanita Tifentale – for their generous and thoughtful collaboration on this study, and PracMed, for allowing us to study their software development efforts.

This work was supported, in part, by Enterprise Ireland and the European Development Fund through grant IR20130022, and by Science Foundation Ireland grants 10/CE/I1855 and 13/RC/2094 to Lero - the Irish Software Research Centre (www.lero.ie).

References

TABLE II
PRACTICAL ASSESSMENT AGAINST GTM RECOMMENDATIONS FOR “DEFINE GLOBAL PROJECT MANAGEMENT” SPECIFIC GOAL.

S 1 Define Global Project Management
SP 1.1 Global Task Management

Sub P. 1.1.1 Determine team and organisational structure between locations.
A (A) Create roles, responsibilities and rules to facilitate coordination and control over geographical, temporal and cultural distance.
A (A) Be aware of problems with unbalanced team sizes; e.g., smaller teams may be threatened and fear job loss.
X (A) Team structure should cater for possibility of dual reporting to management at more than one location, e.g. team structure could be cross divisional or multi-organisational and management remote.
B (A) Ensure that the supervision, support and information needs of all team members are met regardless of location.
A (A) Organisational structure should be documented and available to all team to allow a clear understanding of everyone’s roles and responsibilities within the project.

Sub P. 1.1.2 Determine the approach to task allocation between locations
A (A) Identify and document reason for working with virtual team.
A (A) Identify and document reason for working with virtual team.
C (B) Base task allocation on the organisational requirement, e.g., if proximity to market is reason development team is located in a particular country, then customer-related tasks should be allocated to that team.
X (X) Retain tasks that require frequent communication between groups within collocated teams.
A (A) When GSE team members (e.g., subdivided into work modules, e.g., subdivided into work modules or different parts of the life cycle) management must allocate tasks based on core competencies of each sub-team, and clearly define which stages are carried out at which location.
X (A) Confidential software development activities that provide competitive advantage should be developed within the organisation.
X (X) Related non-confidential development activities can be undertaken by external remote team colleagues.

SP 1.2 Knowledge and Skills

Sub P. 1.2.1 Identify business competencies required by global team members in each location
A (B) Ensure that all team members have a clear understanding of the customer base and the business functions to take full advantage of the proximity of the team to the customer base.
A (B) Modularisation: partition work into modules which have a well defined functional whole
Sub P. 1.2.2 Identify the cultural requirements of each local sub-team
A (A) Cultural diversity: Each team member should be trained to understand the culture of the virtual team. Face-to-face meetings are recommended when and where possible, ideally at the start of the project and/or when a new team member joins. Having individuals visit locations for extended periods can also be a successful strategy and should be fully leveraged at every possible opportunity.

Sub P. 1.2.3 Identify Communication Skills for GSE
A (A) In order to develop the right practice, a new communication protocol needs to be set up. Policies should be put in place to support these new requirements to the satisfaction of all virtual team members. For example in synchronous communication, ensure that link up times are shared between core team working hours in each location.

Sub P. 1.2.4 Establish relevant criteria for training teams
A (C) Effective knowledge transfer: Carry out evaluation of training needs to include cultural and linguistic issues. Undertake training onsite and face-to-face so team members can be directly assessed and training provision tailored to their specific requirements.

SP 1.3 Global Project Management

Sub P. 1.3.1 Identify GSE project management tasks
B (A) Define ability and potential productivity of team: Global project manager should allocate tasks and timescales that are realistic. Where possible, the project manager should be actively involved in the recruitment and selection of team members. Failing this, they should gather all information relating to technical and professional experience of potential and existing team members.
Sub P. 1.3.2 Assign tasks to appropriate team members
A (?) Assign according to one or more of three different approaches; Modularisation; Phase-based approach; and Integrated approach.
A (?) Modularisation: partition work into modules which have a well defined functional whole
X (A) Phase-based approach: Use when phases of the development cycle are relatively independent. Ensure that the team members developing a specific phase have a good understanding of what is required at each specific stage.
X (A) Integrated approach: Set up a protocol to allow handover from one geographic location to another to ensure a successful follow the sun development.

Sub P. 1.3.3 Ensure Awareness of cultural profiles
X (B) National cultural differences should be identified and communicated to the management and team members. Cultural training can be communicated in the following ways.
A (C) Provide training to give all team members an opportunity to learn and understand about each other’s culture.
B (B) Address national, religious and relevant ethnic issues, all team members should understand acceptable and unacceptable forms of behaviour.
B (B) Training should be tailored to team member’s specific needs and location.
? (A) Project managers ensure that cultural profiles for teams are established. E.g., Management and staff should show respect for gender-related cultural values of all colleagues. All employees’ legal rights must be upheld.

Sub P. 1.3.4 Establish cooperation and coordination procedures between locations
B (B) Ensure that a suitable infrastructure, process and management procedures are in place to help establish cooperation and coordination between locations. Achievable milestones should be planned and agreed. Projects should be monitored with reference to costs, time, productivity, quality and risk.

Sub P. 1.3.5 Establish reporting procedures between locations
A (A) Regular formal reporting will help the project manager to remain aware of how project is progressing. Procedure should include and encourage team members to report whether or not they can take on that task in the given time and report any problems before it is too late.
<table>
<thead>
<tr>
<th>S 2</th>
<th>Define Management Between Locations</th>
</tr>
</thead>
<tbody>
<tr>
<td>SP 2.1</td>
<td>Operating Procedures</td>
</tr>
<tr>
<td>Sub P. 2.1.1</td>
<td>Define how conflicts and differences of opinion between locations are addressed and resolved</td>
</tr>
<tr>
<td>C (A)</td>
<td>Set up a strategy to handle, monitor and anticipate where conflict between remote locations may occur. The strategy should include how conflict will be resolved and how a person responsible for that resolution is selected.</td>
</tr>
<tr>
<td>C (B)</td>
<td>When defining the global strategy for dealing with conflict, different types of conflict have to be taken into account, for example conflict due to fear as well as cultural differences.</td>
</tr>
<tr>
<td>Sub P. 2.1.2</td>
<td>Implement a communication strategy for the team</td>
</tr>
<tr>
<td>C (A)</td>
<td>Plan, facilitate, encourage and monitor communication between teams.</td>
</tr>
<tr>
<td>A (A)</td>
<td>Provide training on how best to communicate with remote colleagues, including the effective operation of communication tools and procedures.</td>
</tr>
<tr>
<td>X (A)</td>
<td>Consider linguistic and cultural implications inherent when communicating remotely.</td>
</tr>
<tr>
<td>Sub P. 2.1.3</td>
<td>Establish communication interface points between the team members</td>
</tr>
<tr>
<td>B (A)</td>
<td>Strategies need to be put in place which encourage both formal and informal reporting.</td>
</tr>
<tr>
<td>C (A)</td>
<td>Ensure that relevant team members are made aware of how and when they will receive inputs to products, needs to distribute outputs from and when complete work products are required.</td>
</tr>
<tr>
<td>B (A)</td>
<td>Ensure teams are aware of potential constraints such as legal restrictions and holidays in countries within which they are developing the product.</td>
</tr>
<tr>
<td>A (A)</td>
<td>Ensure that information about each team member is easily accessible by colleagues. Information of an individual’s role within the team and their specific areas of responsibility should be combined with a photograph, their first name, surname, friendly name (if appropriate) and their preferred form of address.</td>
</tr>
<tr>
<td>A (A)</td>
<td>Intranets and wikis can be invaluable for this form of communication.</td>
</tr>
<tr>
<td>Sub P. 2.1.4</td>
<td>Implement strategy for conducting meetings between locations</td>
</tr>
<tr>
<td>A (A)</td>
<td>Identify appropriate virtual meeting technology is used.</td>
</tr>
<tr>
<td>C (A)</td>
<td>Try to ensure all participants are comfortable with virtual meeting and are given opportunity to agree or disagree with points raised, and offer new ideas.</td>
</tr>
<tr>
<td>C (A)</td>
<td>Circulate agenda prior to meeting, and clearly minute actions agreed a meeting</td>
</tr>
<tr>
<td>C (A)</td>
<td>Ensure that no delay occurs between the meeting and the circulation of minutes as people may be waiting for the minutes before implementing the actions.</td>
</tr>
<tr>
<td>SP 2.2</td>
<td>Collaboration between locations</td>
</tr>
<tr>
<td>Sub P. 2.2.1</td>
<td>Identify common goals, objectives and rewards for the global team</td>
</tr>
<tr>
<td>B (B)</td>
<td>Global Project manager sets project goals and objectives.</td>
</tr>
<tr>
<td>B (B)</td>
<td>Goals at project level are common to all locations.</td>
</tr>
<tr>
<td>B (B)</td>
<td>Project goals and objectives communicated, understood and agreed across all team members regardless of location.</td>
</tr>
<tr>
<td>C (B)</td>
<td>The global team is viewed as an entity in its own right, regardless of the location of its team members and its performance should be judged and rewarded accordingly.</td>
</tr>
<tr>
<td>? (B)</td>
<td>Acknowledging team success may require tailoring rewards to the needs of different cultures.</td>
</tr>
<tr>
<td>? (C)</td>
<td>Project Managers need to understand the cultural motivation of the different team members and identify and apply appropriate rewards in each situation when and where relevant.</td>
</tr>
<tr>
<td>? (B)</td>
<td>Consideration should be given to cultural issues, economic situation and income tax laws when planning rewards.</td>
</tr>
<tr>
<td>Sub P. 2.2.2</td>
<td>Collaboratively establish and maintain work product ownership boundaries</td>
</tr>
<tr>
<td>A (A)</td>
<td>Define product ownership boundaries through partitioning of work across GSE teams.</td>
</tr>
<tr>
<td>B (A)</td>
<td>Each location should understand their role within the life cycle of the product.</td>
</tr>
<tr>
<td>C (A)</td>
<td>Each location should understand how their modifications to the product unit can affect the other locations.</td>
</tr>
<tr>
<td>Sub P. 2.2.3</td>
<td>Collaboratively establish and maintain interfaces and processes</td>
</tr>
<tr>
<td>B (B)</td>
<td>Define common process goals across all locations.</td>
</tr>
<tr>
<td>A (A)</td>
<td>Define process ownership - placing ownership with those closest to process where possible.</td>
</tr>
<tr>
<td>A (A)</td>
<td>Seek and encourage input from team members at all locations.</td>
</tr>
<tr>
<td>A (A)</td>
<td>Let team members know their input to process development and ownership is valued.</td>
</tr>
<tr>
<td>C (A)</td>
<td>Processes should address specific challenges associated with GSE.</td>
</tr>
<tr>
<td>A (A)</td>
<td>Processes should take into account the relevant structures and procedures from all sites.</td>
</tr>
<tr>
<td>Sub P. 2.2.4</td>
<td>Collaboratively develop, communicate and distribute work plans.</td>
</tr>
<tr>
<td>B (A)</td>
<td>Achievable milestones should be planned and agreed.</td>
</tr>
<tr>
<td>A (A)</td>
<td>Within the commitments made, team members must explicitly include communication plans to include use synchronous and asynchronous communication tools.</td>
</tr>
<tr>
<td>C (A)</td>
<td>Contingency plans should be in place to address potential risks.</td>
</tr>
<tr>
<td>C (A)</td>
<td>Establish procedures to coordinate implementation of contingencies when and if required.</td>
</tr>
</tbody>
</table>