Coordinating mechanisms for Agile Global Software Development

Emam Hossain
UNSW-NICTA, Australia
Emam.Hossain@nicta.com.au

Abstract

Despite the apparent significant differences between the fundamental principles of agile and Global Software Development (GSD), there is a growing interest in assessing the viability of using agile practices in GSD. But there has been little research in identifying and understanding the challenges involved in integrating agile and GSD and potential strategies to deal with them. One of the key challenges is coordination of the development work. Study asserts that a good understanding of coordination difficulties can help devise strategies to overcome them. The aim of this PhD study is to explore and understand the use of different coordinating mechanisms to coordinate GSD projects while using agile software development practices.

1. Introduction

Many organizations have turned toward globally distributed software development in their quest for cheap, higher-quality software that has a short development cycle [1, 2]. Global Software Development (GSD) also claims to make it possible for organizations to gain time-zone effectiveness, leverage a large skill pool, rapid innovation and exploit low labour costs in certain part of the world [3][4]. However, a GSD team works across geographic, temporal, cultural, political, and organizational boundaries [6], and suffers significant challenges with respect to communication, coordination, and control [7]. Therefore, GSD is recognized as being considerably more complex to manage than even the most complex in-house project [8, 9].

Recently, Agile Software Development (ASD) have gained significant popularity because they promise to handle requirements volatility, promote extensive collaboration between customers and developers, and support early and frequent delivery of the product [34]. Also there has been a growing interest in applying agile approaches in GSD to solve some of the coordination and communication difficulties [7]. A number of experience reports mention the successful implementation of agile practices in different GSD projects [10] [11]. However, researcher’s note that agile approaches are difficult to scale up and a little empirical evidences are available in global arrangements [12] [36]. In some cases, research reports come up with contradictory conclusions about applying agile practices in a distributed environment [7]. Hence, we can argue that although the combination of both agile and GSD is expected to be beneficial, their blending mechanisms are often poorly understood [7]. This lack of understanding requires further in-depth studies [37] [18] [12].

Towards an in depth study, section 2 discusses agility in GSD based on current, state of the art literature. Section 3 discusses the use of different coordinating mechanisms in software development. Section 4 discusses coordinating challenges when agile practices used in GSD. Section 5 presents research questions and propositions. Section 6 describes about the research methodology. Finally section 7 concludes the paper with a brief description of future work.

2. Agility in GSD

Several authors have reported on the successful implementation of agile practices in GSD [14-17] with mentioning a number of implementation barriers. Geographic, temporal and socio-cultural distances are recognized as potential barriers for applying agile practices in GSD [11]. Taylor et al. [12] classify fourteen published papers from 2001 to 2005, which focus on agile practices in GSD. They note that none of these papers addressed agile GSD process issues adequately.

It is necessary to carefully choose appropriate agile practices for offshore development to gain success[14] as agile practices are hard to implement in distributed teams especially when team size is large[15][18].
Among the most known and adopted agile methods are Extreme Programming (XP) [19] and Scrum [20]. Specific Scrum and XP practices are found to be useful for reducing communication, coordination, and control problems associated with GSD [11]. Braithwaite and Joyce [21] report successful application of XP principles to full-scale commercial distributed developments by providing sufficient rich communication, respect, and trust. XP practices in outsourcing are also possible but not proven as effective for fixed-price outsourcing contracts because this requires the need for accurate estimates of scope rather than allowing changes in requirements [38]. Scrum practices can also apply in distributed development by maintaining vision and awareness of work activities throughout teams [22].

Distributed software development can be agile, when the unique characteristics of both agile and distributed development environments are successfully integrated [23] [3]. Careful incorporation of agility in GSD is essential to address several challenges related to communication, control and trust throughout the dispersed teams [23] [3]. Lee et al. [39] argue that agility in GSD can also be achieved by the proper alignment of IT strategy with infrastructure and project management.

Although literatures mention the benefits of using agile practices in GSD but none of them discusses development process issues adequately. This requires further in-depth studies to understand agile GSD process more carefully [18] [37]. To address such inadequacies, this PhD study will investigate the use of different coordinating mechanisms in GSD projects to coordinate team, team members and their associated responsibilities and tasks while using agile practices.

3. Coordinating mechanisms

Mintzberg [24] mentioned that organizational work coordination can described by following three basic coordinating mechanisms.

1. Mutual adjustment - based on the simple process of informal communication;
2. Direct supervision - one person takes responsibility for the work of others by issuing instructions and monitoring their actions;
3. Standardization - of which there are four types: work processes, output, skills (as well as knowledge) and norms.

Mintzberg [24] also indicated that the role of coordinating mechanisms change according to task complexities (Figure 1.).

![Figure 1. Fitting coordinating mechanisms to task complexity [24]](image)

Throughout the history (Figure 2) of software development, different models and approaches have been suggested for tackling the complexity and uncertainty of software development. Traditional development models following the mechanistic worldview are often referred to as plan-driven [28] (e.g. the waterfall model [29] and SW-CMM [30]) and models following the agile approach are often described as change-driven [41].

![Figure 2: Evaluation of software process models [41]](image)

According to Mintzberg [24], plan-driven and change driven development usually preferred “standardization” and “mutual adjustment” as primary coordinating mechanism respectively. But the use of different coordinating mechanisms in agile GSD is not clear because of the combination of both plan-driven and change driven development. Hence the understanding of using different coordinating mechanisms in GSD and agile development will give ideas about the problems and challenges of balancing different coordinating mechanisms in agile GSD.
3.1. Coordinating mechanisms in GSD

The coordination activities in GSD seem difficult because of the negative impact of distance [5]. The effect of software team member distribution is characterized by coordination breakdown, geographic dispersion, loss of communication richness, loss of “teamness”, and cultural differences [9]. Inconsistency in work practices can also reduce cooperation through misunderstandings [13] and socio-cultural distance can also reduce mutual understanding [12].

Plan-driven development reflects a hierarchy involving a command-and-control style of management with clear separation of roles [26] [32]. Usually GSD relies mainly on formal mechanisms (standardization), which exploit detailed architectural design and plans, to address impediments to team communication because of geographical separation [3] [7] [31]. Although methodological standardization is considered as effective coordinating mechanism [9], GSD also need to use mutual understanding with direct supervision to minimize different challenges. For this reason agile practices may benefit GSD projects through using coordinating mechanism “mutual understanding” [10] [11]. Study therefore argue that GSD uses “standardization” as its primary coordinating mechanism and “mutual understanding” and “direct supervision” also used with care.

3.2. Coordinating mechanisms in ASD

Agile Software Development (ASD) is comprised of self-managing teams, relies on people and their creativity rather than on processes [33]. ASD puts less emphasis on up-front plans and strict plan-based control and more on mechanisms for change management during the project [26]. ASD also favours a leadership-and-collaboration style of management where project manager's role is as a facilitator or coordinator [34]. Self-managing teams are also proven as a success factor for innovative projects [25] [35]. Mintzberg [24] also mentioned that mutual adjustment is the preferable coordinating mechanism for innovative organizations. Although the members of a self-organized team is largely on their own but are not uncontrolled [25]. Hence, management should establish enough checkpoints to prevent instability, ambiguity, and tension from turning into chaos. But also should avoid the kind of rigid control that impairs team creativity and spontaneity. Hence, direct supervision and standardization must be used with care in a self-managing team. Thus it can be argue that in ASD, mutual adjustment is used as a primary coordinating mechanism with care of direct supervision and standardization.

4. Coordinating mechanisms in Agile GSD

Despite the apparent significant differences of the role of coordinating mechanisms between agile and GSD, there is a strong motivation of applying agile practices in GSD to solve some of its difficulties [7] [11]. However, study believe that the use of coordinating mechanisms of agile GSD is still not clear. Usually, plan-driven development focuses on command-and-control, formal communication and desire of a mechanistic (bureaucratic with high formalization) organizational structure [32]. Thus GSD prefers standardization as key coordinating mechanism with care of direct supervision and mutual understanding. On the other hand ASD focuses on leadership-and-collaboration, informal communication and desire an organic (flexible and participative, encouraging cooperative social action) organizational form [32]. Hence ASD favours mutual adjustment as key coordinating mechanism with use of standardization and direct supervision. Table 1 summarises the problems with applying agile practices in GSD.

<table>
<thead>
<tr>
<th></th>
<th>GSD</th>
<th>ASD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Standardization</td>
<td>Formal rules, often follows waterfall</td>
<td>Iterative and exploratory, few</td>
</tr>
<tr>
<td></td>
<td>approach, separate design and implementation, rule-driven</td>
<td>formal rules</td>
</tr>
<tr>
<td>Direct supervision</td>
<td>Command and control</td>
<td>Shared leadership</td>
</tr>
<tr>
<td>Mutual adjustment</td>
<td>Individual roles, specialization</td>
<td>Self-organizing teams, informal</td>
</tr>
<tr>
<td></td>
<td></td>
<td>communication</td>
</tr>
</tbody>
</table>

Table 1. Problems with applying agile in GSD

An organization can maximize their development efficiency by balancing both agility and discipline carefully in their development environment [3] [28]. Thus we can also argue that a proper balance of agile development and GSD may bring more benefits. Study found that major challenge of such merging is to balance three different coordinating mechanisms as shown in Figure 3.
Figure 3. Conflict of merging coordinating mechanisms

Also introducing change-driven development in a plan-driven environment may impact several aspects of the organization including structure, culture, and management practices [27]. Thus to establish software development processes as Agile GSD, it is necessary to understand how different coordinating mechanisms are used in that development clearly. As Mintzberg conceptual framework for work coordination [24] provides understanding how work is coordinated in both GSD and agile, we claim that this framework can also be used to understand coordination issues in “Agile GSD“.

5. Research question & Proposition

Given the problem description, the overall goal of this PhD study is to address the following broad research question:

RQ1: How are coordinating mechanisms in GSD projects used to coordinate teams, team members and their associated responsibilities and tasks when applying agile software development practices?

To answer the overarching research question above we propose following sub-questions.

RQ1.1. When is the coordinating mechanism “informal communication” useful in coordinating GSD projects that use agile software practices?

RQ1.2. When is the coordinating mechanism “direct supervision” used in coordinating GSD projects that use agile software practices?

RQ1.3. When is the coordinating mechanism “standardization” used in coordinating GSD projects that use agile software practices?

RQ1.4. what other coordinating mechanisms are used to coordinate GSD projects that use agile practices?

Research will also focus on the role of different coordinating mechanisms depend on the complexity of GSD tasks.

RQ2: How does the complexity of tasks influence the use of different coordinating mechanisms in GSD projects that use agile practices?

Based on the formalized research questions, research produces a set of propositions which will help to identify the relevant information about cases.

Proposition 1
If a GSD project uses agile software development practices then “informal communication” among distributed teams and team members is the primary coordinating mechanism.

Proposition 2
If a GSD project uses agile practices then “direct supervision” among distributed teams and team members is a secondary coordinating mechanism.

Proposition 3
If a GSD project uses agile practices then “standardization” among distributed teams and team members is a secondary coordinating mechanism.

Proposition 4
If a GSD project uses agile practices then coordinating mechanisms other than “mutual adjustment”, “direct supervision”, and “standardization” may also be used.

Proposition 5: If an agile GSD project is very complex then the use of coordinating mechanisms “mutual adjustment” will increase to ensure project success.

6. Research Methodology

This research will use multiple, industry-based single case studies that strive to describe, evaluate, explore or explain the role of different coordinating mechanisms in GSD while using agile practices within real-life context. All case studies will be planned and conducted
according to the case study guidelines (draft) developed by NICTA researchers and Yin [42] case study research outlines. This case study will follow the following broad phases.

**Research Pre-Planning Phase:**

Study already conducted following steps to ensure that sufficient preparation and training have been performed for case study.

- a set of high-level research objectives have been defined based on the identified area of interest
- comprehensive literature analysis (ongoing)
- appropriateness of case study approach has been determined
- a written record of the procedures (protocol template) followed throughout the study is set upped (ongoing)

The necessary legal, ethical, intellectual properties including publishing, scheduling and working condition issues are also been considering by this study.

**Planning Phase:**

This PhD study is currently working on case study planning phase and already performed

- research questions refinement based on further study and expert feedback
- research propositions are clearly defined
- concepts are defined and appropriate unit of analysis are identified

Study now focusing to

- define measures, identify project constraints and boundary of the proposed case studies, result analysis, develop a conceptual framework and baseline for evaluation
- Select appropriate cases and a pilot case

**Design Phase**

This phase of the case study will connect the empirical data to a study’s initial research questions and, ultimately to its conclusions [40]. Steps will involve in this phase:

- Convert propositions to hypotheses
- Identify appropriate method of comparisons
- validate data through four design test including construct validity, internal validity, external validity and reliability
- Define case study plan, quality assurance and also data collection strategy
- Design a case study storage system and produce first draft of the plan
- Check and update protocol plan based on external expert feedback
- Undertake a pilot case study to finalize the required data

**Data Collection Phase**

The research will aim to consider six sources of data: documents, archival records, interviews, direct observation, participant-observation and physical artifacts as main sources of data. However other sources such as films, photography, ethnography and life histories may also be considered if required. Both qualitative and quantitative data will be collected through interviews and observational studies.

**Data Analysis**

Suitable data analysis methods and tools will be used for analyzing data. Coding techniques will be applied to the qualitative data collected from the case study interviews. The aim is to validate the conceptual framework based on the empirical studies.

**Reporting**

Necessary reports and relevant publications will also be developed based on the case study findings with the ultimate goal of addressing the research objectives.

**7. Conclusion**

The conceptual framework developed in this research describing coordinating mechanisms of agile software development practices in GSD will be validated using multiple case studies. Finally the research will conclude with the main findings and the research questions will be addressed. Furthermore, a description of future research will also be provided.

**8. References**


