Coping strategies for temporal, geographical and sociocultural distances in Agile GSD: a Case Study

Dávid Marcell Szabó and Jan-Philipp Steghöfer
Chalmers | University of Gothenburg, Sweden
szabodavidmarcell@gmail.com, jan-philipp.steghofer@gu.se

***Abstract*—Globally distributed software development teams face a number of challenges in their work that are associated with temporal, geographical and sociocultural distances. This case study explores the relationship between agile practices and these three distances in global software development. Data was collected by interviews and secondary data analysis. The results show that the three distances affect agile practices and the case team modifies the agile practices accordingly. Agile practices, in turn, affect the three distances by reducing communication, control and coordination challenges. Non-agile coping strategies such as specialised communication strategies also play an important role to alleviate the effects of the different distances.**

***Index Terms*—Global Software Development, Agile Software Development, Distances**

I. Introduction

Globalisation is increasing in many industries, including the software development industry. Interest in Global Software Development (GSD) is rising where people from different nations, organisational cultures and time zones are involved [1]. There are certain benefits to GSD [2], including that companies gain access to workforce in developing countries, which reduces the labour costs, that the time zone differences allow to develop software nonstop, that businesses can take advantage of the proximity to the market, and that is is possible to exploit opportunities through the quick formation of virtual teams.

On the other hand, GSD needs to address temporal, geographical and sociocultural distances that pose challenges in communication, control and coordination [3], [4]. Temporal distance is a measure of the disruption of time between two people who want to interact, geographical distance is a measure of the effort of one person visiting the other and sociocultural distance is a measure of a person's understanding of another person's values and normative practices [5].

Agile methods in turn address key problems in software development (low quality product, late delivery and high cost of development) by a lightweight and fast development process [2]. Agile practices rely on people, short, iterative software development cycles, collaborative decision making, inclusion of rapid feedback into the processes and development and continuous code integration [6]. XP and Scrum are maybe the two most well-known agile methods. They complement each other as Scum is used for managing and tracking software development and XP provides good engineering practices [7].

Previous research shows that agile methods can alleviate GSD challenges by improving communication, coordination and control [8]. They have been shown to improve communication and reduce coordination and control overhead in GSD [2], [9]. However, key concepts of agile practices are difficult to implement in GSD as a result of the project members being distributed [10]. Face-to-face, informal and synchronous communication is, e.g., difficult to implement when the customer is not on site, team members are geographically distributed or even in different time zones [9].

For practitioners who wish to handle the complexities of GSD, it is crucial to understand the dynamics of the three distances. This study analyses these distances in a distributed development team of a pharmaceutical company. In particular, we focus on the coping strategies currently employed and how they impact the different distances. We distinguish agile practices that are employed and other, not strictly agile, coping strategies that support the agile ones. Our study thus allows practitioners to better decide on concrete strategies, illustrates the possible impact of off-shoring, and helps researchers identify gaps in the current understanding of the combination of agile processes and GSD. We follow the argument that more empirical studies are needed to understand the use of Scrum in GSD [11] and the need to test innovative viewpoints on communication, coordination and control in distributed contexts using agile practices [12]. Our main research question is thus:

RQ1: *What is the relationship between agile practices and temporal, geographical, and sociocultural distances in a globally distributed software engineering project?*

We are also interested in the combination of agile practices with non-agile approaches in more general coping strategies:

RQ2: *Which coping strategies does the team use to address the three distances?*

Concretely, we address the following sub-research questions:

- **RQ1a:** *How are Scrum, XP and other agile practices used by the case study team?*
- **RQ1b:** *How do agile practices affect temporal, geographical and sociocultural distances in the case study team?*
- **RQ1c:** *How do temporal, geographical and sociocultural distances affect agile practices in the case study team?*

Our results show that global software development introduces temporal, sociocultural and geographical distances to the agile teams. Agile practices do indeed affect the three distances and the three distances affect the agile practices, thus establishing a bi-directional relationship. To address the three distances a combination of agile and non-agile practices is necessary.

II. Background and Related Work

A. Globalisation in the Software Industry

With the rise of globalisation, offshoring software development became popular in many companies [9]. In offshoring, globally distributed virtual teams collaborate across countries [13]. To achieve this, the work is outsourced to an offshore third party via *offshore sourcing* [14]. Offshore sourcing also includes the insourcing of work within organisational boundaries to one of the company's department in an offshore location [9]. Increased geographical distance increases management complexity due to problems in communication, coordination and control [15]. It can also cause the offshored parts of the teams to feel like they do not belong to the core team [16]. This can create problems with trusting each other's work and can reduce team morale. Geographically dispersed teams either use asynchronous modes of communication, such as email or synchronous communication methods, such as video chat and teleconferencing [9]. Also, globally distributed teams collaborate with people from different nationalities, thus increasing sociocultural distance [9]. In addition, globally distributed teams are likely to work in different time-zones, thus causing temporal distance.

B. Agile Software Development

Agile software development methodologies focus on responsiveness to customer needs, thus improving software quality and enhancing productivity [17]. Furthermore, increased customer satisfaction is achieved by early and continuous delivery of valuable software. Business representatives and developers ideally work together during the development process and face-to-face communication is preferred over formal communication [18]. Agile assumes that requirements change frequently and flexible requirements management (e.g., through iterative requirements gathering and analysis) is necessary.

1) Scrum: Scrum focuses on project management, rather than on software development [19]. Scrum practices include sprints, daily stand-ups, sprint planning, retrospectives and sprint reviews [20]. A sprint is a short iteration, which lasts usually two to four weeks. In a sprint, certain requirements captured as user stories should be completed. User stories are captured in a product backlog where they are prioritised by the product owner on behalf of the customer. Sprints are planned to identify and estimate the time required to implement target functionality. Team members communicate in stand-up meetings, which last around 15 minutes and give an overview of current and next work and any potential impediments [3]. Scrum teams are self-organising, multidisciplinary "feature" teams who develop small increments of working software [20].

2) Extreme Programming (XP): Extreme programming focuses on the engineering aspect of software development and is based upon four values: "communication, simplicity, feedback and courage" [21]. These values are engrained in twelve development practices: planning game, small releases, metaphor, simple design, test driven development, refactoring, pair programming, collective code ownership, continuous integration, 40-hour week, on-site customer and coding standards.

C. Agile in GSD

The common view is that agile practices are not suitable for GSD because only some of the agile principles can be applied [9]. The three distances of GSD, geographical, temporal and sociocultural, make communication, control and coordination more difficult [2].

As mentioned in Section I, informal, face-to-face communication and co-located teams are not achievable when team members are located in different locales and often in different time-zones. Pair programming, shared code ownership and onsite customers are also difficult to implement in globally distributed projects [18]. Hansen & Baggesen [22] claim that Scrum practices, such as daily scrum, sprint planning and retrospectives improve collaboration among distributed team members, uncover hidden problems, build trust, and develop team spirit. According to Paasvivaara et al. [23], daily stand-ups provide transparency and support informal communication among distributed team members.

Distributed agile teams can choose from a wide array of tools for improving communication. Careful selection of synchronous and asynchronous communication methods minimises the differences in time zones, work day hours, or public holidays [20]. Collaborative project management tools, online meetings, social media and wikis can also help in communication across distributed locations. Periods of co-located work, visits, unofficial meetings, training activities, or distributed documentation help overcome sociocultural distances and can boost team morale [20].

According to Holmström et al. [2] temporal distance was mitigated by pair programming and by increasing the overlap in working time between the distributed developer pair. Joint Scrum sprint planning addressed the geographical distance by increasing the "teamness" feeling. The team published "sticky notes" on a web page to share a visualisation of project progress. These activities helped to reduce sociocultural distance by increasing mutual understanding and collaboration.

Yadav [9] describes a flexible management approach that lies between the traditional plan-based and the pure agile approach. She proposes iterative prototyping to increase product quality, frequent online meetings through web conferencing, lightweight project planning practices, appointed site-coordinators to coordinate work across distributed locations, using cloud-based collaborative technologies for requirement gathering, project wikis to help in coordination, and social media for informal communication.

Lee et al. [8] describe "ambidextrous" coping strategies where agile is balanced with rigour. Effective global software teams need to sense unpredictable external changes rapidly to make timely responses to changes. Frequent communication and increased team/task awareness help to respond to problems. As coping mechanisms, the authors mention "24/7 command centres," "project dashboards," "weekly corrective action meetings," and "onsite customer representatives" for the offshore teams to sense and respond to emergent problems. On the other hand, due to the difficulties of global boundaries,

GSD teams in the study used rigorous processes. Detailed documentation and user requirement specification helped to reduce misunderstandings and ambiguity. Redundant roles in multiple locations helped to reduce single points of failure. Assigning on-site customers to offshore sites improved the effectiveness of communication.

Kircher et al. [24] remove practices that are dependent on co-location such as planning game, pair programming, and on-site customers in "distributed extreme programming". Practices that are independent of co-location such as small releases, testing, refactoring, collective code ownership, 40-hour week, as well as coding standards are encouraged.

Hossain et al. [16] describe eight different mitigation strategies that summarise the strategies mentioned above. They also suggest synchronised work hours, both synchronous and asynchronous communication methods including tool-supported communication through instant messaging, frequent visits, an iterative development approach with frequent opportunities to monitor progress and correct issues, reviews that include all stakeholders and provide feedback, and joint planing to scope the work and available resources, and establish the processes.

III. Case Description

The case is based on a cross-functional software development team developing a platform for internal projects. The team is working in a pharmaceutical setting with strict quality control. The owner of the platform project is a programme manager. The project is financed from internal projects that the platform serves. The internal projects involve smart medical devices, mobile apps, and big data analysis.

The core team is based in Sweden, the internal devOps team is based in the US and in India, and two back-end developers are based in Bulgaria at a supplier company. The Californian team manages the Indian team, but does not work directly with the Swedish team. Also, there is a quality assurance (QA) professional working with compliance in Mexico with non-overlapping working hours with the Swedish team. An overview of the team and its members is given in Table I.

Table I
Platform Team members

Role	Activities	Location
Technology Lead	Manages the Bulgarian team members, gathers and analyses requirements, adapts architecture, develops.	Sweden
Programme Manager	Accountable for the projects the platform serves and the platform itself.	Sweden
Project Manager/Scrum Master	Manages platform team and one of the projects based on the platform. Responsible for regulatory documentation, release notes, and change requests. Works with the compliance coordinator. Supports team as Scrum Master.	Sweden
Product Owner	Requirement gathering and analysis.	Sweden
Back-end Developers (3)	Develop the platform.	Sweden
Mobile Developers (2)	Develop the Android and IOS SDKs.	Sweden
Full-stack Developers (2)	Develop both backend services for the platform and the user interfaces.	Sweden
Configuration Manager	Manages the releases.	Sweden
Automated Tester	Works with automated software testing and back-end development.	Sweden
QA Lead	Works with the compliance coordinator.	Sweden
Tester	System and user acceptance tests.	Sweden
Back-end Developers (2)	Senior and junior; develop the platform.	Bulgaria
devOps Managers (2)	Manage the Indian team, revise the release documents from a quality perspective.	California
devOps engineers (2)	Perform devOps tasks for the platform and for other projects.	India
Compliance Coordinator	Responsible for the regulatory compliance for the platform.	Mexico

The core team follows Scrum with two-week sprints and some XP practices. The Swedish team shares sprints with the Bulgarian one, but daily stand-ups are independent as the Bulgarian team adopted overlapping working hours with the US. It thus starts working at eleven o'clock Swedish time and holds its daily standup in the afternoon, together with two of the developers from Sweden. The core team doesn't have overlapping hours with the Californian team and the Indian team is always online during Swedish working hours. The core team works with the Indian team as needed, but the Indian team follows its own Scrum cycle and also provides devOps services to other projects. Whenever the platform team releases a new version or needs help in devOps activities, they contact the Indian team members and create tasks for them on Jira.

The requirements for the platform are provided by internal projects. The first project — a digital health project with smart medical devices — was the source of 80% of the requirements. This project is classified as a medical device, thus forcing strict regulatory compliance requirements on the platform. This affects the release process that needs to be verified and signed by the compliance coordinator in Mexico. The verification process follows a waterfall approach. Other projects that use the platform mostly reuse existing capabilities.

The core team has a product owner who works 50% with the first digital health project and provides requirements from the customers. The project manager also acts as a Scrum master who helps with coordination and applying the Scrum practices. The technology lead helps in the backlog prioritisation and in managing the Bulgarian developers. The rest of the platform team works with server-side development, testing, configuration management, mobile app development and front-end development.

The team uses a cloud-based agile project management tool for issue tracking (Jira[1]), cloud-based project wikis for team collaboration (Confluence[2]) and an online shared

[1]https://www.atlassian.com/software/jira, project mgmt. and issue tracking
[2]https://www.atlassian.com/software/confluence

Table II
INTERVIEWEE DESCRIPTIONS

Role	Work experience	Years at case company	Location
Project Manager (Pilot)	20 years	1.5 years	Sweden
Technology Lead	18 years	3.5 years	Sweden
Product Owner	20 years	17 years	Sweden
Configuration Manager	9 months	9 months	Sweden
Senior software developer	7 years	1.5 years	Bulgaria
Senior devOps engineer	6.5 years	1.5 years	India

code repository (Bitbucket[3]). For informal communications cloud-based instant messaging tools are used (Slack[4] and Skype[5]). Also, the team uses email for informal and formal communications. There is no physical backlog but a shared online backlog in the cloud-based issue tracking tool.

It is difficult to coordinate the teams due to the temporal, geographical and sociocultural distances (cf Section V). The platform team is stretched with meeting deadlines and communication between the platform team and the devOps team is tense. Also, the internal projects that the platform team is serving find the delivery too slow. The platform team's project manager believes that despite the Scrum project practices and the agile tools, the project is delivered in a waterfall fashion. This can be due to the compliance process. Also, the regulatory requirements on quality control are high, which puts additional burdens to documentation and testing.

IV. RESEARCH METHODOLOGY

We applied a case study methodology [25] to provide an in-depth explanation of the case. The research strategy is qualitative since the research will try to understand a phenomenon and generalize the results. This is an inductive approach where the emphasis is on generation of theories.

A. Data collection

The *primary data* was collected by semi-structured interviews from the case company. The *secondary data* was collected through analysing the platform team's documents.

1) Interviews: We conducted five individual interviews and one pilot interview (cf. Table II). The purpose of the interviews was to gain an understanding of how the team addresses the three distances and how they use the agile practices to mitigate them. We followed the following steps:

Step 1 – Question Forming: The interview guide was based on Bass' [20] study which examined the influences of agile methods in globally distributed software development projects. We added questions to address the research questions about the three distances. To avoid bias and to put the questions in a logical order flow, we iteratively refined the questionnaire.

[3]https://bitbucket.org/

[4]https://slack.com/

[5]https://www.skype.com/en/business/, video chat, voice calls, screen sharing

Step 2 – Pilot Interview: The project manager was interviewed to test the interview structure and content. We received constructive feedback and tested whether the questions were understandable. We also measured the time of the interview to establish a baseline. We took notes of the answers and used these notes to support the analysis. We also clarified some questions based on the feedback for future interviews.

Step 3 – Interviews: We conducted three interviews with the core Swedish team: the product owner, the technology lead, and the configuration manager. Two interviews were conducted with offshored team members: a senior devOps engineer in India and a senior developer in Bulgaria. The tech lead works mostly with the Bulgarian team. The configuration manager works mostly with the devOps team in India. We thus cover all relevant roles for offshore and onshore teams. Each interview took between 45 and 60 minutes and the employees were interviewed individually. The interviews were recorded with permission from the interviewees and then transcribed.

2) Secondary Data: We combined several qualitative methods to avoid reliance on a single approach [25] in analysing secondary data and used it to test the interview data. It stems from the documentation that the project team produces, i.e., tickets in Jira and wiki pages that detail the release process and include information about retrospectives. The user stories in Jira do not strictly follow the standard Scrum user story definition. Descriptions were often very short. All user stories had story points associated with them. We regarded all user stories created between September 2017 and April 2018. The wiki pages provide detailed release instructions instructions intended for the Indian team for each release between May 2017 and June 2018. They were linked to a Jira ticket. Monthly retrospectives were saved on wiki pages between January 2017 and December 2017. We considered all available wiki pages.

3) Direct Contact: We used email to contact the tech lead, the senior offshore devOps engineer and the senior offshore back-end developer to get quick answers to clarification questions. We used this informal method when a formal interview was inappropriate and to conduct member checking.

B. Qualitative Data Analysis

After transcribing the interviews, general statements in them were noted down [26]. The statements were categorised and each category was labelled. Then, the interviews were coded using a mixture of pre-defined and emergent codes. Predefined codes consisted of the labels and additional ones identified from the literature: communication, control and coordination [27]. One of the emergent themes that came up was the bidirectional relationship between agile practices and distances. The codes were categorised into six themes: *geographical*, *temporal*, *sociocultural distance*, *Scrum*, *XP* and *rigor*. Codes were organised in a mind map and cross-relations were identified. The final step was to interpret the findings and lessons learned based on the literature.

C. Threats to Validity

Validity means whether the record truly reflects what happened [28]. To improve validity we used triangulation, which

refers to combining two or more views/approaches/methods in an investigation to get a more accurate view on the phenomena [28]. This was done by analysing secondary data in addition to the semi-structured interviews. Also, for additional feedback, the data was discussed with an agile coach at the case company. In addition, constant comparison technique was used to check the accuracy and consistency of the interpretations. A coding list helps in this technique as each time a passage of text is coded it can be compared to the existing definitions in the list. This helps to avoid definitional drift [28].

Reliability means that the result of the analysis would be acquired if different researchers repeated the study [28]. Reliability was improved by member checking. Respondents were consulted about the interpretations that the data analysis generated. Moreover, evidence is included in form of quotations from interviews. The interviews for this study have been done remotely and not face-to-face. This threat was also mitigated by member checking with the tech lead and the product owner.

V. Results

This section presents the results of the interviews and the secondary data analysis. Table III provides a summary of the issues and the coping strategies described below and relates them to similar issues in the literature. Tables IV, V, and VI provide an overview of different agile practices we found and how they are used by the team.

A. Temporal Distance

The case description shows that the Swedish team does not have overlapping working hours with the US and Mexico. This creates a significant communication delay and slows down the release process. When the product owner is emailing the Mexican team members he has to wait a day for a reply: "*You send something in the afternoon and they respond the next day in their working day and then you can work with it next day and they respond the next day after that*.

The Indian devOps team adjusted its working hours to have overlapping hours with the Swedish team. This was done with working overtime, i.e., between 12 and 16 hours a day. As the technology lead says: "*they are working a lot, they are almost always available when we need them*". One way to mitigate the non-overlapping hours with the US team is that the Swedish team tries to speak with the Indian team members whenever they can instead of the US ones.

The Bulgarian team starts working at 11:00 Swedish time because of the adoption of the US company culture. This causes them to miss the daily stand-ups of the Swedish team in the morning and thus are not updated on the latest issues and cannot contribute to the other team members' problems. As the tech lead says: "*[the Bulgarians] miss a lot and don't get the context*". Participating in the daily stand-ups provides background and context to the project and to the problems that the Swedish team members are working on.

As it can be seen, communication and coordination are challenged by temporal distance. According to the respondents the main problem is the delay in responses. In addition, project control becomes more difficult with no overlapping hours [2].

B. Geographical

The physical distance creates a division among the teams from the different countries. The Bulgarian team does not share social time with the Swedish team and does not take part in the discussions. As the product owner puts it: "*with that kind of distance and not being present all the time, you can sense that they are not part of the sort of core team*". This was acknowledged by both parties. Mitigation strategies include a shared project backlog, occasional visits, and calls.

Also, because of the geographical distance, the Bulgarian and the Indian team are not able to collaborate closely with the Swedish team. In an agile setting, interactions happen quickly and small decision are taken all the time. As the tech lead puts it: "*the team takes small decisions all the time, you need to discuss things very often, small details, and it is very hard to bring them into those discussions*". Arranging a meeting with the offshored teams takes more time than an ad-hoc collaboration or meeting. For example, when there was a problem that the offshored team could not resolve within three weeks, they travelled to the Swedish site and resolved the issue in two hours. However, such visits are not frequent.

In addition, the Swedish team does not have a designated video meeting room for its stand-ups. Additional time would be required to set up a video meeting for each stand-up with the Bulgarian team. The Swedish team thinks that setting up a video meeting would slow down the local team. As the product owner said about a desirable solution: "*at the stand-up have all the screens already connected, just go into the room*".

Moreover, the geographical distance creates a communication lag, too. As the senior Bulgarian developer said: "*when I ask a question it takes quite some time to get the answer because you are not present. You can't just go [. . .] and ask them directly*". The senior Bulgarian developer mentioned that the people who are present have priority in getting their questions answered: "*they get an email from me and somebody goes to the technical lead and of course the person next to the technical lead will be more important, because the person is there, that person will get their answers and after that if there is time we will get our answers in an email*". This issue is mitigated by a designated person who is responsible for delivering the Bulgarian team's questions to the right parties and ask them in person.

Trust issues can also be found. Some of the Swedish team members do not trust the quality of work of the Bulgarian team. The company does not mitigate this by any agile practice. The tech lead mentioned that there may be a trust issue with the Indian team that keeps the development and test environments very closed. The Swedish team does not have full access to these and can not execute certain commands on them. This may be related to the security of the environments and the fact that the Indian team is accountable for them, but impedes progress for the Swedish team.

C. Sociocultural

A major issue that exacerbates the sociocultural distance is language. The Swedes report that members of both the Bulgarian and Indian team have accents that are difficult to

Table III
SUMMARY OF THE ISSUES AND COPING STRATEGIES FOUND IN THE CASE TEAM. HIGHLY SUCCESSFUL STRATEGIES ARE *highlighted*.

Issue in the team	**Issue source**	**Similar issue in literature**	**Coping strategies in the team**
Communication delay	• Offshore senior developer • Configuration manager • Offshore Senior devOps Engineer • Product owner	Holmström et al. [2]; Yadav [9]; Lee et al. [8]	• speak with the offshore team with extended working hours • *designated person to answer offshored team's questions* • *visits* • *teleconference, phone calls*
Missed daily stand-ups	• Tech lead • Product owner • Offshore senior developer		• separate stand-up with the offshored team • *visits* • iterations • reviews
One team feeling	• Tech lead • Product owner • Offshore senior developer	Holmström et al. [2]	• *shared project backlog* • *visits* • Skype calls
Close collaboration	• Tech lead • Product owner	Yadav [9]	• *visits* • ad-hoc meetings
No designated meeting room	• Product owner	Hossain et al. [11]	• *visits*
Trust	• Tech lead	Lee et al. [8]	• *frequent communication* (e.g., on Slack) • *visits* • sprint reviews • give complex tasks to the offshore back-end team
Language	• Tech lead • Product owner • Configuration Manager	Holmström et al. [2]; Lee et al. [8]	• asynchronous communication • *frequent communication* • get used to the accents • have a shared glossary of terms
Offshore team not taking decisions	• Tech lead	Highsmith [6]	• reviews • *frequent communication* • delegating work in work packages
No shared calendar	• Offshore senior developer	Lee et al. [8]	None
Not challenging the work tasks / misunderstandings	• Tech lead • Product owner • Configuration Manager	Ågerfalk and Fitzgerald [15]	• ad-hoc meetings • *visits* • sprint planning and reviews • provide and iterate detailed instructions in advance
Bureaucracy in decision making	• Configuration Manager	Ågerfalk and Fitzgerald [15]; Lee et al. [8]	• shared understanding on how each party makes decisions • *frequent communication*
Summer vacation and public holidays	• Tech lead • Product owner	Bass [20]	• dedicated member of Swedish team present during vacation time
Internal politics	• Tech lead • Configuration Manager	Bass [20]	• escalate to programme manager

understand. This can create misunderstandings in the team. The offshored team members in turn did not mention a problem with understanding the Swedes' accent.

Second, according to the tech lead: "*[the Bulgarian developers] are also very keen on that we take decisions"* and "*[. . .] you need to have developers that are independent and can take decisions themselves.*". This includes class names and detailed questions where the tech lead would expect autonomous decisions. This is mitigated by the Swedish team defining work packages for the Bulgarian team, which slows down the Swedish team. On the other hand, the Bulgarian team believes that architectural changes need to be approved by the tech lead because he is responsible for the platform while they only give advice. They say that this is a contractual agreement between the Bulgarian vendor company and the case company.

Moreover, because the offshored team works in a different company team members do not have access to the Swedish team's calendar. This makes it difficult to book meetings to resolve problems together. The project manager needs to find a time slot for joint meetings with a Swedish team member. This problem is coupled with the fact that the Swedish team does not always answer emails immediately when they receive a question from the Bulgarian team: "*when they get questions they say, well, I will answer the question in about an hour [. . .] and it takes several days because they forget*". Booking a meeting can take a day for the Bulgarian team: "*until they answer me the working day will be over and I won't be able to book a meeting time*". For these reasons, the Bulgarian team members switch to a different task until they get an answer.

Third, according to the tech lead the Indian team members "*tend not to challenge what you say. They often say, yeah, we will fix it, but they don't understand what they will fix*". This affects the quality of their work as the product owner puts it:

"*The quality of their delivery is not as we expect it to be*". This is somewhat mitigated: "*We tend to have a 2 weeks period time when we send instructions to them and iterate it between us, until we feel comfortable that they have understood those instructions*". On the other hand, the Bulgarian developers according to the tech lead "*won't give up until they understand what we mean.*"

Fourth, the Indian team is more bureaucratic. According to the configuration manager there is a "*differing view on management structure and who takes what decision and how many layers of management do we need to go through before we make a decision*". The Swedish team is organised in a less hierarchical way. This can create problems in reaching agreements which increases lead times.

Fifth, public holidays are not addressed in a good way. As the tech lead puts it: "*We are almost never aware when they have public holidays in California and India. Suddenly, they are not reachable*". Also, the Swedish team has a contiguous summer vacation from middle of July to end of August and this slows down the development of the offshored teams.

Sixth, office politics create tension between the Californian team and the Swedish team. As the configuration manager put it: "*when we need to communicate it can be a bit like two heads bashing together*". This issue is, however, outside of the Swedish team's jurisdiction and is addressed on the level of the programme manager.

VI. Discussion

This research explores the relationship between temporal, sociocultural and geographical distances and agile practices in a globally distributed software development team in a pharmaceutical setting. We interviewed six onshore and offshore team members. The results show that there is a bidirectional relationship between the agile practices and the distances. An agile practice can be affected by a distance and the agile practice can have an effect on a distance.

A. Agile practices to address the distances

The team uses various agile practices in a GSD setting (**RQ1a:** *How are Scrum, XP and other agile practices used by the case study team?*). We can see that the platform team cherry-picks certain XP practices (cf. Table IV). According to Appelo et al., complete adoption of agile practices is not necessary [29], validating this approach.

In contrast to Smite et al. [18], shared code ownership was partly used by the platform team for back-end software development, but not for front-end. Of the XP practices suggested by Kircher et al. [24] for GSD, the platform team uses metaphors and 40-hour work week partly and simple design, refactoring, and collective code ownership fully. They do not use small releases, test-driven development, or coding standards. Since planning game, and on-site customers are dependent on co-location [18], they are not used. Pair programming is only used on-site.

From Scrum, the Swedish team uses daily stand-ups, sprint reviews, sprint planning, and retrospectives which is in line with suggested practices for distributed agile [23], [22], [2] (cf. Table V). The stand-up practice is used partly by having an extra stand-up with the Bulgarian team. User stories do not all follow the structure proposed by Scrum and metaphors are not used in the tasks or user stories either. This can create misunderstandings and rework.

Moreover, of the agile practices suggested in Appelo et al. [29] (cf. Table VI), the platform team uses developer wikis, synchronous (phone calls, videoconferences, Skype calls) and asynchronous communication (emails, Slack and Skype messages), visits, iterations, and synchronised working hours which is in line with suggested scrum practices in GSD [16]. Also, the team practices version control, test automation, unit testing, and build verification testing fully. Code reviews are partially adopted.

Table IV
Summary of XP practices

Practice	Practised?	Comment
Test-driven development	No	Test-driven development is not practiced. As the Bulgarian senior developer puts it: "*usually the business wants something developed fast and test-driven development usually means spending more time on creating a test first*".
Metaphors	Partly	This practice is used among the developers but not in the user stories (secondary data).
Small releases	No	The team releases every other sprint. The production and QA environments are handled by the Indian devOps team and are under regulatory approval by the Mexican team.
Planning game	No	The developers use the Scrum planning practice.
Coding standards	No	Coding standards are currently being implemented.
On-site customers	No	The customer is represented through the Product Owner.
Pair programming	Partly	Pair programming is practised within teams but not between the Swedish and offshored teams.
Refactoring	Partly	According to the senior Bulgarian developer: "*small refactorings are done once per a couple of months at most*" and "*two bigger refactorings [have been going on for] almost a year now and they are not yet implemented*". According to user stories, one to two refactorings are completed per month.
Collective code ownership	Partly	Collective code ownership exists at the platform team, but not on the mobile or front-end development.
Simple design	Yes	The senior Bulgarian developer said that "*we usually have some kind of design initially. We try to use that and just add additional functionality using that design*" and he agreed that "*design is an ongoing activity, simplify existing work, backlog is not stagnant, don't add functionality before it is scheduled*".
Continuous integration	Yes	Continuous integration is handled by the Indian devOps team.
40 hour workweek	Partly	The Swedish and the Bulgarian teams do not work overtime but the Indian team does.

Table V
SUMMARY OF SCRUM PRACTICES

Practice	Practised?	Comment
Daily stand-up	Yes	The Bulgarian team has a daily stand-up at 13.00 Swedish time with the tech lead while the core team has its daily stand-up at 9.30. The Indian team does not participate in the core or Bulgarian Scrum meetings.
Sprint reviews	Yes	Only the PO is involved in the sprint reviews, not the customers. The configuration manager sees room for improvement: "*It's usually been: 'okay, you done that, you done that, that's good. I don't really know what that is'*".
Sprint planning	Yes	According to the senior offshore back-end developer, sprint planning takes too long since all developers participate in the story point estimation.
Retrospectives	Yes	Notes from team retrospectives are part of the secondary data. The team allocates all available time and does not leave slack: "*Should we plan 100% of resources time or 90-95%?*". This can create a problem when the team needs to do unplanned work: "*still have too much work coming in from outside the Jira. (unplanned)*". Also, actions are not taken on the retrospectives, impeding improvements of team performance: "*We haven't had any reflection/actions on the retrospective comments issued*".
Iterations	Yes	The platform team uses Scrum sprints and iterative development.
User Stories	Partly	The product owner comments: "*In the backlog we are not working with user stories, the user stories are in a shape of an epic, if I have a new user story, we create a new epic. And the epic will have story [tickets] connected to it where we work with the actual analysis and implementation of the features*". According to the tech lead: "*I would recommend having very well specified tasks, simple to understand what you say. We don't have that.*" The configuration manager's point of view: "*Very seldom a description or who ordered this, what use case are we trying to solve.*". According to the secondary data, the team has tasks, user stories, epics and bugs. The tasks are developer-centric, the user stories may come from developers, from the project manager or from the product owner. User stories sometimes have acceptance criteria or are connected to a customer. Some stories are extremely brief and do not always provide enough information to the developer. They are not always connected to an epic. Some epics are currently worked on but are not "In Progress". According to the notes from the retrospectives the Jira tasks/user stories are not detailed enough: "*Write more crispy Jiras, i.e., complete one-liners with more information before bringing the Jiras into the sprints*" and "*Requirements are vague or totally missing, e.g., goals*". This issue was also mentioned in the interview with the tech lead.

To answer RQ1a, we see that the platform team cherry-picks XP, Scrum, and other agile practices that the three distances and the organisational barriers (separate devOps team, internal politics) allow them to use. What both XP and Scrum practices suggest are on-going retrospectives with actionable results [29]. However, as profit is connected to the delivery of working software, billable hours and the delivery of new features might be prioritised over the learning.

B. Effect of agile practices on distances

A number of agile practices affect the three distances (**RQ1b:** *How do agile practices affect temporal, geographical and sociocultural distances in the case study team?*). We observed that *sprint planning and review* reduce the sociocultural distance since they are used to clarify misunderstandings and thus, improving communication and coordination [16]. Also, these practices affect the geographical distance even when meetings are held using online collaboration tools, such as video conferencing and screen sharing. These tools make it possible to have the meeting across borders, improving communication and coordination.

We also observed that *daily stand-ups* had an effect on the sociocultural distance because frequent communication helps the team to sense and respond to problems which improves communication, as also observed in [8]. They also help with the geographical distance as they increase task awareness (coordination) and help to convey the vision and strategy (control) [16].

The *shared online backlog* used by the team affects the geographical distance since it increases coordination by allowing everyone to keep track of their tasks. Our observations are thus similar to those in [2]. Also, it helps management to control and to keep track of the team's progress according to the interviews.

Retrospectives as used by the team affect the temporal distance by using synchronous communication tools to reflect upon what works and what does not (communication). Also, they affect the geographical distance as they increase critical task awareness (coordination). These communication tools also help with sociocultural distance as they reduce conflicts arising from misunderstandings (coordination) [16].

Continuous Integration affects the geographical and temporal distance. The build server is up 24 hours a day and the developers can push code to it whenever they want to. We observed that this supports control by providing feedback on the quality of the integrated code in real time.

Automated unit and build verification tests reduce the geographical and temporal distance since the tests are executed automatically and the results are available online. We observed that the team executes the tests and monitors their status from anywhere at any time, reducing coordination and control costs according to the interviews.

On-site visits reduce the sociocultural distance as the visits improve collaboration and improve communication according to our data. They support informal communication by allowing the team members to get to know each other better, thus improving commincation that takes place over a distance at later points in time. Lack of informal communication was found a problem in Holmström et al.'s [2] study.

As we can see from the discussion above, the answer to RQ1b boils down to several scrum ceremonies and other agile

Table VI
SUMMARY OF OTHER AGILE PRACTICES ACCORDING TO APPELO [29]

Practice	Practised?	Comment
Version control	Yes	As the senior devOps engineer said: "*everytime a developer does a checkin [...]it automatically triggers a build and it deploys it to the dev environment. We do version it and push it to the shared online repository.*"
Build verification testing	Yes	According to the senior devOps engineer: "*...for each CI build they have test cases*" and "*When the test case failed the build fails automatically and they have to fix it.*"
Test automation	Yes	The secondary data shows that the team uses end-to-end testing so that all APIs can be tested by running test scripts.
Unit testing	Yes	From the secondary data review, the team uses JUnit testing.
Code reviews	Partly	Secondary data shows that the team started to use code reviews frequently from May 2018 but only with the Bulgarian team.
Developer documentation (wikis)	Yes	Confluence holds developer documentation about the architecture, branching strategy, developer how-tos, cookie policy, onboarding a new project, and developer guidelines.
Synchronised work hours	Partly	The Indian team works overtime and synchronised its working hours with the Swedish team. The Bulgarian team has overlapping work hours with the US team.
Synchronous communication	Yes	The Indian and Bulgarian teams use individual or conference calls with the Swedish team.
Asynchronous communication	Yes	The Indian and Bulgarian teams use email, instant messaging, and wikis with the Swedish team.
Visits	Partly	The Bulgarian team has visited the Swedish team several times in the past two years.
Frequent communication	Yes	Ad-hoc communication, e.g., on Slack, is common among the team members.

practices as these help with the three distances. Also, there are some tools (such as online collaboration tools, backlog in the cloud, build server) that are not necessary agile, but support these agile practices, and help reduce the three distances.

C. Effect of distances on agile practices

The three distances have an effect on how the agile practices can be used and how effective they are (**RQ1c:** *How do temporal, geographical and sociocultural distances affect agile practices in the case study team?*).

Sprint planning and review are affected by geographical distance by the lack of a dedicated meeting room. Therefore, it takes longer to set up the meeting than to set up a face-to-face meeting and coordination costs increase. Similar issues have been observed by Hossain et al. [11].

Daily stand-ups are affected by the temporal distance since two meetings need to be conducted, one with the Swedish team and another meeting with the Bulgarian team. This increases the coordination effort [11].

User stories are affected by the sociocultural distance since the Indian team tends to agree to tasks without deeper discussion. There are also persistent misunderstandings and language problems. To mitigate this and reach agreements, the Indian team gets very detailed instructions. This increases the communication effort.

Simple design is affected by the sociocultural distance by not making decisions without approval of the tech lead. This increases the communication effort.

40-hour work week is affected by the temporal distance as the Indian team works overtime to have overlapping working hours with the Swedish team. This also has a negative impact on *sustainable pace*.

Pair programming is affected by the geographical distance. Coordination costs are increased as it is more difficult to organise meetings without shared calendars (coordination). Also, the team members have to communicate through online tools and not face-to-face. On the other hand, Holmström et al. [2] found pair programming a practice to increase understanding and time overlap between the onshore and offshore colleagues.

Release planning is affected by all three distances. Temporal, as the Indian team deploys the code to production and the Mexican QA lead verifies the signed off documents. Geographical and socio-cultural, because the Swedish team works with offshore teams. This increases communication, control and coordination costs and the platform team does not deliver a release in every sprint. It was found in Bass' [20] study that teams in similar situations also struggle to deliver functionality in every sprint.

Co-located teams are affected by geographical distance by not being able to sit at the same place. This increases communication, control and coordination costs.

The points mentioned above show how the three distances affect the feasibility and efficacy of some of the agile practices the team is practising. All distances affect the communication, coordination, and control costs in a negative way.

D. Coping strategies

There are different coping strategies that the team uses to mitigate the three distances (**RQ2:** *Which coping strategies does the team use to address the three distances?*, cf. Table III). The Indian team synchronised its working hours with the Swedish team [16]. On the other hand, the Bulgarian team's working hours are not synchronised with the Swedish team's working hours. This creates problems in communication, coordination and control. The team uses ICT-mediated synchronous communication for formal and informal communication through phone calls and conference calls. These tools are available regardless of methodology [16]. The platform team also uses ICT-mediated asynchronous communication, such as instant messaging through Slack and Skype and project wikis through Confluence. These tools help to minimise the disruption in time-zones, work day hours, or public holiday differences [20], [9]. On-site visits by the Bulgarian team help to build relationships, trust and to get to know each other's cultures [20]. Frequent

communication enables detecting changes quickly, clear up misunderstandings and to solve problems in a collaborative fashion. Iterations are important to monitor progress and resolve issues in a repetitive way. The team uses Scrum release cycles of a month with two week sprints. The shorter the sprints the more iterations the team goes through that help in monitoring progress and resolving issues. Reflections in the team occur during Sprint retrospectives, but the team does not always act on the reflections. This inhibits learning and performance improvement in the team. Sprint planning helps with scoping the work, resourcing, and scheduling. The team goes through planning once a month which they perceive as increasing the amount of work. According to the senior Bulgarian developer the planning takes up too much time.

There are coping strategies that are not categorised as agile in the literature but are used by the team to address the three distances. They include having a separate stand up with the Bulgarian team to mitigate the context loss, assigning a designated person to answer the offshored team's questions to mitigate the communication delay, giving complex tasks to the offshore back-end team to mitigate trust problems, giving detailed instructions to the Indian devOps team two weeks in advance, iterating the questions until sufficient understanding is reached to mitigate misunderstandings, escalate issues to the programme manager to deal with internal politics and to synchronise working hours to deal with the time difference. As we can see, purely agile practices do not solve all the problems this globally distributed team has and other strategies are needed.

VII. Conclusion

In summary, we concluded that the three distances — temporal, geographical and sociocultural — have an effect on agile practices and that agile practices have an effect on the three distances. We report how the team uses agile practices with the offshore team members in a modified way. Some practices are used only partially or not at all. Moreover, the platform team uses agile and rigorous coping strategies that arise from the three distances with a varying degree of success. We thus provide guidance for practitioners as to which agile and non-agile practices can be combined to address the distances, in how far issues are addressed, and which challenges persist.

For future research it would be interesting to measure the benefits vs. the cost of the distance-agile relationship in terms of communication, control and coordination. Also, the relationship of traditional, non-agile practices and the three distances is an interesting subject of further investigation.

References

[1] S. Sahay, "Global software alliances: the challenge of 'standardization'," *Scandinavian Journal of Information Systems*, vol. 15, no. 1, p. 11, 2003.

[2] H. Holmström, B. Fitzgerald, P. J. Ågerfalk, and E. Ó. Conchúir, "Agile practices reduce distance in global software development," *Information systems management*, vol. 23, no. 3, pp. 7–18, 2006.

[3] J. D. Herbsleb and A. Mockus, "An empirical study of speed and communication in globally distributed software development," *IEEE Transactions on software engineering*, vol. 29, no. 6, pp. 481–494, 2003.

[4] D. Damian, "Workshop on global software development," *SIGSOFT Software. Eng. Notes*, vol. 27, no. 5, 2002.

[5] P. J. Agerfalk, B. Fitzgerald, H. Holmstrom Olsson, B. Lings, B. Lundell, and E. Ó Conchúir, "A framework for considering opportunities and threats in distributed software development," 2005.

[6] J. Highsmith, "The great methodologies debate: Part 2," *Cutter IT Journal*, vol. 15, no. 1, 2002.

[7] B. Fitzgerald, G. Hartnett, and K. Conboy, "Customising agile methods to software practices at intel shannon," *European Journal of Information Systems*, vol. 15, no. 2, pp. 200–213, 2006.

[8] G. Lee, W. DeLone, and J. A. Espinosa, "Ambidextrous coping strategies in globally distributed software development projects," *Communications of the ACM*, vol. 49, no. 10, pp. 35–40, 2006.

[9] V. Yadav, "A flexible management approach for globally distributed software projects," *Global Journal of Flexible Systems Management*, vol. 17, no. 1, pp. 29–40, 2016.

[10] H. Holtz and F. Maurer, "Knowledge management support for distributed agile processes," in *Proceedings of the Workshop on Learning Software Organizations (LSO)*, 2002.

[11] E. Hossain, M. A. Babar, and H.-y. Paik, "Using scrum in global software development: a systematic literature review," in *4th IEEE Int. Conf. on Global Software Engineering (ICGSE 2009)*. IEEE, 2009, pp. 175–184.

[12] G. K. Hanssen, D. Šmite, and N. B. Moe, "Signs of agile trends in global software engineering research: A tertiary study," in *6th IEEE Int. Conf. on Global Software Engineering Workshop (ICGSEW)*. IEEE, 2011, pp. 17–23.

[13] E. Carmel and P. Tjia, *Offshoring information technology: Sourcing and outsourcing to a global workforce*. Cambridge University Press, 2005.

[14] E. Carmel and R. Agarwal, "Tactical approaches for alleviating distance in global software development," *IEEE Software*, vol. 18, no. 2, pp. 22–29, 2001.

[15] J. Ågerfalk and B. Fitzgerald, "Flexible and distributed software processes: old petunias in new bowls," *Communications of the ACM*, vol. 49, pp. 27–34, 2006.

[16] E. Hossain, P. L. Bannerman, and D. R. Jeffery, "Scrum practices in global software development: a research framework," in *International Conference on Product Focused Software Process Improvement (PROFES)*. Springer, 2011, pp. 88–102.

[17] K. Kaur, A. Jajoo *et al.*, "Applying agile methodologies in industry projects: Benefits and challenges," in *Int. Conf. on Computing Communication Control and Automation (ICCUBEA)*. IEEE, 2015, pp. 832–836.

[18] D. Šmite, N. B. Moe, and P. J. Ågerfalk, *Agility across time and space: Implementing agile methods in global software projects*. Springer Science & Business Media, 2010.

[19] K. Schwaber, *Agile project management with Scrum*. Microsoft press, 2004.

[20] J. M. Bass, "Influences on agile practice tailoring in enterprise software development," in *AGILE India*. IEEE, 2012, pp. 1–9.

[21] C. Larman, *Agile and iterative development: a manager's guide*. Addison-Wesley Professional, 2004.

[22] M. T. Hansen and H. Baggesen, "From cmmi and isolation to scrum, agile, lean and collaboration," in *Agile Conference (AGILE'09)*. IEEE, 2009, pp. 283–288.

[23] M. Paasivaara, S. Durasiewicz, and C. Lassenius, "Distributed agile development: Using scrum in a large project," in *IEEE Int. Conf. on Global Software Engineering (ICGSE)*. IEEE, 2008, pp. 87–95.

[24] M. Kircher, P. Jain, A. Corsaro, and D. Levine, "Distributed extreme programming," *Proceedings of the Internation Conference on Extreme Programming and Flexible Processes in Software Engineering (XP 2001)*, pp. 66–71, 2001.

[25] A. Bryman and E. Bell, *Business research methods*. Oxford University Press, USA, 2015.

[26] J. W. Creswell and J. D. Creswell, *Research design: Qualitative, quantitative, and mixed methods approaches*. Sage publications, 2014, 4th edition.

[27] A. Fruhling and G.-J. D. Vreede, "Field experiences with extreme programming: developing an emergency response system," *Journal of Management Information Systems*, vol. 22, no. 4, pp. 39–68, 2006.

[28] G. Gibbs, C. Taylor, and A. Lewins, "Online QDA – Quality of qualitative analysis," accessed 28 April 2018. [Online]. Available: http://onlineqda.hud.ac.uk/Intro_QDA/quality.php

[29] J. Appelo, "The big list of agile practices," accessed 18 June 2018. [Online]. Available: https://dzone.com/articles/big-list-agile-practices