Evaluation of A Communication Practice

Ruxandra-Stefania Tudose

IV1303: Modern Software Development, Spring 2024

Abstract

Advanced technical projects and problems are commensurate with complex situations that especially when given a certain limited time frame, cannot be dealt with by one engineer only. Therefore, for an effective approach, peers have to collaborate and work together in order to find the best possible solutions for the encountered problems. In order to do so, communication lies at the core behind resolving them and bringing the final product or vision to fruition. This paper's purpose is to highlight the impact of communication within an engineering team caused by the developers' gap in their level of mastering the foreign language they collaborate in and to suggest a possible solution in that sense. Having said that, based on research papers, discoveries combined with the personal observations made while working in such a team, it seems that regardless of how well a person is technically trained, the fact that the communication bridge between two or more peers may be missing can turn into a major deterrent in the evolution of the project.

Keywords

Technical peer collaboration, foreign language skill gap among developers, exchange ideas, idea explanation approaches

Contents

1 Introduction

Ever since the integration of technology and more precisely of computers into any individual's everyday tasks, from the simple to the more complex ones, collaboration between engineers making this technology a reality and naturally, constantly improving it has turned into an essential aspect, which is part of today's modern society. Therefore, considering its importance, this is precisely why it has become a research subject upon which, attention has constantly been drawn. Having said that, this idea is best encapsulated by the following phrase:*"Software engineering research has looked at humans and their collaborations from its very beginning." [1]*

Applied sciences are often viewed from an external standpoint as a demanding field that requires attention to detail, a large set of skills which have to be well mastered, consistency and naturally, passion for the subject. However, when it comes to actually explaining a certain technical concept to a person with little to no background on the problem, this can become even more challenging and the moment the communication tool is not used at its best, the problem difficulty can easily see an exponential increase.

To put it briefly, this paper aims to highlight to what extent the gap among any team's developers in terms of their skills of mastering the foreign language they communicate and collaborate in, can affect the progress of the project they have to finalise together. Having said that, the assessment will be done by monitoring the communication practice in the above mentioned context and by comparing the findings with other researchers' international discoveries.

Moreover, the outline of this report is as follows. Apart from the introduction, there can be identified 5 well distinguished and at the same time interconnected parts. First and foremost, part 2 is meant put the reader into the context of the chosen practice, namely communication as well as create an understanding of the environment it has been evaluated in. Next, part 3 comes as a follow-up since it outlines the steps that have been taken in order to systematically conduct the research phases, while part 4 represents its continuation as it highlights the raw results of the different findings. Next, the fifth part complements the previous one as it presents a thorough analysis of the obtained results in order to integrate, compare and contrast them with other international findings. And last but not least, part 6 is indicative of the paper's final conclusions and at the same time, as far as the chosen topic is concerned, it puts into the spotlight the possibilities of future work within this field of study.

2 Background

2.1 Introduction

When working on a complex idea or project in the field of software engineering, if the tasks are not approached in a systematic, well planned way, this can easily lead to the developers finding themselves lost in details or even skipping important technical aspects that should have been taken into account along the way. As a result, in order to avoid these unwanted situations, software development methods should be put into practice since their implementation does make a difference in that sense.

2.2 Software Development Methods

As far as the software development methods are concerned, they represent by definition, "structured approaches to some work (software development) which include process models and system models, notations, rules, design advice" [7] and at the same time, they can be split into two main categories: the traditional and the agile ones. To begin with, the traditional ones can be summed up to being defined by measurable results, which enhance predictability, as well as strictness on flexibility along the entire process, facts that emphasise the need of high detailed documentation, which in turn leads to slow progress. On the other hand, the agile ones present quite the opposite properties: they gravitate around very little to nothing being fixed, they are open to changes at any given time, they avoid documentation as much as possible, therefore the reason why the progress is much faster and last but not least, in turn, they lead to less overall predictability. Having said that, by highlighting these brief descriptions, one can perceive them as being indicative of the fact that the main difference between the agile and traditional methods represents the type of situation in which one is used over the other. In other words, complex problems present multiple unknowns that most of the time are discovered along the way, which in turn lead to less predictability. That being said, this idea can be resumed to the following conclusion: the more complex a problem presents itself to be, the less likely does it become to tackle it using a traditional method as under these circumstances, given the lack of having full control over the situation in order to plan it in detail, such a method cannot be implemented [6].

2.3 Software Development Practices

Furthermore, a practice is by definition a "customary, habitual, or expected procedure or way of doing of something"[7]. In other words, in the context of engineering, this refers to the concept of developing and planning technical tasks on a consistent routine, which is indicative of well defined steps that should be taken in order to achieve the desired goal. Having said that,

as far as software development is concerned, the most known practices in that sense are: pair programming, test-first development, communication, retrospectives, user stories, story points, planning and risk management [8]. Having said that, out of all the aforementioned practices, the report will focus on one only, namely communication.

2.4 The Communication Practice

In simple terms, communication can easily be perceived or understood as the exchange of different pieces of information between two or more people. Naturally, in everyday life, communication can take different shapes based on the model or pattern it follows [9]. To illustrate this, it can either be verbal, non-verbal and so on and so forth. Having said that, this paper will put into the spotlight one type of communication only, that can or should be identified in any modern software development environment. This is also known as *coordination communication* and it occurs when one or more developers have to coordinate the way they exchange technical details or other important dependencies of their work, manage possible conflicts and discuss the generally approved task strategy in order to in the end, succeed in combining their personally developed pieces. In other words, by putting those pieces together, they contribute to a significant part that adds up to the final product itself [1].

2.5 The Project Course

Last but not least, as far as the environment this practice has been evaluated in, it should be taken into account that the personal findings and perspectives are brought to this paper through my participation in the *II1305 Project in Information and Communication Technology* course offered at KTH Royal Institute of Technology (KTH) during the 2024 spring semester. Having said that, for a month, I was part of the English speaking 8 member team called Albatross, whose goal was to develop within this time frame an application. The final created software app is supposed to enable users to find out predictions of their English accent, concluded by analyzing the submitted recorded or uploaded audio file. The only evident user constraints are that the audio files have to be between 5 and 60 seconds long and that they should only contain content in English with as few background noise as possible. In addition, the software development method used throughout the project course was an agile one, namely Scrum. From a technical standpoint, the project represents a combination between front-end programming by using Android Studio for the app's framework with back-end programming through the implementation of the server and the creation of the audio classifier in the context of the artificial intelligence neural network that would predict the user's accent. In that sense, Python, more precisely

the TensorFlow library, an open source instrument provided by Google, was used. Naturally, in order to ensure the health and progress of the project, a series of practices and tools have been used such as: communication (oral and written), user stories, pair programming from time to time (especially for the periodic code review), spring retrospectives and of course planning, which most of the time led to backlog refinement. Each working day was concluded by the Daily Scrum meeting in order to keep everyone updated on the work and tasks that were both accomplished and that were left to be completed during the upcoming days.

3 Research Method

3.1 Introduction

In order to in the end, ensure high quality conclusions and results of the studied chosen practice, as it can be noticed in *Figure 1*, a systematic approach of the research phases has been designed and followed step by step. This way, new pieces of information were included and gathered consistently, avoiding time gaps.

3.2 The Research Steps

The first and most natural step in getting started with the research method was re-reading the lecture slides on the chosen topic provided in the IV1303 Modern Software Development course offered in the spring semester 2024 at KTH, followed by going through other several research papers in order to get even more acquainted with the communication practice. In order to find and access the necessary scientific literature on this subject, the online KTH Library website has been used. Having said that, once the big picture, goal and purpose of the picked subject were clear, the actual monitoring of the practice within the project course began. The monitoring was done during 5 consecutive work days with two exceptions caused by three present legal days off in the schedule. However, in total, all this time can be summed up to 4 complete organised Scrum Sprints within the project course. The goal was to observe the communication flow in English within the team, especially in the moments where two or more peers were interacting namely, by debating, analyzing the possibility of a new solution or exchanging ideas. Most of the times, those were the exact moments when multiple individual project parts were to be combined together, which ensured the progress and development of the final product.

After each working day, more precisely during the Daily Scrum meetings, scratch notes and personal observations were taken, which at the end of each week were put together in the written form. As far as the members'

English study background is concerned, it should be taken into account that no direct data collection was made in that sense. And that was because my intention was not to make anyone feel pressured or continuously evaluated, knowing that I, a member of the team itself was taking note of this particular aspect. Had I done that, I fear it might have negatively affected the project's health and evolution, which was naturally not a desired outcome. Therefore, the presented data is solely based on personal observations and the pieces of information provided on the KTH website related to the English entry requirements necessary for the Bachelor's program in Information and Communication Technology (BSc. ICT). At the end of the final Sprint, all the gathered results were compiled together and more research papers on the topic were continued to be read. After having done that, it came as a smooth next step to integrate and compare my findings with the data and pieces of information drawn from the read literature.

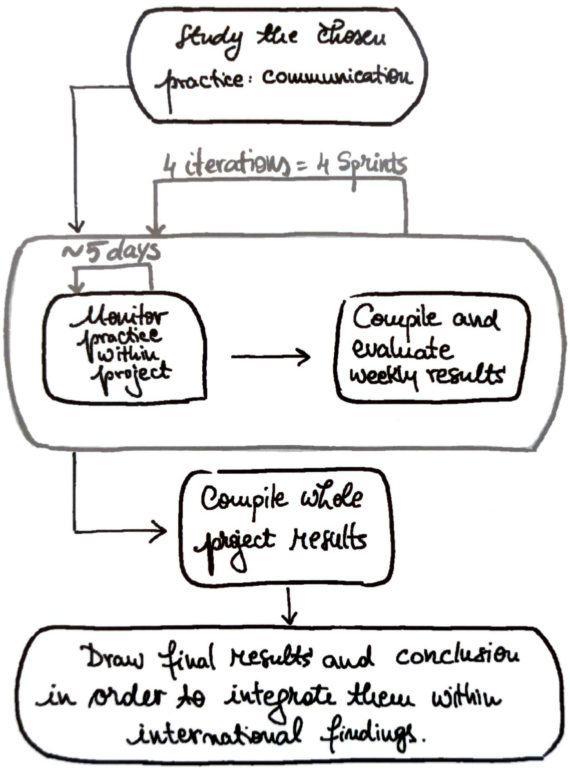

Figure 1: The design of the research method phases.

3.3 The Evaluation Criteria

Foreign language fluency refers to the extent to which an individual can express themselves clearly, with a precise goal, without too many pauses in their speech caused by the unnecessary use of interjections meant to hide poor vocabulary, lack of language skills, inability to use sentence connectors or linking words [2]. Any bilingual or multilingual individual's proficiency in the foreign languages they master can be evaluated by making use of the different available international scales in that sense [3]. In this report, we will assess whether and how the varying skills in mastering English language proficiency among developers impact the progress of the team's technical project. In other words, significant milestones in the project's evolution are evaluated first and foremost, during the instances where various app components created by different developers have to be integrated. From a communication standpoint, these identified moments are assessed based on how well peers manage them by explaining their own parts and by understanding the mutual requirements in order to put them together.

4 Project Results

4.1 Introduction

Since the research method phases have now been introduced to, it is time to dive into the raw collected results obtained by having followed the steps highlighted in *Figure 1.* Having said that, the weekly findings will first and foremost be presented in four distinct parts, which align and correspond to the four Scrum Sprints. Consequently, a comprehensive overview of the final results will be provided under the form of a direct or inverse proportional relationship between the communication quality in English and the overall error rates, individual tiredness and peer collaboration. It should also be taken into account that in accordance with the presented evaluation criteria in part 3, every week, at least two essential moments were identified when the project would not see any evolution caused by the poor coordination communication. As the weeks went by, naturally, the tasks would increase in difficulty as the product was becoming more and more complex, therefore the way communication impacted the project's work flow and the encountered impediments were followed as presented below:

4.2 The Weekly Compiled Results

To begin with, the first week was marked by the peers getting accustomed to the pace and each individual's work style, as not everyone knew each other beforehand. The most important milestones were the discussions on the establishment of the team's social contract, the user stories and the server schematic debate marked by misunderstandings within the team due to the

often use of unnecessary interjections in their speech. In other words, the first Sprint was marked by chaotic communication due to a waterfall ideas presented in an unsystematic way without using linking words, sudden work interruptions caused by frequent peer to peer technical questions formulated without using the proper technical vocabulary, increased tiredness by the middle of the week and slight team division. More precisely, as far as the frequency of the posed questions is concerned, on average, a question was being addressed to the same team member *every 15 minutes*, while if one of the peers could not make themselves understood in English after *3 times*, either the listener or the speaker would give up on the explanation and temporarily abandon the subject entirely.

Furthermore, the second Sprint was even more intense. Following the user stories, during the week, multiple individually app developed parts were created and were ready to be assembled, however great impediments were faced in exchanging the pieces of information related to what one peer required from the other in order to integrate their code accordingly. To illustrate this, the exported .json data files had a specific format, the server needed a clear data header of the audio file size before the actual content was being sent and the app to server TCP connection had to follow certain steps. During this sprint, there were still frequent technical questions, approximately every *20 to 25 minutes*, however if one of the peers could not make themselves understood in English after *the first time*, either the listener or the speaker would give up on the explanation, which led to increased tiredness much earlier in the week, lack of personal motivation, less patience with each other, frustration and in the end to an extensive meeting during which, the entire social contract was redefined.

To continue with, the third week has, naturally seen different results. The social contract had already been revisited, so the expectations were clearer and since there was more understanding of each individual's way of working, less tiredness, more interaction was noticed on a team level and more patience with each other was taken account of. Furthermore, in our technical answers it was agreed upon doing our utmost to use more detailed descriptions and appropriate technical vocabulary. As a result, because of previous lack of coordination communication, during the second Sprint, the server to app connection had been delayed for 4 days, time during which the project stagnated because of a large series of errors. Now, in the third Sprint the aforementioned technical issue was made functional in approximately half a day of constant work and that was because a clear peer to peer schedule was created in terms of exchanging ideas, while all questions, which were posed *every hour or so* were fully and in detail clarified either verbally or with the aid of extensive schematics on the board, using less interjections and more precise technical English words. The successful server connection also

boosted team motivation and excitement related to getting closer and closer to finalising the application. Last but not least, the fourth Sprint results are similar to the ones obtained in the third and will therefore, only be briefly mentioned. For similar reasons, they include noticeable fast progress with no major errors, better technical explanations complemented by schematics and more in depth discussions related to the mutual necessary requirements before diving into the work. By and large, it was observed that the more complicated a technical issue was, the longer and more extensive explanations were needed. As noticed, the same challenging idea was presented, on average, *at least 2 to 3 times* until it was fully grasped.

4.3 Overall Compiled Results

According to the overall compiled result, there exists *a direct proportional* relationship between the quality of the coordination communication in English and error rates, team motivation, tiredness, collaboration and productivity. In other words, once this practice is used at its best, it significantly impacts and improves the other aforementioned aspects. The overall compiled results also reflect an essential and at the same time, interesting aspect that was not fully concluded until all four Sprint Demos took place. Having said that, it has been noticed that from a technical communication standpoint, there exists an increased peer tendency in assuming that others who have little to no background on the problem one is working on, are fully aware of the details one has been developing for an extensive time. As a result, similar to the atmosphere within the team, especially in the beginning of the course, the explanations lack details and a clear, logical structure with linking words or connectors and consist in minimum clarifications using the adequate technical descriptions. Not only was there very little to none interaction with the audience during the Sprint Demos, but there was also very little personal enthusiasm communicating about the product, which in turn led to those listening to the presentation lose interest into a highly interesting product. In addition, it should be taken into account that the audience consisted in the course responsible and the other enrolled students.

5 Results Analysis

5.1 Social Context

Considering the project results presented in part 4, it can easily be observed that it took two weeks of constant effort before the team members broke the ice and found the best approach in making themselves understood in English. After all, in any field of study it takes time for people to get accustomed to the work style and personality of their team members. However, judging by the results of my assessment and my overall undergraduate

experience at KTH, it is clear to me that social collaboration in software engineering is a huge obstacle that is most of the time very difficult if not impossible to overcome. In other words, it is easier to grasp when one understands that this fact is linked to the type of personality that is suited to this field. According to the ideas highlighted in *"Collaborative Software Engineering"* by André van der Hoek, software development is indicative of individuals who most likely prefer an isolated environment where they can fully concentrate for an extensive time frame, with little or no interaction, so that they can achieve their goal. That is to say, this profession tends to be suitable for introverts rather than extroverts [1]. On the other hand, no matter how much some would prefer to work alone, it is well-known that nothing great can be achieved by one engineer only. And that is precisely why, high quality peer to peer coordination communication in the language they work in (in our case, English) should be encouraged, which in turn leads to active collaboration and better technical results.

On a society level, it seems to me it that there has been a long period of time, during which from an external standpoint, engineers have only been encouraged to expand their technical skills, followed by the claim that those are the abilities that truly help them achieve a great career in this field of study, while ignoring other essential know-hows. Those know-hows are part of the human nature, such as effective communication in an international foreign language, such as English and when used properly, it can actually contribute equally or even more to the product's success. As it can be noticed, after my team and I had redefined the social contract, became aware of the fact that most of our errors' source was miscommunication in the English language and changed the explanation style by including more details and looking up for technical vocabulary in English, that was the exact moment when the project itself saw huge progress. Not only did we advance much faster compared to the first two weeks, but there was also a more pleasant work environment. Interestingly enough, my findings and encapsulated conclusions seem to also align with the following idea that is highlighted in the article *"8 Communication Techniques Engineering Leaders Need to Succeed"*, published by Harvard Business School:*"Whether connecting with clients, a new employer, or peers, your communication skills can be just as important as your technical knowledge"*[10].

5.2 The Results Into Perspective

The team I was part of consisted in peers from all walks of life and as mentioned in part 2, although no data collection was made on the developers' English study background, team members do share a common starting point in that sense. In other words, according to the KTH and the Swedish UniversityAdmissions website, all students enrolled in the KTH BSc. pro-

gramme in ICT should meet the necessary English proficiency requirements either through an international test or through the Swedish upper secondary course, namely English 6 [11][12]. In spite of this common ground, many differences in their spoken language, vocabulary diversity, presentation skills and grammar use were noticed.

Looking back and by having read corresponding scientific literature in that sense, it turns out, the reason why developers in my team gave up very fast or had a really difficult time making themselves understood was the lack of extended English vocabulary, linking words and grammar structures, fact which led to frustration and last but not least, abandoning expressing themselves or answering any technical questions. In addition, the frequent unnecessary pauses or misuse of interjections led to less peer patience and avoidance in asking for further clarifications, if needed. Therefore, the different levels in mastering the English language in this case, highly affected the understanding of the technical goals in the first two weeks, which in the end, apart from other negative outcomes, it significantly impacted the developer's tiredness. And that is because in those confusing moments, not only is brain power needed to link the scientific facts, but also to strive to understand what the other developers are grammatically saying in English.

By and large, according to the findings highlighted in the Harvard Business Review article, *"Global Business Speaks English"*, these issues seem to be the root cause of the noticed wide spread team division and tendency in expecting colleague engineers to be aware of the specific technical details part of their peers' tasks even when they are not directly involved, which leads to self-isolation from the team: *"Many may feel at a disadvantage if their English isn't as good as others', team dynamics and performance can suffer[...][4]"*

You might now be wondering what a possible solution given this situation would be. Well, to begin with, it is not always possible that highly skilled technical developers with the exact same English and communication skills are brought together in order to collaborate. Therefore, one possible alternative I have monitored was the implementation of detailed schematics whenever there is lack of vocabulary or whenever one can simply not find the proper words to express themselves in English. To illustrate this, one colleague explaining a confusing idea in front of the whiteboard would mention: *"If this was supposed to be the file you wanted to connect to the server, where should it go next? What data should I merge in the header here?"*, while drawing simple squares representing the files and the server. Therefore, as put in the spotlight by the *"How to craft a narrative that matters"* in the Harvard Business Review, by simplifying the steps that are to be taken next, similar to a story, it becomes visually easier to understand the 'narra-

tive' from a technical standpoint: *"Research has shown that storytelling has a remarkable ability to connect people and inspire them to take action. [5]"*.

Looking back, this exact strategy has shown to be the source of the seen improved results during the third and fourth Sprint in my team. And all in all, the result analysis combined with other relevant international findings indicates that one of the biggest impediment in the health and progress of a project are all the negative outcomes generated by poor coordination communication, being in this case generated by the lack of mastering the foreign language the team uses when collaborating, namely English. After all, nothing can be manufactured, coded, designed or assembled if the technical context or requirements are not fully grasped, therefore the reason why high quality communication in that sense plays an essential role in achieving the project's success.

6 Conclusions and Future Work

By and large, throughout this paper, important findings and conclusions related to the communication practice have been highlighted by integrating personal gathered results as a participant in a project course with other international, relevant remarks on the topic. As a result, the conducted research has put into the spotlight this practice's importance in any working environment within the software engineering field of study and how much it can negatively influence the product's or vision's development if the foreign language is not used at its best by the speakers. Having said that, I very much hope that this report will help with further research in this area and that at the same time, it will serve as a good source of information based on which peer to peer interaction in technical teams can be improved from now on, so that any project's health and progress can be maintained. However, in spite of these thorough findings, I do believe, there is without a doubt, so much left to look ahead.

To my mind, in order to increase the accuracy of the result analysis, one point of interest within future work includes the integration of this paper's conclusions with peer collected data about their English study background, as this new piece of information would give even more valuable insights into this practice. In addition, this could also be indicative of what could be improved or changed in order to ensure that the moment coworkers begin their collaboration, their foreign language level and skills are as close as possible, ensuring smooth technical communication.

7 References

[1] *"Collaborative Software Engineering"* by André van der Hoek, [online], available: https://shorturl.at/arY12

[2] *Fluency* in the British Council [online], available: https://shorturl.at/hkx07

[3] *Proficiency Scales* in Language Testing International, [online], available: https://shorturl.at/iKOV9

[4] *"Global Business Speaks English"* in the Harvard Business Review, [online], available: https://hbr.org/2012/05/global-business-speaks-english

[5] *"Storytelling That Drives Bold Change"*, in the Harvard Business Review, [online], available: https://hbr.org/2023/11/storytelling-that-drives-bold-change

[6] *Difference between Traditional and Agile Software Development*, [onine], available, https://www.geeksforgeeks.org/difference-between-traditional-and-agile-software-development/

[7] KTH, IV1303: Modern Software Development, Spring 2024: Lecture 1

[8] KTH, IV1303: Modern Software Development, Spring 2024: Report instructions PDF

[9] *Communication* in the Wikipedia, [online], available: https://en.wikipedia.org/wiki/Communication

[10] *8 Communication Techniques Engineering Leaders Need To Succeed* in the Harvard Business School, [online], available at: https://online.hbs.edu/blog/post/communication-techniques

[11] *Entry requirements for Information and Communication Technology* on the KTH website, [online], available:https://www.kth.se/en/studies/bachelor/information-communication-technology/entry-requirements-for-information-and-communication-technology-1.450313

[12] *English language requirements* on UniversityAdmissions website, [online], available: https://www.universityadmissions.se/en/entry-requirements/english-language-requirements/#meeting-the-english-requirement-with-an-english-test