Implementing a Knowledge & System transfer template in the Application Management team

Nguyen Nguyen

2023 Laurea

Laurea University of Applied Sciences

Implementing Knowledge & System transfer template in Application Management team

Nguyen Nguyen
Business Information Technology
Thesis
04, 2023

Laurea University of Applied Sciences **Abstract**

Business Information Technology

Bachelor of Business Administration

Nguyen Nguyen

Implementing Knowledge & System transfer template in Application Maintenance team

Year 2023 Number of pages 46

Working in a team has many advantages and disadvantages. On the one hand, a person is rewarded with the knowledge and experiences from being a part of a diverse, professional environment. On the other hand, it is a struggle to align with various ways of working, and if a common ground of understanding is not well established, people may find it difficult to keep up with the speed of collaboration and this may result in decreasing productivity. This thesis is dedicated to eliminating at least one common problem that a team within the client company, -Nordcloud, is facing. Specifically, the author explores how the Knowledge & System transfer process is being conducted in the team, and from there introduce a basic guideline under the form of a Jira template to help people perform that routine task efficiently and consensually.

Putting the Application Management team members at the core, with the focus on productivity, experiences and expectations, the goal of the thesis project was to listen to the developers telling their stories, sharing their experiences and future expectations. Once the facts are surfaced, the next step was to check if their team leaders are aligned with the practice of the squads. From forming the theoretical frameworks to recognizing the feasibility of the thesis's objectives, several approaches were used. The first approach utilized a qualitative research methodology and involved two actions: 1-on-1 interviews and a Continuous Improvement meeting. The second approach utilized via using collaboration tools with to the team. Here: Miro and Mentimeter were employed to pick out the most crucial reflection of the chosen topic.

The main findings indicate a lack of mutual understanding on the details discussed in a Knowledge & System transfer meeting, while the team leaders and their teammates held contrasting opinions on how the process should be implemented. Moreover, it is proven to be impossible to have a one-size-fit-all transfer template, due to the conflicting requirements of the projects.

Based on the results collected, a sample template was created to promote the general procedures when performing a Knowledge & System transfer process. Using the common collaboration tool Jira, the responsible parties of the process have a clear picture of what needs to be covered, who are the participants, timeline, and other associated factors. The templates then subjected to a testing period, to evaluate whether they have the potential to turn into a standard practice for the benefit of the team in the future.

Keywords: Knowledge, system, transfer, experiences, template, Jira, software maintenance

Table of Contents

1 Introduction

Working in a team brings along many advantages. You have the chance to work with many individuals from different backgrounds and expertise, share the amount of workload, solve the problems together, create potential for innovation, and enhance personal growth. Being in a happy team makes people feel more motivated and boost productivity, hence more contribution towards the company growth (Middleton 2022).

As a member of the Application Management team, the author gets the opportunity to learn from the strengths of multiple talented individuals, collaborate with them in delivering smooth operational management service to the customers, and boosting each other softs kills. We also share the same pain points, go through the similar struggles in our work life. We constantly make improvements because that is how we can become a healthy and well-functioning team. Therefore, when noticing that the Knowledge & System transfer process within the team is facing some obstacles, the author feels the urge to pick up the topic and start investigating on what has gone wrong, and how can we make the experience better for everyone. Furthermore, a practical solution is ultimately the final goal of the thesis. The solution may not be a perfect one, but it is essential that we acknowledge the problem and seek to replace it with a more efficient, applicable approach.

1.1 Problem identification

The core of this thesis is on implementing a practical template for Knowledge and System transfer process as a part of Continuous Improvement effort of the Application Management team, under a Jira ticket format. When a person leaves the team, the knowledge and system transfer sessions are taken place, which heavily depend on the memory of the main developers to decide what information to be shared with other team members. This creates a confusion because the context of each session is different, and sometimes important details are forgotten during the transfer. The new team members, on the other hand have no way to track if he or she has captured all necessary knowledge to start with the new project. In addition, the sessions are always taken place closely to the exit time of the developers, which is another constraint because it is not possible to share as much details as everyone wants, since it is not possible to squeeze in all the knowledge of the project within a few meetings. In other occasions, new contexts emerge after the transfer process, and it is too late to ask questions or go through the system again, because the people have left the team. This causes not only confusion for the everyone in the team, but also creates a negative image to our customers, for us having lack of transfer process practice in place. It is essential to remind people that an aligned understanding is needed about the steps included in the transfer

sessions and how to implement them sufficiently. This is to make the handover between people not only smoother but also boost the efficiency and transparency within the team.

1.2 Thesis outline

The thesis consists of seven chapters, starting with a short introduction about the topic and its objectives, following up with the second chapter presenting the commissioned company-Nordcloud and the background of the Application Management team where the author was working. The next chapter provided information on the theoretical frameworks in influence including the operational focus-Software maintenance, focusing on supporting and improving development productivities, experiences and how does it look comparing to the current set up of the team. The fourth chapter specifies which research methodologies were applied in this case, which is Qualitative research. Fifth chapter covers the steps to conduct the research, how the data is collected and analyze. The results are presented in the sixth chapter which leads to the proposal template. In chapter seven, the author concludes all the findings during the thesis process with recognition on the existing limitation and open suggestions.

2 Nordcloud -Application Management team

This chapter provides detailed information about the commissioned company and the background of the team where the author is currently working. In addition, the concept of teamwork will also be introduced and analyzed from different perspectives.

2.1 Nordcloud

The history of Nordcloud dated back in 2006, when a cloud-native and web application company is formed, but the name Nordcloud only came to life in 2011. Since then, the company has grown rapidly, first opening its own foreign office in Sweden, then entering the markets of Denmark, UK, Netherlands before the expanding in Poland. Throughout the operational years, not only Nordcloud got to scale up the existence in various locations with a skyrocket number of employees, but the company also got the pleasure to be called Partner of the years by Microsoft, received recognition by Gartner Magic Quadrant for Public Cloud Infrastructure Professionals and Managed Services (Nordcloud n.d.).

In 2020, IBM made an announcement to acquire Nordcloud to gain a deep foothold within the cloud-service industry. With this acquisition, IBM received unfathomable expertise that not only work for their current customers on the digital transformation paths, but also to target future advocacy of the hybrid cloud platforms (Shead 2020). Two years after joining IBM, Nordcloud now becomes Nordcloud, and IBM company with 1349 employees, spreading across

twelves countries, with headquarter in Helsinki, Finland. Having close to a hundred solutions and services provided, with multi-cloud platforms option, such as AWS, Azure, and Google Cloud, customers can rest assured that not only they are guided throughout the strategic shift into cloud, but also to see their business empowered and blossom within the fast pace of digital migration and modernization (Nordcloud n.d.).

2.2 AM Team

AM stands for Application Management, which is a team of nineteen people of different level of expertise. AM team members are locating in various countries, which strategically contributes to a diversity workplace and cultural aspects. The whole team is divided into four manageable squads, with their own number of accounts to attend to. The squads share some common practicality way of working, however, each squad is allowed the independency in defining what would be the most suitable process and comfort, to not only keeping up with the expectations of the customers, but also to ensure a smooth flow and transparent coordination within the team. As the AM team grows, the needs to define a common ground of the basic tasks should be taken into more consideration for establishing a clear understanding on what to be carried out in every situation not only within one specific squad, but to target all squads in AM. This practice allows the people to improve the process of collaborative work, guidelines are clear, and procedures are followed throughout.

As a member of AM, the author aims to learn for the alternatives, outside of the already recognized process. The purpose is not only to deliver the best performance towards our customers, but also to make the work itself meaningful for the members of the team. Teamwork in AM, like any other organization is an essential contribution towards the success of the company. It is not viable for a single person to perform all the work, nor should they. Having multiple people working together towards a common goal is essential, instead of dividing all the tasks and work independently boosts the effectiveness of a team. There are four main advantages of what teamwork can deliver. First, teamwork means motivation and inspiration. Instead of focusing on oneself, working together improves productivity, with milestones evidently reached, and challenges are more manageable to overcome. Second advantage is productive conflict management. Being in a team meaning that you must work with people from different backgrounds, diversities and naturally, different points of views. To reach the common goals, the team members must conquer these obstacles, both by challenging each other or compromise to reach an agreement. This means not only are problems solved, but the team conflict management skill is also evolved. Another benefit of teamwork is the meaningfulness of team development. Working together allows us to learn and understand each other, from their expertise skills to the individual's personalities. The stronger the connection there is, the more developed the team can be. The final benefit of teamwork is the one common thing there is: to reach the goals. In fact, the more effective

work a team can do, the bigger goals discovered. A successful team would not want to stay still for long, instead they would explore new goals where new challenges can be conquered and new skills to be built up (Waters 2022).

Although participating in a team brings many benefits, it does not guarantee that you would always have effective teamwork. Not all type of works or tooling fit into the team, nor having the same process, methods for a long period is ideal for teamwork. Every day there are many changes in the working life, which enforces us to keep evaluating the way we work within a team, whereas what used to be effective in teamwork today could still be valid in the future, should we eliminate some procedures and implement better, more productive one to adapt to the rapid changes around us, hence, bringing in the concept of Continuous improvement (Brealey 2017).

3 Theoretical frameworks

We are living in a world, where technology is advanced by the minutes. Everything we do has at least a bit of technology trace (Bulao 2023). As a member of the society, we must adapt to this growth by constantly learning and upskilling ourselves, and since we are working in the software development industry, it is essential to acknowledge the values and goals that we aim to reach to make our software product or service relevant and favorable to the customers. However, in this thesis, the customers are not the main audience here, but the developers are, because we are a part of the value chain that delivers to the customers. In this chapter, the author hopes to circle back and highlight the definition of Software Maintenance, to walk people back on the essence of the maintenance work, while at the same time assessing what we can do to lighten the pressure in our daily work. The goal is to help all team members collaborate easier, with more visual, well-defined, and time-saving instruction. This can be learnt better examining the advantages of a few recommended frameworks, in this case are the SPACE framework, Developer Experience Framework and Software Development Life Cycle methodology. The purpose is not to try forcing one framework into the team operation, but to choose which elements are the healthiest to implement.

3.1 Software Development Life Cycle

Software Development Life Cycle, or SDLC is a process of phases involved in any given software creation. SDLC consist of seven primary phases, which are: Planning, Analysis, Design, Implementation, Testing & Integration and Maintenance.

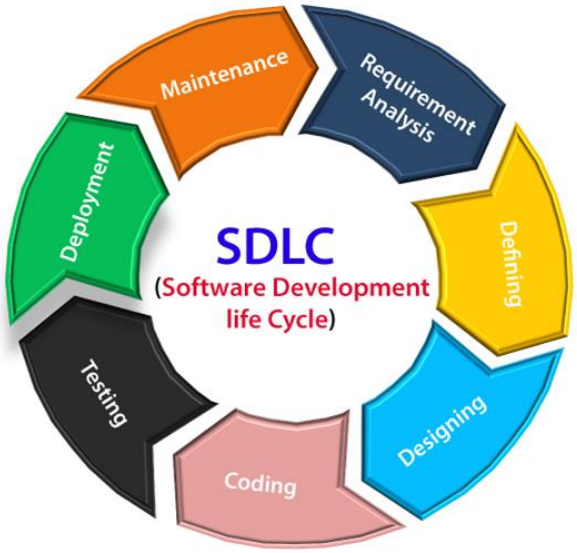

Figure 1: Software Development Life Cycle (Java Point n.d.)

Many software organizations are in favor of applying SDLC strategies into their business practices to create a customized guidance to all the teams throughout the stages of development (Velimirovic 2022). By providing specifications for each phase of creating and deploying software product, companies can assure that the software product is delivered within timely manner, budget is compiled with and viable future investment. Companies can choose their own SDLC methodologies that suits their needs, from Waterfall, V-Model, Big Bang to Agile, Spiral, DevOps and Iterative (Service Now n.d.).

Benjamin Franklin used to say, quote “If you fail to plan, you are planning to fail” (Kellar 2020). It means that whatever you do, having a plan in place is essential. Any project without a proper planning would lead to many failures, developers don’t know what features they should work on, managers cannot keep track of the project progress, and the organization fails to evaluate if the final products are bringing in any profit or not. That is why the Software Development Life Cycle methodology comes into the picture and removes the doubts. Each of the organization or project team has all the freedom to choose another methodology suiting their business, whereas the traditional Waterfall model, or Agile approach or a combination of a few, the benefits of SDLC is clearly visible: defining common guideline for each of the development phase, determined communication channels include all relevant parties, both internals and externals. In addition, the responsibilities between members of the projects, designers, business analysts are explicitly clarified. Expected outcomes and actual results can be monitored throughout in every step, and the completion of each phase is confirmed via the determination of Definition of Done (Swersky 2022).

Software maintenance is the final phase of the Software Development Life Cycle model. Once a software product is successfully launched, its life does not stop there. Instead, it should be

constantly observed and maintained to ensure the competitiveness and relevancy. It is defined as the process of “changing, modifying, and updating to keep up with customer needs. Software maintenance is done after the product has launched for several reasons including improving the software overall, correcting issues of bugs, to boost performance, and more” (Thales n.d.). Software maintenance carries the same importance as software development, in fact, it is stated by Robert Glass in his book “Facts and fallacies of software engineering” that 60% of the cost is upheld by maintenance, and solution enhancement is responsible for 60% of total software maintenance cost (Gadhavi 2022). However, based on Erlikh (2000), the highest estimation for maintenance phase expense can reach over 90 % of total lifecycle cost.

There are four types of software maintenance: Adaptative maintenance, Perfective maintenance, Corrective maintenance, and Preventative maintenance. The first type- Adaptative Maintenance aims at software adjustment to ensure the compatibility whenever there are changes in business requirement and technology transformation. Software framework is the focus of this maintenance, to allow the continuity and responsiveness with the new operating systems, hardware, and platform (Gadhavi 2022).

The next maintenance option is Perfective maintenance, which has an impact on all elements, functionalities while boosting the system operation and performance. The current software functionality will be adjusted to improve its receptiveness and usability. If the purpose is to fix existing errors, then Corrective maintenance is another suitable maintenance type to choose. In contrast of Perfective maintenance where a whole process is altered, Corrective maintenance job is fixing bugs or getting rid of any issues in the software. The maintenance effort can happen frequently, under the formation of small updates (Gadhavi 2022).

The last maintenance type in the list is Preventive Maintenance. Focusing on solving vulnerabilities, this kind of maintenance deliver improvement solutions towards the software to enable a defense for future use, while eliminating any alternation that may harm the product. In addition, Preventative maintenance can also support for scaling, maintaining, and handling the legacy systems (Gadhavi 2022).

Depending on the purpose and goals, an organization may enhance one or multiple maintenance types to their SDLC. Nevertheless, in general, software maintenance is essential for every single organization, as they carry many advantages, from reducing costs when new features and services added on, to well-preparation before any given problematic situations emerge, data security is strengthened to allow the business core is always in the focus (Radcliffe n.d.).

With the focus on software maintenance as the main service delivery, it is crucial for the Application Management team to understand what maintenance practices bring benefits to the team and investigate measures to improve the whole operation. From applying the suitable type of maintenance for different customer projects to select the right tools for collaboration of having a well-structured way of working is vital to the liveliness of the team, which resulting in time and cost saving and more business values to the customers.

3.2 SPACE framework

In this subchapter, the concept of SPACE Framework would be explored as one of the potential implementation for the future team practice. We first look at how the framework is defined, what are the main factors, its impacts on teamwork and then dive into how the framework implementation conducted in real life.

3.2.1 Definition

The SPACE framework is a research-based developed by GitHub, Microsoft, and the University of Victoria (Canada) researchers, focusing on measuring, understanding, and improving developer productivity, which "encourages engineering leaders to have a holistic approach to productivity". The framework has five dimensions: Satisfaction & Well-being, Performance, Activity, Communication & Collaboration, Efficient & Flow (Gralha 2022).

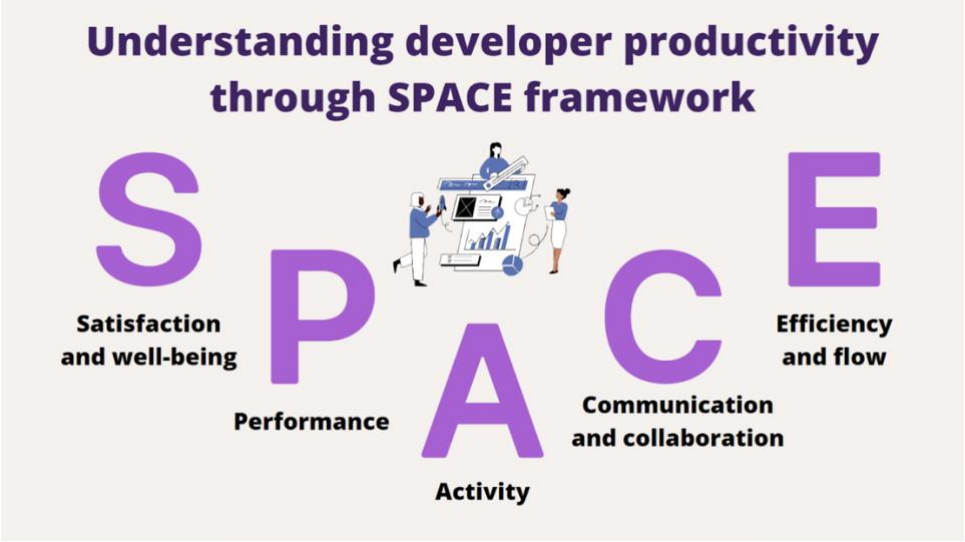

Figure 2: SPACE Framework (Cortex, No date)

The first dimension, Satisfaction & Well-being (S) assess the satisfaction, happiness, and positive habits of the people in the team via employee surveys. It is believed that future productivity can be predicted through the crucial outcome results, and if no improvements are done, it would lead to dissatisfaction and burnout of the developers. One way to learn

about how people feel is via interviews, polls, and surveys as a qualitative method. People who take pride in their work, feel happy with their daily tasks, and aim for the best results are the one who prefer to stay with the same company rather than switching jobs, and that company can benefit from the contribution of these personnel's innovation (Simic n.d.).

Performance (P) is the second dimension, which focuses on the team or workflow efficiency. Via the evaluation of performance measurements, the team leaders can draw the connection between people's action and the produced outcomes. The evaluation can be done by using dashboards to track a couple of metrics such as code reviews approval, code quality and static code analysis because the health of a service depends on the developer's coding skills. The goal is not only to understand how well the team is working, but it also creates an impact on other aspects, for example customer satisfaction, product acceptance, cost management and so (Circei 2022).

The next focus is on Activity (A). This dimension concentrates on the developers' outputs, with straightforward measurements such as work items, pull requests, deployment frequency and more (Gralha 2022). Communication & Collaboration (C) measures the efficiency of how transparent and clear the communication within a team is. By using the metrics such as Documentation quality, Work integration speed, Work contribution quality, Network metric etc. the developers can better align when setting priorities, fit their work efforts into a bigger initiatives and learn from each other (Nussbaum 2022).

Efficiency (E) is the last dimension, which is in relations to all dimensions of SPACE framework. It catches "the ability to complete work or make progress on it with minimal interruptions or delays, whether individually or through a system." (Forgren et al. 2021). According to Koponen (2022), a few common metrics used to calculate Efficiency include Team health checks, Workflow observation and completion, Investment balance, Code commitment and Restoration time.

The SPACE framework allows both the developers, and the team leaders to understand the developer productivity from a more holistic angle, meaning that the productivity level is not done by any individual's effort, but it is from the whole collaboration of everyone in the team. Using the suitable metrics and insights allows everyone to address the key points which leads to fruitful engineering and top-notch values delivered to the customers (Pedro 2022).

3.2.2 Team practice

Since the start, the developers of the team are considered the main objectives, therefore when selecting a suitable framework for further study, the author focused on what are the most important elements to clarify. It was not about the final products or services that the customers received, but the process that led to that outcome. The questions that were being

repeated all the time were usually about how well they were doing, what were the obstacles they were facing and what could we do to make their work easier and more productive. In short, the goal is to find out what could we do to make our development team happier.

In that sense, the SPACE Framework fits quite well into the picture. The framework reminds us that we should explore the meaning of development productivity in a broader scope with holistic manner, not by the number of codes lines or fixed bugs, or how many tickets have been completed. The framework acts more as a guidance on how to select the appropriate metrics to evaluate the developer's productivity and fill in the tension gap between the management team and development team. If the causes that affect how the developers perform could be visualized better, the more assistance their team leaders can provide to ensure a more productive, purposeful environment (Ayanleke 2022).

There are a few methods chosen within the team to approach this concept, depending on the team level. As the Application Management team is divided into four squads, each squad decides how to pursue this concern differently. The most common practice is a monthly retrospective, where all squad members could discuss about their work history, with feedbacks spit into four categories:

1. What has been done well?
2. What should we stop doing?
3. What more should we do?
4. Future action points

By sticking with these categories, all squad members have the same chance to reflect their personal work performance and use the meeting as a channel to communicate with the rest of the team. They could contribute to defining a better mean of work, or simply share how they struggled with specific tasks. This allows everyone to recognize what aspects are essential to remember and be able to reach out to others for help or raise their concerns. The collected feedbacks and opinions could be put under further evaluation, and no matter what action is chosen, the only goal is to improve the working environment and enhance productivity.

3.3 Developer Experience framework

Developer Experience framework, or DX is the framework subjected for further study here. In this part, the goals are understanding the concept of DX, its deliverable values and effectiveness in practice.

3.3.1 Definition

Developer Experience is a comparable narrative of User Experience, but instead of the customers, the developers are the one who use the products (Andrzejewski 2022). A good DX

framework allows the developers to feel happier, encourage them to proudly promote and keep up with the products that are useful for them (Cavalcante 2019) by putting yourselves into the developers' shoes, searching for ways to improve their working methods. An exemplary DX means that focus time is supported, all information is documented properly, ramp-up time reducing, and soft skills are paid attention to (Tsuei n.d.).

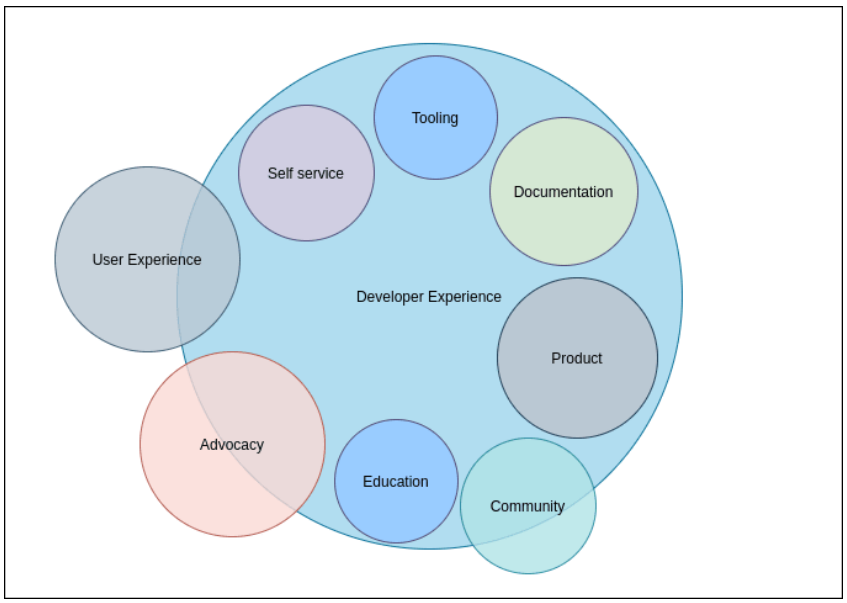

Figure 3: Developer Experience (Sasidharan 2021)

Knowing the importance of DX is one thing, executing the plan is another. It is not up to the developers to conduct the implementation themselves, but their leaders or managers should be the one who drive that process and prove that they have the full understanding on what leads to the productivity of the developers, what make them happy and ensure them that all their needs are met to produce a quality software product (Hrzenjak n.d.).

There are different methods to measure Developer Experience. The first group of pillars includes findability, usability, and credibility. Findability means how easy it is for the developers to find the information, tools, systems needed for their work, which can be achieved by delivering contextual, structure, functional and organized documentation, together with unlimited access to essential tools. Usability focuses more on how comfortable the developers are when using codes, documentation, and tools. The last pillar – Credibility means how much trust the developers have on the products and their benefits (Towns n.d.).

Cavalcante (2020) recommends a similar list of pillars, with the add on of Accessibility. A few metrics to use for measurement including from guerilla test, lab test, in-person/remote interview for measuring Usability, Journey map for Accessibility. Asking the question of how much time it takes to complete a task is accounted for Findability. As for Credibility, it is suggested to measure bugs that occur in the product and their critical levels.

Another set of metrics can be used to measure developer productivity including Cycle time, Deployment frequency, Merge frequency, Investment profile, and Planning Accuracy. Cycle time means the time from when the work starts until it is completed. The shorter the cycle time is, the better performance the team has. Deployment frequency tracks the regularity of codes deployed to production environment. Frequent deployments allow higher quality products reach to the customers. Merge frequency measures the number of pull requests during a period. The number of code lines and pull requests creation do not bring any value if nothing is to be merged. The next metric is Investment profile, which calculates how much time allocated to different type of work. By applying this metric properly, the team would have a clear understanding if they were aligned with the business requirements or else. The last metric- Planning accuracy allows the developers to know how much issues or story points have been finished, comparing to the planned iteration. It helps to recognize any hidden unplanned and deferred work, and from then the team's scope could be better adjusted (Pauly 2022).

Although there are different approaches on what DX measurements to implement, they all share the same goal of improvement developers' working life. Learning what truly affects them allows the managers/team leaders to fill in the gap between what you think they need and what they really want (Pluralsight 2022).

3.3.2 Team practice

The term of Developer Experience is completely new to the team, and for the author, it was the first-time hearing about the framework. As a consulting company, the customers in some way receive more attention than the developers, in fact there have been plenty discussions and guidance about how to improve Customer Experience within the company, but not the other way around. When looking into the framework model, there are a few similarities in focus points in the team's operation that could be enhanced further. Since there is no one-size-fit-all structure on how to implement this framework in practice, we need to see if there is a match between the recommended pillars and the reality of the team. The first pillar to be considered is Findability- how challenging it is for the team members to find the useful information; do they have enough tools to perform their task or how well the details are defined. Usability is the next term that comes to mind. Are we all happy with the context that we have so far? Is the tooling working well for our needs? Do we feel struggle with any part of our job? Moreover, do we have enough trust with the current practice that we are exercising or is it more doubts that we cast- which is what the last pillar Credibility represents.

By following the proposals of this framework and place the attention into “Experience”, the author wishes to get to understand the team members more conclusively and turn their experiences into a more positive, efficient path.

3.4 Continuous Improvement

Continuous improvement is defined as “an ongoing effort to improve all elements of an organization-processes, tools, products, services, etc.” The size of these improvements can vary, but they all meet at one point of being frequent (Dewar et al. 2019).

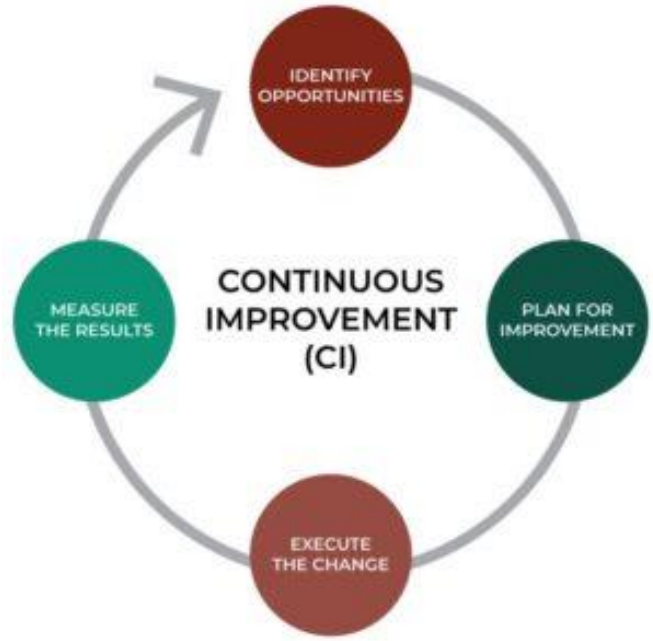

Figure 4: Continuous Improvement (CI) (VMEC 2018)

The Continuous Improvement (CI) consists of four main elements, which are Identify Opportunities, Plan for Improvement, Execute the Plan, Measure Results. In addition, there are few key terminologies that useful to remember, for example Value added, meaning the activities which would boost the functionality of the product or service. Non-value added are the contrast activity which should be review for reducing or eliminating, simplifying. Continuous flow focuses on the course of information and materials without being hindered (VMEC 2018).

The Continuous Improvement approach is for all organizations, regardless of size, area of trades and geographical aspects. Everyone can benefit from applying continuous improvement methodology for reduce the risks in operational work, higher level of employees’ engagement, customer satisfaction while being cost effective and stay competitive in their market to overcome all obstacles given by customers. This approach can come to aid in

various ways, including quality improvement, processing time reduction, boosted morale, excessive people engagement, decreased employee turnover and intensive professional growth. To be able to gain these benefits, there are a few elements that one must remember when introducing this concept to the team or organization, with the first point is to align and clear vision set throughout the organization. While choosing the metrics is important, it is even more essential to ensure that the right measurement is in place. Secondly, there is no need to build up a large measurement but instead staring with a small approach. Running a small test, finding the individuals who are willing to participate in the trial and observe the change over time. The method is there to assist, not an asset to win, therefore it is important to remind people that failing is acceptable, as it is a learning opportunity. The number of failures is not viable, but the lessons come out of them. Finally, as said many times, goal setting is the key element. Having goals set clearly is essential as it determine the success of the organization (VMEC 2018).

The concept of Continuous Improvement is not a stranger to the Application Management team, instead, the team has organized a bi-weekly session, with different topics for each session to brush up the current knowledges of team members. There is one main facilitator, and individuals can suggest which topics they want to elaborate further, with no limitation from general operation standard to introducing new tooling, experience sharing, technical workshop and updates between the team with other divisions of the company.

The average length of the session is one and a half hour, depending on the number of issues to be discussed. The sessions are held at the same time every month to reduce complexity. Beside the core nineteen members of the team, the facilitator also invites a few more colleagues outside of the team, either acting as the main speakers or assisting in further clarification of the topics. The attendees are always encouraged to participate to ensure that the knowledge is spread evenly. In case of absence, the team members can research the shared materials or watch the recorded sessions to keep them up to date with the contexts.

3.4.1 Knowledge transfer

Knowledge transfer is defined as a “systematic and purposeful strategy for capturing critical knowledge from key personnel to store and share within an organization for maximum efficiency”. It is typically carried out when a person decides to leave the team and instead of letting all the information faded away, the rest of the organization must trigger multiple meetings to capture as much knowledge as possible before the departure of the personnel. Nevertheless, knowledge transfer process can go beyond its original attention by turning into a developed strategy to have enough knowledge goes to the people who need it. This allows the organization to create a foundation of teams with people who are capable of growing and developing despite of personnel change (Maestro 2020).

To approach the concept of knowledge transfer in a simpler, more visual way is to look at the Triangle of Wisdom. The triangle consists of four levels: Data, Information, Knowledge, and Wisdom. The goal is to move up from Data into Wisdom to gain tactic knowledge which contribute greatly to the strategy of planning, coaching, and mentoring (Carruthers 2021).

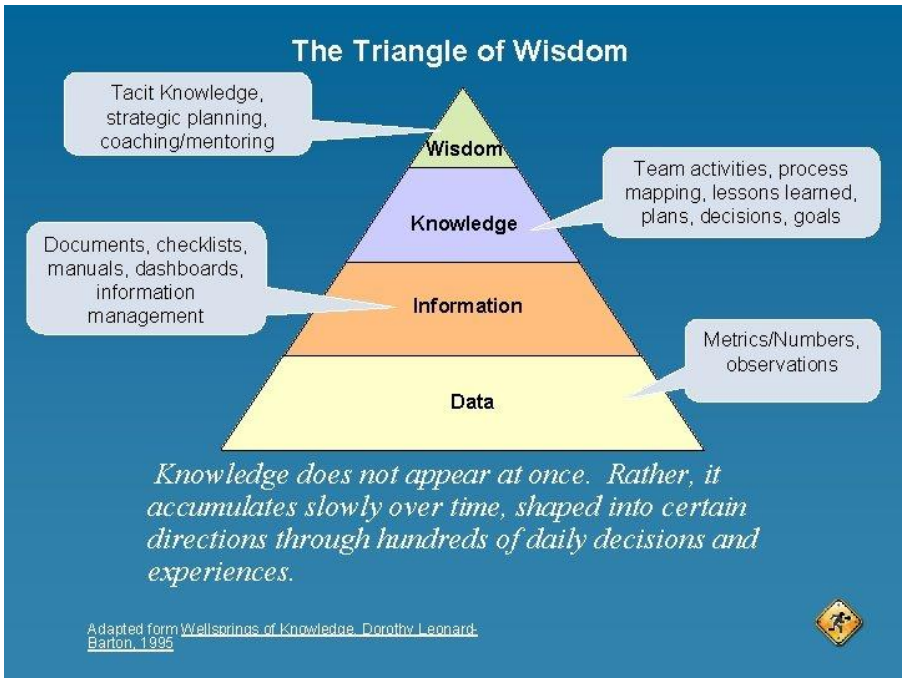

Figure 5: The Triangle of Wisdom (Carruthers 2021)

Knowledge transferring can sometimes be referred at knowledge sharing, while these two concepts are very much alike, they are still incompatible. Knowledge sharing has a limited focus, with the goal is to stimulate the innovation and enhance cross-teams collaboration by passing the insights and ideas to other members. Knowledge transferring, on the other hand focuses on succession planning, building the bridge of information between the experienced members and the new joiners or people who would take on the new roles within the organization (Carruthers 2021).

When an organization recognizes the importance of Knowledge transferring process, the next step is to define how to execute it effectively. The knowledge is kept in the people's mind, which can be shared by telling, showing, or writing. Depend on the senders and receivers, it is recommended to exercise different approaches and tools in the five steps of Knowledge transfer process: Identify & Collect Knowledge, Capture & Store Knowledge, Transfer & Share Knowledge, Apply Knowledge & Show results, Create New Knowledge (Brown 2021).

Knowledge transfer is a task that has been performed many times by members of the team and for various reasons, from when a person exits the team or when a new member onboards to the project. If it is a standard process, then the expectation that everyone should be

familiar with which topics to be covered, how much time should we spend on Knowledge transfer and where they can find all the necessary documentation. Unfortunately, this is not the case. While everyone is aware of a few elements of the transfer process, many other aspects are still be left out, resulting in documents are outdated, important talking points are forgotten, team leaders 'supports not arrive at the ideal time and the team members are lost on what and how to proceed further. If there is a clearer instruction, better-defined formula to attend to every step of performing the Knowledge transfer progress, the less confusion and pressure brought to the team.

3.4.2 System transfer

System transfer is a term that regularly used by the Application Management team, besides the concept of Knowledge transfer. In other situations, people may even prefer using the name Project handover to describe the change in application ownership. Instead of assigning the customers who uses the products as the new owner of the system, at least a new developer would take over that responsibility, as we call it primary developer, and in other cases new project manager would join force too. Regardless the reason of the transfer, how it is executed is much more important, because it determines whereas the project would continue to thrive or turn into failure for whoever takes over (Gavrilovska 2021).

Wójcik (2023) introduces a list of seven points need to be checked when performing a handover between the old and new teams, which are Code audit, Detailed documentation, Knowledge sharing sessions, Establishing communication standards, Gradual transition, and Progress monitoring. When conducting Code audit, there are a couple of options to choose from, either continue working on the existing codes, or conveying the issues by refactoring. If these are not possible, re-writing the codes from scratch should be considered. Knowledge sharing is essentially included as a part of the process, which can be done by pair programming, team code reviews and even hackathons (Azorin n.d.). A detailed, updated documentation allows the new developers to get the full understanding of the system's features and functionality, with the focus on two key points: process and product documentation. Process documentation targets testing standards, plans, timelines, and meeting notes, while product documentation answers questions such as product requirements, product architecture, functions' explanation, release notes and more (Vasin 2023).

When conducting the collaboration and communication effort, taking into consideration what is the best approach. The old and new developers can meet face to face if the situation allows. Otherwise, there are a few tool choices for collaboration such as Jira, Teams, Zoom, Slack to help connecting with people from different location and time zone (Dziuba 2021). Choosing the right tools ensure that everyone in the team can participate effectively from

anywhere in the world and balancing the awareness on the tasks, decisions, and resources (Needle 2022). As for the Application Team, Slack is the main communication and collaboration tool, which have been working sufficiently in exchanging conversations and cooperation on multiple projects.

When the time comes for the team or team members to start working on the tasks, they may face certain difficulties, therefore it is important to decide which tasks they should start with first to allow the well-structured transition, for example maintenance work could be an ideal option, to get acquainted with the code base. When they get a good sense from the source codes, they can move on to more complicated tasks and proceed deeper with the growth of the projects. Moreover, the old developers need to keep monitoring the progress, to avoid any hidden hassles and help steering the people into the right direction, not by having an exhaustive dominance but by being in touch with the team and answering all issues called for (Wójcik 2023).

In the AM team case, the objective is not to understand about Knowledge on a phenomenal level, but to break down the concept and investigate what are considered as Knowledge from the developers' perspective, based on the practical work that they perform to make the decision on what to be transferred to the new owners of that knowledge. For example, as a customer-focused company, the customer engagement is very important. What we consider as knowledge when it comes to this element is who are the stakeholders, our mutual way of working, communication channels are all parts of customer relationship. From a developer's perspective, knowledge is about the system's domain, for example industrial business, utility operation, retails, together with the architecture behind them, the development phases or the bugs that have been solved. This is the knowledge that matters the most to the team.

4 Research Methodologies

As Albert Szent-Gyogryi, the Nobel-winner in philosophy used to say, quote "Research is to see what everyone else has seen, and to think what nobody else has thought" (1957), it is essential to conduct in-depth analysis on the problem and try to get a useful result out of it. Doing the research is not even close to a no-brainer, but in fact it is quite challenging and laborious. However, conducting research is also a rewarding outcomes, where you can put your skills and expertise in place when pursuing deeply into the interested topic (Reddy n.d.).

In this chapter, we will focus on implementing qualitative research methodology, by applying to kinds of methods: 1-on-1 interviews and surveys.

4.1 Qualitative research

When searching for the best approach, the thesis author recognized that questions here to be asked were not relatively close to numerical perspective, but rather seeking the answers for “how” and “why” to capture a thoughtful understanding of experiences, context, and phenomena (Cleland 2017, 61-71). In fact, because the author was interested in learning from a small group of colleagues, to fathom whereas our way of working was still functional, or should we seek for another alternatives. In addition, the author wished to apply earned experiences and observation into practice, therefore qualitative research is the most applicable choice for that goal.

4.2 Data collection method

The first method chosen is 1-on-1 interview. Due to natural course of the environment, a semi-structured interview style was followed. While having an underlined purpose of what to ask, there was no order to the questions to allow a more flexible, pleasant atmosphere for the interviewees. In addition, it helped to prevent any accidents in asking leading questions and assisted in exploring the topic in interest candidly (George 2022). The interviews were set up as a conversational event, with the meetings organized through selected options of communication and no set limit in interview length between the author and the interviewees. A total of 4 people were participated in the interview process in a period of two weeks.

Focus group is the second method that fits into the scenario. Reason for choosing this methodology is because it allows the author to explore whereas if the proposed idea has any link to the practicality. With multiple criteria in place, this method supports further clarification, expansion of data and getting feedbacks reported back to the research participants (Gill et all. 2008, 291-295). Since the target audience group size is not large, it suits ideally, allowing everyone a chance to share their thoughts, while eliminating any chaos or frustration that may come from a much larger group of participants. In this case, the focus group includes all nineteen developers of AM team, four squad leads and two team managers.

In addition to the two mentioned methods, the author also applies personal experiences in use, by participating in the transfer meetings as a mean for collecting more materials for further evaluation.

4.3 Data analysis method

The data analysis is done by combining two methodologies: Narrative analysis and Interpretive Phenomenological Analysis (IPA). Narrative analysis is the kind of story telling tactic, and by listening to people sharing their points of view on specific contexts, we can have a clearer acknowledgement of how the stories really appear, comparing to how we thought or told it

should be. The goal of applying this type of analysis is to gain powerful insights from the people's mindsets and to paint a clear picture of their perspectives (Warren 2020).

Since all the team members would encounter a Knowledge & System transfer at least once during their working time with Nordcloud, the author put the focus on studying from people's experiences and expectations in what elements they feel are important. Drawing a Transfer guideline template by involving Interpretive Phenomenological Analysis into the process is essential. IPA is an experimental qualitative research method developed in 1996 by Jonathan Smith, with its main objective is to gain insights of the experiences through the people who promptly encounter them (Rodrigruez 2022). By applying the practice of IPA method, the author seeks to intercept how the Knowledge & System transfer is conducted from the developers' points of view and makes meaning of the experiences they have gone through. Finding the answers for What are their stories are the goals here.

5 Conducting Research

At the early stage of conducting the research, there was no interviews nor questionnaires designed yet. At the time when the thesis topic was initially emerged, there were a few situations in the team, including multiple small teams, in which developers left their positions and must transfer the system ownership to the next person so at least one knowledge and system transfer meeting was facilitated. The author got the chance to participate, observe the transfer meetings and collecting the steps on how the meetings were organized and what details were discussed. After that, the next step was to have the discussions with the team leading to clarify how each team proceeded with the transfer or handover without any inputs from the authors. After completing these steps, the topic was introduced to the rest of the developer team with a sample template revealed, using Google Meet as the core communication tool.

5.1 Research setup

As mentioned, the research phase was divided into three parts. The first part was about gathering information on how the current transfer sessions were set up based on personal experience and team situations. The second part was focusing more on 1-on-1 interviews between the thesis author and different team leads to capture deeper insights from their own experiences and mindsets. The next step was to design multiple questionnaires, in addition to drafting a sampling template based on the data collected. After completion, the idea was presented using the Continuous Improvement meeting to meet up with all team members, where the author could get deeper insights on how people perform the transfer in practice,

what has worked well for them and what changes did they wish to apply. Moreover, the sampling template was disclosed for further discussion and feedbacks.

5.2 Data collecting

The researching data was collected in different phases. During the first phase, the author participated in the internal knowledge and system transfer between the teams, together with the current and replaced developers. The meetings were facilitated using Google Meet, with time span from between 45 minutes to 60 minutes per session. All the talking points including the transfer process, number of participants, length of sessions, list of materials and common practices aspects discussed during the meetings were recorded using Miro board template's Sticky Notes for visual and tracking purpose.

Figure 6: Sticky Note (Miro Board)

Once the meetings were completed, the author continued with the interview process with the team leaders. The notes taken during the first phase of research were used as questionnaires during the interviews. All the interviews were conducted during the period of two weeks. Again, the answers were logged in using Miro, and besides the four team leads interviewed, the author also wrote down personal observation for researching purpose. At this stage, there were two groups of contents, one belonged to the developer group, the other was from the team lead group, which led to the next phase of transferring the common pain points into Idea Napkin framework to for ideas comparison and assessments.

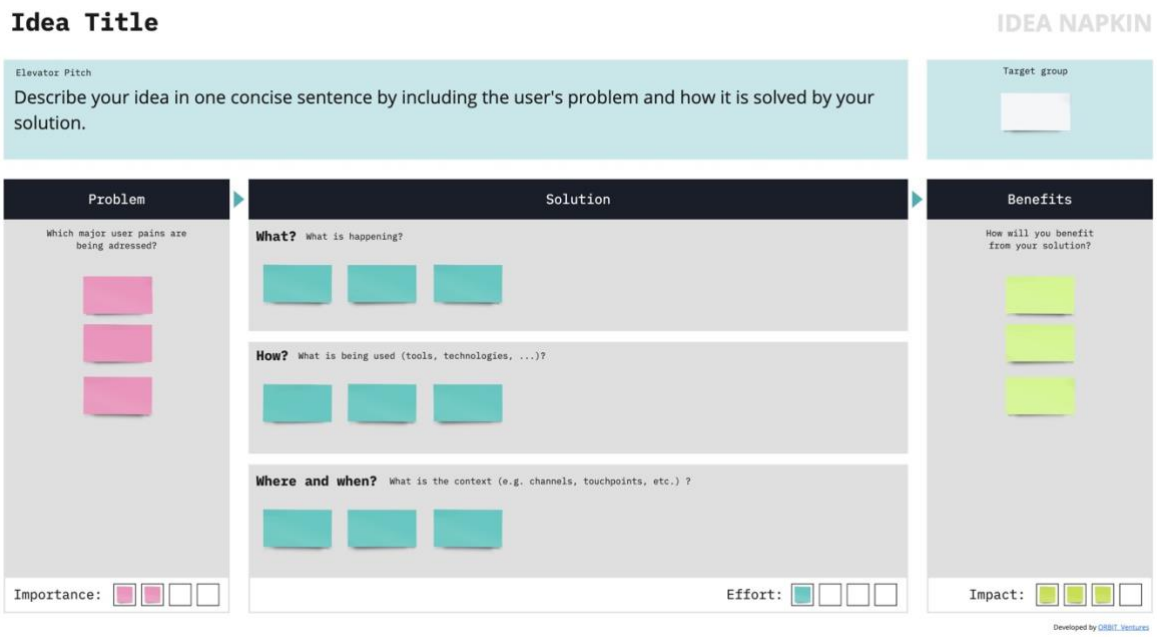

Figure 7: The Idea Napkin (Miro Board)

With the details gathered from the first two phases, a draft template was designed, using Jira tool, and put into testing. All the developers involved in the transfers and their team leads were introduced and encouraged to track the process of the tickets. The testing phase was launched from mid-March and still ongoing.

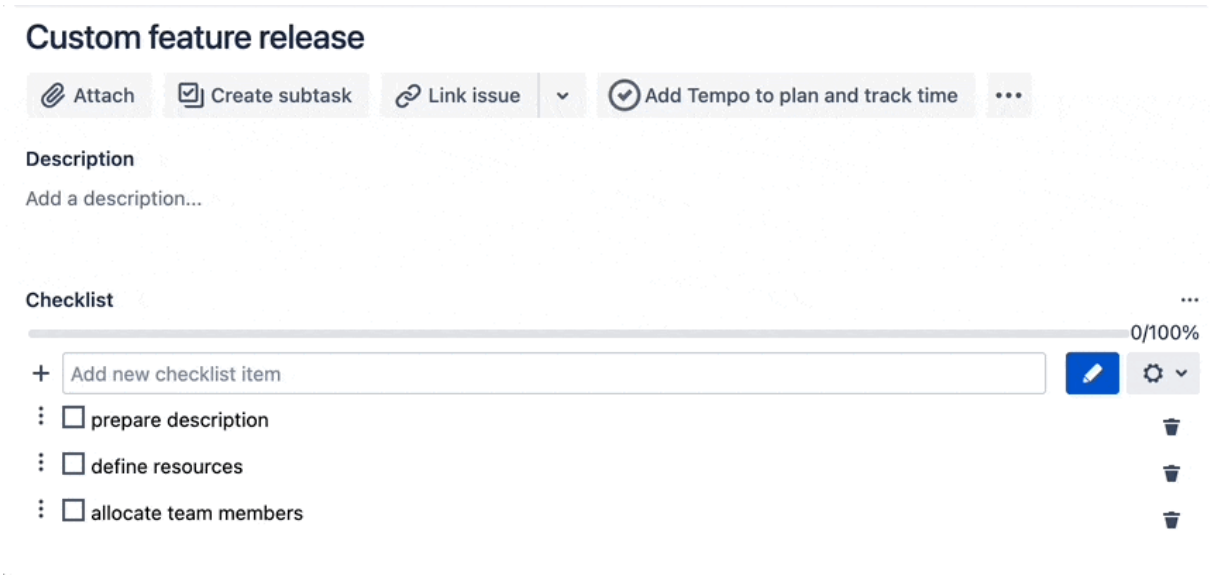

Figure 8: Jira template (Atlassian Community)

The next step was to present the ideas during the Continuous Improvement meeting on 31.03.2023, with the participation of nine developers and three team leads. A questionnaire

with eight multichoice questions and two open questions was prepared using Mentimeter-a tool for presentation and immediate feedbacks (Edward 2021) for collecting further opinions, and the whole presentation was performed within thirty minutes.

As the data was collected via multiple occasions, the last step to perform was to continue fine-tune the knowledge/system transfer template and turned it into a ready-to-use version, covering all the basic elements and pain points.

5.3 Data analysis

During the first phase of collecting data, the author joined the knowledge/system transfer meetings as an active participant and facilitator, with the real intention hidden, addition to staying in the shadow, meaning not providing any personal opinions or interfering into the discussions. The focus point at this time was to write down what was discussed, what was considered as important information to handover. The atmosphere and overall impression of the participants were also observed carefully.

The interviews set with the team leads on the other hand had more focuses on how the Knowledge and System transfer was conducted in squad level specifically, to learn to which extent the people understood about the steps included, the materials required, relevant parties, and other practicalities matters. Together with personal experiences as one of the team lead in Application Team, the author began to anticipate in the process of evaluating of what needed to be included in the draft sample Jira template, what tasks or talking points the developers should include within the Knowledge/System transfer meetings, and whereas the template would be practical and beneficial for the team. At least one person said that it was not possible to define how many meetings should be held, while the rest of the interviewees thought 2-4 meetings would be efficient. The ideal meeting length would be about 1-1,5 hours during morning time.

6 Research Result

In this chapter, we discussed the conclusion of the interviews and CI meeting processes. In addition, the questionnaire's results were presented to understand the developers' experiences and what would they expect in the future.

6.1 Interview results

All the interviews were held with the squad leads of the Application Management team. During the meetings, the thesis's topic was presented to the interviewees and then proceeded with a list of questionnaires. All the interviews were conducted online and although there was

a timebox, the interviewees were assured that the length of the meeting was flexible, which allowed them to feel comfortable and comprehensively focused on the details in that meeting. During the first meeting, seven questions were asked then gradually increased into nine questions in total during the other three meetings. The average length was 31,75 minutes with the short interview lasted for 21 minutes, and the longest one lasted for 45 minutes. All the interviews were recorded and later reviewed, with results added as Sticky Note using Miro.

The first question was asked if the concept of Knowledge transfer and System transfer could be considered as one, which returned in contrast responses. One person said that they never notice about the differences between the two factors, while another said that they understood Knowledge transfer process well, but System transfer was a completely strange concept. The third person was unsure about what Knowledge transfer was for, and in fact in their opinion knowledge could not be transferred, but rather the information. The last interviewee shared that they considered that those two concepts were somewhat related but could still be standing alone because they served different purposes.

The next focus was on learning about what should be included in the Knowledge transfer and System transfer. To simplify the question, the interviewees were encouraged to combine the two concepts at once, and shared all the topics they considered as important to explore during the transfer. All the interviewees shared the mutual agreement that Runbooks, Customer engagement rule, System domains were the one that needed to be walked through. In addition, Business domains, System Architecture, SoW, Code bases were essential to discussed too. Three out of four interviewees praised the concern that during the transfer process, we should spend time checking all the open tickets, reviewing common bugs and blockers, while one interviewee focused on what were the business values, integration set up, pair programming or shadowing organized between the developers.

Once the transfer process's contexts were clarified, it was vital to understand how many of those meetings that required, how long should each of them last, the core participants and whereas recording was necessary. At least one person thought that there should not be any limitation to the number of transfer meetings while the other three agreed that 2-4 meetings should be efficient. The meeting length was recommended to be about 1 hour during the morning time and recording the meetings was necessary or even made it as a rule. The core participants were the old and new pair of primary and secondary developers, and while everyone was welcome to participate, the priority should always go to the focus group because of budget constraint. The primary developers were always the people held accountable for the transfer progress, while the team lead's role was more on the assisting side. The transfer meetings were expected to occur when a person onboarded to or exited from the team, or during project rotation. Most of the people agreed that the meetings

should happen as early as possible and more frequently, while one person thought that we did not have an urgency for having the transfer meetings.

When being asked what the other suggestions were, they wanted their team members to acknowledge, the team leads shared that people should have the understanding that a good Knowledge transfer meant good reputation in front of the customers, everyone should know what contexts needed to be transferred, the goals were to be defined clearly, and transparent communication should be kept. Furthermore, each system was unique in their own, so avoiding using the same procedure for all and paid attention to the small details, because those could be easily forgotten.

The last interview question concentrated more on the team lead themselves, to understand their points of view toward their own team. Most of them agreed that they have full trust on their team members, and there was no necessity to take full control of the transfer progress, as people could be responsible for the tasks in consideration of them being fully aware of the execution.

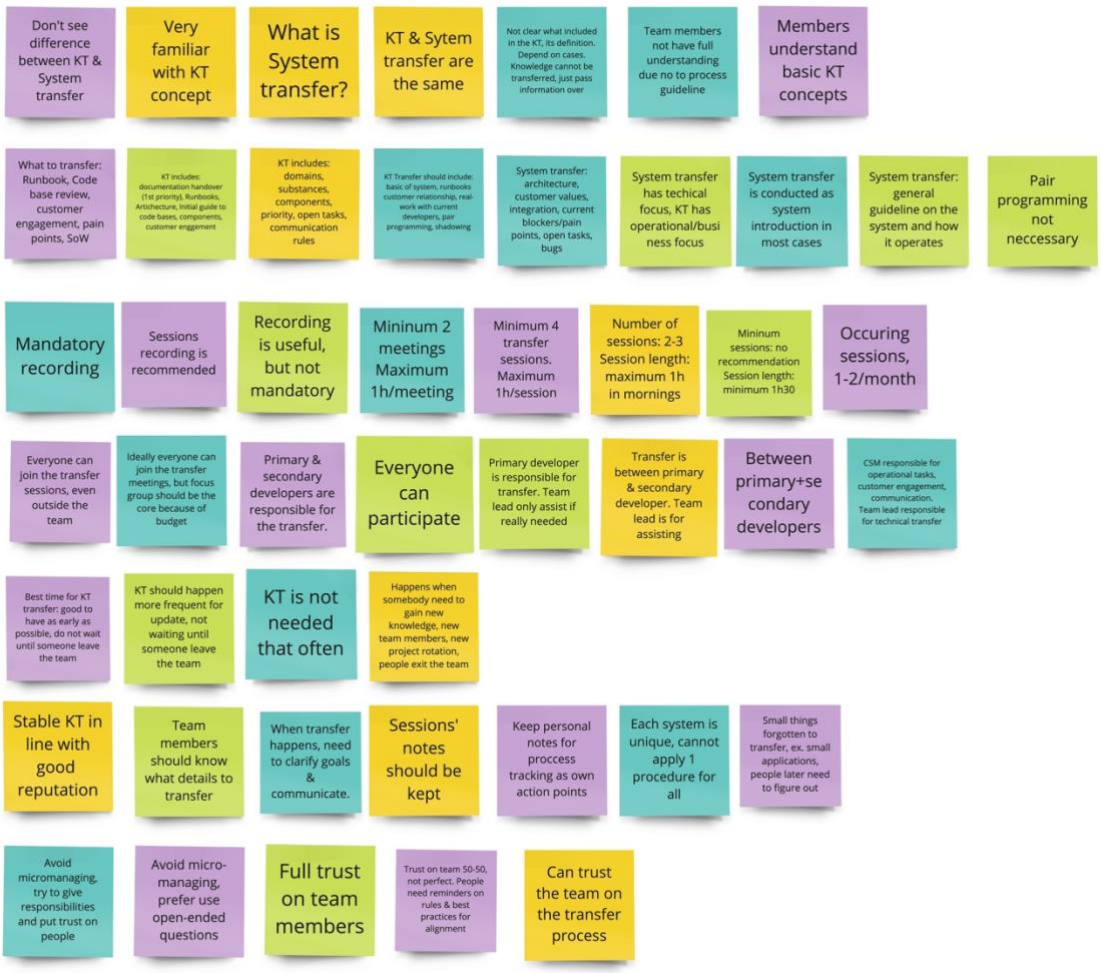

Figure 9: Interview results (Miro Board)

6.2 CI meeting results

During the Continuous Improvement meeting, there were eight questions presented and their responses collected for evaluation, including nine developers and three team leads. At first, we started with the Knowledge and System transfer concept whereas they would see any differences between them.

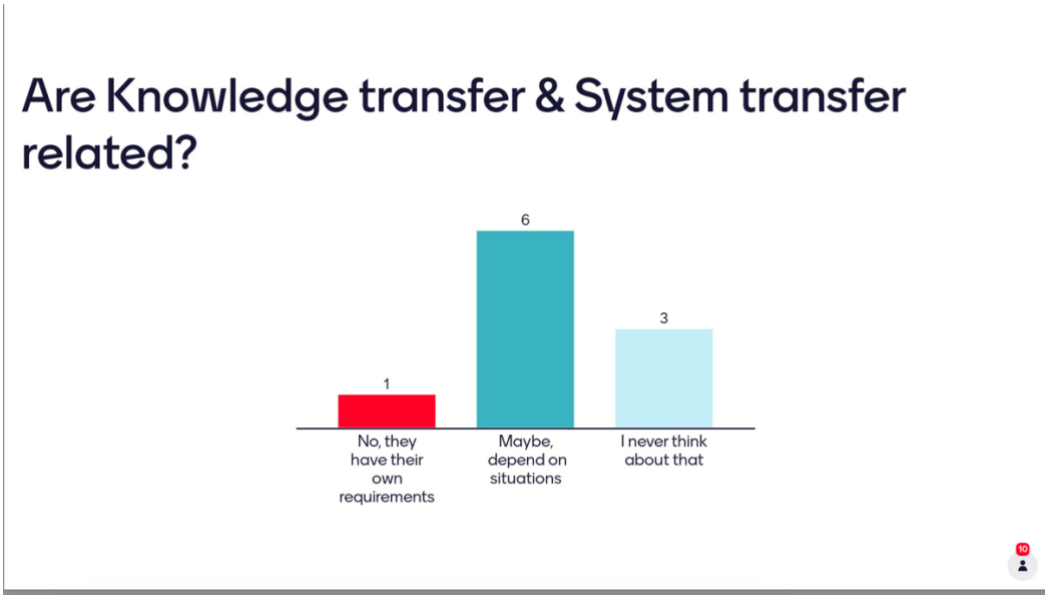

Figure 10: Results from the questionnaire during Continuous Improvement meeting (Mentimeter)

We received 10 answers, with 6 out of 10 said that the concepts are somewhere relatable in some contexts, 1 person argued that they should not be considered the same because of different requirements, while the rest shared that they never put much thought on the subject.

Next question was about the time to conduct the Knowledge transfer, with 9 answers in total. No one agreed that Knowledge transfer should occur when a person exited the team, while 2 people said that it should happen when onboarding members. The majority emphasized that Knowledge transfer should happen any given time, regardless the reason.

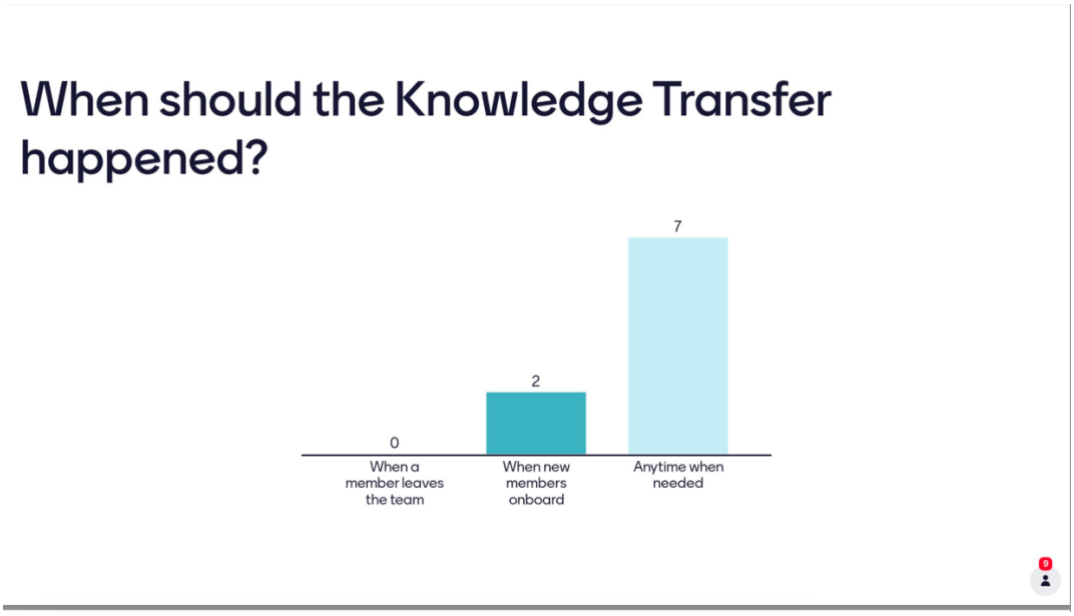

Figure 11: Results from the questionnaire during Continuous Improvement meeting (Mentimeter)

Since everyone in the team had at least once participated in a transfer meeting, they were asked how many sessions they would need, and for how long those meetings should last before them losing their focus, the responses were received as followed:

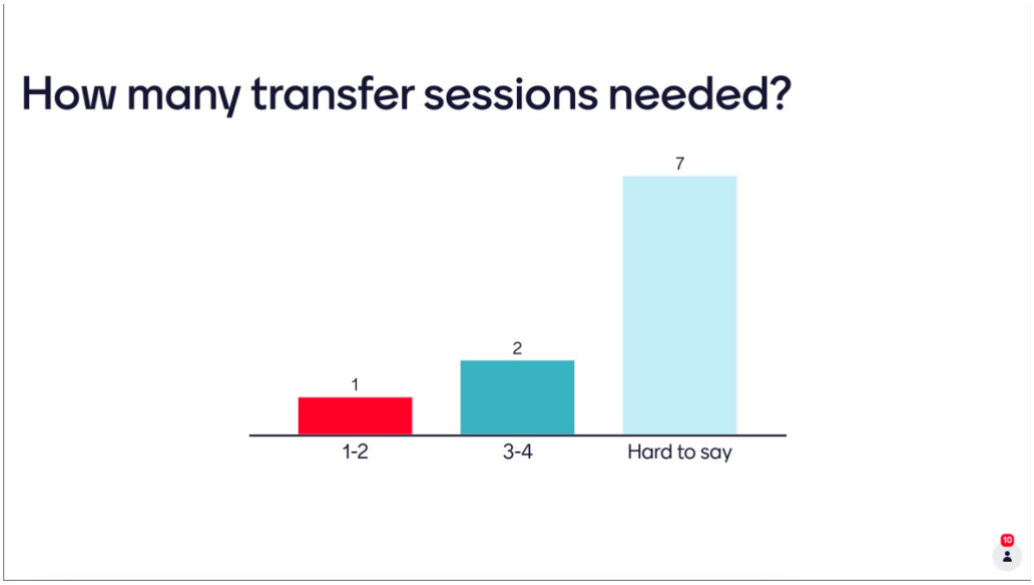

Figure 12: Results from the questionnaire during Continuous Improvement meeting (Mentimeter)

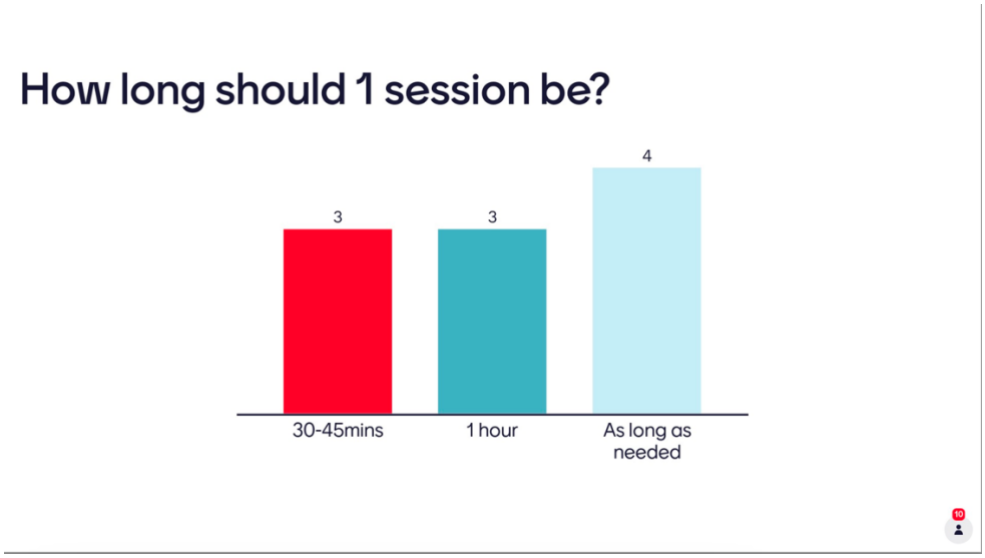

Figure 13: Results from the questionnaire during Continuous Improvement meeting (Mentimeter)

Out of 10 responders, only 1 person said that one meeting was needed, while 2 people thought 3-4 meetings were required. The other 7 people said that it was hard to predict a specific number of meetings to be held for the transfer progress. When it came to the meeting length, there was a tight between 30-45 minutes meeting and 1 hour meeting, with 3 people in favor for each, while the other 4 people said that there was no constraint to the time limit, meaning that they should last as long as insisting.

Figure 14: Results from the questionnaire during Continuous Improvement meeting (Mentimeter)

Sometimes, it would be hard to remember all the details discussed within the meetings, therefore the participants were asked whereas they felt that those meetings should be recorded and stored for later use, only 1 person replied that they did not mind about that, while the other 9 thought it was essential to record the transfer meetings so that they could access to them later as references.

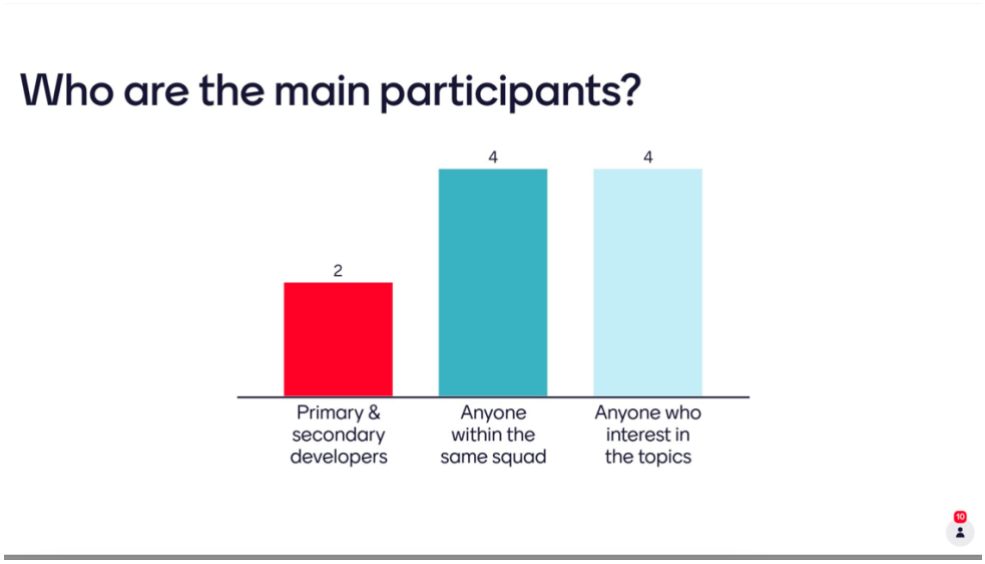

Figure 15: Results from the questionnaire during Continuous Improvement meeting (Mentimeter)

Moving on to checking who should be invited to the meetings, we learnt that 2 people chose that only the primary and secondary developers should participate the transfer progress, while the number of responses were equal between all squad members were allowed to join, and any person, regardless which squad they belonged to should have the chance to participate.

Figure 16: Results from the questionnaire during Continuous Improvement meeting (Mentimeter)

Because there were contrast opinions about the definition of Knowledge transfer and System transfer, we decided to ask what people thought each of those approaches should contain. Instead of showing the options under multiple choice format, we used the Word Cloud question type, so that people can provided their own points of view. For Knowledge transfer, the results showed different aspects, from Project overview, Customer communication to more system specific, such as Coding walkthrough, Technical stack, Environment set up etc.

Figure 17: Results from the questionnaire during Continuous Improvement meeting (Mentimeter)

As for System transfer, the results were mostly in favor of technical perspective. The participants expected that we should cover the System architecture, Popular risks, Integration and triggers, CI CD guidelines, Deployment process and more. Based on these assumption, we could see that people believed that System transfer progress referred to any technical activities performed within the system.

6.3 Results assessment

With the data collected after participating in multiple transfer meetings, combining with the interview and CI meetings, the most important pain points were picked and transferred to the Idea Napkin by Miro. In the board, there were three categories: Problem, Solution and Benefits. For Problem clarification, we could see that currently, there was no clear instruction on how to execute the Knowledge & System transfer process, the suggested participants were not defined, and the individuals' responsibilities during the transfer were missing. Moreover, the process was not transparent as it should be, leaving the people lost in tracking the states, making it impossible to recognize hidden issues and communication blocking.

Once the problem was clarified, the next step was to start with the implementation of what to change. A solution was proposed with various characters, including writing clear transfer instruction for the team, creating a sample template, placing it under testing and gradually requesting feedbacks from the users for improvement. The template should be created using Jira template format-a well-known collaboration tool using by all members of the AM team and could be considered as a continuous progress. By putting the template in use, we aimed to achieve a few goals: time and efforts saving, smooth team collaboration, straightforward tasks' visuality while limiting any inherent risks, such as forgotten documents, unsolved bugs, or lack of customer engagement awareness.

Initially, the author had the intention to separate Knowledge transfer and System transfer as two separate processes, however, due to the reaction of other team members from the interviews and CI meeting, it was proved that dividing the two concepts were unnecessary, as it would create more confusion than validation. Therefore, the template's name would precisely be Knowledge & System transfer.

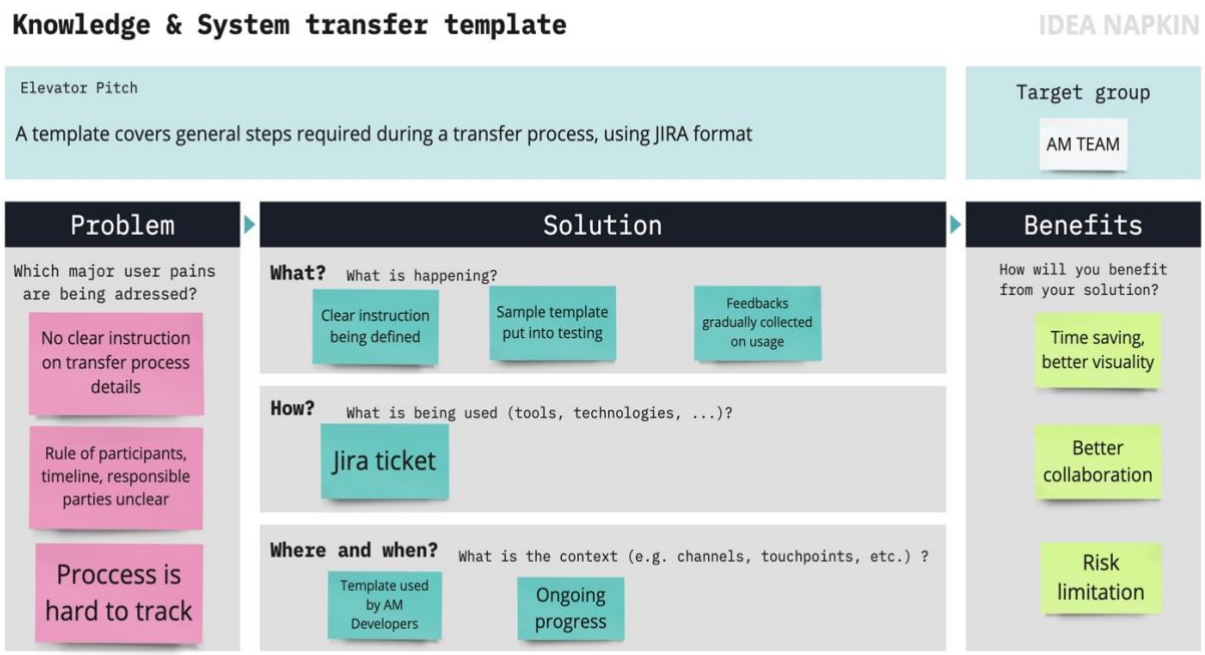

Figure 18: The Idea Napkin for assessment (Miro Board)

6.4 Sample transfer template

We were now heading to the final step: creating a Knowledge & Transfer template and testing its practicality and relevancy. The template was designed using Jira tool and follow a standard procedure. The Jira ticket would have the Customer & System name displayed, with a short Description provided. A checklist was added, with all the general TO DO items that any individual who were a part of the transfer progress must follow for checkup. The ticket would be assigned to the responsible person for updating purpose, and from the feedbacks gathered, that was the primary developers with the assistance from their team leads when required. The number of meetings would be minimum 2 sessions, with the average of one hour per session. Recording those meetings was encouraged for later use. When performing a task in the list with another member, i.e., Project management then a tag linked that individual should be added. All the tasks were listed based on their priority order, and the transfer progress should be started as soon as the transfer confirmation was announced.

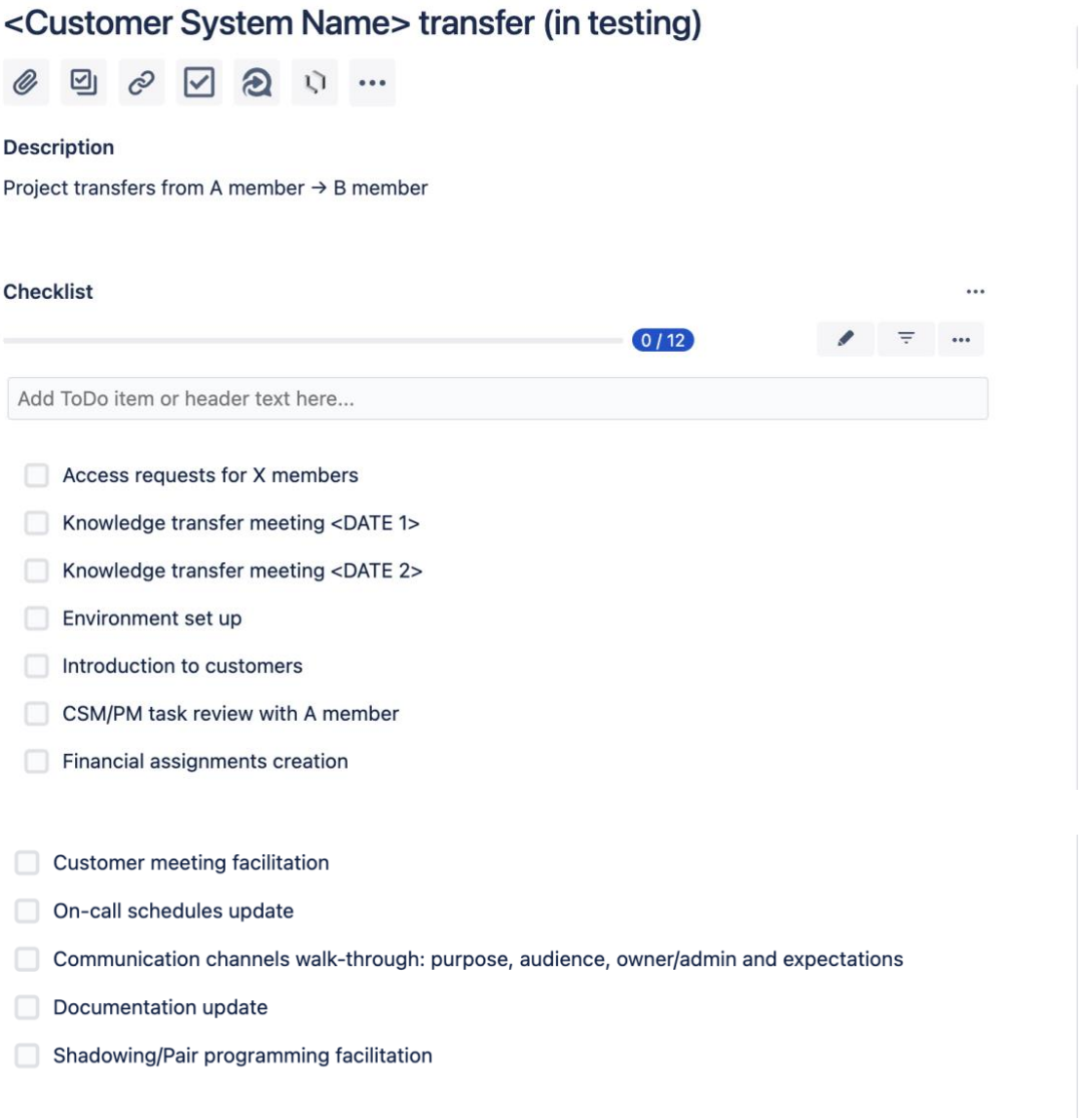

Figure 19: Sample Transfer Template design (Atlassian)

At the time of concluding the thesis process, there are a few Knowledge & System Transfer Jira tickets under testing by 2 different squads and 5-7 developers. People use the ticket checklist as references, and once completed, the tasks would be crossed out. People with specific roles also tagged to the ticket, either to be assigned to a specific task or to provide further clarification.

7 Conclusion

The results collected from the personal observation, 1-on-1 interviews and team's Continuous Improvement meeting showed that people are interested in having a different approach, when it comes to Knowledge & System transfer. From the 1-on-1 interview, we learn how the transfer process is operated in every squad, what are the current understanding and future expectations. It also shows the team lead's viewpoints, whereas they are satisfied with the current transfer arrangement or do they prefer making some improvements. From the results of Continuous Improvement meeting, we dig deeper on the practicalities of the process, based on the developers' experiences: the number of participants, accountability, timeline, materials, and collaboration tools.

From the results, we make a comparison between the management and the developers' team to find the common problems and misunderstanding between the two groups. When the problems defined, we continue with specifying what is the suitable solution: what would we do next? How would we do it? Where and when it would happen? The ambition here is to gain benefits from the changes we make or at least raising the awareness of the problem, while filling the gap between the management and development team.

It takes the author by surprise that many team members have a different approach when discussing about the transfer process, its contents, and requirements- which has been a part of teamwork for a long time. Moreover, the team leaders and their members do not share similar expectations when walking through the specifics. Nevertheless, we receive positive impressions by the people on the usefulness of the transfer template. Being a team with different backgrounds and located in various countries is already a challenge, therefore if we can agree on the same procedures, number of steps including during the transfer, the materials to walk through, responsibilities and expected outcomes, we can benefit from having a shared understanding, smoother collaboration, timesaving, and more independency. In short, we can be productive while enjoying our work together, as a team.

Of all the theoretical frameworks mentioned, only Continuous Improvement model is the most familiar with the Application Management team. The other frameworks such as SPACE and DX are somewhat unheard of, which makes the efforts to implement them are not fully feasible for the time being, because it is still debatable on what is the best approach. However, it is worth introducing and diving into the benefits of them to determine which path we should follow in the future to secure the robustness of the Application Management team, whereas it is a from specific model or the comprehensions of them all.

7.1 Limitation

One of the first limitation when starting the thesis scope is the time constraint. The author had a tight schedule of two months for personal observation, doing the interviews and conducting the questionnaires with the whole team. We also acknowledged the contrast in every squad' operation structure, making it difficult to propose a solution that would fit well into everyone's needs. Of course, as a whole Application Management team, we share common values and goals, but it is up to the squad level in defining the specifications, due to the situation of the squad.

Another limitation that surfaced that despite as much effort as we tried, implementing one template that covered all requirements was not doable, because every project was different than the others. When needed, extra steps or clarifications would be added to the template for better visuality and tracking. The responsibilities of the parties in the transfer process would also be affected by the scopes and purposes. All the team leaders when being asked said that micro-management is not their favorite practice, therefore they hope that when people are assigned with the task, they should be able to deliver and be held accountable for the outcome results.

The thesis objective included a small focus group containing of 22 people, however, we only received responses from a total of 12 developers and 4 team leads, making the results not fully reliable. We cannot force everyone to participate in the process if they do not enjoy it, and with the others who participated, we can at most make the recommendation on starting to use the template but not placing it as a rule that everyone must follow. In addition, the sample templates are currently under testing, and while it shows some positive affection, it is quite early to determine the success of the implementation. Based on the current situation, we need to wait for at least 1-2 months to see how effectively the template helps with the Knowledge & Transfer process, and only then we can conclude on whereas the template can be officially put into use.

7.2 Future suggestion

We recognized a few points that can enhance the practicality of the template. First, as a development team, we aimed to align our way of working, meaning that instead of creating the transfer template for every needed occasion, we can automate the template creation process. By using a script, we could take in team members or project names and then generate the Jira ticket automatically, including all the necessary Transfer items check list.

In addition, we can investigate which projects share the similarities and modify the template contents based on those requirements, for example some projects may require background checks, which we can include in the instruction or guidance in the designated transfer ticket.

Furthermore, the transfer tickets can be shared in the Team level board to make it easier with tracking the progress and better visuality. All other relevant discussions can be done by adding the details in the Comment section within the ticket, rather than communicating in other channels. This helps to bring full attention to all relevant members and keep them up to date with the transfer process.

References

Electronic

Nordcloud. About us. Accessed 28 February 2023. https://nordcloud.com/

Andrzejewski, J. 2022. What is Developer Experience? (DX). Accessed 25 March 2023. https://dev.to/jacobandrewsky/what-is-developer-experience-2lh8

Ayanleke, O. 2022. Why Developers Are Using The SPACE Framework For Productivity. Accessed 27 March 2023. https://www.scatterspoke.com/post/why-developers-use-space-framework

Azorin, P. No date. Best Knowledge transfer Methods for Development teams. Accessed 25 March 2023. https://www.bairesdev.com/blog/development-teams-knowledge-transfer/

Brearley, B. 2017. Why your team needs to keep improving. Accessed 28 March 2023. https://www.thoughtfulleader.com/continuous-improvement-matters/

Bulao, J. 2023. How Fast Is Technology Advancing in 2023? Accessed 28 February 2023. https://techjury.net/blog/how-fast-is-technology-growing/#gref

Carruthers, R. 2021. Knowledge transfer: Keeping critical know-how within the organization. Accessed 25 March 2023. https://www.togetherplatform.com/blog/knowledge-transfer

Cavalcante, A. 2019. What is DX? (Developer Experience). Accessed 25 March 2023. https://medium.com/swlh/what-is-dx-developer-experience-401a0e44a9d9

Cavalcante, A. 2020. Developer Experience Metrics. Accessed 25 March 2023. https://medium.com/@albertcavalcante/developer-experience-metrics-46b1d087811d

Circei, A. 2022. The SPACE framework For Software Developer Productivity. Accessed 30 March 2023. https://www.forbes.com/sites/forbestechcouncil/2022/12/16/the-space-framework-for-software-developer-productivity/?sh=3de55bc229d5

Cleland, J. 2017. The qualitative orientation in medical education research. Korean Journal of Medication Education, 29(2), 61-71. doi: 10.3946/kjme.2017.53

Dewar, C. Doucette, R. & Epstein, B. 2019. How continuous improvement can build a competitive edge? Accessed 28 March 2023. https://www.mckinsey.com/capabilities/people-and-organizational-performance/our-insights/the-organization-blog/how-continuous-improvement-can-build-a-competitive-edge

Dziuba, A. 2021. Effective Knowledge Transfer Between Software Teams: Methods and Tips. Accessed 25 March 2023. https://relevant.software/blog/effective-knowledge-transfer-between-software-teams/#10_Communication_and_collaboration_tools

Edwards, L. 2021. What is Mentimeter and How Can It Be Used for Teaching? Tips and Tricks. Accessed 05 April 2023. https://www.techlearning.com/how-to/what-is-mentimeter-and-how-can-it-be-used-for-teaching-tips-and-tricks

Erilkh, L. 2000. Leveraging legacy dollars system for e-business.IT Professional, 2(3), 17-23. doi: 10.1109/6294.846201

Gadhavi, M. 2022. Why Software Maintenance Is Necessary? Accessed 30 March 2023. https://radixweb.com/blog/why-software-maintenance-is-necessary

Gavrilovska, A. 2021. How to successfully hand over systems? Accessed 01 April 2023. https://developers.soundcloud.com/blog/how-to-successfully-hand-over-systems

George, T. 2022. Types of Interviews in Research. Guide and Examples. Accessed 05 April 2023. https://www.scribbr.com/methodology/interviews-research/

Gill, P. Stewart, K. Treasure, E. Chadwick, B. 2008. Methods of data collection in qualitative research: interviews and focus group. Accessed 05 April 2023. https://www.nature.com/articles/bdj.2008.192

Hrzenjak, B. Best practices for creating a great developer experience (DX). Accessed 25 March 2023. https://www.shakebugs.com/blog/developer-experience-best-practices/

Atlassian Community. Accessed 25 April 2023. https://community.atlassian.com/t5/Jira-Software-questions/Checklist-in-jira-ticket/qaq-p/1342697

Kellar, W. 2020. “If you fail to plan, you are planning to fail”. Accessed 30 March 2023. https://www.humaninvesting.com/450-journal/if-you-fail-to-plan-you-are-planning-to-fail

Koponen, A. 2022. Your organization’s guide to the SPACE framework. Accessed 30 March 2023. https://www.swarmia.com/blog/space-framework/

Maestro 2020. How to Effectively Complete a Knowledge Transfer Plan. Accessed 28 March 2023. https://maestrolearning.com/blogs/how-to-effectively-complete-a-knowledge-transfer-plan/

Middleton, T. 2022. The importance of teamwork (as proven by science). Atlassian. Accessed 01 April 2023. https://www.atlassian.com/blog/teamwork/the-importance-of-teamwork

Needle, D. 2022. 4 reasons why business need to use collaboration tools. Accessed 07 April 2023. https://www.techtarget.com/whatis/feature/4-reasons-why-businesses-need-to-use-collaboration-tools

Forsgren, N., Storey, E., Maddila, C., Zimmermann, T., Houck, B., Butler, J. 2021. The SPACE of Developer Productivity, 19(1). Accessed 29 March 2023. https://queue.acm.org/detail.cfm?id=3454124

Nussbaum, H 2022. Understand and Optimize Developer Productivity with the SPACE framework. Code Climate. Accessed 29 March 2023. https://codeclimate.com/blog/the-space-framework

Pauly, E. 2022. 5 Best Developer Productivity Metrics & How to track them. Accessed 30 March 2023. https://linearb.io/blog/developer-productivity-metrics/

Pedro, T. 2022. Boost your developer productivity with SPACE framework. Accessed 30 March 2023. https://linearb.io/blog/space-framework/

Pluralsight. 2022. Measuring developer experience: The superpower of today. Accessed 28 March 2023. https://www.pluralsight.com/blog/software-development/measuring-developer-experience

Radcliffe, A. No date. What is Software Maintenance and why it is essential in the current crisis? Accessed 05 March 2023. https://spyro-soft.com/blog/software-maintenance-services-benefits

Reddy, C. No date. Why Research is Importance for Students, Humans, Education. Accessed 10 April. https://content.wisestep.com/research-important-students-humans-education/

Rodriguez, E. 2022. What is Interpretative Phenomenological Analysis (IPA)? Accessed 20 April. https://doctorelenagr.com/meet-elena/

Sasidharan, D. 2021. What is Developer Experience and why should we care? Accessed 26 March 2023. https://dev.to/adyen/what-is-developer-experience-and-why-should-we-care-1k9i

Service Now, No date. What is the Software Development Life Cycle? Accessed 30 March 2023. https://www.servicenow.com/products/devops/what-is-sdlc.html

Shead, S. 2020. IBM snaps up Finnish cloud firm Nordcloud as battle with AWS, Microsoft and Google heats up. Accessed 28 February 203. https://www.cnbc.com/2020/12/21/ibm-buys-nordcloud-as-cloud-wars-with-google-aws-microsoft-heat-up.html

Simic, P. Guide to SPACE framework and metrics for developer productivity. Accessed 07 March 2023. https://www.shakebugs.com/blog/dev-space-framework/#:~:text=what%20they%20are!-,SPACE%20metrics,Source%3A%20Shake

Swersky, D. 2022. The SDLC: phases, popular models, benefits and more. Accessed 01 March 2023. https://raygun.com/blog/software-development-life-cycle/

Towns, A. No date. How to measure developer experience (with metrics!). Accessed 27 March 2023. https://www.getclockwise.com/blog/measure-developer-experience-metrics#:~:text=Examining%20lead%20time%2C%20automation%2C%20and,to%20understand%20the%20developer%20experience.

Tsuei, J. No date. Developer Experience (DX) and why it matters. Accessed 29 March 2023. https://www.getclockwise.com/blog/what-is-developer-experience

Vasin, A. 2023. Knowledge transfer methods for Software team: How to Ensure the Smooth Onboarding of New Hires. Accessed 02 April. https://youteam.io/blog/knowledge-transfer-methods-for-software-teams/

Velimirovic, A. 2022. What is SDLC? Understand the Software Development Life Cycle. Accessed 02 March 2023. https://phoenixnap.com/blog/software-development-life-cycle#:~:text=An%20SDLC%20(software%20development%20life,all%20major%20stages%20of%20development.

VMEC. 2018. Using Continuous Improvement (CI) to Benefit Your Business. Accessed 20 March 2023. https://vmec.org/continuous-improvement-benefit-business/

Warren, K. 2020. Qualitative Data Analysis Methods 101: The “Big 6” Methods and Examples. Accessed 20 April 2023. https://gradcoach.com/qualitative-data-analysis-methods/

Waters, S. 2022. What will make or break your next role? Find out why teamwork matters? Better Up. Accessed 05 April 2023. https://www.betterup.com/blog/what-is-teamwork

What is a Software Maintenance Process? 4 Types of Software Maintenance. Accessed 02 March 2023. https://cpl.thalesgroup.com/software-monetization/four-types-of-software-maintenance#:~:text=Software%20maintenance%20is%20the%20process,to%20boost%20performance%2C%20and%20more.

Wójcik, S. 2023. This is not what I expected-how to hand over the software project to new developers? Accessed 10 April 2023. https://crustlab.com/blog/software-project-handover/

Zainal, Z. 2007. Case study as a research method. Accessed 15 April 2023.https://jurnalkemanusiaan.utm.my/index.php/kemanusiaan/article/view/165/158

Figures

Appendices

Appendix 1: Interview questions

1. Would you like to share a bit about yourself?
2. How long have you been in your role?
3. What are your daily responsibilities?
4. There are 3 identified problems: the first is not having enough work to do, the second problem is Knowledge transfer, and the last problem is System transfer process. In your opinion, are these problems relevant with your squad?
5. How do you understand about the concept of Knowledge transfer?
6. Is it clear for all your team members to know what to do during a knowledge transfer session? Are they happy with the current process set up?
7. When is the best time to conduct the Knowledge transfer meeting?
8. What about the System transfer process? Is the concept clear to you?
9. Is it clear for all your team members to know what to do during a System transfer session? Are they happy with the current process set up?
10. When is the best time to conduct the System transfer meeting?
11. Do you need to have follow up with these transfer process, based on your squad situation? If yes, which methods do you use, i.e., personal notes, official documentation?
12. Do you feel the need to micro-manage or the team members should have the freedom in conducting the transfer process?
13. In the squad, is the communication transparent enough?
14. Are the team members confident with asking for help?
15. What changes do you want to see?
16. Other comments? Questions you want to answer?

Appendix 2: Topic presentation during CI meeting

31.03.2023

Knowledge transfer & System Handover Guideline

Thesis topic: Implementing KT transfer & System Handover as a part of Continuous Improvement in Software Maintenance

Target group: AM team.

Goals:

- Transparent handover process
- All important handover items are performed within timeline
- Smooth collaboration
- Higher productivity, less conflicts
- Automated process
- More visibility

Nordcloud an IBM Company

How?

- MCA-1809
- MCA-1808
- Jira ticket format
- Items sorted by priority
- Automated process
- Basic requirements covered
- Links to sub-tasks for project-specific clarification

Nordcloud an IBM Company