TALLINN UNIVERSITY OF TECHNOLOGY
Faculty of Information Technology
Department of Software Science

Ain-Joonas Toose 134301IAPB

Analysis of the new Swedbank software development process

Bachelor's thesis

Supervisors: Gunnar Piho, PhD

Sofia Trummal

Tallinn
2017

TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond
Tarkvarateaduse instituut

Ain-Joonas Toose 134301IAPB

Swedbanki uue arendusprotsessi analüüs

Bakalaureusetöö

Juhendajad: Gunnar Piho, PhD,

Sofia Trummal

Tallinn
2017

Autorideklaratsioon

Deklareerin, et käesolev lõputöö on minu töötulemus ja seda ei ole kellegi teise poolt varem kaitsmisele esitatud

Kuupäev: ……………………………………………….

Lõputöö kaitsja allkiri: ………………………………………………

Contents

Annotation

Ain-Joonas Toose (2016), Analysis of the new Swedbank software development process, Tallinn University of Technology, institute of Informatics. Tallinn. Manuscript.

The goal of this body was to

1. Compare the new development process to a repeatable CMMI model.
2. Compare the new development process to the process used before and to compare to the popular agile development platform Scrum.
3. Compare the new development process to the one used previously.
4. Study the reception from developers using the process and their comparison of it to the previous.

The study found the new development process rated as a level 2 maturity development process due to the flexibility of the process, the documentation, repeatability and the continuous integration.

Compared to the old process, the new process was found superior in its way of operation to the old. The study also found the elements of Scrum that resemble the new process, but also the differences mainly in how the teams are structured, the team size and the release planning process.

The study found the developers to be highly receptive to the new process with more positive feedback towards it than to the old process. With greater oversight, better expectation management, planning and communication between parties, the new process was deemed as a positive while the old was claimed to be lacking in all the properties counted earlier.

Keywords: software development processes, CMMI, Scrum, agile, waterfall

34 pages. 21 references used. 11 Illustrations. 2 extras.

Annotatsioon

Ain-Joonas Toose (2016), Swedbanki uue arendusprotsessi analüüs, Tallinna Tehnikaülikool,Informaatikainstituut. Käsikiri. Tallinn. 32 lehekülge. 20 kasutatud allikat. 7 illustratsiooni. 2 lisa.

Töö eesmärgiks oli

1. Võrrelda uut arendusprotsessi teise astme CMMI mudeliga.
2. Võrrelda uut arendusprotsessi eelnevalt kasutatuga kui ka populaarse agiilse tarkvara arenduse protsessiga *Scrum*.
3. Võrrelda uut arendusprotsessi temale eelnevalt kasutuses olevaga.
4. Uurida tarkvara arendajate arvamust uuest protsessist ning nende võrdlust eelnevaga.

Uurimusest leiti, et uues tarkvara arenduse protsess on võrreldav neljanda taseme valmidusega tarkvara arenduse protsessiks tänu süsteemi paindlikkusele, dokumentatsioonile, korratavusele ning jätkuvintegreerimisele.
Võrdluses eelmise protsessiga, leiti uus protsess suurema valmiduse astmega protsessina ning peamised erinevused vana süsteemiga. Uurimusest samuti leidusid sarnasused *Scrum* protsessiga ning erinevused, mis olid implementeeritud uude protsessi. Nendest peamised seisnesid tiimi suuruses, struktuuris ning tarkvara reliisi planeerimise protsessis.

Uuringust leiti, et arendajad on võtnud uue protsessi hästi vastu. Tänu paremale ülevaatlikkusele, paremate eelduste tegemisele, planeerimisele ja kommunikatsioonile osapoolte vahel, leiti uus protsess hea arenguna eelneva üle, nähes selle puuduseid peamiselt eelnevalt mainitud omadustes.

Märksõnad: tarkvara arenduse protsessid, *agile, waterfall,*CMMI,*Scrum*

34 lehekülge. 21 kasutatud allikat. 11 illustratsiooni. 2 lisa.

1.Introduction

In the modern world, the need for new software has risen tremendously. With the need for new programs, databases, interfaces and websites, many of the development processes employed in other fields or even in the software company market have proven themselves cumbersome, with software coming out at points where they could already be facing extinction. To face this problem, many of these companies shifted their focus to use quicker ways to deliver software while not having to make compromises in the quality of the product. The result of that are the different agile processes commonly used today, *Scrum* and *Kanban* being the prime examples of popular, iterative, software development processes.

In 2016, Swedbank, though largely a financial company, holds a dedicated software development group, whose primary objective is to create software to provide many different services to its customers, starting from services allowing customers to check their finances online to large business data analysis and business intelligence. Due to the compromising nature of the work, most of the people working on these projects are directly employed by Swedbank, making them at least partly, a software company.

Due to the competitive nature of the marketplace, Swedbank also realized a need the solutions provided by agile development processes.

In 2015, Swedbank introduced a new software development process to use in developing new business software, with deployment starting in several internal groups of the organization, of which one the author is a part of. With its own documentation and vision, the process is reminiscent of Scrum and states directly in its Document the inspiration from other agile and lean processes [1].

The Swedbank Agile Process(SWAP) is a development process derived from agile and lean principles with the purpose to enable greater adjustability to the changing state of the market and to enable a faster time to release with the focus of delivering as much business value as possible by continuous prioritization of tasks with quality and stability in mind [1].

SWAP is designed to replace the old DP (Development Process) of Swedbank, which is a development process reminiscent of Waterfall. Swedbank has set a vision to replace all groups using the DP process to start using SWAP, employing guiding staff and using internal methods of validating the use.

There has yet to have been a validation of the maturity level of the SWAP process and its implementation in a team. Therefore, a problem is found – there has yet to have been an analysis of the efficiency of SWAP compared to its precursor nor a metric by which one could determine the efficiency of the new system.

The purpose of this work is to assess the maturity of the Swedbank Agile Process using CMMI, comparing to the popular agile development process Scrum, the previous development process used and the input gathered from the members of the development team present when the group transitioned from DP to SWAP.

The research is split into three parts:

1. Information about the methods used: CMMI, SWAP, DP, Scrum.
2. Work done, the comparison of SWAP to CMMI and Scrum, input from developers.
3. Analysis of the work done.

2.Theoretical bases and methods

2.1 SWAP

Swedbank Agile Process, SWAP, is the development process used for agile stable cross functional teams, with end-to-end delivery responsibility. [1] A process based on agile and lean principles with the purpose to enable a faster time to market and enable better conditions to shift focus on different developments to deliver more business value from the products developed. To create such an environment, SWAP is handled in teams called „cross-functional teams “, teams comprised of developers and business personnel who deal with delivery in a stable manner, meaning the composition of the teams are not based on the projects handed, but the projects are handled by the cross-functional teams. In the current system, SWAP teams exist intermittently with teams using the DP process. SWAP teams are allowed to differ from each other in their way of adapting the system.

The key concepts of SWAP, as described in the documentation are [1]:

- Agile iterative and incremental development processes
- Cross Functional Teams
- Fixed monthly invoicing
- Multiple Minimum Viable Product deployments for each product
- Checkpoints and artefacts

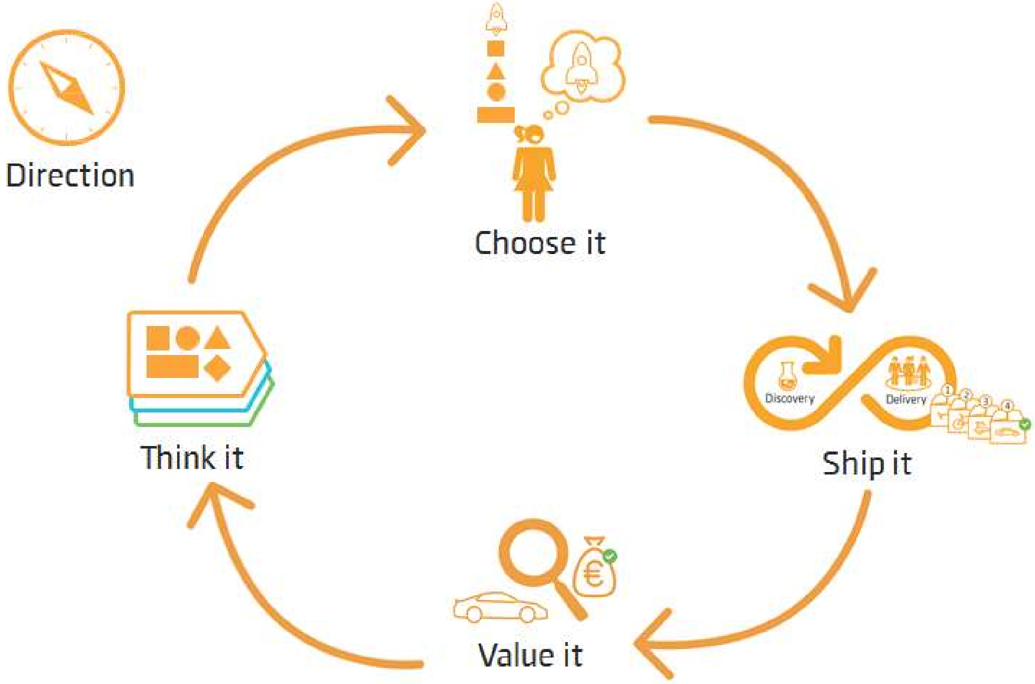
The guiding values of SWAP are derived from the Agile manifesto [2] and lean thinking [3] with the guiding principles named accordingly:

- **Focus on customer value** – ensure the best cost and benefit to the customers and users
- **Continuous planning that welcomes change** – prioritize according to outside changes.

- **Cross Functional teams**
- **Fixed cost** – teams have a fixed budget with the purpose to focus less on managing cost and resources.
- **Deliver fast and get feedback often**
- **Lean Quality Assurance** – products are tested early on with focus on test automation and testing in production-like environments.
- **Improve continuously**
- **Continuous integration and delivery**

SWAP products are continuously refined with meetings where business presents new investments (user stories) with the purpose of having the development part of the team analyze, value, prioritize and ultimately solve.

SWAP works iteratively, typical teams have 2-4 week sprints where the items solved in the sprint are chosen at the start of the iteration. In this context, the iteration and sprint share the same exact meaning. Swedbank has defined the model with the following four steps: Think it, choose it, Ship it and Value it.



Guiding principles of SWAP [1]

A typical SWAP team has 4 different types of meetings.

- Dailies, a daily meeting where all members of the team are gathered to describe the work of the day before and the current.
- Refinement, meetings where the team is gathered to present and evaluate the user stories created.
- Planning, a meeting at the start of every sprint where refined user stories are chosen for development.
- Retro, a meeting at the end of every sprint, where the work done in the sprint is discussed and evaluated.

Cross functional teams are comprised of business specialists, developers and product owners, who take input from stakeholders and a Scrum Master, a team member with the added responsibility of team management.

SWAP teams are often focused on many different products at the same time. Often a sprint is consistent of different ongoing, larger projects causing the focus of the team split between them, with more different projects sitting in the backlog, waiting to be or already refined.

Though SWAP intends to forego many of the waterfall-*esque* principles, it still shares some of the artefacts with the old system employed. Though a pre-study is not required for every product, the SWAP document still suggests to study opportunities and create a delivery plan, a master plan, describe risks and other actions that usually should present themselves naturally in the incremental and iterative nature of a typical agile process.

2.2 DP

The Swedbank Development Process(DP) is the official development process used in the daily operation of Swedbank. As SWAP belongs to Development Process at its nature and is even described in the official documentations, the process taken under look in this research is the older method, dubbed internally as the Waterfall method.
Development Process guides all participants in project work through a disciplined, reliable approach distilled from expert experience.

It provides easy-to-use techniques and a common language that promote good engineering discipline.

It is designed to improve performance (quality and productivity) of both individuals and teams involved in systems development activities. [13]

DP involves 3 levels of process with 4 different iterations in which a product is developed.

The 3 levels of value are described as.

- Decision level, where the main task is for the chairman to assess the business value, risks, results, costs and plans with respect to the deciding party. A steering committee representing the deciding body's interest oversees the process.
- Quality level, where the quality of the product in development is ensured by the project manager and the team(s) it governs
- Activity level where the project manager ensures the efficiency and holds responsibility to the project being finished in lieu of the markers set by the steering committee and the chairman.

The 4 iterations are described as

- Business Needs Analysis, where the main objective is to figure out the business needs and plan validity to follow into the next phase
- Pre-study, where the project business model is examined, solutions or suggestions are formed, the architecture is described with possible risks. Business cases are refined by the team to achieve greater awareness until the project is ready for development.
- Project development, where 4 independent iterations are carried through in standard Waterfall measures, meaning initiation, development, verification and delivery are all done at separate times.
- Value Realization, the post study of the developed product where new measures are taken about optimization and later business value.

In the DP process, all work is project based.

While highly optimized and standardized, the DP process carries almost all the properties of a standard Waterfall process with all the limitations found and described in previous studies. Teams are function based and testing is detached from development, a model proving itself to be with a velocity unsuitable for development in the modern world.

2.3 CMMI

Capability Maturity Model Integration is a process for improvement training and appraisal administered and marketed by Carnegie Mellon University (CMU) with the main objective of guiding process improvement across a project, division or an entire organization. CMMI defines different maturity levels for processes for assessment. Some of the more known users of CMMI are NASA, the Department of Defense(US) and their contractors. The version used for this research is CMMI-DEV 1.3. CMMI for Development is a reference model that covers activities for developing both products and services [4].

The main objective of CMMI is to reduce the cost of implementing improvements in processes by eliminating possible inconsistencies and establishing guidelines to assist organizations at various stages of a software project (planning, management, and others). The architecture is comprised by defining a set of 22 process areas, in which they are grouped into five levels of capability: Initial, Repeatable, Defined, Managed and Optimizing.

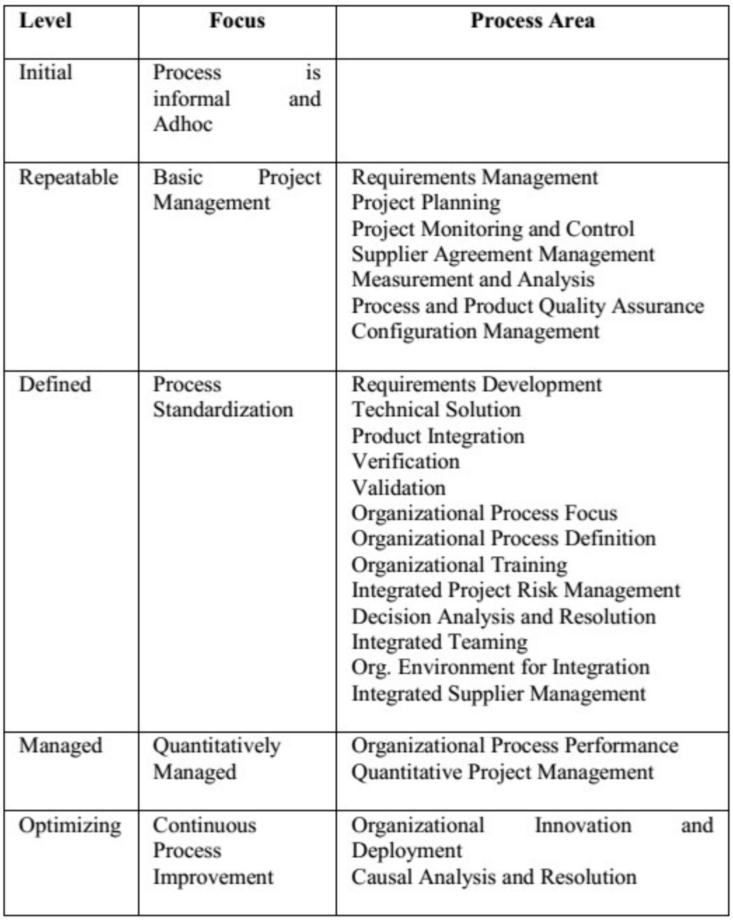
CMMI level 1 describes as an ad-hoc, chaotic model where no process has been repeated nor are there any rules to follow. Since SWAP operates in managed teams with clear vision and oversight with a strict release system in the background, SWAP definitely exceeds the first level expectation.

CMMI level 2 describes a system where steps are documented sufficiently enough that repeated attempts are seen as possible. While the process discipline is unlikely to be rigorous, the process is maintainable enough to be trusted during stress.

CMMI level 3 describes a defined process, where standard procedures are documented and established with the capability of improvement over time. These processes are stable and consistent with members of a process in this level are aware and consistent in their performance. The way projects are examined and developed are always in the same way with management being able to reliably identify problems and members able to solve them.

CMMI level 4 describes a managed system. Using process metrics, the team management can effectively control the process as-is and identify ways to adjust and adapt to the project without measurable losses in quality. As such, process capability is set at this level. Processes are easily modified to allow better performance without documentation suffering, meaning the teams can effectively “bend” their process around the project.

CMMI level 5 characterizes a level 4 process where the focus is on continuously improving process performance both by incremental and innovative technological changes. The process is able to address statistical common causes of process variation and changing the process without the need to change documentation radically nor upset the performance of a team. A level 5 process is continually improving in markers that can be proven effectively. [14]



Level	Focus	Process Area
Initial	Process is informal and Adhoc	
Repeatable	Basic Project Management	Requirements Management Project Planning Project Monitoring and Control Supplier Agreement Management Measurement and Analysis Process and Product Quality Assurance Configuration Management
Defined	Process Standardization	Requirements Development Technical Solution Product Integration Verification Validation Organizational Process Focus Organizational Process Definition Organizational Training Integrated Project Risk Management Decision Analysis and Resolution Integrated Teaming Org. Environment for Integration Integrated Supplier Management
Managed	Quantitatively Managed	Organizational Process Performance Quantitative Project Management
Optimizing	Continuous Process Improvement	Organizational Innovation and Deployment Causal Analysis and Resolution

Table 1: CMMI model [4]

Its use has been widespread in assisting the conduct of software projects in organizations. [5]

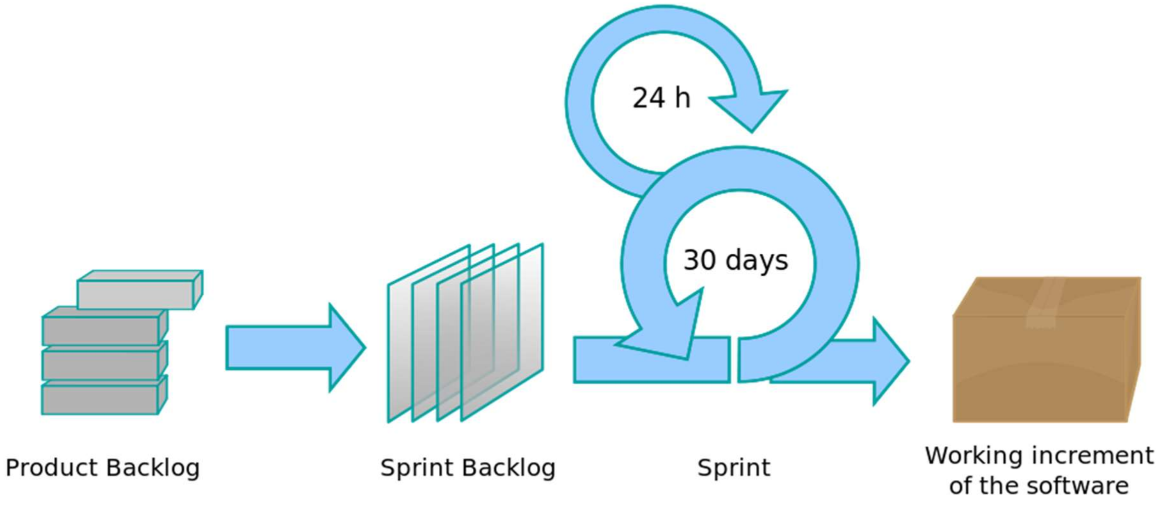
This way of assessing maturity of a process has been employed before [19], with a notable use in BI process assessment in Iranian banking industry. [18]

2.4 Scrum

Scrum is an iterative, incremental agile software development framework for managing product development [6]. Scrum structures work in iterations called Sprints, which are typically between 2 to 4 weeks in length and take place one after the other for the duration of a project.

As does SWAP, Scrum teams consist of cross-functional teams working on user stories in a time boxed iteration [7].

At the beginning of each Sprint, a cross-functional team selects items from a prioritized list of requirements called the Backlog; the list is created and maintained by the product owner(s), who ranks them according to the business value they hold.



The Scrum process [7]

Each sprint the team has a plan meeting where they decide together what needs to be done for the next sprint. The result of it is the sprint backlog. During the execution of each Sprint, the teams gather daily for a ~15-minute meeting called Daily to track the work progress by answering three simple questions – “What have I done since the last scrum meeting?”, “What will I do until next scrum meeting?” and “What prevents me from doing my work as efficiently as possible?” [9]

At the end of the Sprint, the team holds a Sprint Review, a meeting where the new developments are demonstrated and feedback from the product owner(s) which they can incorporate into the product backlog for future Sprints. The team also holds a Sprint Retrospective, where teams’ way of working is revised and changes in work are agreed on. Each Sprint is self-contained, meaning that new stories being added to the sprint backlog in the middle of the sprint are discouraged. The Product Owner can make whatever changes they like to the Product Backlog though: adding, removing or reprioritizing items at will.

Scrum is the most used agile process used in the software industry [8], seen in many areas, from academic institutes and financial companies to software development companies as the de facto way of developing in an agile way.

2.5 Background and methods

2.5.1 Background of the organization

Swedbank is a Nordic-Baltic banking group offering mainly retail banking, asset management and financial services. Other services include, for instance, insurance. Swedbank grew out of a savings banks group in Sweden and since its formation has reached out to Norway, Estonia, Latvia and Lithuania where the largest hold on the market is in Estonia. Today, Swedbank has 9.5 million retail customers and over 600 000 corporate customers with a 40% market share in Estonia [10], it is also a large employer, employing more than 16 000 employees across all its operations and is currently one of the most sought after employers in Estonia. [11]

In the Baltics, Swedbank gained market by buying out the Estonian bank Hansa Pank, later called Hansabank, to form Swedbank Baltics. As Hansa Pank had a great background in info technology, the services developed were kept and the organization itself renamed Swedbank Baltics.

The organization is split into many groups classified by their main focus and furthermore to groups based on the products the team work on. As an example, the team viewed in this research is specialized in loan services, having ownership of every product associated with loans.   Development teams are split among countries. While most of the group is located at a single site, it is not irregular to see offsite teams such as business analysts in Latvia and Lithuania or development teams in other cities.

2.5.2 The teams

The teams under view in this research, Loans Origination and Loans core, hold about 30 employees, mostly systems analysts or developers. These teams work on all the Baltic loan products Swedbank offers, from signing loan applications to delivering loan offers and credit cards with the focus on mainly backend services in Loans Core to developing internet bank and client manager specific software in Loans Origination.

The teams consist of roughly 10-15 people with most of them being developers and testers and 2-3 business analysts. The team leader is chosen inside the team and does not have to be the team manager, the person who deals mostly with managing the team outside of the work process and handles communications with other teams besides other managerial duties.

With many focuses, the teams are often split by different ongoing projects handled by separate product owners with a single product owner often having multiple projects. It is common to see at least 2 different projects on a teams' task list on a single sprint. As such, the line between a business analyst and a product owner can often grow awfully slim.

In the spring of 2015, both of the teams transitioned from the Waterfall-schemed Swedbank System, called the Development Process, to an agile approach when SWAP was announced, with the focus of using Scrum as the base of their agile way of working because of distinct lacks in the documentation about what iterative method of agile development was supposed to be used or a description of the system on a team level.

2.5.3 Current working process

The bank operates the loading of new software to production servers through monthly releases, where all groups participating in that release are given a development deadline, a testing deadline and a release deadline, with development time typically sitting at 3 weeks. Testing and bug fixing is usually given a week because most of the testing is expected to be done in development time, leaving the testing week mostly for regression tests.

Because every team operates under different development systems, with some using the Waterfall scheme and others employing different agile methods with different iteration times, a global release system is necessary to have an oversight on the changes made in the production environment.

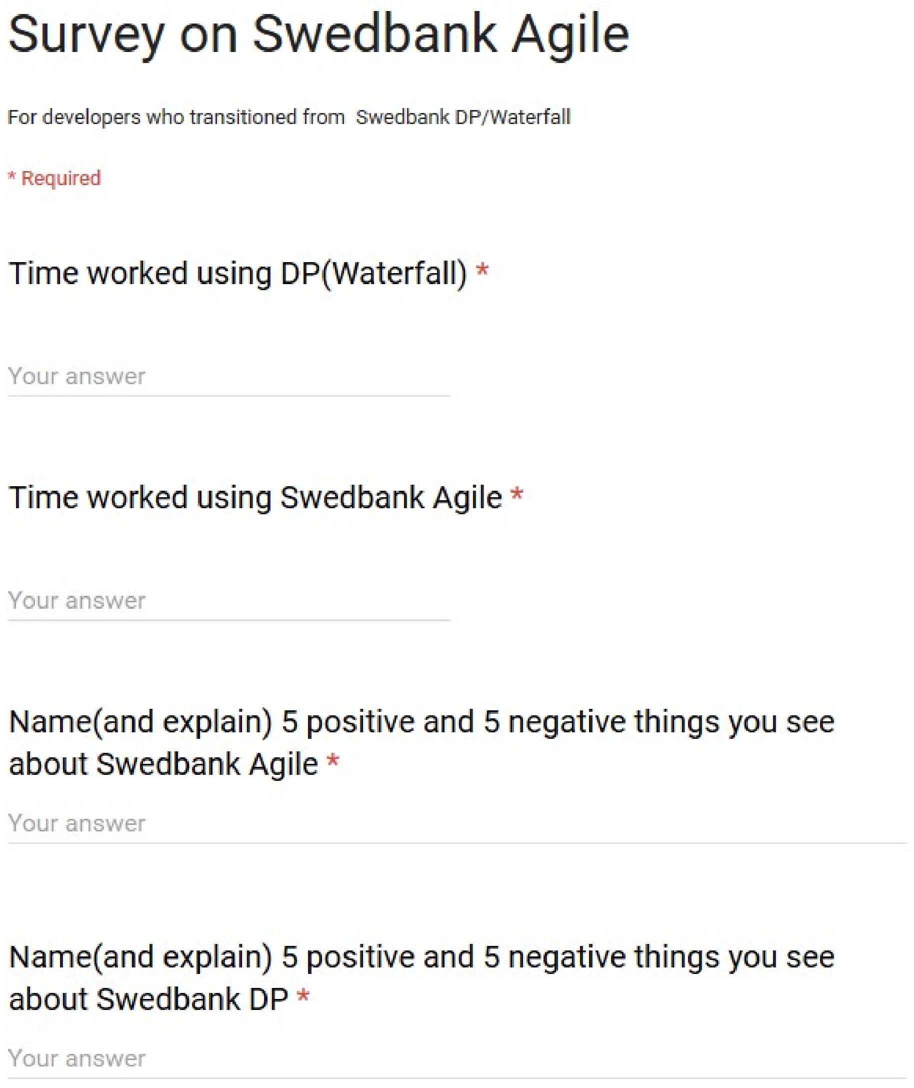
Development is mostly made in separate development environments with some being shared between teams and an environment dedicated to continuous integration. All of the pre-production testing is done in two standard environments with one holding constantly a copy of the current production environment and the other holding new developments for immediate access by all teams. In the testing period of each release, a new production copy environment is loaded with the developments approved for release loaded also onto it.

The teams are funded by projects, requiring the teams to sign time worked based on projects rather than as a team. Since SWAP requires teams to be funded by teams, the current state of the implementation is still ongoing. As the projects are all funded individually, the releases are project based also, though the release structure is not as rigid as one would see in a Waterfall system, products are released with agility in mind. From the point of proposal to development teams, it is not uncommon to see projects released to production in the span of a few months, a large improvement from the waterfall system where many of projects of same scope could take at least 6 months to production.

2.5.4 Questionnaire for the developers' opinion

For the analysis, the opinions of 4 separate developers were gathered from a base of 20 developers in the acceptance range, the range criterions being experience in both DP and Waterfall. With experience in DP ranging from a year to more than 10 years and the experience in agile development being in the range of 2 years, the people asked were deemed sufficient in their metrics to answer the questions provided.

The questionnaire consisted of 4 questions – 2 to classify the two qualifications, them being the time worked using DP and time worked using SWAP. For the next 2 questions, the developers were asked to name and explain the 5 positive and 5 negative aspects they have about SWAP and DP



Survey on Swedbank Agile

For developers who transitioned from Swedbank DP/Waterfall

* Required

Time worked using DP(Waterfall) *

Your answer

Time worked using Swedbank Agile *

Your answer

Name(and explain) 5 positive and 5 negative things you see about Swedbank Agile *

Your answer

Name(and explain) 5 positive and 5 negative things you see about Swedbank DP *

Your answer

Form presented to the developers

The developers were given the possibility to define their own shortcomings and praises to the processes due to the investigative nature of the form – the questionnaire was not designed to frame the processes in any light for the developers due to the need of hearing the developers’ own opinion and not any that could possibly come out of the architecture of the form.

Defining the 5 positive and 5 negative aspects of SWAP, explained in the form as Swedbank Agile were required to see the developers’ understanding of SWAP and how they feel its implementation to be. Some developers did not produce exactly 5 aspects of each, but all developers shared at least 2.

Defining the 5 negative and positive aspects of DP were required to gather a quick analysis of the old development process with the goal of finding further material for comparison.

An example of a response in the questionnaire is available in the extras.

2.5.5 Comparison

A large part of the analysis of the new system revolves around comparing it to other systems. As a way of finding the maturity of the software development process, SWAP is compared to CMMI.

In this comparison, SWAP will be compared to the CMMI process at level 2(repeatable). A level 2 CMMI implementation is chosen because of the relative amount of markers to compare against. Whereas the process may fit some parts of the description of a level 3 CMMI process, but will not be under view in the analysis of the system at this time. A better recognized version of this comparison has been done to compare Extreme Programming (XP) to CMMI, where the third level of key process areas were also analyzed. [21]

Because SWAP at a first glance is reminiscent of Scrum, the process will be compared to the popular agile development process by comparing the teams, the planning process and the release process.

Due to the fact that SWAP is built as a replacement for the Waterfall-based Development Process, the third comparison will be descriptive, bringing a comparative overview of the changes between the two processes.

3 Methodical comparison

3.1 Current working process

The bank operates the loading of new software to production servers through monthly releases, where all groups participating in that release are given a development deadline, a testing deadline and a release deadline, with development time typically sitting at 3 weeks. Testing and bug fixing is usually given a week because most of the testing is expected to be done in development time, leaving the testing week mostly for regression tests.

Because every team operates under different development systems, with some using the Waterfall scheme and others employing different agile methods with different iteration times, a global release system is necessary to have an oversight on the changes made in the production environment.

Development is mostly made in separate development environments with some being shared between teams and an environment dedicated to continuous integration. All of the pre-production testing is done in two standard environments with one holding constantly a copy of the current production environment and the other holding new developments for immediate access by all teams. In the testing period of each release, a new production copy environment is loaded with the developments approved for release loaded also onto it.

The teams are funded by projects, requiring the teams to sign time worked based on projects rather than as a team. Since SWAP requires teams to be funded by teams, the current state of the implementation is still ongoing. As the projects are all funded individually, the releases are project based also, though the release structure is not as rigid as one would see in a Waterfall system, products are released with agility in mind. From the point of proposal to development teams, it is not uncommon to see projects released to production in the span of a few months, a large improvement from the waterfall system where many of projects of same scope could take at least 6 months to production.

3.2 Comparison of SWAP to DP

The initiation of a product development follows similar methods in both processes. The business analysis with the risks and needs with business rules and viability are gathered, analyzed, developed and presented in both systems. What follows is the process of communicating the project to the development team. This is the point where both systems though diverge. In the DP style of work, a pre-study follows, where the architectural assessment is made, while in the SWAP system, the plan is communicated to the cross-functional team with experience in development and architecture to the team.

DP is rigid in its process, the goals are set early in the analyzing process and change difficult to present. At the prestudy phase, most of the requirements are set, technologies, team members chosen and put into the waterfall system. In SWAP, the business requirements can always change and follow the implications set by the real life changes around them. The

Waterfall process in that category has the high chance of becoming "bubbled" – the model picked in the prestudy can quickly become outdated, making the world the product is developed for a non-existent one. [20]

DP follows the Waterfall principle of differentiating iterations as stages reserved for development, quality assurance and business analysis as some of the larger to mention. SWAP does not experience such phases, all iterations are incremental and technological and business assessment is made weekly to the objectives the team is trying to achieve.

In the DP system, teams are chosen by project, meaning that though expertise is always set to the project at hand, employees can be subjected to different projects, making the process of keeping track of work confusing. SWAP addresses to this by creating teams that work together on the projects handed to them. SWAP teams always have a clear objective, from the grand scheme to the day-to-day operation. As such, development becomes more agile because of the transparent way of development. SWAP offers stories with quick, motivating goals whereas DP, especially in its Waterfall stage, may demotivate the employee due to the time it takes to reach their objective.

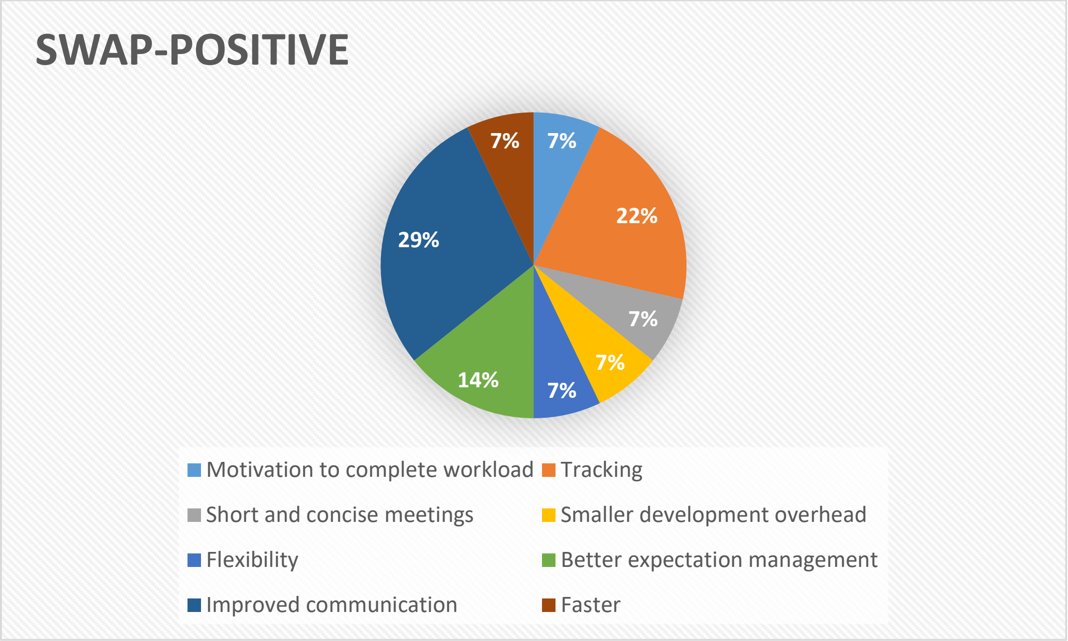
SWAP provides ample resources and moments where the software can be made to fit the needs of the end result and dynamically change to the real life changes around the project, but often carries a larger load of meetings for the development team who are given a larger role in the development of the projects. DP, though more static in its goals sets less requirements on the development team as the project manager and the shareholders are responsible for the planning and organizing of the iterations, giving the development team only the requirement of staying in their deadlines.

3.3 Developers' opinion

The analysis on the developers' opinion is described by the responses gathered in the questionnaire. Each of the aspect mentioned are represented with the amount of times any of the developers have mentioned them.

The results are presented as tables in the extras. Under this part of the analysis, the results will be presented as pie charts describing the amounts of times a single aspect was mentioned.

The positive aspects were next:



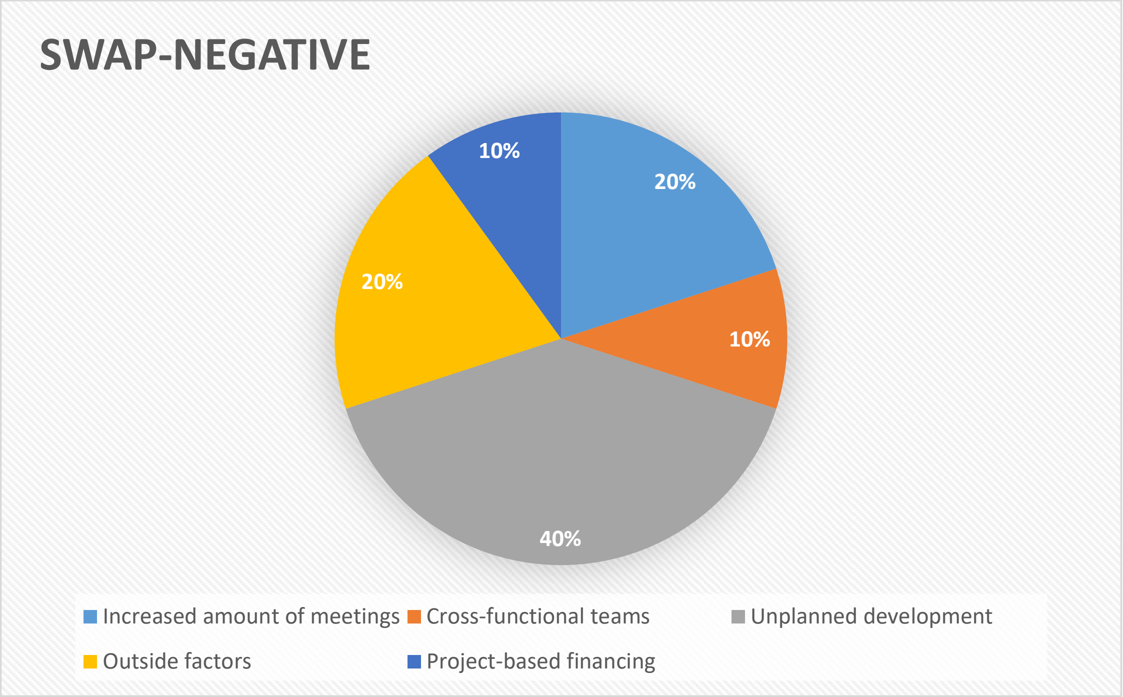
Positive aspects of SWAP

The main 3 aspects that received more mention were the benefits of tracking – the projects seem easier to track because of the storyboarding and the fact that teams take part of the planning directly for themselves.

The developers also praised the improvement in communication – because of the way the teams are set up, all parts that bear responsibility in a project are always represented in the same relative, physical area, meaning that communication does not need to take way by e-mails or other slower methods of communication because the system negates it. This was also the single point all developers mentioned in their responses.

One of the more mentioned questions was expectation management. The developers seem to find the process to be better at communicating with the requesting party of the system – the business analysts and the product owner than it was in the earlier process.

The negative aspects of SWAP were as followed.



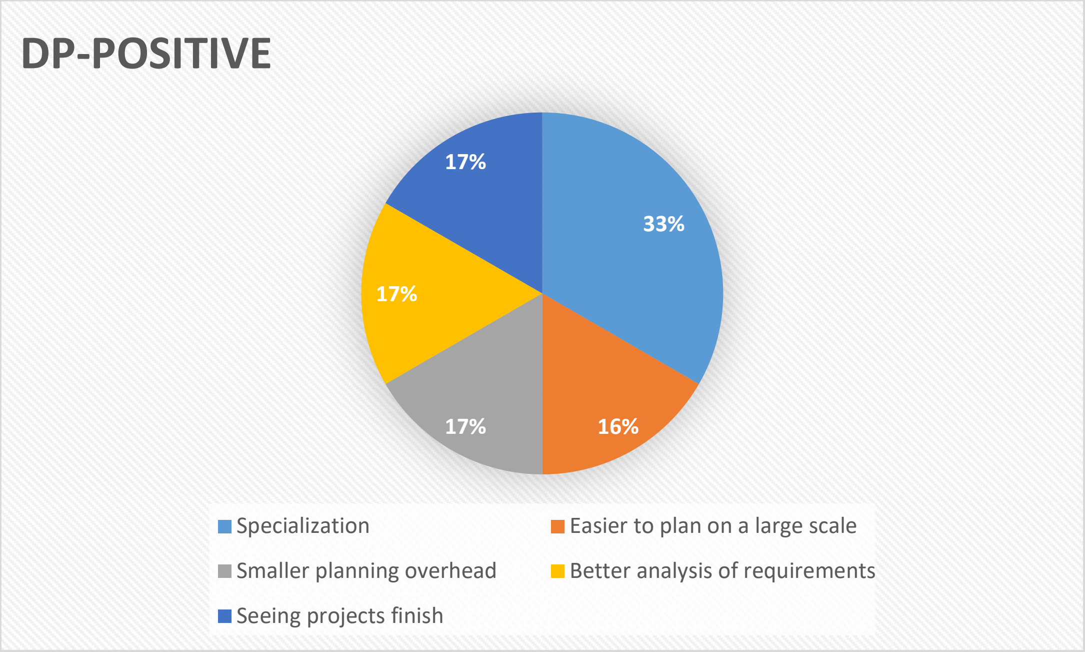
Negative aspects of SWAP

The negative aspects part of the questionnaire received typically less answers than the positives, but more consistent at the same time. SWAP was seen as more time consuming in the meeting room. SWAP requires at least 2 biweekly meetings, a weekly meeting and a 15-minute daily meeting. The meeting time in DP depended on the projects and typically had less meeting requirements for the developer.

SWAP seems to present itself with the limitation of having less business analysis than the process before, resulting in more development that seemingly have not had as much analysis as before, but still require swift work towards release. As the relationship between business analysis and project presentation has grown thinner, some developments can come in as seemingly unplanned.

Outside factors and project-based financing were mentioned as strains. Due to the method of budgeting, the teams have no clear budget over themselves, but only by project. This has been addressed in the documentation and is set to change, yet at this moment, presents itself as cumbersome to the team. The outside factors can be explained as other teams not following the process, hindering the possibility of making a truly agile environment.

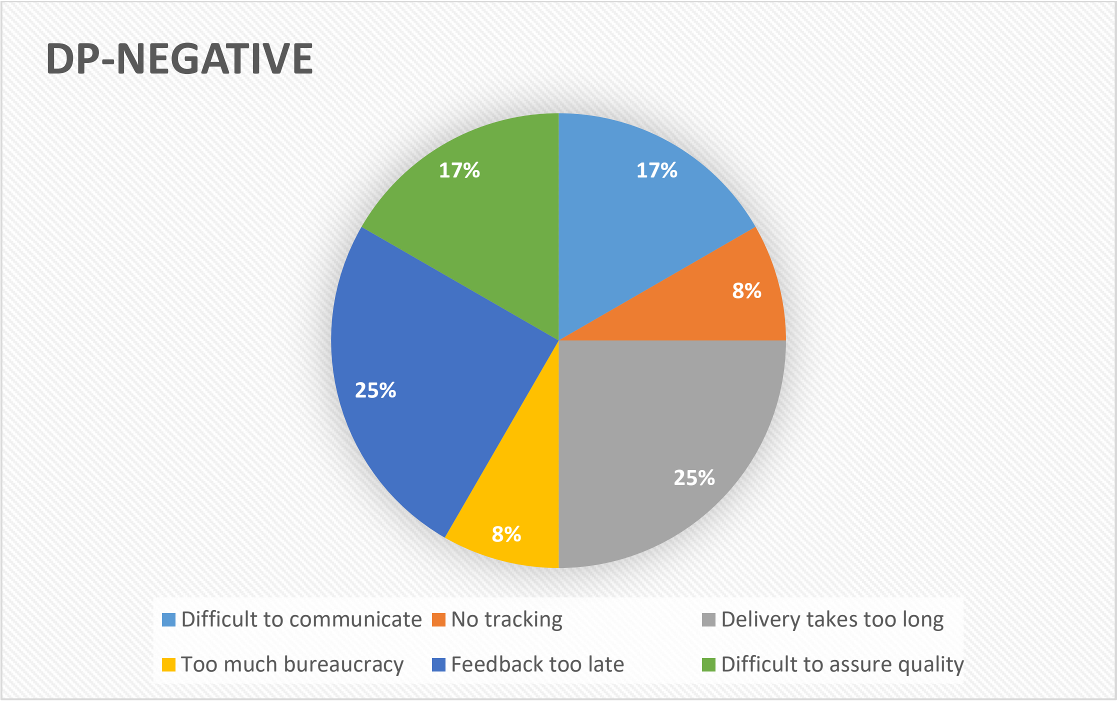
With the responses toward SWAP answered, the questionnaire turned towards DP. The positive aspects of DP are described.



Positive aspects of DP

The most mentioned article in praise of DP is specialization. Developers felt that in the DP process, their work was more towards a single environment or solution, meaning greater expertise in a single area of development. The aspects of business analysis and planning were also mentioned. The aspect of architecting larger systems being easier is one of the more unforeseen aspects mentioned. In the developers' opinion large systems are easier to architect and develop in the DP process than SWAP. Because of the longer time frame of projects, it is possible that DP allowed more architectural analysis to the systems developed, while negating the incremental nature preferred today to the slower method producing more details.

The negative aspects of DP were found to have more common answer.



Negative aspects of DP

The most common answers were with time. Both project delivery being too slow and feedback being too late both indicate the disconnect between teams and the longer planning process where developers had little input. The answers also point towards issues with communication – developers felt disconnected from the product. Communication issues were apparent – both answers about feedback being too late, difficulties to assure quality and even directly mentioned difficulty to communicate were common answers, outlining the largest negative aspect of DP – lack of communication.

SWAP gained the most different positive aspects with 8 separate instances that could not have been grouped together, where communication was mentioned by all. DP had more negative aspects defined than SWAP, though most of the negative aspects of DP revolved around communication.

On both results, communication was the most common argument used. The agile development process succeeds at bringing people together as teams. Due to the nature of solving objectives and finishing projects are required to be done together, everyone has a better view of the common subject, allowing all parts of the team to develop more towards a solution, rather than waiting for the different phases as it was in the old process. The immediate connect between parties seems to have brought a positive change to the development cycle.

Neither of the processes seem to be perfect. The result of the questionnaire produces few unanimously accepted aspect, yet one of them was found in the negative aspects of SWAP. Because the incremental nature of SWAP, business analysis seems to have grown smaller as more developments seem to be produced at minimal notice or outside of sprint planning. As these objectives appear, the developers find themselves producing results they did not agree to do in the planning, producing the most negative aspect of SWAP.

The least answers were collected for DP. All developers did not find 5 separate positive aspects for the process and the aspects collected had the least connect. The largest pointed out was specialization and seems to be the only more common argument towards positives for the process.

In conclusion, SWAP seems to have better acceptance than its predecessor did, which on its' own merits is a good sign – the developers have, overall, accepted SWAP and see it as a software development process with more positive effects than negative and a process with overall more positive and less negative aspects than its predecessor.

3.4 Comparison to Scrum

In this comparison, the system is compared by setting the predicate that the team has selected a Scrum style of sprint planning in the process of the framework selection suggested by the SWAP documentation [1]. Meaning that even though the documentation has its own overall process of getting a product to release, the overall framework will rely on the steps and planning material common to Scrum. The predicate is proven through the use of SWAP in the team and the yearlong experience of practice.

A SWAP team consists of developers, testers, business analysts, an agile team master called the scrum master and a manager. SWAP encourages team members to share roles where possible – business analysts take part of the testing process, focusing mostly on business values and rules being filled correctly while testers are in charge of confirming technical quality assurance. Developers are often suggested to participate in testing.

Scrum teams also use the assumption of shared disciplines. Roles are often seen as shared with the same goal in mind, with accountability and skills being balanced in the scrum team.

The first point where Scrum and SWAP differ is in the team size. Scrum teams are typically suggested to be 7 +/- 2 people [12]. The SWAP teams can grow considerably larger, with both teams under view currently holding more than 10 people in either teams. This mostly grows from the workload necessary for the organization where the requirements for specialization and amount of projects would make bringing any business value to the projects considerably more difficult.

When comparing the workload of a standard sprint, differences become obvious – a scrum team specializes on a single product until the development is finished, before starting work on new business value through products. SWAP teams constantly hold more than one project at a single time, with different parts of the whole system developed on often independently. It is common for a SWAP team to be internally split on at least two different systems concerning different projects.

The way both process' backlogs are formed are identical. In both systems, backlogs are prioritized by the product owners. Sprint planning is done by the team at the start of each sprint, where the sprint length and velocity are also determined. The main difference at this part comes at the end of the sprint. Where Scrum typically requires a demo meeting, a meeting where the developments are demonstrated to the product owners, a SWAP team often has the product owner testing the development throughout the sprint. Since the client is always the product owner to the development team, the demo meeting is cut from the SWAP team model. In the documentation, a process for a demonstration meeting is described, but real life observation has concluded the demo meeting to be a missing part from the SWAP system. At the end of every sprint a retro meeting is held, concluding the sprint and providing the team insight over the completion process with the positive and negative aspects of every sprint brought out. Instead, SWAP teams hold two refinement meetings, where stories are presented by the product owner(s) and allowed to be refined by declaring Story Points to the stories by employing the Scrum Planning Poker.

SWAP stories are completely different from Scrum user stories. The suggested practice of a Scrum story requires the role of the receiver of business value to be described with the synopsis of the story. In SWAP, stories are described by accurate and short descriptions of the story through a concise title, with most of the content described in the description of the story. SWAP stories define the terms of completion in the story loosely, often a story is declared finished when the requested development, often presented in standalone meetings or during refinement meetings, are tested by the product owners or testers. There are no official or standard terms of completion as these requirements are allowed to grow naturally in the development process and following the story description. SWAP stories are often suggested to be made only for technical tasks.

One of the largest points where SWAP and Scrum diverge in are the way releases are planned. Since Swedbank employs many teams with different development frameworks employed, the releases are not designed in the Scrum way of release planning by setting the individual team release date or plan. Every bank release is released monthly with a development and testing period described earlier. All SWAP sprint planning is heavily dependent on this type of release planning because of the availability of environments to test in and schedules set for the products. Every team is required to propose changes based on the projects they are developing for to the release. While team velocity is important to the SWAP team to measure sprint performance and guiding the planning process, it is not important for release planning as releases are handled by a separate release team with the purpose of release planning, testing and feedback.

As a result, SWAP and Scrum are different on all levels though share multiple similarities that makes SWAP seem alike to the Scrum way of development. With larger team sizes to accommodate the large amount of ongoing projects, changes in meetings and meeting types to the way the backlog is organized, one can certainly find flavors of Scrum embedded into the process, but the overall way of release planning, team composition and product ownership methods differ from Scrum completely, making the system employed at Swedbank unique.

3.5 Comparison to Repeatable CMMI

SWAP is an architected system with incremental iterations and cross-functional teams with a commonly decided framework of backlog processing, though the main steps are always common. The way the development process is carried through is heavily documented with room for team decision.

The teams can quickly adapt to new objectives set in the project and have power over the objectives filled in their time boxed iterations, the length of which is decided inside the team. The teams are cross-functional, with architectural, development, testing and business capabilities.

The developments of the team are in line with the general release system that surrounds them, including many different projects from different teams, requiring continuous integration and general testing to assure the quality of the whole system while not excessively disrupting the developments of the teams that declare their work ready for releases.

The teams are managed inside with the Scrum Master being a team member. While the teams aren't defined by clear leaders, teams still have a manager with the objective to oversee the teams' actions and carry responsibility. The manager ensures that the team is capable of hitting the deadlines presented by the release system. The duties of handling resource allocation are in the realm of the manager.

Team members are capable of spreading their capability. Developers are allowed and suggested to take part of the testing process, as are the business members. Testers are allowed to develop and are actively helping with continuous integration in the environments set up to allow the progress of the whole system, making the integration of new products to the system streamlined and documented.

Teams are able to co-operate with each other, employing shared meetings between teams working on the same project. As such, shared responsibilities are able to be addressed with ease.

Observations conclude that dealing with the workload presented to the teams are, unlike to Scrum, often from multiple projects, meaning the teams can handle the workload of projects without the need to organize separate teams based on projects, meaning more elasticity to the team's ability to perform. The velocity of each iteration, as gathered through the Story Points assigned to each objective, also known in SWAP as a Story, is constant with the average velocity easily assumed. The meeting process is documented and accomplished, every team member is aware of their objectives at all times through the planning environments used.

For a process to be a level 2 CMMI, it has to fill 7 key process areas: Requirements Management, Project Planning, Project Monitoring and Control, Measurement and Analysis, Process and Product Quality Assurance, Configuration Management and Supplier Agreement Management.

In SWAP, projects are planned by user stories, sets of objectives and tasks refined through the use of “Planning Poker”, where the tasks are presented by the product owner for the team to decide the difficulty of the objective through story points. The objectives are assembled into sprints, the length of which is decided by the team. As such, the requirements are managed. There are issues with requirements management, as indicated by a questionnaire filled by developers using the system, with objectives in the story sometimes coming up unplanned.

Project planning is done through the use of visual environments with Atlassian JIRA used by the teams under observation. The project planning process has 2 biweekly meetings, a weekly meeting and a daily 15-minute meeting dedicated to it. This process gives a clear oversight of the current status to all people involved and provides possibilities to re-evaluate the difficulty of objectives.

As Project Planning involves the whole team, there is also a factor of monitoring and having clear overview. To achieve this, Daily meetings are implemented into the system. Meetings where every team member gives an overview of what they had done the day before, what they will do that day and optionally what they could do to make work easier. All tools for both monitoring and controlling the development process therefore exist. Though these solutions exist and are used, having control over the iterations is a shortcoming as factors outside of project planning are often brought into the sphere of interest due to outside factors, production bugs or solutions due to a need to accord to local law changes.

Measurement and analysis in SWAP is solved partly. As data is collected and feedback given, the process is not all inherently clear on the methods used. A release management team handles the release management process with its own guidelines and feedback. The solution is not always clear, the team members get overview on their progress on the release, but no specific, standard measures can be found in documentation.

Quality Assurance on the product level is solved inside the cross-functional teams for immediate quality assurance, with the main objectives to determine the product viability and technical solution. Another part of the cross-functional team also tests accordance to business rules. After the objective has been tested by the team, all solutions ready for release are tested by the release management team in testing periods, where all teams also take part in, most notably by regression testing. As such, one can determine a specific SQA element and a testing process also inside the original team.

Configuration management is the process of gaining absolute control over the surrounding system code is implemented into. Especially important in an agile environment, a process with good configuration management must have multiple environments for developing and testing, automated scripts for building, frequent builds and multiple baselines. The Swedbank supported systems are all built using automated building scripts, multiple builds can be requested daily, multiple continuous integration processes are constantly in use with production-like environments supported also.

The last key process area for a repeatable process is supplier agreement management. This is a process established for the acquisition of products or services. The company uses third party services in the way of outside consultants working in the teams. Product acquisition is solved on a case-by-case method due to the relative lack of a need for doing many acquisitions. For the use of services, agreements exist between parties. The acquisition of products is not documented in the SWAP documentation.

In the table below are the KPA-s a model has to reach to be equivalent to a level 2 CMMI model. Parts that are filled completely are marked with “++”, parts that are filled partly are marked as “+” and parts not filled are marked with “-”.

SWAP -"Repeatable" requirements	Status
Requirements Management	+
Project Planning	++
Project Monitoring and Control	+
Measurement and Analysis	+
Process and Product Quality Assurance	++
Configuration Management	++
Supplier Agreement Management	-

Table of process areas of a level 2 CMMI
+ for existing, ++ for completely filled.

Though SWAP fulfills the requirements of a level 2 system, level 3 becomes difficult to achieve. A defined system implements risk management, does not explain a technical solution nor has any implementation for decision analysis. To compare a level 3 CMMI to SWAP would see requirements development filled, product integration and technical solutions filled completely, but find issue with many of the parts that define a Defined CMMI model.

In conclusion, SWAP can be compared against a level 2 CMMI model, where the only short point of the process lies in the supplier agreement management section, which, while existing, is not as well documented to be put forth as analysis.

4.Conclusion

SWAP is an agile development process with leniency in the choice of team framework, but with specific steps in its realization with inspiration from other agile development processes, mainly with Scrum being inspiration drawn from

SWAP does not utilize the release planning presented in the Scrum processes, but utilizes its own instruments of release with a separate team controlling the process, asserting guidelines the agile teams must follow, making the release process and optimized and streamlined process.

With the release process aspect and the overall agile way of thinking employed, SWAP presents itself as a mature system, comparing to a level 2 CMMI.

SWAP is a process inherently more developed system than the methods used earlier. In comparison with DP, SWAP is a more mature system with more positive feedback from the development team, a better planning process and better efficiency.

6.References

[1] – Swedbank , SWAP v1.1

[2]- Agile manifesto, www.agilemanifesto.com

[3] – Lean thinking, Womack, James P., Daniel, T. Jones (1996)

[4] – CMMI, for Development, CMMI Product Team, Carnegie Mellon University version 1.3, (2010)

[5] - Using CMMI together with agile software development: A systematic review , Fernando Selleri Silva, F.S.F. Soares, A.L Peres, I.M de Azevedo, AP.L.F Vasconcelos, F.K. Kamei, S.R.de Lemos Meira, Information and Software Technology 58 (2015)

[6] – https://www.scrumalliance.org/why-scrum

[7] – The Scrum Primer p.Deener,G.Benefield, C.Larman, B.Vodde http://www.scrumprimer.org/scrumprimer20.pdf

[8] - VersionOne surveys Agile development for 2015 http://searchsoftwarequality.techtarget.com/photostory/4500251249/VersionOne-surveys-Agile-development-for-2015/5/How-popular-is-the-Scrum-process

[9] – M.Beedle,M.Devos, Y.Sharon,K.Schwaber, J.Sutherland, "Scrum: A Pattern Language for Hyperproductive Software Development"

[10] – Pankade turuosad, Estonian Banking Association, http://www.pangaliit.ee/et/kasulik/pangandusinfo

[11] – "Eesti Ihaldusväärseim tööpakkuja 2016", CV Keskus, https://www.cvkeskus.ee/career.php?menu=2&lastmenu=1439&text_id=2052&career_style=2

[12] -Scrum Roles and Teams, International Scrum Institute, http://www.scrum-institute.org/Scrum_Roles_The_Scrum_Team.php

[13]-Development Process overview, Swedbank

[14] - Paulk, Mark C.; Weber, Charles V; Curtis, Bill; Chrissis, Mary Beth (February 1993). "Capability Maturity Model for Software (Version 1.1)" . Technical Report (Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University)

[15] – Z.Bougroun,A.Zeaaraoui,T.Bouchentouf,"The projection of specific practices of the third level of CMMI model in agile methods: Scrum XP and Kanban," 2014 Third IEEE International Colloquium in Information Science and Technology (CIST)

[16] – K.Sägesser,B.Joseph,R.Grau,"Introducing an Iterative Life-Cycle Model at Credit Suisse IT Switzerland " IEE Software (Volume 30, Issue 2)

[17] – G.Philip Rogers,P.Miles “Manual testing's newfound place in the automated testing world “ IEE AUTOTESTCON 2015

[18] – M.Najmi,M.Sepehri,S.Hashemi, “The Evaluation of Business Intelligence Maturity Level in Iranian Banking Industry” Industrial Engineering and Engineering Management (IE&EM), 2010 IEEE 17Th International Conference

[19] – I.Stojanov,O.Turetken, Jos J.M Trienekens, “A Maturity Model for Scaling Agile Development”

[20] – K.Sägesser, B.Joseph, R.Grau, “Introducing an Iterative Life-Cycle Model at Credit Suisse IT Switzerland”, IEEE Software (Volume:30 , Issue: 2)

[21] - Mark C. Paulk, “Extreme Programming from a CMM Perspective”, IEEE Software (Volume: 18 , Issue: 6)

Extras

[1] – Example of a response gathered in the questionnaire

1 year	2 years	Negative: *not fully agile (business, projects, financing) *too different environment for pure agile (there's no one concrete topis to dealt with, many different things, dependencies, releases) Positive: *more visible workflow *better planning (almost no overtime after going waterfall -> agile) *easier testing (can see what's happening and test smaller parts) *MVP - we can challenge to develop not-used-features	Negative: *testing was difficult - hard to kepp track (at the end too big scope at once) *too late feedback due to lifecycle *planning was bad - hard to estimate and release Positive: *was possible to focus on one thing/topic (no content switching so much) *projects were finished (now we are finishing only some parts of many projects and they are hanging in our board quite long time)

[2] – results of questionnaire presented towards Swedbank developers in teams Loans Origination and Loans Core.

SWAP - Positive aspects	Times mentioned
Motivation to complete workload	1
Tracking	3
Short and concise meetings	1
Smaller development overhead	1
Flexibility	1
Better expectation management	2
Improved communication	4
Faster	1

SWAP - Negative aspects	Times mentioned
Increased amount of meetings	2
Cross-functional teams	1
Unplanned development	4
Outside factors	2
Project-based financing	1

DP - Positive aspects	Times mentioned
Specialization	2
Easier to plan on a large scale	1
Smaller planning overhead	1
Better analysis of requirements	1
Seeing projects finish	1

DP - Negative aspects	Times mentioned
Difficult to communicate	2
No tracking	1
Delivery takes too long	3
Too much bureaucracy	1
Feedback too late	3
Difficult to assure quality	2