G

- SPI practices lack detail, categorisation poorly motivated, discusses applicability to 1st WS
- Improvement steps disconnected from goals
- 4 papers referenced

Final Report

ANONYMOUS AUTHOR
Examination in DIT347 Software Development Methodologies
Bachelor Program Software Engineering and Management
Department of Computer Science and Engineering
University of Gothenburg
Gothenburg, Sweden

In this report I will introduce and discuss about Software Processes and Software Process Improvement (SPI). I will also discuss how I have used these within the course of DIT347 Software Development Methodologies taught at the University of Gothenburg. While I've been in contact with some of these practices before, this course has had a more in-depth assessment with a lot of practical teaching moments performed in smaller groups.

I. Introduction

Software Process and Software Process Improvement (SPI) are two common methodologies used in software development. The main goal of these is to help and improve software development as a whole, always having the customer value in mind. Software Process can be explained as a set of activities that together, shows and helps the development of a software from start to finish. These activities can be anything from the actual work itself to an interacting with the stakeholder, e.g., "Collaborate together to produce good quality code that meet the requirements" or "Build a system by implementing, testing and integrating one or more system elements. This includes bug fixing and unit testing" shown in alpha cards from our first lecture [1].

Software Process Improvement (SPI) is on the other hand about improving the efficiency and quality of the software development. These are also presented as tasks and techniques that can be implemented in any software development.

These methodologies we're introduced and taught to us through mostly group activities, with the focus being to simulate a real-life organization by having two major Workshops. Inside these Workshops and outside of class, we got to prepare and use the tools of Software Process and Software Process Improvement (SPI) to see the benefits at first hand.

II. Process Applied in the Scrum Workshop

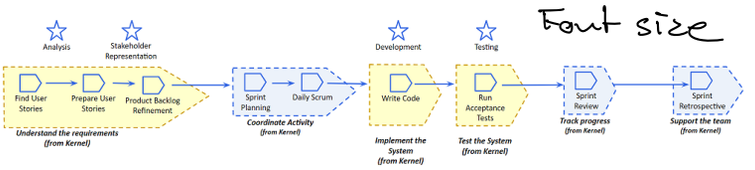
The process we defined and used for the Scrum Workshop was the Scrum practice combined with a User Story practice. We chose this approach because the Workshop stated that it would be performed in sprints with user stories already in place, making a Scrum and user story practice very applicable.

The Scrum approach also benefits an iterative approach and breaking down into smaller teams (three to nine) where one Scrum team focuses on one derived product together [2]. Since the class already consisted of smaller teams of six and user stories already being in place, it made sense choosing the Scrum practice above the other ones.

When choosing which practice to combine with the Scrum practice, it was between the user story or the incremental delivery practice. As aforementioned, having the user stories already derived, we choose the user story practice instead of incremental delivery.

The following diagram shows our process plan with these two practices combined in activities [3].

Font size



Our defined process plan consisted of first *understanding the requirement* by finding and preparing a user story followed by refining the product backlog. This would then lead onto the *coordinate activity* where the sprint planning and daily Scrum would partake. After this, the actual *implementing of the system* would begin by having the team working on the product backlog items. This also includes testing the system and backlog items implemented. Each sprint would round off by *tracking the progress* through a sprint review and lastly give the *team support* by having a sprint retrospective.

For our process plan our team roles were the Scrum master, resource gatherer and builder.

We applied this to the Workshop by planning to have three sprints during the three-hour Workshop session, making each sprint around one hour each. On the team level, each Scrum team would have a Scrum master which would be responsible for attending the Scrum of Scrums together with the product owners and Scrum masters from the other teams.

During the Scrum of Scrums, the Scrum master would collect team backlog and then give it to his/ hers Scrum team. With this backlog, the Scrum team would then conduct the work to fulfill the requirements needed by building, testing and delivering each backlog item. The Scrum master, who still helped with the Scrum team work, also helped by coaching the team and having an open communication with the other Scrum teams and product owners.

How?

After this, the Scrum team would perform a sprint review together with the responsible product owner. Here, depending on the requirements being fulfilled or not, the team would either pass on a backlog item as delivered or needing more work on.

Finally, each sprint would end with a retrospective. Here the Scrum team would together go through what had went well, what could have been done better and things to keep in mind for the next sprint.

While our defined process plan and what we actually applied in the Workshop where very similar on paper, the first schedule

of the workshop didn't really live up to the process plan we had first drafted. This ultimately came down to us being inexperienced and underestimating some the Scrum steps and how much time they actually required. This further resulted in reduced work efficiency and spiraled into having less time for next sprints and even having to skip some sprint reviews and/ or retrospectives.

We also skipped most of the *understanding the requirements* activities since all of the user stories had already been drafted prior to the workshop. This didn't affect our team to much and all-in-all even helped in saving some time.

III. Experiences in the Scrum Workshop

To put it mildly, the very first Scrum Workshop did not go as well as planned. Before the Workshop had even started; the Scrum teams were supposed to draft a schedule and Definition of Done together. With having less than half the Scrum teams participating in the creation of these, the Workshop had a very shaky start with some teams not even knowing or/ and had not given their opinion on these drafted documents.

For this Workshop I was assigned as the teams Scrum master and for our part, were involved in the mentioned discussions and creations of these documents. Due to this, it luckily didn't affect us too much on our team level, but it still created a knowledge gap between the different Scrum teams.

At the start of the first sprint, I together with the rest of the Scrum masters performed the Scrum of Scrums together with the product owners. Since it was the very first sprint of the Workshop, we had a bit of a slow start with how to choose and divide the different user stories. We finally managed to decide to go with a pick-and-choose approach for this. At first glance, this wasn't much of a problem but would later on create a split of work between the product owners where for some sprints, all of the user stories being worked on had the same product owner and would cause overburden on him/ her and further set us back in time.

When it came to the actual work part, it went very well for our team. We were one of the few teams that managed to not only finish one user story per sprint, but also communicate well between the Scrum teams and with the program level. This was mostly thanks to most of us already being experienced with *Minecraft* and not having to learn how to actually use the tools that were given to us.

I also think that we were in general very well prepared and everyone in our team participated and gave their all.

For each sprint review, we changed and improved something every time. For the first one, we had all teams gather, listen and conduct the review together. This turned out to be a big time sink and taking up to thirty minutes instead of the planned ten.

As mentioned in section II, this was both due to us being inexperienced and underestimating how much time it would actually require. It also spiraled with having less time for the sprints after.

For the second sprint, we instead had the assigned product owner conduct a personal sprint review together with each Scrum team. This turned out to be a bit better than the first sprint review in both time and work efficiency, but still took more time than planned. And since prior to the second sprint we'd decided to always stick to the schedule (even if some teams had not had their sprint review done yet), it resulted in some teams having to miss their sprint review.

For the third sprint we decided to shorten the work time greatly and increase the sprint review time. We also combined the sprint review step with the sprint retrospective, so if a team had to wait for their product owner for a sprint review, they could save some time and do their sprint retrospective meanwhile.

What we can conclude from the mentioned issues above is that we had to spend a lot of time readjusting the schedule. For each sprint, we managed to improve but at the cost of shortening each sprint. I personally see this first Workshop as a learn-by-failing where my main takeaways from it is the importance of planning, communicating (between teams and to the program level) and sticking to the schedule.

The issue with communication between teams that we encountered during the Workshop is a common issue that can also be found in and compared with similar low-level companies. In a paper released in 2003, they discuss issues with internal communication that arise between low and high maturity level companies [4]. This can be compared to our class being at a low maturity level, which where one of our most dominant issues was the lack of communication.

IV. Software Process Improvement Techniques

There are several known Software Process Improvement (SPI) methods, models and techniques that can be used. While these differ in some way with some advantages and/ or disadvantages, discussed in a paper from 2008, most of them cyclic with four main steps: an evaluation of the current practices, planning for improvements, implementation of the improvements and an evaluation of the effects of the improvements [5].

These SPIs can further be categorized into two different main ones, a prescriptive (top-down) and inductive (bottom-up). The inductive methods focus more on step one and four, where we evaluate the current practices and the improvement of the effects, while instead the prescriptive models instead focus on step two and three, where we use concrete practices to plan for the improvement and then implement them.

As we can see, using an inductive framework combined with a prescriptive framework we manage to encapsulate the cycle and use all four major steps of the SPI. Therefore, using these two together is a powerful way of doing improvements as a whole.

Some SPI examples we were taught during the lectures are [6]:

QIP (Quality Improvement Paradigm). It's an inductive framework which starts of by understanding the current situation and identify the most critical issues for the specified organization. Its goal is to help organizations identify their most critical issues and needs.

iFLAP (Improvement framework utilizing lightweight assessment and improvement planning). It's an inductive framework and a lightweight version of QIP with similar goals but instead uses multiple data sources to asses needs for improvement and then triangulates them to confirm its issues.

CMMI (Capability Maturity Model Integration). It's a prescriptive framework that collects and defines process areas based on a collection of best practices. While it's one of the most

used frameworks and can provide a gold standard that the organization can compare and make changes with, it doesn't consider the organizations needs and can be too heavy for some to implement.

SPICE (Software Process Improvement and Capability Determination). It's a prescriptive framework that contains a generic process model that needs refinement for a specific domain. It's advantage above CMMI is that it being domain focused, considers issues specific to that domain which could otherwise be missed.

With these differences in mind, one could compare these four examples of SPI frameworks and place them into the following categories shown in the picture.

Why a pyramid? Source?

How do you arrive at that mapping?

Where QIP would land under the *understand,* iFLAP under *plan*, SPICE under *implement* and CMMI under *check*.

Not the point

For our first Workshop, having a better understanding on our major issues could have helped greatly to counter some of the ones we encountered, mainly communication. Implementing and using something like QIP or iFLAP could have early on shown this and given us time to fix it sooner. I can also see how implementing and using a specific SPICE framework for our domain could have even generated some process models for us to use to maybe not even have some of the issues we had at all. Although CMMI being a gold standard with a collection of best practices, I think it and maybe even SPICE could have been too burdensome for us to implement because they don't take into consideration smaller organizations and their limit to time and recourses.

V. SPI IN INDUSTRY

Some Software Process Improvements (SPI) in industry's that we got in contact with was with 1928 diagnostics and Siemens. These companies both had a guest lecture where they discussed about their take on Software Process and Software Process Improvements (SPI) and how they implemented and use it daily within their organization [7].

At 1928 Diagnostics, their process is combined by an Agile/ Scrum work-way together with a Quality Management System (QMS). By applying the QMS, they first measure performance through feedback from the end user together with internal data gathering from revisions and reviews. This would be their sprint refinement step which will also provide requirements for future sprints. After this, a sprint planning is held using the QMS models supporting and management processes. This includes an internal revision together with reviews of managements for documents, changes and risks.

With these steps complete, they continue improving and implementing the software by applying the Agile/ Scrum technique. Once the sprint development is complete, it is demoed and a new round of review and feedback processes will start. With this new information collected, the model can be adjusted for an improved one for the next cycle.

Since 1928 Diagnostics is involved in sensitive medical devices, they also were bound to have a strict regulation regarding their processes. This is because outside factors and requirements such as laws and FDAs limited on how and what processes and improvements can be implemented. This can be related with similar issues we encountered for our Workshops. By having assignments and a grade outside of these Workshops, we also had to keep in mind of these factors and requirements. It not being as dramatic as laws and FDAs, we still had to adhere to these and think of them when doing our planning and what we could apply.

?

At Siemens, before making changes to their Processes and SPI, they used a waterfall process for their development. While it did provide them with a good process for developing software with getting things done at the right time, it overlooked the focus of completing the correct things instead. Since it being a waterfall process, it also didn't allow any changes to be made during the process [8], which limited any improvements they might have wanted to implement, such as having a stakeholder wanting to change a requirement. Due to this, they adapted more incremental model, consisting of multiple waterfall iterations. While it allowed them to be more flexible, it still didn't manage to solve all the problems that comes with a waterfall process.

To tackle this, the organization made a drastic change and switched to an upside-down agile process. With this new process, instead of having the CEO at the top of the hierarchical structure with developers supporting upwards, the CEO instead is at the bottom, supporting the developers. By doing this, the focus shifts an agile culture with collaboration and creation at the top.

This was mostly implemented through a so called S4 tier of Scrum meetings. Starting at S1, the team discusses if any help is needed, which then is passed up all the way to S4, the Scrum of Scrums. Here they bring up the progression of the teams and products. In case any team needs extra support, they can be provided with it with a more direct support from the lower levels inside the organization.

This was of solving their hierarchy issue closely relates to how we solved some of our issues regarding some of the bigger user stories during the Workshop. Since some of the user stories involved many steps and almost a continues integration in the sense of adding more roads and lampposts to every building created, we also solved this by bringing it up on our Scrum of Scrums meeting where it could be addressed and have another team sent to support them.

VI. SPI PROPOSAL FOR FUTURE SCRUM DEVELOPMENT EFFORTS

Our SPI proposal consists of three concrete improvement goals. These are accompanied with the Project Planning (PP) and Requirements Managements (REQM) process areas [9] written in concrete steps in one of our assignments [10]:

PP will help to improve the efficiency and scheduling of the sprint reviews that appeared as an issue in the first Scrum

Why not start with the goals?
-> disconnect between improvement goals and concrete steps

Workshop. Relating to SP2.2, identify and analyze project risks; and the collaborative planning agile practice, we will be establishing a directive that any planning affecting the schedule shall be discussed in the scrum master communication channel. This will be done with the continuous improvement and the reflect and tune the process agile practices, where during the workshop scrum of scrums meetings after any and all issues pertaining to the process have been addressed during the individual retrospectives. This is addressed in SP2.4 identify and analyze project risks, should any of these issues go beyond the development team level, the scrum master shall elevate the issue to the product owner(s) through the dedicated communication channel, as they shall be included during the development processes.

REQM pertains to increasing the knowledge of the requirements from the team's viewpoint. During the sprint planning phase, after the team's scrum master has selected which sprint task our team shall be assigned to, we as a team will discuss the requirements of the task to ensure clear understanding throughout and, as described in SP1.2, that a commitment exists between the requirements and the project participants. Should there be any uncertainty regarding the requirements we will be utilizing the planning at different levels agile practice to communicate with the product owner(s) to gain clarification of the specific requirement(s), and simultaneously addressing SP1.1 where we are developing an understanding of the meaning of the requirements with the providers. Before the sprint takes place, we shall keep an audit log of which team members will contribute to a particular requirement of the project task, this will be done in a Google Docs or similar application. Whilst the work phase of the sprint has gone underway, any upcoming hindrances specified in SP1.3, managing changes to requirements as they evolve during the project, will be resolved through the collaborative teams and knowledge sharing tools agile practices. We will be using the dedicated communication channel previously explained in the first goal to ensure no issues have been left unattended as well as sharing potentially useful knowledge between our team members as well as other teams' members. Once we achieve the sprint review stage, we want to ensure that project plans and work products remain aligned with the requirements as addressed by SP1.5. This will be done by going through the requirements checklist with the product owner(s) and taking note of any dissatisfactory results provided; using these results to correct the work product during the next sprint.

Our improvement goals are the following written in GQM [11] for one of our assignments [12]:

Goal 1.

Improve the efficiency and scheduling of the sprint reviews from the team's viewpoint.

The purpose of this goal is to make the reviews more efficient as well as improve the scheduling. When conducting the interview, it became apparent that this was one of the parts of the workshop that had the biggest improvement potential.

Purpose: Improve
Issue: the efficiency and scheduling of the
Object: sprint reviews
Viewpoint: From the team's viewpoint

This QGM goal would be answered through the following two questions:

Q1. How much time was spent being idle waiting for the product owner to perform a review

Q2. How many minutes did the review go over the time limit?

With these questions, we want to see how much time was lost spent being idle waiting for conducting a review since this was one of the major issues that appeared in the first Scrum Workshop along with how much actual time the review took to get an understanding if the allocated scheduled time for the reviews was enough.

These questions would be measured in the following metrics:

Q1M1. Amount of time in minutes spent waiting for the product owner to conduct the sprint review, this data is collected from documentation provided by the teams after each sprint.

Q1M2. Amount of time in minutes between the building being shut off and the review being conducted, this data is collected from documentation provided by the teams after each sprint.

Q2M1. Median and standard deviation from the allocated review time in minutes.

Q2M1. Current average time for sprint reviews in minutes / Allocated time for sprint reviews in minutes * 100.

With these metrics, the goal is to get an all-around coverage on how much time spent in minutes and percentage when it came to waiting for or conducting a review.

Goal 2.

Increase the level of communication between the teams from the team's viewpoint.

The purpose of this goal is to increase the communication between the teams; during the interview, there was a strong emphasis on the lack of communication between teams which affected the work being done.

Purpose: Increase
Issue: the level of
Object: communication between the teams
Viewpoint: From the team's viewpoint

This QGM goal would be answered through the following two questions:

Q1. How many channels were frequently monitored by the team members?

Q2. Is the communication visibly improving?

With these questions, we want to know how many communication tools were monitored by the team to see the difference in communication between team and what tools they actually end up using along with if they felt like the communication was continuously improving since there was a lack of it at the first Scrum Workshop.

These questions would be measured in the following metrics:

Q1M1. Number of channels monitored during every new sprint. This data comes from the amount of designated communication channels (Slack, in-game, Zoom, etc.)

Q1M2. How many times during each sprint are the channels monitored. This data comes from interviewing team members about how often they utilized the communication channels during each sprint.

Q2M1. How many participants are communicating in the channels every sprint. This data comes from the monitored communication channels.

Q2M2. A Likert scale of how well the communication was between teams and their product owners. This data comes from interviews with the teams.

These metrics goals are to see what tools are being used, how many team members and teams uses them and how the satisfaction level was when using them.

Goal 3.

Increase the knowledge of the requirements from the viewpoint of the Team.

In order to find out if the team has the understandability needed to start working, we have to find out if the team fully understands the requirements before the work starts. This is one problem that we found during our interviews since some teams spent more than one sprint for some user stories.

Purpose: Increase the
Issue: knowledge of
Object: the requirements
Viewpoint: from the Teams viewpoint

This QGM goal would be answered through the following two questions:
Q1. How many minutes were spent on discussing the requirements of user stories before each sprint?
Q2. How many user stories were accepted by the end of each sprint?

With these questions, we want to figure out how much time a group spends in discussing the requirements of a user story to see if they spend any time to understanding it properly along with how many user stories got accepted to see if any correlation can be done between the two.

These questions would be measured in the following metrics:

Q1M1. Minutes spent on understanding the requirements of user stories. This data comes from each team's record of time (schedule).

Q2M1. Number of completed user stories that were accepted by the product owner after each sprint. This data comes from the teams finished backlog items for each sprint.

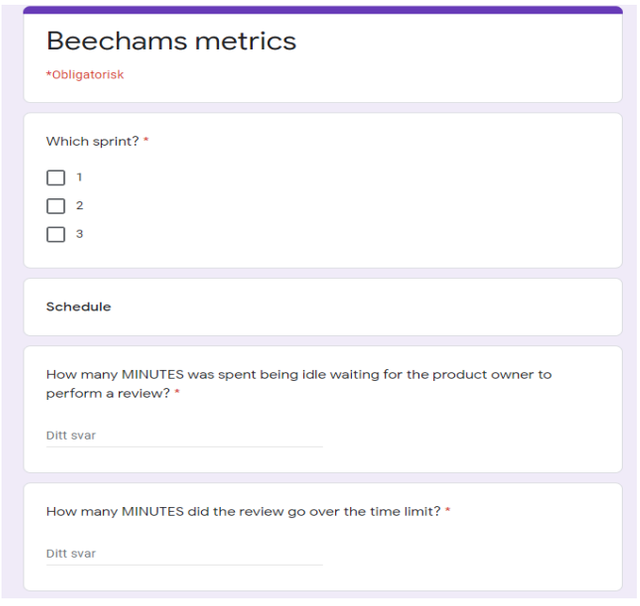
Q1&2M2. Correlation between time in minutes spent on understanding the requirements during the sprint and number of completed user stories that were accepted by the product owner after the sprint.

These metrics goals are to see how much time is spent in understanding the requirements and how many backlog items are completed every sprint. With the last same metric for both questions, we get a clear view if any correlation is made between the first metrics.

VII. Implementation of an SPI Initiative

For the second scrum Workshop, all teams had a more collective knowledge on what didn't work so well in the first Scrum Workshop. It started off by having all Scrum masters participate and create a schedule and Definition of done together. This compared to the first Workshop went much better and since we now had more experience, the schedule was much better done and had more time allocated for the actual sprint review, which had previously been a complete mess. It was also decided that the sprint reviews would be done in groups of product owners, as in having all the teams with a user story that belonged to the same product owner gather together and conduct them in groups. This had two major benefits; first being it was much more time efficient that the two previous methods we had tried. Second it also forced us to divide the user stories properly among teams so the product owner had an evenly split on user stories being worked on. The communication among teams and product owners was also improved. Teams talked much more among themselves and we had a proper communication tool setup for this through slack. With having a more focus on talking with the product owner for this Workshop, there was also a wait-in-line system implemented which meant that you could sign up to talk and ask questions for a product owner. While this system meant that there could be a waiting time, it still provided a much more structured way of handling the communication which in the end helped us. Teams were also much more encouraged to talk and ask about the user stories and their requirements with the product owners to counteract any chance of misinterpretation. This was shown as there were more user stories and a higher quality over-all around the completion of them.

We also had a survey created to measure these improvements. The questions were the same as the QGM in section VI and we shared it through the communication channels and asking other teams to give their feedback. While we didn't get as much feedback as we had wanted, it still showed some interesting results.



Some interesting takeaways were the spread among teams when it came to waiting in minutes to conduct a sprint review. Here, three teams answered 0, one team answered 1, one 10

mins+ and two teams 5. While the 10 mins+ might seem concerning, combined with the second questions of how many minutes the review went over the time limit where's all team answered 0, it becomes clear that even though one or two teams had to wait for their turn during the sprint reviews, it still didn't go over time. This can directly be pointed towards the new improved schedule and sprint review.

Another interesting result is how many communication channels were frequently monitored by the team members. Here, none answer 0 and there was an almost evenly split between 1-4 channels. This together with the second question about if the communication was visibly improving, where there was an agreement with all responses that it was better than the first workshop. Since most teams used the in-game or slack chat, they were somehow always monitoring some communication channel which was a big improvement since the last workshop.

VIII. Summary and Lessons Learned

From combining all of these viewpoints presented in this report, it becomes clear that Software Process and Software Process Improvement (SPI) plays a great part in efficiency and quality assurance when developing. Comparing the first Scrum workshop with the second is almost day and night. While this could also boil down to us having more experience with the game and how things are done inside the Workshops, I still stand my ground in the importance of Software Process and SPI. By not only taking our experiences from these Workshops into consideration, but also looking at the concrete examples like the one Dulce presented during her guest lecture about the transformation of Siemens and their processes, it shows that Software Process and Software Process Improvement (SPI) has their impact. It also comes down to how the organization decides to use these tools that are presented and which one fits the best which organization the best.

If I were to another iteration of SPI, I would have wanted to try to do it with a more focus on the program level. Since most of the implementations and planning we did was on a team level, many of effects and changes I would had wanted to see never got through. Other than that, I'm quite happy with how the second Workshop went and I really got to see the benefits of proper SPI and first hand.

References

[1] Jan-Philipp Steghöfer, "Lecture 1 – Building Blocks of Software Processes", Lecture Slides, DIT347 H20 Software Development Methodologies, Software Engineering and Management Program, University of Gothenburg, 2019

[2] Jan-Philipp Steghöfer, "Lecture 4 – Scrum", Lecture Slides, DIT347 H20 Software Development Methodologies, Software Engineering and Management Program, University of Gothenburg, 2019

[3] [Undisclosed Authors], "Process Plan", Assignment 2 in course DIT347 H20, Software Engineering Program, University of Gothenburg, Sweden, 2020

[4] Beecham, Sarah & Hall, Tracy & Rainer, Austen, "Software Process Improvement Problems in Twelve Software Companies: An Empirical Analysis", 2003

[5] F. Pettersson, M. Ivarsson, T. Gorschek, and P. Öman, "A practitioner's guide to light weight software process assessment and improvement planning" Journal of Systems and Software, 81(6), pp. 972-995, 2008.

[6] Jan-Philipp Steghöfer, "Lecture 6 – Inductive and Prescriptive SPI Methods", Lecture Slides, DIT347 H20 Software Development Methodologies, Software Engineering and Management Program, University of Gothenburg, 2019

[7] [Undisclosed Authors], "Process in the organisational context", Assignment 5 in course DIT347 H20, Software Engineering Program, University of Gothenburg, Sweden, 2020

[8] Kai Petersen, Claes Wohlin, and Dejan baca, "The Waterfall Model in Large-Scale Development", 2009 Bib data

[9] Jan-Philipp Steghöfer, "Lecture 8 – CMMI", Lecture Slides, DIT347 H20 Software Development Methodologies, Software Engineering and Management Program, University of Gothenburg, 2019

[10] [Undisclosed Authors], "Concrete process improvement", Assignment 6 in course DIT347 H20, Software Engineering Program, University of Gothenburg, Sweden, 2020

[11] Victor R. Basili, Gianluigi Caldiera, H. Dieter Rombach, "The Goal Question Metric Approach", 1994 Bib data

[12] [Undisclosed Authors], "Process Analysis and Improvement Goal Formulation", Assignment 4 in course DIT347 H20, Software Engineering Program, University of Gothenburg, Sweden, 2020

[13] Dulce Goncalves, "Guest Lecture – Dulce Goncalves", Lecture Slides, DIT347 H20 Software Development Methodologies, Software Engineering and Management Program, University of Gothenburg, 2019