G

+ Good general desc. of SPI
- No details about SPI techniques
- SPI in industry very short
+ Detailed improvement plan
- No questions
+ Concrete measurements
- Little discussion of effects
+ References & papers...
- ...but no analysis

Starting up SPI

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

Abstract—A class of students develop a process, apply it in a simulated scenario and then try again after extensive SPI practices. The first time around did not go so well, struggles such as lack of communication with the product owner and bad requirement tracking were prevalent. After applying SPI techniques things improved measurably but still showed obvious flaws.

I. Introduction

This report provides a unique view on "forcing" inexperienced bachelor students to develop and apply a home made agile process. This process was applied during 2 workshops where a whole class had to work together to build various things in the block game MineTest. After the first workshop SPI techniques were introduced to make an improvement plan with concrete measurement points to see if things actually improved.

II. Process Applied in the Scrum Workshop

The planned process used was heavily based on the agile SCUM framework, SCRUM introduces concepts such as sprints, sprint planning, sprint retrospective and review, the product and sprint backlog, scrum masters and "daily" scrum. In addition to SCRUM, the processes User stories and Incremental delivery were incorporated.

User stories are a form of task description where a wanted feature is explained in natural language instead of listing systematic formal requirements for example and are often used to make introducing new features to a system more accessible to stakeholders. In our process User stories were used in the SCRUM product back log.

Incremental delivery is a release model where the developed product is released to production in an incomplete but in feature expanding state and is used to elicit feedback faster. In our process after each SCRUM sprint the system will be deployed as is before the sprint retrospective such that stakeholders can give feedback which might influence backlog refinement.

?

review

Abstract in practice.

Team level: Work starts with sprint planning where the SCRUM master and the team will decide what to work on. This planning is based on the user stories contained in the product backlog.

When the team decided on what to work on the respective user story is broken up in concrete requirements and tasks are moved into the team backlog. The SCRUM master then has the responsibility to assign tasks to individuals in the team.

Once the sprint planning finalized the sprint commences and iterative work gets done. At the end of the sprint the working state of the product gets released as per the incremental delivery practice and relevant feedback gets collected. Then the sprint retrospective will conclude the sprint. In the sprint retrospective the whole team will discuss performance, progress, potential requirement issues and other matters that might affect or improve the next sprint and or the product backlog. this cycle repeats until the product completion.

$\rightarrow$ review

relation?

Program level: The program level process is a scaled up version of the team level process, where you see a member being part of a team, you see a team being part of a program. Likewise, sprints consist of 2 to 4 team sprints. Teams are represented by their SCRUM master, at the beginning of a sprint the representatives come together to pick tasks from the program backlog, a scaled up version of the product backlog, for their own team's product backlog and start the team level sprint planning.

Once the sprint ends a program level sprint review occurs where all team representatives discuss the last sprint.

Reasoning

In our case, multiple teams building a city, fast response to requirement changes is very important. For example, the geographical state of the designated build site is not always known, the product owner could their mind as they see the first building starts to take shape or as dependencies start to emerge such as road networks. That is why SCRUM is used as the core of the process, it allows for fast iteration and thus provides ample opportunity to change direction when needed.

?

To exercise that agility, Incremental delivery is used to elicit feedback from stakeholders at the pace of the SCRUM iterations. This feedback will improve the existing requirements[5] and help discover hidden requirements not known before[4].

Role of citations?

To properly process the feedback of stakeholders User stories are used to track tasks in the SCRUM backlog. User stories enable stake holders to express their requirements in a structured but high level manner such that deep knowledge about the system is not required.

Reality in practice

In reality the process explained above did not come to fruition, it did not get the chance to.

The process we ended up using during the workshop could be described as a semi structured replication of the SCRUM process in the sense that our process included repeated key elements that make up SCRUM, but there was no consistency.

How is that different from plan?

On a team level, the process mainly worked on the idea of sprints, at the beginning of these sprints the team stops with whatever they are doing discuss what has to be done in the upcoming sprint. The SCRUM master, the appointed leader, then assigns tasks to each team member for that particular sprint in our case, three team members were assigned to collect resources(miners) and two other team members were builders (who are responsible for building the structures). Once the sprint deadline is hit the team does a sprint retrospective to discuss progress and potential issues. The sprint retrospective is directly followed by the sprint planning, completing the sprint cycle.

On the program level, the process deviated differently from the SCRUM process. At the beginning of each sprint, all scrum masters from the respective teams came together to select a user story their team wanted to handle, with emphasis ”on their team wanted”, there took very little planning on the program level took place. While occasional non planned meetings occurred to collaborate, this wasn't planned and officially no structured meetings beyond this user story selection took place. Which went against the originally planned SCRUM ”but up scaled” idea where there would be a real sprint planning and retrospective for each sprint.

III. Experiences in the Scrum Workshop

As stated in II the process that was used in reality felt very unstructured and missed the point on how we initially envisioned the process, which was to actively engage the stakeholder. By not actively engaging and eliciting feedback from our product owner quite a few mistakes were made that could have been avoided. More interestingly though, this seemingly lack in interest of engagement caried over to even understanding the initial requirements which caused most of the aforementioned mistakes.

Ignoring the original process was likely done to speed up the development of the thing at hand, there was a clear goal: ”Build something in Minetest”. Speeding up development to hit a target is a bad idea[7]. The feeling of fast tracking the process produces faults that cost time to fix in the end, resulting in a net loss.

Concretely?

Planning got done in an informal manner, for example, a resource collector got tasked with collecting sand, but never got a concrete amount and would always result in over collection or after the fact questions. These issues did get better with each sprint, we were still working in an iterative manner, feedback slowly but surely improved the situation.

Speaking from personal experience, the original process outlined in II was not entirely made for genuine usage but was made for hitting a deadline. Part of the failure of this workshop can be attributed to this lack of motivation[2]. Students are still students and don't always have the same level of motivation that paid employees might have.

How is that a motivation issue?

Something that did not get better till the end was the scheduling of the program level activities, it got quickly decided on right before the kick-start of the project and never got changed. It was planned that each team would demonstrate their progress to the whole program and get stakeholder feedback on the spot sequentially. While it was a bonus that all teams got to see progress and feedback for each part of the system it introduced massive overhead as it forced each team to wait to elicit their own feedback and begin their team sprint retrospective. This time waste negatively impacted real sprint time, reducing the amount of real work that could have been done.

The cause for this is that each participating team had practically no experience planning at large scale, this problem is not unique and similar problems can be found[2]. Aside from lack of experience the fact that nobody stopped to think about reforming the system is also a known problem[6] that spells out doom for a process.

What does that mean?

Lack of motivation resulted in overly complex text book processes meant for already high functioning teams, applying such processes to low maturity teams is found to be pointless[1].

IV. Software Process Improvement Techniques

In the world of SPI there are two major ways of achieving SPI. Inductive SPI frameworks work on a bottom up principle; you have to fully understand the current situation first before making decisions about implementing improvement tactics. This way, solutions are hand tailored to help remedy the exact problems identified during the situation assessment. Frameworks like this help create tailored processes that work perfectly for existing teams, it make individuals think about their flaws and helps maturing a team faster than applying disruptive process changes. The downside is that the end result is not sustainable for future projects and teams as it was tailor made based on past problems and struggles and not based on empirically proven good processes; You end up reinventing the wheel every time.

A well known example of a bottom up SPI framework is QIP, which works in an iterative manner by setting clear goals and base judgment for what works on measurements taken from real experiences.

Contrary to inductive SPI methods, prescriptive methods apply a top down view to SPI. When applying a prescriptive method, known good process methods are introduced to the existing framework in an organization if the replacement is proven to be more efficient regardless of situation or needs[3]. That's why in this model, there is focus on gathering information about the existing process to see what practices have priority to be replaced by a better process. Applying "guaranteed to be good" processes like this saves a lot of time that would be spent figuring out what would prove be good like in inductive methods. Also this makes a process more reusable and generic, instead of applying different processes to teams the whole organization will follow the newly introduced processes as they are not tailored to solve specific problems. Good

A well known example of a top down SPI framework is CMMI, which defines a standard of practices that are known to be good and tiers them appropriately such that an organization can gradually improve.

These SPI frameworks both strive to help organizations and teams to work better, when applied properly they could be found to be beneficial to our own struggles as outlined in III, but a mix of inductive and prescriptive methods have to be used in this case. Students need ? to be spoon fed proper teachings, but also need to experience gradual improvement to actually believe it works, otherwise students will think they can do it better in their own way.

V. SPI in Industry

Agile transformation - Dulce Goncalves

When starting with agile more is less, when transforming companies to agile the goal was to restructure the existing into a more leaner process. As structure often means more complexity and overhead. The complexity of a product often mirrors the dysfunctional complexity of an organization.

Agile culture is for organizations where all employees actually do want to make a difference. Even when people want to work in an agile way and take responsibility these teams become immature very quickly. To succeed they have to really understand what commitment means and take accountability seriously.

During the agile transformation it was seen very quickly that there was a big lack in trust and that people feared that they might be incompetent. For a manager in an agile environment it is extremely important to be engaged in the workplace itself, as the role of a manager is shifted towards a coaching role rather than a commanding role. Transforming to the agile way has nothing to do with technology, it simply has to do with the way you think. But the right conditions have to be in place for it to happen, otherwise nothing will come from the ground. During the transformation it's important to involve your product consumers to get feedback and change along the way. A lot of companies end up just checking boxes instead of actually understanding what agile means, which ends up producing no value to the company at all. You have to understand why you do certain things to extract the value.

1928diagnostics - Robert Engberg: As a company in the medical field quality and holding up regulations is extremely important. For that 1928Diagnostics implements an iterative way of QMS a a highly defined process to keep quality high and more importantly be consistent. When constrained by regulations and privacy concerns following ISO standards are most of the time the only option forcing a top down SPI framework as there is no room to hand Not really... tailor processes. Implementing a new process is in itself a process. Over documentation is directly correlated to reduced effectiveness but gradual SPI reduces this initial overhead over time.

VI. SPI Proposal for future Scrum Development Efforts

Before setting up a concrete SPI plan 3 improvement goals have been decided upon based on the experiences from the first workshop III:

1) Improve the strategy for task distribution from the scrum master's perspective to reduce time spend on planning but also to make sure each team member knows that to do and that each tasks is properly broken up.
2) Increase the proficiency in using the tools from the developers perspective to remove the barrier for more SPI further down the road when skill is not holding back the team anymore.
3) Improve the timeliness of task delivery from the scrum master's perspective to help the team meet deadlines for each sprint and to give a better sense of progression.

The goals were chosen not to reach the top, but to remedy fatal flaws. Goal 1) will give each team member a clear goal to work towards during sprints, to stop wasting time and boost morale as it increases task completion (even if more tasks will get produced). Goal 2) is necessary to build up the bottom line of the team, high skill will often make things easier. But also helps the team identify personal problems which can be taken into account during sprint planning and task distribution. Goal 3) is there to speed the whole process up and hit deadlines more frequently, the description is relatively broad but this is so that not one specific piece of the puzzle suffers due to time pressure[7]. This SPI proposal was made following the CMMI for development document[8] and taken from past assignment[9].

Requirements Management (ML2)

The purpose of Requirements Management (REQM) is to manage requirements in a way that no information gets lost and that ALL requirements are properly documented and known such that no misunderstandings happen.

Specific Goals

SG1: Requirements are managed and inconsistencies with project plans and work products are identified.

Specific practices

SP1.1: Develop an understanding with the requirements providers on the meaning of the requirements.

Improvement steps during sprint planning / review

1) Establish objective criteria for the evaluation and acceptance of requirements. Examples of evaluation and acceptance criteria include the following: Clearly and properly stated, Complete, Consistent, Appropriate to implement, Verifiable, Traceable, Achievable, Identified as a priority for the customer
2) Analyze requirements to ensure that established criteria are met.
3) Reach an understanding of requirements with requirements providers so that project participants can commit to them. (This step is directly related to GP2.7 seen below)

SP1.4: Maintain bidirectional traceability among requirements and work products.

Improvement during sprint planning:

1) Link each task to the acceptance criteria of user story
2) Create a board to keep track of which tasks are created, and are assigned to which team member
3) Make sure each team member understands which acceptance criteria they are working towards

SP1.5: Ensure that project plans and work products remain aligned with requirements.

Improvement steps during sprint review

1) Review project plans, activities, and work products for consistency with requirements and changes made to them.
2) Identify the source of the inconsistency (if any).
3) Identify any changes that should be made to plans and work products resulting from changes to the requirements baseline.
4) Initiate any necessary corrective actions. (This includes adding additional tasks or refining tasks which are currently in progress)

Project planning (ML2)

The purpose of Project Planning (PP) is to establish and maintain plans such that no time is wasted discussing what to do next and time estimations are more accurate.

Specific Goals:

SG1: Estimates of project planning parameters are established and maintained.

Specific Practices:

SP1.3: Define project life cycle phases on which to scope the planning effort.

Improvement plan during sprint planning

1) Create a defined schedule for the entire workshop
2) Create a defined schedule for each sprint
3) Planning will occur during sprint planning and sprint retrospectives

notes: 2) See schedule below, 3) Retrospective included to make changes on the fly.

SPI1.4: Estimate the project's effort and cost for work products and tasks based on estimation rationale.

Improvement plan during sprint planning:

1) Planning poker during sprint planning
2) Understand requirements and break them down into relevant tasks
3) Based upon tasks and story points, assign an estimated time to each task
4) Total time for all tasks in a sprint

SG2: A project plan is established and maintained as the basis for managing the project.

Specfic practices

SP2.5: Plan for knowledge and skills needed to perform the project.

Improvement plan:

1) Identify the knowledge and skills needed to perform the project. (Based past experience)
2) Assess the knowledge and skills available. (Team members can point out what is difficult, i.e. mining, sprinting, etc.)
3) Experienced team members can guide others on what they need to do before the workshop.

SP2.6 Plan the involvement of identified stakeholders

Improvement plan:

1) Stakeholder involvement plan (Can be seen below in GP2.7 as it needs to be on a program level).

Generic goals: GP2.2 Establish and maintain the plan for performing the process. why:

- As this generic practice closely relates to 2 of our GQM goals:
  - Improve the strategy for task distribution from the scrum master's perspective.
  - Improve the timeliness of task delivery from the scrum master's perspective.
- Following the planned process more rigidly will allow us to spend more time doing and less time not knowing what to do.
- Previously, we established a plan but did not follow or maintain it as all members were not aware of it.

how:

- Sprint schedule should be agreed upon at a program level. All teams will have the same process to ensure collaboration is possible.
- Scrum master's from every team meet and reach a consensus for what the plan will look like prior to the beginning of the first sprint
- Plan out a process that every team member agrees to and is committed on following
- involve developers as well as PO in planning process
- Create a realistic schedule where:
  - the schedule accounts for delays

- the schedule accounts for technical issues

GP2.7 Identify and involve the relevant stakeholders of the process as planned. related to SP 2.6 Plan for the involvement of identified stakeholders.

why:

- In the previous workshop we identified the stakeholders but failed to involve them in our process
- Involving stakeholders such as the product owner in our sprint planning would aid task distribution (Improvement goal).
- The product owner can help define the acceptance criteria more clearly, therefore aiding task creation and story points estimation

how:

- Create a program level schedule in collaboration with other teams (regarding which team will communicate with the PO at what time), to ensure product owner is equally available for all teams, for relevant user stories during sprint planning or sprint review
- Create a schedule for the Sprint plan where relevant time is allotted to the Product owner
- Document acceptance criteria shared by the Product owner
- Adapt to the feedback given by the Product owner

Measuring process improvement

- Satisfaction of the scrum master with the task distribution on a Likert scale
  - Justification: This metric will show if the product owner thinks that the product development process is going according to plan or if it needs improvement.
  - When: In each Sprint Review
  - How: in sprint review the POs can answer the sample questions like: in the likert scale how satisfied are you with the product development process.
- Satisfaction of the product owner with the product development process on a Likert scale
  - Justification: This metric will help show how much the team worked in each sprint and if the task distribution is related to the velocity, i.e. if the story points in the backlog match the velocity, it means that the scrum master considered previous progress during task distribution.
  - When: Sprint planning and sprint retrospect
  - How: The sum of developers' velocity will be the team velocity.
- Team Velocity in each sprint
  - Justification: as developers get more comfortable with the tools they are using, they become faster and can deliver more user stories per sprint, and with it the process improves.
  - When: Sprint review
  - How: Total number user stories the team were able to demonstrate to PO in sprint review.
- Number of user story points delivered per sprint
  - Justification: This metric will show much change there needs to be regarding task delivery. If the satisfaction is too low it's the first symptom of bad task delivery.
  - When: Sprint retrospective
  - How: During the retrospect they can express their satisfaction and based on that the team can adjust.
- Tasks completion per sprint
  - Justification: This metric measures the present of tasks left uncompleted per sprint, it shows if there are too many or too little tasks. More importantly this metric shows if tasks are too big, smaller tasks would allow a higher completion rate when not looking at story points
  - When: Sprint retrospect, sprint review
  - How: Number of planned user stories delivery minus number of delivered user stories.
- Number of hours/minutes spent on task distribution (minutes / hours)
  - Justification: This metric will show how long the task distribution took and if it required more time to be done properly.
  - When: during implementation
  - How: By comparing the time spent on task distribution and schedule.

Schedule:

- Sprint 1: 9:30 - 10:30, Planning: 9:30 - 9:35, Implementation: 9:35 - 10:05, Review: 10:05 - 10:15, Retrospective: 10:15 - 10:20
- Break: 10:20 - 10:30
- Sprint 2: 10:30 - 11:25, planning: 10:30 - 10:35, Implementation: 10:35 - 11:05, Review: 11:05 - 11:15, Retrospective: 11:15 - 11:20
- Break: 11:20 - 11:25
- Sprint 3: 11:25 - 12:15, Planning: 11:25 - 11:30, Implementation: 11:30 - 12:00, Review: 12:00 - 12:10, Retrospective: 12:10 - 12:15

If there are any delays, Implementation time can be decreased by up to 10 minutes for each sprint.

VII. Implementation of an SPI Initiative

The second workshop started off with a few extra constraints: we were to build a castle, all buildings had to be tightly packed as walls had to be present at all times. This presented a slight shift to increased pressure on the organizational level, and affected the results we were aiming for. But none the less, we have implemented outlined practices in VI. how?

Requirements management

Effect?

SP1.1 Failed to implement, rather than analyzing requirement criteria we opted to simply confirm with the product owner that which what we made followed the requirements.

SP1.4 Was partially completed, calling it bidirectional tractability would be rather rich a separate team Trello board was made to track tasks derived from the taken user story but the information presented on the board was less than ideal. This could be because due to the fact we were breaking up tasks and writing them down further requirements were not desired and thus forgotten about.

SP1.5 Frequent checks were made with the product owner to make sure requirements were met, the results respectively 2, 5, 3 were produced respectfully to each sprint. In the 2nd sprint a major user story was completed resulting in complete satisfaction, which shows that requirements often have to be completed in full to convince the product owner. Requirement changes did happen and they got handled but no meaningful measurements were taken for that particular scenario.

Project planning

SP1.3 Has been more or less achieved, a schedule had been created beforehand and sprint planning was deemed to be satisfactory by the SCRUM master with a score of 3, 5 and 4 respectfully to each sprint. The sudden shift from a 3 in the first sprint to a 5 in the 2nd can be attributed to the organizational overhead and start up pains decreasing.

SP1.4 Partially, planning poker did not occur and time estimations were discarded as well but rudimentary tracking of tasks did occur yielding a team velocity of 4.3 and task completion of 1/3, 12/12 and 1/3 each sprint respectfully. Same issue as SP1.4, it was deemed unnecessary during the workshop.

SP2.5 Partially, while experienced members helped out, assessment and identification of skill level did not happen. Lack of skill was less of a problem this time around and did not come up beyond quick questions.

SP2.6 Completed, a new system compared to the last workshop was implemented where each team had to approach their respective product owners instead of the other way around. This resulted in less overhead for both sides which was a good success. Though it could be said that no real time was allotted to meet with the product owner but instead become irregular meetings.

With varying results of the implementation of these specific practices we can say that improvement goal 1 and 2 have been hit while the 3rd improvement goal is debatable, as the new challenges introduced make it hard to directly compare against the first workshop.

VIII. Summary and Lessons Learned

We started off designing a questionable process, failing to implement said process and crawling back up with SPI after defining what SPI actually is and showcasing 2 industry examples.

The conclusion of this experiment is that SPI does indeed improve the performance of a team. Understanding a team's problems with an inductive SPI framework, in our case GQM, helps build the basis for improvement as understanding and motivation increases in the face of knowing you can do better. Then applying a prescriptive framework such as CMMI to target the set goals with known good practices undoubtedly shows success. Mixing these SPI frameworks yields the best of both worlds. That said though, it is critical that before applying SPI the whole team has trust and is willing to move forward in the process V, sloppiness in practice will downplay a lot of the improvements being implemented.

A small nitpick with our process improvement plan is that we barely touched on the organizational aspects of the process, which turned out the be quite detrimental to the whole experience. We also lacked a way of measuring product owner engagement even though it was quite an important part of our plan. The thing I would want to do differently next time around is swaying the organization more, only a small portion of the class actively engaged and this clearly has its limits as the active members also have to focus on their own team. If each person in an organization chimed in a lot could be done.

Good

Making the process even more simple should also be a priority, I found that while nice, these improvement plans are sometimes very complicated to understand which lead to them not being fully implemented in realityVII.

References

[1] A. Cater-Steel, M. Toleman, and T. Rout, "Process Improvement for Small Firms: An Evaluation of the RAPID Assessment-Based Method". Information and Software Technology, vol. 48, no. 5, pp. 323- 334, 2006.

[2] P. Abrahamsson and K. Kautz, "Personal Software Process: Classroom Experiences from Finland". Proc. Software Quality – ESCQ '02, pp. 177-181, Springer, 2002.

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

[4] J. Sutherland et al. "Scrum and CMMI Level 5: The Magic Potion for Code Warriors". Proc. Hawaii International Conference on System Sciences pp. 1-1

[5] A. Sidkey and J. Arthur "A Disciplined Approach to Adopting Agile Practices: The Agile Adoption Framework". Virginia Tech, pp. 3-3

[6] J .S. Reel, "Critical Success Factors in Software Projects", IEEE Software, 16, 3, 1999, pp. 18-23.

[7] Chappell, D. (n.d.). "The three aspects of software quality: functional, structural and process". davidchappell.com, pp. 5-6

[8] CMMI Product Team, "CMMI ® for Development, Version 1.3" Carnegie Mellon University, Software Engineering Process Management Program, pp. 281-299, 341-347, 2010

[9] Anonymous authors, "Process Plan", Assignment 6 in course DIT347 H20, Software Engineering Program, University of Gothenburg, Sweden, 2020