From Scrum to Scrumban: A Case Study of a Process Transition

Natalja Nikitina, Mira Kajko-Mattsson
School of Information and Communication Technology
KTH Royal Institute of Technology
Stockholm, Sweden
nikitina@kth.se, mekm2@kth.se

Magnus Stråle
Affiliation omitted due to confidentiality agreement
magnus@strale.se

***Abstract*—Transitioning from one development method to another has become a common routine for many companies. Despite this, very few reports describe how the process transition has been carried out, and provide suggestions for how to define a process transition model. This paper reports on a process transition from Scrum to Scrumban in one software development company. The paper gives an account on the process transition process, changes done to the development process undergoing the transition and the improvements achieved. It rounds up with lessons learned.**

***Keywords*-Process improvement; method adoption; process introduction; Kanban**

I. Introduction

Despite low empirical evidence, the newly risen agile software development methods are regarded as a silver bullet to solve many organizational and/or process related problems [1][11][17]. Hence, transitioning to them has become a common routine for many software companies today [1][9] [17].

Transitioning to another development method is not a risk-free adventure, however. It may be challenging and it may not always bring the desired and long-lasting results [9] [19]. This may be due to two main reasons. First, one puts too much trust into the goodness of the new methods. Second, one is not aware of the scale, complexity and impact of transitioning from one method to another [17][19]. Such awareness may only be gained by creating a suitable and realistic transition plan and strategy, as well as a proper process transition model.

So far, the majority of the reports dealing with the transition discuss the transition from heavyweight methods to agile methods [5][7][10][16], or they report on the adoption of agile methods into projects that do not follow any method at all [14][19]. They mainly focus on the results of the process transition and on the positive characteristics of the newly adopted methods [4][7]. Except for one report that has been written by the first two authors of this paper [15], none of the reports give an account on how process transition has been carried out and none of the reports provide suggestions for how to define a process transition model as part of a software process improvement effort.

In this paper, we report on a transition from Scrum to Scrumban in the Vietnamese office of one software development company. The transition was done as part of software process improvement effort. The company has previously conducted a similar process transition in its Swedish development office [15]. This paper also presents a process transition process as executed in the company and describes the lessons learned.

The remainder of this paper is as follows. Section 2 briefly presents Scrum, Kanban and Scrumban. Section 3 describes the organization studied. Section 4 reports on our research method. Sections 5 – 7 present the process transition process and describes problems within the organization before and after process transition, respectively. Finally, Section 8 rounds up the paper with the conclusions and lessons learned from the process transition.

II. Scrum, Kanban and Scrumban

Scrum and Kanban are light-weight agile methods, most often used in small or middle size development organizations [8]. They center around the informative workspaces, involve collaborative teamwork and acknowledge the importance of teams' self-organization [8].

Scrum is a well-known iterative and incremental method resulting in a piece of working software to be delivered at the end of each development iteration, called sprint. It is focused on managing software development projects by means of strictly defined roles, meetings and process artifacts. [18]

Kanban is a new method that is gaining an increasing popularity. It is based on Just-In-Time and Lean production system. In contrast to Scrum, it is less prescriptive. It does not include development iterations such as sprints, does not define roles such as Scrum Master, does not define meetings such as daily standups, demos or retrospectives, and finally, it does not contain process artifacts, such as release backlog [8]. Kanban's core constituents are (1) visualization of the workflow, (2) limitation of work in progress, and (3) measurement of lead time [8].

Similarities between Scrum and Kanban allow combining the two methods by implementing some of the Scrum practices and Kanban principles. Such a combination is being referred to as Scrumban [3].

III. The Organization Studied and its Context

The organization studied is a medium-sized Swedish company developing and maintaining a content management system. Due to the fact that the company wishes to stay incognito, we use its fictitious name instead,

which is *CMSiPro*. The company has sales and marketing offices in ten countries all over the world. Its software development is geographically distributed between two development offices, one in Stockholm, Sweden and the other one in Hanoi, Vietnam. Its main management, product design and product owners (POs) are located in Sweden and Denmark.

The company's technical staff includes one team of developers located in Sweden, one team of developers and one team of testers located in Vietnam. Each team consists of 15 to 20 engineers. In this paper, we refer to those teams as *Swedish development team*, *Vietnamese development team* and *Vietnamese testing team*. The *Vietnamese testing team* supports both development teams with testing services.

At the beginning of 2010, both development offices followed Scrum poorly. The *Swedish development team* experienced many process related problems and low developer motivation. For this reason, in March 2010, the *Swedish development team* transitioned from Scrum to Scrumban [15]. This implies that the organization has implemented Kanban's continuous development flow, but kept some of the Scrum's elements such as daily standups, demos, release planning and retrospectives.

Although the transition had led to positive results, it introduced synchronization problems between the *Swedish development team* and the *Vietnamese testing team*, which, in turn, had led to significant communication problems between the two teams. The *Vietnamese development team*, on the other hand, was not affected by the process transition in Sweden at all. Still, however, it constituted an isolated island of developers running their own Scrum race.

To introduce organization-wide process uniformity in the whole *CMSiPro* and to remove the above-mentioned problems, the company management decided to introduce Scrumban in Vietnam as well. The decision was made in December 2010 and the transition from Scrum to Scrumban was conducted in the period of February 2011 till July 2011.

The *Vietnamese process transition* consisted of three consecutive stages: (1) *Pre-transition*, during which the company planned and prepared for the transition, (2) *Process transition,* during which major changes to the process were made and a transition to the new process model was realized*,* and (3) *Post-transition*, during which the newly introduced process model was tuned and continuously improved.

IV. Research Method

The research method used to conduct this study was action research. Both the researchers and the company were engaged in the *intended study in action* [2]. The third author of this paper was a *Transition manager* at *CMSiPro*, and the other two authors were researchers, who were participant observers of the process transition. The role of the *Transition manager* was to lead the process transition while the role of the researchers was to observe the process during the *Process transition* stage, participate in some of its activities, identify problems, guide the process changes and reflect upon them. In this section, we first present our research method and then discuss the validity of our results.

A. Research Method Steps

Overall, our research method consisted of two main phases: *Initial study* and *Software process transition.* Before this study was commenced, the researchers were already well acquainted with the company. Despite this, they had to supplement their overall company knowledge with the information about the development and testing processes as run in Vietnam and their interface to the processes as run in Sweden. During, the *Initial study* the researchers observed and participated in the software development process in Sweden for more than one year. They also conducted a number of unstructured interviews with the developers and testers from Sweden and Vietnam.

Regarding the *Software Process Transition* phase, we followed all its stages with different sets of responsibilities. Being the *Transition manager*, the third author of this paper had the executive role throughout the whole transition process. It was he who was responsible for carrying out the process changes and for making sure that they were properly conducted and managed.

The roles of the other two authors, on the other hand, varied across the process transition. During the *Pre-Transition* stage, the researchers were only observant participants. Their role was to collect data on the process assessment, interview the technical staff to be affected by the process transition, lead some training sessions and document all the pre-transition activities. During the *Process transition* stage, the researchers were participant observers, whose role was to observe a chain of events taking place in the process transition, participate in some of those events and give recommendations on transition strategy and its implementation. During the last stage*, Post-transition,* the researchers did not observe the process directly. However, they collected feedback by frequent and unstructured interviews, emails and online conversations with the *Transition manager*, developers and testers in Vietnam. Their role was to document a chain of events taking place during the iterative process tuning.

B. Validity

All the qualitative research methods, including case studies and action research, encounter validity threats [2], [13]. Those threats concern construct validity, internal validity, external validity, and conclusion validity.

Construct validity refers to the degree to which inference can be made from the operational definition of a variable to the theoretical constructs [20]. The main threat to construct validity is to guarantee that the right measures have been chosen for the study. Regarding the construct validity, the risk was that the researchers might use wrong measures, and as a result, they might misinterpret the transition process and its results. To minimize this threat, multiple sources for collecting data were employed. Those included semi-structured interviews with different roles involved in the process, data gathered during participatory observations,

unstructured interviews with the managers, developers and testers, and study of the internal documentation.

Internal validity refers to the degree of inferences of the cause-effect or causal relationships in the study [20]. Even though internal validity is not relevant in most observational or descriptive studies [20], it is still relevant to this study. Main threats to internal validity here concerned the social intervention that might lead to the situations in which some of the observed individuals might behave differently when being observed. Those threats were minimized by means of triangulations of data, use of different methods and involvement of various roles. However, due to the high involvement of the authors, we cannot claim full objectivity of the evaluation results.

External validity refers to the degree of whenever the sample findings can be generalized [20]. The main external validity threat to our study was the fact that the process transition and elicitation of the process transition process was made within only one company. Due to the nature and limitations of the method used in this study, there is no formal foundation for data generalization. However, we believe that the findings and conclusions of this study can be found useful for software development companies, which plan to include software transition as part of their process improvement initiatives. We also believe that our elicited process transition process can provide a platform for developing process transition methods.

Conclusion validity refers to the degree to which the conclusions are based on the correct interpretation of the relationships of the data [20]. The conclusion validity threat to our study was that the conclusions would not be related to the data. To minimize the threat, we based our conclusions on the multiple data sources such as data gathered during the interviews and data gathered during the observations throughout the whole transition process.

V. Problems within the Organization before the Process Transition

During the *Pre-transition* stage, the following process related problems have been identified:

Problem 1: Lack of communication pattern: Even though the company had globally distributed development and testing teams, they did not follow any communication pattern. Communication inside each development and testing team was good. However, the communication among the teams and the Product Owners (POs) was limited. They communicated via emails, and rarely via conference calls.

The poor communication had led to various delays and miscommunications. For instance, the *Vietnamese testing team* was not always notified on time when a user story was developed, and therefore, its testing results were frequently delayed. In addition, the *Swedish* and *Vietnamese development teams* working on different sub-products did not share enough information about the product, solutions used, or ideas proposed. This limited the exchange of important information that was essential for achieving uniform system structure and quality and for achieving successful collaboration and effective inter-team learning and knowledge sharing. The six-hour time difference and the limited availability of the POs had led to long waiting times thus contributing to inefficient development process.

Problem 2: Miscoordination of work: The *Swedish development team* followed Scrumban that was driven by a continuous flow, and the *Vietnamese development team* was still using Scrum that was driven by iterations. This had created challenges to the *Vietnamese testing team*. They had to follow two different processes: (1) Scrumban when collaborating with the Swedish team and (2) Scrum when collaborating with the Vietnamese team.

Testing worked well in the context of Scrumban. The testers worked side-by-side with the developers. While the developers were implementing the user stories, the testers were writing the test cases for them. As a result, the testers managed to provide fast feedback to the *Swedish development team*.

In the context of Scrum, on the other hand, testing did not work well. Although the managers expected the testers to work side-by-side with the Vietnamese developers, the developers preferred the testers to write test cases only after development. As a compromise, the testers wrote only a subset of test cases, if any, during development, leaving most of the work to be done after development. As a result, the testers felt frustrated and overburdened with testing after development and their testing feedback was always substantially delayed.

Problem 3: Poor requirements change traceability: Product requirements were expressed as user stories. They were created by the POs and documented and stored in TFS, a common organization-wide collaboration platform [12]. Each user story was briefly formulated in one or two sentences and assigned a unique identifier. Larger and more complex user stories were supplemented with detailed textual descriptions.

The user stories were often not ready for implementation when they were assigned to the developers. They either missed important information, or they were poorly verbalized, or they were not sufficiently explored and validated. Some of their details were often changed by the POs during sprints and the changes often got recorded as new user stories without being linked to the original stories whatsoever. This had led to the difficulties in tracing the requirements changes, and thereby, to confusion among the developers and testers, and consequently, a substantial waste of time. The problem applied to about 20% of the user stories.

Problem 4: Lack of definition of "done": The definition of when a user story was considered completed – *"done"* in agile parlance was not clear enough to all the stakeholders. Quality management activities such as unit tests, performance tests, code documentation and the like were not uniformly performed. Sometimes, they were performed, and sometimes they got forgotten. As a result, untested or undocumented code was often checked in to the main code branch. This had created quality and management related problems, such as low code quality and confusion upon its release readiness and status.

Problem 5: Prioritization of short-term solutions: High importance of delivering all the planned features by sprint deadline made the Vietnamese developers focus on short-term deliveries and neglect long-term strategic goals and plans. The developers were pressured to compromise on code quality for the sake of feature quantity since it contributed to delivering the sprint goals on time. Even though they intended to attend to all the quality problems in later sprints, they did not always do so, due to the low prioritization of quality improvement tasks and lack of time.

Problem 6: Inertia towards changes in the merged code: Code merges and integration were done only at the end of the sprints thus resulting in highly time consuming, complicated and problematic code merges. To save time and to avoid introducing new merging problems, the developers were unwilling to change the merged code. As a result, very few code improvements were done after code merges.

Problem 7: Slow feedback from management: The efficiency and effectiveness of the Vietnamese employees strongly depended on how promptly they got feedback from the Swedish office. The requests they provided to the Swedish office did not always receive enough attention and timely feedback. The Vietnamese employees felt that the Swedish office did not always prioritize their requests and often delivered their responses with a substantial delay.

Problem 8: Lack of process knowledge and commitment: The Vietnamese developers and testers have not undergone any training in Agile methods. They did not have clear understanding of the process they were performing, and therefore, they could not easily grasp the reasons behind and implications of changing it. As a result, their support, engagement and commitment to the process and process transition were very low. The need for process training in the current and candidate development method was evident.

Problem 9: Lack of openness: Due to their cultural nature, the majority of the Vietnamese people are shy and humble [6]. This could be clearly visible in the context of the company studied. The Vietnamese employees did not actively share their opinions and ideas with their managers. They did not dare to express their dissatisfaction and they were not courageous enough to initiate process changes. Finally, due to the strong culturally ingrained power structure in Vietnam, the Vietnamese developers were intimidated to speak up in front of the Swedish managers.

Problem 10: Increase of technical debt: Due to *Problems 2, 5* and *6*, the quality of the system was deteriorating. Since the quality management activities were not prioritized and many ad hoc changes were done on a frequent basis, the system quality and maintainability started to decrease. It became more and more challenging and time consuming to make changes to the system.

Problem 11: Low prioritization of process related activities: Little attention was paid to the process and process related activities. There were no resources reserved for tasks such as defining a process, enacting, improving, and documenting it. This had led to most of the above-listed problems, especially to lack of process knowledge (see *Problem 8*), lack of communication pattern (see *Problem 1*), miscoordination of work (see *Problem 2*) and poor requirements change traceability (see *Problem 3*).

Problem 12: Inefficient development process flow: Due to the majority of the above-listed problems, the development process was wasteful of time and resources and not efficient enough in producing the intended results and quality. It got often blocked or disrupted due to lack of timely information on the solutions and their status.

VI. Vietnamese Process Transition

The main stages of the *Vietnamese process transition* were: *Pre-transition*, *Process transition* and *Post-transition*. These are illustrated in Figure 1 and their activities are listed in Figure 2. Due to space restrictions, we cannot describe the whole process in detail. Hence, when describing them in this section, we only focus on the main activities. For more detailed information, we advise our reader to study Figure 2.

A. Pre-transition Stage

The *Pre-transition* stage lasted for three months. It consisted of the following five steps:

1) Transition Initiation: Driven by the goal of aligning the two offices and their processes, the company management had made a decision to change the Vietnamese process from Scrum to Scrumban. To initiate it, they defined and assigned the role of *Transition manager*, whose responsibility was to prepare for and lead the process transition. After it, the *Transition manager* briefly informed the personnel about the process transition and explained his role in the transition process.

2) Pre-assessment: To be able to determine the future directions of the process transition, the *Transition manager* made an initial process assessment. He first studied the process for one week, in order to identify its strengths and weaknesses. He then had individual interviews with all the developers and testers, during which, he explained the process transition process and its purpose, and he assessed developers and testers' knowledge of the process. While doing this, the *Transition manager* had observed lack of knowledge of the process, and lack of interest and even some inertia towards the process change.

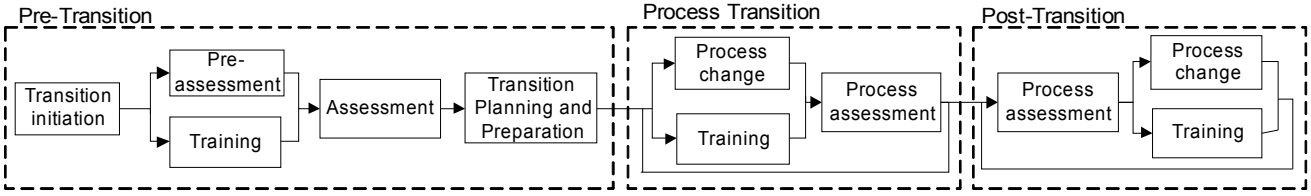

Figure 1. Vietnamese Process Transition

Pre-transition Stage

Transition initiation :

A-1.1	**Initiate process transition**
A-1.1.1	Identify needs for process transition
A-1.1.2	Set (preliminary) goals for process transition
A-1.2	**Get management support for process transition**
A-1.3	**Assign a role of the *Transition Manager***
A-1.4	**Announce the process transition**

Pre-assessment:

A-1.5	**(Initially) assess current process**
A-1.5.1	Study current process
A-1.5.1.1	Identify the strengths of the process
A-1.5.1.2	Identify the weaknesses of the process
A-1.5.1.3	Identify gaps and overlaps in responsibilities
A-1.5.2	Assess stakeholders' knowledge of current process
A-1.5.3	Assess stakeholders' knowledge in candidate process model for transition
A-1.5.4	Assess stakeholders ' satisfaction with the process

Training:

A-1.6	**Train the stakeholders in process transition**
A-1.6.1	Present experiences from previous process transition
A-1.6.2	Present the need and potential benefits of the process transition
A-1.7	**Train stakeholders in process**
A.1.7.1	Determine educational needs
A-1.7.2	Design educational program according to the educational needs
A-1.7.3	Train the stakeholders in candidate process model for transition

Assessment:

A-1.8	**Assess current process**
A-1.8.1	Analyze current process
A-1.8.2	List, analyze and prioritize process related problems
A-1.8.3	Identify problem causes
A-1.8.4	Define solutions for the identified problems
A-1.8.5	Measure the process
A-1.8.6	Evaluate readiness towards p rocess transition

Transition planning and preparation :

A-1.9	**Establish a transition team**
A-1.9.1	Identify the "supporters" of the process transition
A-1.9.2	Include the "supporters" in the transition team
A-1.10	**Plan transition**
A-1.10.1	Set goals for process transition
A-1.10.2	Identify the common parts and gaps between the current and candidate process models
A-1.10.3	Determine process transition changes
A-1.10.4	Identify impact of process transition
A-1.10.5	Determine the strategy for the process transition
A-1.10.6	Define a model for *Process transition* stage
A-1.10.7	Define a model for *Post-transition* stage
A-1.10.8	Create transition plan
A-1.10.9	Determine the transition budget
A-1.10.10	Announce process transition plan and schedule
A-1.11	**Prepare for transition**

Process Transition Stage

Repeat those steps for each iteration :

Process change:

A-2.1	**Implement the determined process change (s)**

Training:

A-2.2	**Train and coach the stakeholders in the change (s) to the process**

Process assessment:

A-2.3	**Review the changed process**
A-2.4	**Identify and prioritize/reprioritize the process related problems**
A-2.5	**Update the list of the prioritized problems**
A-2.6	**Determine the next process change(s)**
A-2.7	**Update transition plan and schedule**

Post-transition Stage

Repeat those steps for each iteration :

Process assessment:

A-3.1	**Review the changed process**
A-3.2	**Identify and prioritize/reprioritize the process related problems**
A-3.3	**Review the structure of the process review and revise it , if needed**
A-3.4	**Assess stakeholders' satisfaction with the process**
A-3.5	**Measure the process**
A-3.6	**Determine the next process change(s)**
A-3.6.1	Propose solutions for the top priority problems
A-3.6.2	Identify the next set of process change(s)
A-3.6.3	Assign people to the role of the *Change Manager*
A-3.6.4	Plan for how to implement the determined process change(s)

Process change:

A-3.7	**Implement the determined process change (s)**

Training:

A-3.8	**Train and coach the stakeholders in the change(s) to the process**

Figure 2. Process transition process as executed in the Vietnamese office

Last, using the questionnaire in Table 1, the *Transition manager* surveyed the people's overall motivation and satisfaction with the current process. The developers and testers were asked to answer fourteen motivational questions structured into the following four clusters:

1. *process related motivational factors* referring to process related problems that affect developers' motivation. They concern clarity of responsibilities, project goals and requirements, as well as developers' productivity and satisfaction with the development process.
2. *intrinsic motivational factors* refer to personal interests and to developers creativity and learning opportunities.
3. *extrinsic motivation factors* referring to motivation stimulated by the outside factors and concerning developers' satisfaction with the company, rewards for their work and social events organized by the company.
4. *social factors* concerning social aspects that affect motivation and focus on relationship with colleagues.

The engineers positioned their answers on an ordinal scale from 1 to 5, (where "1" corresponded to the most dissatisfied and "5" corresponded to the most satisfied). The results of the evaluation as presented in Table 1 show that the developers were highly satisfied with the process. Only one question (No.12) scored below 3, representing engineers' dissatisfaction with their salaries. The average evaluation for all the other questions scored in the interval between 3 and 5, representing high satisfaction with the process. This might explain the aforementioned lack of interest in and inertia towards process transition.

3) Training: To increase people's engagement and interest in the process transition, the *Transition manager* gave two educational sessions. During the first session, he aimed at educating developers and testers in the process transition. Here, he presented the process transition in Stockholm and its positive results, and explained the need for and potential benefits of process transition. During the second session, the *Transition manager* trained the developers and testers in the candidate process model for

TABLE I. MOTIVATION QUESTIONNAIRE

Motivation questions	$\bar{x}$
Process related factors	
Q1: Are your responsibilities clear to you?	3,3
Q2: To what extend does the frequent requirements change disrupt your work?	3,4
Q3: Are the Product Owners (POs) available to clarify the requirements?	3,7
Q4: Are the company vision and the goals of the project clear to you?	4,3
Q5: To what extend do the meetings and other non-development work affect your productivity?	4,1
Q6: Are the responsibilities of the teams and of the POs clear to both parties?	4,6
Q7: Are you satisfied with the development process that is currently used?	4,0
Intrinsic motivation factors	
Q8: How interesting and creative is your work?	4,4
Q9: Are you continuously improving your skills and knowledge?	4,5
Extrinsic motivation factors	
Q10: Are you satisfied with the company you are working for?	4,3
Q11: Does the company provide social activities to a satisfactory extent?	3,1
Q12: Are you sufficiently rewarded for your work?	2,8
Social factors	
Q13: How strong is the team-cohesion in your team?	4,4
Q14: How good is your relationship with your colleagues?	4,8

TABLE II. PROCESS ASSESSMENT QUESTIONNAIRE

1. What are the benefits of current software process?
2. What are the problems of current software process?
3. What do you think is the reason of the process transition?
4. What are your expectations of the process transition?
5. What are your worries about the process transition?
6. Are you satisfied with the current software process? To what extend?
7. Do you support the process transition? To what extend?

transition. Here, he presented Agile development methods and explained the importance of working as a team. The two sessions had contributed to the improved understanding of and improved attitude towards the process transition.

4) Assessment: As a next step, the researchers assessed and analyzed the development process in Vietnam. They first observed the process for two weeks. The goal here was to acquire an impartial and unbiased understanding of the process. They then interviewed all the Vietnamese developers and testers using semi-structured interviews. As shown in Table 2, the interviews aimed at identifying and analyzing the process related problems and benefits. This was done in order to define solutions for process related problems. The interviews have also contributed to evaluating the level of readiness to process transition which, in turn, would provide feedback for transition planning and preparation.

5) Transition Planning and Preparation: Based on the lack of readiness to process transition and the observed inertia to change, the *Transition manager* decided to have a two-stage process transition. The first stage would be conducted during the *Process transition* steps and it would consist of a set of process transition iterations that were essential for going over from Scrum to Scrumban. The second stage would be conducted during the *Post-Transition* steps and it would encompass iterative process tuning and improvements. Process transition was planned and prepared for by the *Transition manager*. During the transition planning, transition goals were identified, future process changes were determined and implementation of those changes was planned for. Finally, to support the *Transition manager*, a transition team got created. It included the supporters of the process transition.

B. Process Transition Stage

The second stage, *Process transition,* was led by the *Transition manager*. It lasted for five months and it consisted of a series of process changes interwoven with training sessions and followed by the process assessments. This stage lasted until the desired process state was reached. Below, we describe its steps.

1) Process Change: The changes that had been conducted during the process change step included the following:

Change 1: Establish a new process infrastructure: The existing Scrum process infrastructure was continuously transformed to the infrastructure necessary for supporting Scrumban. The electronic Scrum dashboard was first transferred to a physical Scrum board and, after that, it was continuously adapted to the changing process to finally result in the Kanban board.

Change 2: Change the structure of the meetings: The Scrum meeting structure was changed and adapted to fit the new process model. The two meetings that were affected the most were daily standups and retrospectives. Regarding the daily standups, they were continuously evolving towards the desired structure. The desired structure changed the focus from the three standard Scrum daily meeting questions [18] to the sequence of steps including (1) announcement of the top priority tasks by the meeting facilitator, (2) self-assignment of tasks to developers, (3) arrangement of pairs and groups responsible for implementing the tasks, and finally, (4) discussions of the work related problems, if any.

Regarding the retrospective meetings, they were completely neglected and almost non-existent before the transition. They were considered to be waste of time and the few that had been conducted did not have any significant effect. During process transition, the company decided to make retrospectives become a driving forum for the iterative process tuning. The retrospectives were planned to be conducted on a regular basis during which the process would be reviewed, its problems would be identified and their possible solutions would be suggested.

Change 3: Terminate Scrum iterations: Scrum iterations (sprints) concerning the planning activities and the presentation of the sprint deliverables were stopped. Since the developers and testers were not ready for this change, the cancellation of sprints was suggested to become an experiment and, if proven unsuccessful, then the team would have an opportunity to go back to sprints.

A number of activities, such as retrospectives, product planning and demonstrations, were dependent on the sprints'

schedule. Since the sprints were now canceled, the team had to agree on new ways for scheduling those activities. The following was decided:

- The retrospective meetings whose goal was to review the process were scheduled to be done every two weeks.
- The planning meetings whose goal was to define the highest priority user stories were scheduled to be performed every week.
- The dates for product releases were created on a demand basis. They were not planned in advance.

Change 4: Establish a new communication pattern between testers and developers: In response to *Problem 1*, a new communication pattern was suggested in order to improve the communication between the developers and testers. According to it, the representatives of the *Vietnamese testing team* should attend the daily standup meetings of the *Vietnamese development team* during which they inform each other about the status of their work and their daily goals. In addition, to be updated in the development status in Sweden, the representatives of the *Vietnamese testing* and *development teams* should take part in distributed daily standup meetings of the *Swedish development team.*

Change 5: Establish a definition of done: In response to *Problem 4*, a definition of when a user story is considered to be implemented (definition of *"done"*) was established. The definition was inherited from the Scrumban development process in Sweden. As shown in its template in Figure 3, the definition focused on writing good quality code and following coding guidelines and quality standards.

Change 6: Introduce continuous code integration: In response to *Problem 6*, continuous code integration was initiated. Source code merges have become a daily routine.

2) Training: To increase the understanding and support of the process transition, a number of training sessions were conducted. They aimed at educating the developers and testers in the changes to the process. The training sessions were led by the *Transition manager*, researcher and the development team leader, and lasted in average for one hour. They covered the following topics:

- *Agile and lean development methods* focusing on the main agile principles and the importance of working as a team and not as an individual.
- *Scrum and Kanban* aiming at increasing the overall understanding of the process transition and the methods to be used.
- *Organizational rules and roles* aiming at evening out the understanding of the organizational rules and policies and at clarifying the responsibilities of each employee.
- *Web security* aiming at evening up the testers' knowledge about web security.
- *Code refactoring* primarily targeting the developers and aiming at encouraging them to improve the code quality and to decrease the technical debt.
- *Definition of "done"* targeting both the development and testing teams and aiming at introducing a definition of "*done"*. The goal was to make people understand its purpose and to encourage them to follow it dedicatedly.

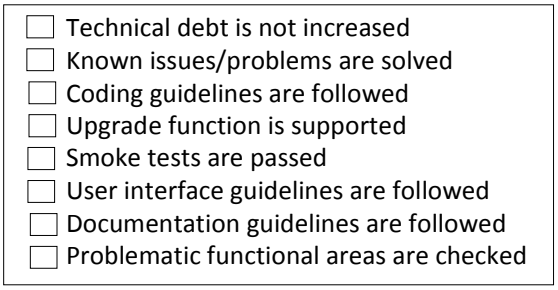

Figure 3. Template of definition of "done" as identified by *CMSiPro*

- *Unit testing* aiming at encouraging the developers to write unit code in a structured and uniform manner.

The training sessions were additionally supplemented by a daily coaching that was led by the *Transition manager.* The goal of the daily coaching was to ensure that the stakeholders understand the established process and properly follow it.

3) Process Assessment: The process assessments were done during the bi-weekly retrospectives. Here the results of the process transition were reviewed, main problems and obstacles were identified and suggestions for new process changes were added to the process transition plan.

C. Post-transition Stage

The *Post-transition* stage has encompassed continuous process tuning and improvement. It started in July 2011 and it still continues. The stage has consisted of continuous iterations of the following three steps: (1) *Process assessment*, (2) *Process change* and (3) *Training*. During the process assessments, the process changes have been identified in a consensus-driven and reactive manner. They have then been implemented and supported with training sessions, if necessary. Below, we briefly describe the steps.

1) Process Assessment: The process assessments were continued on the bi-weekly basis during the *Post-Transition* stage. They became the main forum for driving iterative process improvements, consisting of the following steps: 1) recreate a historical chain of events since the last retrospective by answering the question *"what has happened during the last two weeks?"*, 2) evaluate positive and/or negative impact of each event by means of an individual scoring, 3) discuss the negative events or experiences in detail and propose suggestions for improvements, 4) create a list of the main process problems or impediments by answering the question *"what prevents you from performing as good as you can?"*, 5) prioritize or reprioritize the problems in the list, 6) discuss in detail various solutions to the problems, and finally, 7) suggest solutions for the top three problems to be implemented until the next retrospective and identify volunteers within the teams who would be responsible for their implementation. Focusing on only three process problems has helped the team to stay more focused, structured and efficient.

2) Process Change: During the *Process change* step, the changes to tackle the afore-discovered top three process problems have been implemented by the volunteers, who in Figure 2 are referred to as *Change Managers.* The *Transition*

manager still has had the overall responsibility for the process change implementation. His role has been to assure that the volunteering *Change Managers* perform the process change tasks and to support them by removing all the obstacles hindering their task performance.

The process related changes that had been conducted from July 2011 until November 2011 (the time when the paper was written) concerned the following:

Change 7: Introduction of coding guidelines: Coding guidelines were introduced in order to support the unified and structured product quality standards, and thereby, to contribute to the improvement of product quality and to the decrease of technical debt.

Change 8: Adjustment of the Work In Progress (WIP) limit: WIP defines how many user stories the development team can work on at the same time. The initially set WIP limit was five. Soon however, it was discovered that it was too high and, as a result, it had led to the negligence of some user stories, usually the stories that were problematic and required extra attention. To remedy the problem, the WIP limit was first decreased from five to three. Soon thereafter, the company realized that working on just three user stories was too little, because in some cases developers did not have any pending tasks to be performed. For this reason, they have changed WIP to four, which has been found optimal for the size and velocity of the development team.

Change 9: Introduction of team building activities: In order to support the team spirit, the company had added a number of team building activities into the process. Examples of them are a two-days long kick-off workshop and introduction of common Fridays' team lunches.

3) Training: The training sessions that commenced during the *Process transition* stage got re-initialized during the *Post-transition* stage. Their goal has been to improve the process and product quality, to support the introduction of process changes and to brief the new hires about the development method used. In average, the training sessions were given every month. However, they were not planned in advance but rather created on an as needed basis for the purpose of supporting the undergoing changes. The sessions have lasted for one hour and have dealt with coding guidelines, and current development process. In addition, the *Transition manager* continued to coach the *Vietnamese development* and *testing teams* in order to ensure process adherence, good process understanding and continuous process dedication.

VII. Status of Problems within the Organization after the Process Transition

The initially discovered process problems have been partially or fully solved during the *Process transition* phase.

Problem 1: Lack of communication pattern: This problem has been partially solved. The communication between the distributed development and testing teams has been significantly improved. This has been achieved thanks to the new communication pattern (see *Change 4*), according to which, the representatives from each team have been present on all the daily standup meetings. This together with the updated Kanban boards has contributed to better visibility into the process and better communication and collaboration between the teams.

The communication between the POs and the *Vietnamese development* and *testing teams* has not been improved however. It is still limited, because some of the POs have limited time or are still not fully dedicated to the process. To avoid unproductive delays when waiting for their answers, the *Vietnamese development team* has started to make decisions on their own. These decisions are always communicated to the POs and adjusted to, if deemed necessary. They mainly concern the details of the user stories.

Problem 2: Miscoordination of work: This problem has been fully solved. After the *Process transition* stage, all the teams within the organization follow Scrumban. Both the testing and development team leaders participate in the meetings, during which, new user stories are being discussed. This implies that the testers have now access to the information about all the new user stories and are able to work simultaneously with the developers. Presently, most of the test cases are being written during development, which has substantially reduced the feedback loop between the testing and development teams, and thereby, the overall testing time and effort.

Problem 3: Poor requirements change traceability: This problem has been fully solved thanks to the new policy requiring that all the information about each user story is recorded in TFS [12]. The new policy has been dedicatedly followed by the developers and testers. In cases when POs have not managed to update their stories on time, the developers and testers have done it instead as soon as they were given new information. As a result, the requirements and information about their implementation have been kept updated in a structured and easily tracked manner.

Problem 4: Lack of definition of "done": This problem has been fully solved, when the new definition of *"done"* has been introduced (see *Change 5*). It is now being continuously reviewed, updated and communicated to all the team members. The *Vietnamese development team*'s leader has been responsible for checking that each developed user story satisfies all the criteria for the definition of *"done"*, before handing it over to the testing team.

Problem 5: Prioritization of short-term solutions: This problem has been partially solved, mainly thanks to the increased emphasis on the product architecture and quality, and thanks to the cancellation of sprints (see *Changes 3* and *5*). The *Vietnamese development team* has now a better focus on the long-term goals. The architecture and solutions for implementing each story are now being discussed from the long-term perspective, leaving little or no space for short-term solutions.

Problem 6: Inertia towards changes in the merged code: This problem has been fully solved. After the *Process transition* stage, the *Vietnamese development team* has been merging and updating the source code on a daily basis, (see *Change 6*). Code merges are no longer as time consuming or

problematic as before. Therefore, the developers are now no longer afraid of improving the code after it got merged.

Problem 7: Slow feedback from management: This problem has remained unchanged. Although the *Swedish development team* is getting immediate attention and response from the POs and Swedish management, the *Vietnamese development team* still has to wait for three to four days in average to receive the feedback.

Problem 8: Lack of process knowledge and commitment: This problem has been partially solved. The training sessions and daily coaching have improved the understanding of the agile practices and techniques, as well as the understanding of and adherence to the current process. Still, however, developers and testers do not have in-deep knowledge of the process and agile principles. As a result, nonstandard situations have not always been handled in an efficient manner.

Problem 9: Lack of openness: This problem has been partially solved, thanks to the changed structure of the meetings (see *Change 2*), and thanks to the fact that more attention has been given to the developers and testers' feedback during process assessments. After the *Process transition* stage, the development teams have been encouraged to point out process related problems and suggest solutions for resolving them during the retrospectives. Their suggestions and feedback have been taken seriously and acted upon by the managers. This has empowered the developers and testers to be more outgoing and share their feedback and complains more openly.

Problem 10: Increase of technical debt: This problem is on its way to be solved, thanks to the fact that the company has defined code quality standards, and has assured that they have been followed (see *Changes 5* and *7*). The new code is now being written according to the quality standards. Even the old code, when being modified, is being continuously improved and refactored. As a result, the technical debt is not increasing any more. However, the company needs to define ways of measuring it.

Problem 11: Low prioritization of process related activities: This problem has been solved when the organization has initiated process transition and provided resources for defining a process and for improving it. High focus on the process related activities was sustained thanks to the frequent retrospectives, continuous process changes and the positive results brought by the changes to the process.

Problem 12: Inefficient development process flow: This problem has been partially solved. After the process transition, the development process has become more transparent thanks to Kanban information board and clear definition of *"done"* (see *Change 1* and *5*). This has not however contributed to the efficiency of the process and on time deliveries. The main development project, on which the *Vietnamese development team* was working during the process transition period, has had significant delays. There are several reasons to it: (1) product demonstrations took longer time than planned, (2) the communication between the *Vietnamese development teams* and the POs has not been improved, thus, leading to even longer waiting time, see *Problem 1* and *7*, (3) a large amount of newly recruited developers has slowed down the development pace.

The process transition has also brought a new process related problem to the Vietnamese development office.

New Problem 1: The process is becoming heavyweight: After the process transition, the *Transition manager* and the *Vietnamese development* and *testing teams* have supplemented the initial definition of done with more criteria, adding more quality assurance and testing activities to the process. This has, however, resulted in a complex and, in some cases, heavyweight process. The *Transition manager* is planning to review all the parts of the process in order to suggest solutions for process simplifications.

VIII. Conclusions and Lessons Learned

In this paper, we have reported on the software process improvement effort that has been realized in form of a process transition from Scrum to Scrumban. While participating in the process transition, we have tried to mirror the process transition process.

The process transition process described in this paper is only preliminary, since it was based on only one executed process transition instance. However, it already provides valuable support for the software organizations wishing to transition to another software development method in an iterative manner. It also provides a valuable feedback for the researchers wishing to conduct similar studies or to develop process transition models.

During the process transition, the company has also learned the following lessons that are worth highlighting:

- Organization's readiness to the process transition needs to be assessed prior to determining the transition strategy and designing the process transition. This lesson has been materialized in the Activity A-1.8.6 in Figure 2.
- Training in software process is an important ingredient in the process transition process. This lesson learned is reflected in the Activities A-1.7, A-2.2 and A-3.8 in Figure 2. It is especially important in organizations where the overall process understanding is low.
- To achieve a more in-depth understanding of the principles behind the process and changes made to it, coaching and mentoring in software process should complement the training sessions. The importance of coaching has been incorporated in the Activities A-2.2 and A-3.8 in Figure 2.
- To ensure the sustainability of the positive results from process transition and the improvements effort, the process undergoing the transition should be continuously reviewed. A regular forum for continuous process reviews should be established and continuously revised. This lesson learned is materialized in Activities A-1.10.7, A-3.1 and A-3.3 in Figure 2.
- All the stakeholders, including managers, developers and testers, should participate in process reviews, provide constructive feedback and propose process improvements.

- The continuous process improvement activities should be incorporated into the process transition. This lesson learned has been materialized in the Activities A-3.1 - A-3.7 in Figure 2.

Our biggest lesson learned, however, is the fact that the problems that were first encountered in Scrum were not related to the Scrum method itself, but rather to its poor implementation at *CMSiPro*. Moreover, the positive results achieved by the process transition were not so much thanks to the introduction of Kanban principles, but rather thanks to introducing an appropriate forum for continuous process reviews and improvements.

Effective implementation of the mechanisms for continuous process review and improvements can improve the process and bring fast and positive results without any process transition or new method adoption. Introducing the mechanisms for continuous process improvements is, however, not enough. All the stakeholders involved in the process have to be adequately trained, mentored and coached, as well as they have to commit to continuous process improvement. For this reason, ***the main conclusion of this study is that two pivotal elements for solving process related and/or organizational problems are 1) to establish mechanisms for continuous process improvement, and 2) to have well-trained and committed personnel. Just those two can alone improve the process and bring long lasting and sustainable results, with or without transitioning to or adopting new methods.***

References

[1] S. W. Ambler, "Agile adoption rate survey results," February 2008. Retrieved January 17, 2011, from: http://www.ambysoft.com/surveys/agileFebruary2008.html

[2] D. Coghlan and T. Brannik, Doing Action Research in Your Own Organization, 2nd ed. SAGE publications, London, 2005.

[3] L. Corey, Scrumban - Essays on Kanban Systems for Lean Software Development. Modus Cooperandi Press, 2009.

[4] T. Dybå and T. Dingsoyr, "Empirical studies of agile software development: A systematic review," J. Inform. and Soft. Tech., 50(9-10), 2008, pp. 833-859, doi: 10.1016/j.infsof.2008.01.006.

[5] M. T. Hansen and H. Baggesen, "From CMMI and isolation to Scrum, Agile, Lean and collaboration," Proc. Agile 2009 Conf., IEEE Press, 2009, pp. 283-288, doi: 10.1109/AGILE.2009.18.

[6] G. Hofstede, Culture's Consequences: Comparing Values, Behaviors, Institutions and Organizations Across Nations. SAGE publications, Thousand Oaks, 2001.

[7] R. Jochems and S. Rodgers, "The rollercoaster of required agile transition," Proc. Agile 2007 Conf., IEEE Press, 2007, pp. 229-233, doi: 10.1109/AGILE.2007.59.

[8] H. Kniberg and M. Skarin, Kanban and Scrum: making the most of both. C4Media Inc, USA, 2010.

[9] M. Laanti, O. Salo, and P. Abrahamsson, "Agile methods rapidly replacing traditional methods at Nokia: A survey of opinions on agile transformation," J. Inform. and Soft. Tech., 53 (3), 2011, pp. 276-290, doi: 10.1016/j.infsof.2010.11.010.

[10] J. Li, N. B. Moe, and T. Dybå, "Transition from a plan-driven process to Scrum – Longitudinal case study on software quality," Proc. Int. Symp. on Empirical Software Engineering and Measurement (ESEM'10), ACM, 2010, pp. 1-10, doi: 10.1145/1852786.1852804.

[11] R. Martens, "Five reasons why CIOs should consider Agile development", August 2010, Rally Software. Retrieved January 17, 2011, from: http://www.rallydev.com/agileblog/2010/08

[12] Microsoft, "Team Foundation Server 2010," 2010, Retrieved January 17, 2011, from: http://msdn.microsoft.com/en-us/vstudio/ff637362

[13] M. D. Myers, Qualitative Research in Business & Management. Sage Publications, London, 2009.

[14] N. Nikitina and M. Kajko-Mattsson, "Historical perspective of two process transitions," Proc. Int. Conf. on Software Engineering Advances (ICSEA'09), IEEE Press, 2009, pp. 289-298, doi: 10.1109/ICSEA.2009.49.

[15] N. Nikitina and M. Kajko-Mattsson, "Developer-driven big-bang process transition from Scrum to Kanban," Proc. Int. Conf. on Software and Systems Process (ICSSP'11), ACM, 2011, pp. 159-168, doi: 10.1145/1987875.1987901.

[16] G. Roche and B. Vaquesz-McCall, "The amazing team race – a team based on the agile adoption," Proc. 2009 Agile Conf., IEEE Press, 2009, pp. 141-146, doi: 10.1109/AGILE.2009.67.

[17] C. Schwaber, G. Leganza, and D. D'Silva, The Truth about agile Processes: Frank answers to frequently asked questions. Forrester Research, 2007.

[18] K. Schwaber and M. Beedle, Agile Software Development with Scrum, 1 ed. Prentice Hall, 2001.

[19] H. Svensson and M. Höst, "Introducing an agile process in a software maintenance and evolution organization," Proc. Conf. on Software Maintenance and Reengineering (CSMR'05), IEEE Press, 2005, pp. 256-264, doi: 10.1109/csmr.2005.33.

[20] W. M. K. Trochim, Research Methods: The Concise Knowledge Base. Cengage Learning, 2004.