The Giraf Scrum Manual

Manual for Managing a Student Multi-Project in a Scrum of Scrums Environment

Aalborg University

Preface

This manual was originally written by the Scrum Masters of the Giraf multi-project in the spring semester of 2016. The Scrum manual is not the quintessential way of implementing Scrum in a multi-project and should rather serve as guidance based on previous experiences.

Acknowledgement

We want to give special thanks to Henrik Olsson Dalum, project manager (Scrum Master) at Consignor (A software company, with HQ in Oslo, Norway) who contributed with suggestions for improvements to the Scrum methods.

Authors

This Scrum manual is originally written by:
Group Sw613f16:
Casper Møller Bartholomæussen - cbarth13@student.aau.dk
Kaj Printz Madsen - kpma11@student.aau.dk
Martin Raunkjær Andersen - marand13@student.aau.dk
Rene Mejer Lauritsen - rlauri13@student.aau.dk

Editors

There are currently no editors, but we hope that in the future, students improve the development method and make new versions of the manual, writing about these improvements.

License

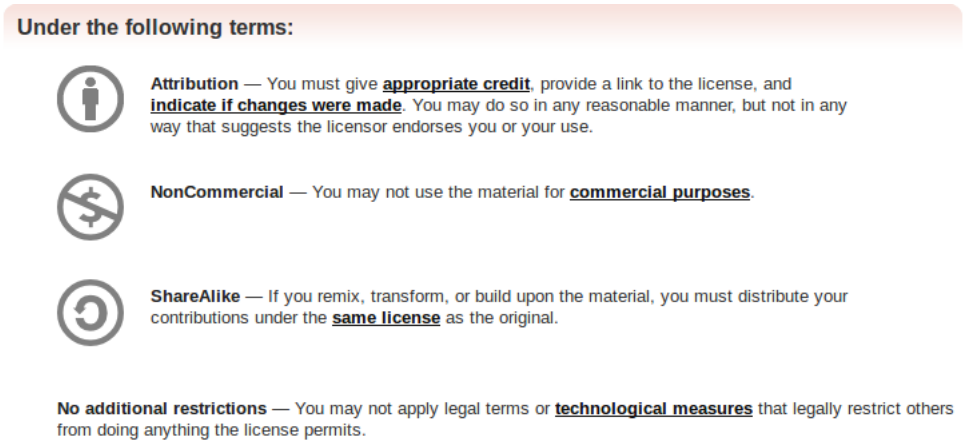

Figure 1: The condensed version of the license

Reading guide

In the manual, there are several *Do* and *Avoid* boxes. These boxes represent recommendations and are based on previous Scrum Master experiences.

> **Do** This box is an example of a Do box

> **Avoid** This box is an example of an Avoid Box

The *Do* boxes represent a good advice which you should consider doing and *Avoid* boxes correspondingly represent pitfalls which you should avoid doing.

Keywords

- **Burn-down chart:** A diagram used for checking progress in a Sprint.
- **Citizen:** Child or adult diagnosed with Autism Spectrum Disorder (ASD)
- **Development team:** All the groups in the Giraf multi-project
- **Guardian:** The legal or institutional guardian of a citizen
- **Task:** Is an assignment that, when completed, brings the project closer to completing a user story. A task is a subset of a user story.
- **User Story:** Used to define a feature in a sentence or two from the perspective of a stakeholder who wants the feature. User stories typically has the following format: As a <type of user>, I want <some goal> so that <some reason>.

Contents

1. Preliminary

1.1 Introduction

Graphical Interface Resources for Autistic Folk (Giraf) is the software multi-project on which this manual is based upon. Giraf has been developed as a multi-project for students at Aalborg University as a part of the 6th semester. Giraf consist of different apps which strive to make the life of caretakers (guardians) of people with Autism Spectrum Disorder (ASD), and the citizens with ASD, easier.

In the spring of 2016, group sw613f16 joined the Giraf development team. At that point, the Giraf project had been under development for more than five years, and was yet to be properly deployed for use by citizens and guardians. Based on experiences and recommendations of development team from the previous semester, it was decided to structure the multi-project as a *Scrum of Scrums*, with a group acting as Scrum Master and one as the Product Owner. Group sw613f16 took on the responsibility as Scrum Master, and throughout the spring semester, they tried to streamline the development process of Giraf. This manual is based on this experience, and can be used to quickly get started using Scrum of Scrums in a semester multi-project for software students (read more about Scrum of Scrums and the group structure in Section 2.1).

This manual is not the be-all end-all definitive way to manage Giraf, but rather a collection of recommendations, experiences and pitfalls. Feel free to modify and continue the development of the process, to better fit your needs. We highly encourage the continued improvement of this manual, so it stays relevant to later projects. If you (most likely Scrum Master) feel like you have improved upon the methods described in this manual, feel free to make a new version, which includes your changes. The continued updates to the manual will hopefully help the Giraf project in the long run.

2. Group Structure and Organisation

You are about to start on an epic journey, together with your fellow students. A journey that consists of collaboration, meetings and probably a lot of frustrations based on the existing code base; you are about to start on a multi-project. A multi-project needs a certain kind of management in order to be productive. This is why you will be needing a solid structure and a way of organising the groups in the multi-project, which is what we will be doing in this chapter.

2.1 Group Structure

A critical part of starting a multi-project well, is choosing a group structure that fits. This section explains and evaluates the group structure employed by the Giraf development team of 2016, aiming to give the future developers of Giraf some perspective into choosing a group structure, even if they do not make the same choices we have.

We chose to go with a *functionality over responsibility* approach. In other words, every group was free to work on whatever task that could be found on the backlog. This was mainly chosen as a response to difficulties experienced by former semesters, which were organised in a sub-project structure. They had three sub-projects: Build & Deployment, GUI and Database. A group belonging to one sub-project could not take tasks belonging to the domain of another sub-project. This caused some time costly obstacles and frustrations.

With the *functionality over responsibility* approach, we hoped to avoid these obstacles and frustrations between the groups, by allowing groups that were blocked by other groups to take on the blocking tasks themselves. However, this approach brought its own set of concerns:

- **Less responsibility:** If groups work on various tasks all over the system, the responsibility they feel towards any single component may decrease, as they only share a little part of the responsibility for this component's quality, and may never see code from the component again after completing their task. This could lead to bad makeshift solutions and bad code quality.
- **Differing interests:** By giving groups the choice of what they want to work on, it may cause a situation where numerous groups want to work on the same parts of Giraf, even though there are other more highly ranked assignments available.
- **Less speciality:** If some groups take tasks from a number of different components, their expertise and familiarity with each part of the components inevitably decrease.

How we alleviated these concerns will be the focus point of the rest of this section. In retrospective, it might not have been necessary to do away entirely with the sub-project group structure, to receive the benefits of *functionality over responsibility*, as long as there exists procedures for preemption of tasks and easy restructuring of the sub-project structure to counter shifting prioritisation.

Do Allow for easy switching of group work domains, to focus on critical domains.

Do Allow for preemption of tasks from overworked groups.

2.1.1 Responsibility Areas

One of the problems we foresaw with the *functionality over responsibility* approach, was the possibility that a degree of specialisation would disappear, since groups were allowed to take a broad range of tasks within multiple domains. To keep an amount of specialisation, and to make the transition period for groups new to a domain shorter, we did two things: We advocated for groups to try to pick tasks mainly within the same domains, and we came up with the concept of *responsibility areas*.

In *responsibility areas*, a number of important areas of expertise are identified. These areas are then split between the groups. Sometimes an area, like server management, is large enough for two groups or more to be assigned to it. Sometimes areas are small enough, that one group can handle more than one. When a group is assigned to an area, that group is required to specialise in that area. This does not mean that they must take tasks within this domain, but it does mean that they have to keep up-to-date on the status and specifics of this domain, so they quickly can solve upcoming problems within the area. Furthermore, they need to be able and available to answer questions about the area.

Do Make sure that all groups are easy to make contact with. Suggestion: Every group are required to have at least one member available on the chosen communication medium between a time range each work day.

Responsibility areas support the *functionality over responsibility* concept, because it allows groups that has taken tasks inside a new domain to skip a part of the learning curve, by having continuous contact with a group that has existing knowledge within the new domain.

In 2016, we identified the areas of responsibility seen in Table 2.1. If subsequent Giraf development teams follow a group structure similar to ours, they should probably try to identify new areas, as the state of Giraf, and consequently the necessary areas of responsibility, may have changed. For instance, we had two groups assigned to server maintenance as their main responsibility, because extensive server updates and migration were done this semester. It is recommended to have at least two server groups (they can both have extra responsibilities), since the servers are very central to the project, and it is more likely that one group is available, if something were to break on the server-side.

The later parts of this manual will explore some of the more management oriented responsibility areas: The Product Owner role (Chapter 3) and the Scrum Master role (Chapter 4).

2.1.2 Peer Review

A risk that the dissolution of sub-projects was thought to bring, was that the responsibility that each group felt for a domain, and consequently the quality and care with which they produced code would decrease, because there may be cases where groups would work on a specific domain only once.

To combat this risk, a peer review system was put into practice. According to this system, when a team member is finished with a task, a least one other member has to review it, by compiling the code and testing it manually, as well as looking through the code, looking for agreed upon code standard violations. Only when the code is accepted by the reviewers, can it be pushed to the shared repository. It is up to the member who writes the code to find reviewers. It is encouraged to find

Responsibility area	Responsibility
Server & Database	Ensure server stability, handling of server permissions and server updates
Git	Make sure git is running standby to help with git issues
Phabricator	Ensure Phabricator stability and provide Phabricator guides and help
Jenkins	Make sure Jenkins is running and provide help with related issues
Product Owner	Have contact with the partnering institutions and prioritise the backlog
Scrum Master	Devise and ensure development methods are implemented, facilitate meetings and help prioritise the backlog
Safety	Research the requirements from the Danish Act on Processing of Personal Data
Graphics	Create a graphics guideline and make new pictograms upon request
Unit and Integration Tests	Establish unit and integration test rules and wiki guides about these
Usability Tests	Conduct usability tests
Documentation	Create rules code documentation and migrate the Redmine wiki to Phabricator
Social/HR	Arrange social events for building up relations between the groups
Google Analytics	Have access to the Google Analytics account and bug reports
Sound	Prime responsible for how sound should be handled in Giraf

Table 2.1: Description of the responsibility areas identified by the 2016 Giraf team

reviewers outside the group, but this is not necessary. The reviewers should at least be people who have not been working on the code that is being reviewed. In the last part of the semester, the group assigned to the documentation responsibility area was automatically added as reviewers to each task, to make sure that everything was documented correctly.

Do Use the peer review workflow built into Phabricator[1]

This peer review system worked well, because it added an element of responsibility for the code, since programmers associated their names with a piece of code. Also, it ensured a high code quality and a low number of bugs, as many bugs and standard violations were caught before they were pushed to the shared repository.

In retrospective, the risk of low quality caused by low responsibility was probably unfounded. It turned out, that most of the groups preferred to stay on as few domains as possible, because it was easier to be productive in a known code base.

But peer reviewing is still a good idea to implement, because of the increase in code quality it brings.

[1]The Giraf Phabricator project is found at http://web.giraf.cs.aau.dk

Do Use peer reviews. It makes discovery of bugs in the code more likely, saving time in the long term.

All in all, we think that the *functionality over responsibility* approach has served us well. This is no doubt due to our willingness to actively evaluate and change our processes, to optimise work flow. We have also been actively seeking feedback from the other groups, and acting decisively on that feedback. We think this has reduced frustrations with the work flow changes, as the groups can see immediate reactions to the feedback they bring to the table, and that they have a say in the changes. The feedback gathering mechanics are explained in the meetings chapter (Chapter 5).

Do Decide upon work flow procedures together as a team. People are more likely to follow a procedure if they know why it is necessary, and some procedures, like peer reviewing, becomes ineffectual if not everyone follows or understands them.

2.2 Backlog

One of the key parts of Scrum is the backlog. The backlog (also found at Phabricator) is a prioritised list of the system requirements for Giraf. Consider it as the big to-do list for the whole Giraf project. The Product Owner manages the backlog, prioritising the items, which in the case of Giraf is in the form of user stories and bugs. A Sprint also has its own backlog, a so-called Sprint backlog, which contains all the tasks that should be done during the Sprint. Each group individually has a group backlog, as well as a group Sprint backlog. The Product Owner by means of the backlog decides what should be done in a Sprint (more about the Product Owner in Chapter 3).

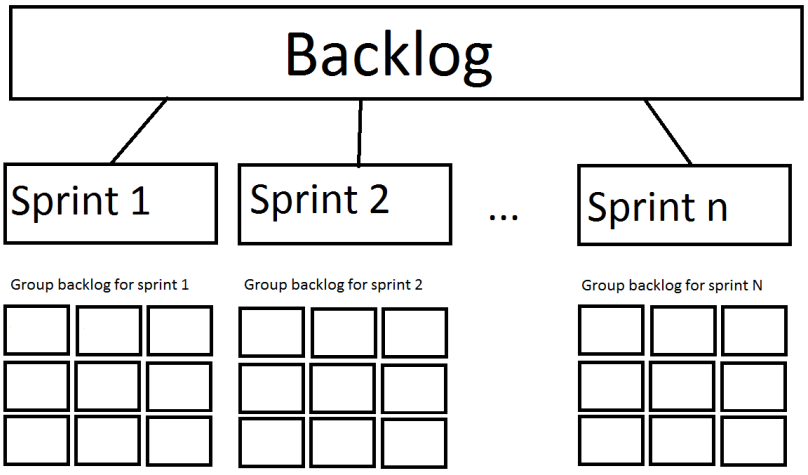

Figure 2.1: The structure of the different backlogs

At the beginning of a Sprint each group is assigned some user stories, which they must resolve during the Sprint. This happens at the Sprint Planning meeting (see Section 5.2). A user story is a way to describe a sequence of actions that needs to be possible in the system. User stories are typically expressed in the following form: "As a <type of user>, I want <some goal> so that <some reason>." The "<some reason>" part of a user story helps other people understand why the user story is needed. In our experience most people did not express that part of the user story, but they did often write a nice description of the user story, so it was actually not needed.

Before a user story can be marked as completed, its subtasks have to be completed as well. A subtask is a subset of the user story, which describes a part required to resolve the user story. A user story can consist of one to many subtasks. Subtasks for an assigned user story are usually decided

by the development group who is assigned the user story. The process of assigning user stories and creating subtasks is detailed in Section 5.2.
Below is an example of a user story and two of its subtasks:
User Story: "As a user I want clear instruction on what I can do when I am editing tracks in Voice Game."
Subtask: "Implement the help button for the edit track screen."
Subtask: "Create a help option"

2.3 Communication

Communication plays a central role in any software project. The development team should decide on a communication platform that is used by all teams. The platform should let you quickly send messages between groups, as well as individual persons. Ideally, the platform also has features to post announcements about things such as meetings, server updates and general information relevant to the entirety of the multi-project.
Some might be inclined to use Facebook as a communication platform, but this is not advisable. Facebook works as an excellent communicating tool between friends, but it is not well suited to be used in a professional environment. Facebook is not well suited since most students already use Facebook to communicate with friends and family. The messages from friends and relatives would only distract them from their work on Giraf. The Giraf team of 2016 has had good experiences with Slack[2], as the main tool of communication.

Avoid Using common social media, such as Facebook, as the primary means of communication.

Rules

When you have decided upon a communication platform, the next thing to discuss are the rules in regards to both the selected platform, but also the general way of communication.

To get you started, think about how to answer the following questions:

- At what times should people be available for communication?
- Where should information about meetings be posted?
- Should all developer groups have their responsibility area in their name?
- Should user names follow a certain naming scheme?
- Does users have to use their personal picture as profile picture?

2.4 Sprints

Scrum works by dividing a period of development up into sequential Sprints. Many Scrum sources would advise the Scrum Masters to only plan one Sprint ahead, but the context differs seeing as this is a multi-project environment [6] and students follow courses as well. This different context is why the number of Sprints needs to be settled early. By doing so it will also be easier for the customers and the Product Owner to plan meetings.

Avoid Being strict on your initial Sprint dates.

When preparing the length of a Sprint, there are a lot of factors that play into settling on the dates, which in many cases require the Scrum Master to be flexible with the already planned Sprint dates. In 2016, we had a total of five Sprints, consisting of an initial Sprint and four regular Sprints.

[2] https://slack.com/

The purpose of the initial Sprint was to get familiar with the development environment as well as look for potential bugs, though it was not permitted to make changes to the code. This form of initial Sprint was not very effective, and many people in the team agreed, that it was mostly a waste of time, since we looked at random domains and could not make any changes to them. An initial Sprint may be effective, if it at least allowed for clean-up in code and the domains were of relevance to each group. There are a lot of unused dependencies in Giraf currently. It would allow the teams to focus better when coding, if all this clutter was gone.

Avoid Arranging an initial Sprint that do not permit code changes.

2.5 Goals

Having goals helps guide the development of Giraf and lets you assess the development progress. The goals should be established early and should be a list of specific items that you commit to finishing by the end of the semester. The creation of these goals should be based on customer wishes, and with the entire development team participating.

In 2016, the following goals were agreed upon:

- Test code coverage of ∼75% for new code and 100% for system critical code
- A high level of stability
- The following applications should be ready for customer testing:
  - Launcher
  - Week Schedule
  - Picto Search
  - Voice Game
- Make *Voice Game* stand-alone from the *Launcher*
- Improve the way Giraf handles personal data, so it adheres to the Danish Act on Processing of Personal Data. [3]
- Documentation of all new code

The goals listed above were not all completed in a satisfactory manner. The *test code coverage of ∼75% for new code and 100% for critical code* goal turned out to be unfeasible, because of existing application architecture, which did not facilitate testing. We were not able to reach the goal of *No sudden crashes* as there are still sporadic crashes in Giraf, but fewer than when we started.

Avoid Picking unrealistic goals.

We did not initially have goals, in 2016, to help guide the development. We more or less just started programming on the highest prioritised user stories and ended up working on different areas of Giraf in an unstructured way, instead of concentrating our efforts in an efficient manner. This lack of a goal-driven development was recognised early and we immediately fixed it at the next Scrum meeting.

Do Consider selecting applications to concentrate development efforts on.

[3] More information about the act at https://www.retsinformation.dk/Forms/R0710.aspx?id=828 and http://eur-lex.europa.eu/legal-content/DA/TXT/HTML/?uri=CELEX:32016R0679

Collaboration between Scrum Master and Product Owner

The Product Owner has the final say when determining the goals of a Sprint and what user stories are included in the Sprint backlog. Having said that it is a good idea if the Scrum Master collaborates closely with the Product Owner since they have knowledge about how much work is feasibly completed by the development team in a Sprint.

A close collaboration between the Product Owner and the Scrum Master is vital to getting a multi-project like Giraf running smoothly and these two parties should meet and discuss at least once each Sprint.

3. Product Owner

The Product Owner role is essential to Scrum and is one of the larger responsibility areas. As the Product Owner in Giraf, you are the link between the customers and the development team, as well as the deciders of the development direction.

Most of this "Scrum Manual" is based on first-hand experience from the Scrum Master group of 2016, but much of the information in this Product Owner chapter can be attributed to the Product Owner group of the same semester (SW614).

3.1 Responsibilities

Some of the important Product Owner responsibilities are listed below:

- Stay in contact with the customer
- Arrange customer meetings
- Represent the customers' interest
- Set goals for the development
- Manage the backlog

Stay in contact with the customer

The most important aspect of being Product Owner is that you are the only with direct contact to the customer. As the Product Owner you are the medium where every message between the development team and customer is passed. The customers from the collaborating institutions are hardworking people, who do not necessarily check their email daily. So do not expect instantaneous feedback from the customers.

> **Avoid** Sending out emails that require a quick response from the customers.

Arrange customer meetings

At the end of every Sprint, the Product Owner meets with the customer, and present the new functionality that has been added to Giraf during the recently ended Sprint. The customer meeting is a corner stone of Scrum , as it delivers direct feedback on the project and gives an opportunity to ask the customers specific questions. The customer meeting is further described in Section 3.2.

Represent the customers' interest

As a team we continue to work on Giraf, because we know there is an external interest in the project. It is important that we do not only develop the parts that interests the development team, but try to satisfy the wishes from the customer. The Product Owner should represent the customers' interest and deliver their requests to the development team.

Set goals for the development

To ensure that the team is working on domains that are in the customers' interest, the Product Owner must set a number of goals for the semester and each Sprint. The semester goals, as the name indicates, span the whole semester and can be general e.g. "Have *Week Schedule* in a finished state, where it can be used by the collaborating institutions". The Sprint goals should be a subset of the semester goals, so by completing the Sprint goals, the team is getting closer to completing the goals of the semester. The Product Owner defines the Sprint goals, but should contact the Scrum Master group, who can agree on them or question their importance to the semester goals. Section 2.5 further describes the importance of goals and the collaboration between Product Owner and Scrum Master.

> **Do** Limit the amount of goals to a few domains that are in the customers' interest.

Manage the backlog

It is the Product Owner's responsibility to manage the tasks and user stories in the backlog. The backlog can become overwhelmingly large, so unnecessary items has to be pruned and reprioritised now and then. Between Sprints would be a good time to do this. The Product Owner creates user stories based on the feedback from the customers, but not all user stories originate from the customers. For instance, usability tests also return a lot of great feedback, so the developers should also be allowed to add new items to the backlog. The Product Owner has an overview of the backlog, so when a new item is added, they should approve it to avoid duplicate, obsolete and badly phrased user stories. When approving user stories and tasks, the Product Owner can also prioritise them according to the importance towards the goals.

> **Do** "Refactor" the backlog by reprioritising user stories/tasks and clean up duplicates and obsolete user stories/tasks.

3.2 Customer Meeting

The attendees of this meeting are the Product Owner and the customers. We highly discourage other members of the development team from attending the meeting, since the meeting should be kept small and intimate, making it easier for the customers to speak their mind.

The meeting is held at the end of each Sprint, so the customers can get progress updates on the newly added features and provide feedback to these. This feedback can be turned into user stories, that may be finished in the upcoming Sprints.

You cannot expect all customers to be represented at each meeting. From our experience it varies a lot e.g. At the first meeting there were five out of six representatives, second Sprint only three, third Sprint four representatives and at the presentation at the end of the semester, only two out of the six contacts were available. It can be very hard to find a date between each Sprint end and start, where the customers are also available. Also, from our experience, the meetings are usually around one and a half hour to two hours.

The rest of this section is split into three parts. Each part respectively describes things to do or remember before the customer meeting, at the meeting and after the meeting.

Before the meeting

Finding a date for the meeting can be troublesome, since it has to fit in-between a Sprint end and start. The Scrum Masters schedule Sprint dates (Section 4.1), so collaborate with them, and find a range of dates where the customer meeting can be held. Make sure to find the dates early in the Sprint or perhaps before it starts, since there is a greater chance of the customers being available, if

they receive the proposed dates weeks before the meeting. The proposed range of dates should be sent out by email to all customers.

Do Use Doodle [1]to find a date where the most customers are available (Do not expect everyone to answer it)

When you have found a fitting date, send out a confirmation email with date and time to all customers. Even if not everyone has responded to the proposed days, they might still make it to the meeting.

Do Send out email reminders with the time and date of the meeting at least a week before the meeting but also a few days before.

The Product Owner represents the development team at the meeting, so it is important you are up-to-date on the latest developments and ask the questions that the team would like answered. A great way of getting up-to-date, is to contact each group, preferably in person, and hear what they have done the last Sprint and if they have any questions for the customer.

Bringing a tablet to the meeting is essential, since this will let the customers experience the worked upon features firsthand. This tablet should have the latest versions of the applications/libraries that you would like to talk about at the meeting. If some features are not completed yet or are very unstable, you can bring screenshots.

Do Bring at least one backup tablet with the latest versions of the applications that you present at the meeting.

At the meeting

At the meeting you can mirror the tablet screen onto a projector, making it easier for everyone to see what is going on[2]. While going through each new feature, allow the customers to comment on it, to hear if they are satisfied or if the feature could use a few modifications. After showing the new features, you can ask the questions that you have collected from the development team. If the questions are related to a new feature, allow the customers to try it out on a tablet (another reason to bring more than one tablet). At the end of the meeting, you can discuss what should be done in the upcoming Sprint.

In general, try to keep the discussion focused on Giraf and not on other tools the customers use. Remember to have a person writing a summary during the meeting, which should be shared with the development team. Also remember to use the correct autism domain terms, otherwise you run the risk of saying something that can be interpreted as offensive.

After the meeting

There are only a few things to do after the meeting, but you are on a tight schedule, since they are expected to be ready for the upcoming Sprint Planning meeting. The meeting summary needs to be shared with the team, depending on the tools you use, make it available on a shared repository/drive and tell people it is available using the communication tool of choice.

For the next Sprint you need to have a set of goals ready, including a set of user stories that you and the customers would like to be finished during the Sprint. This should be your highest priority, since the rest of the team depends on this.

[1]http://doodle.com - A tool for scheduling among groups of people

[2]We have used MirrorGo with success: http://www.wondershare.net/android-mirror/

4. Scrum Master

Being Scrum Master in Giraf is a great responsibility, that requires a lot of work. The Scrum Master role is pivotal to the Giraf Scrum of Scrums structure, since it enforces the methods and techniques of Scrum.

The foundation of the role is based on the description in "The Scrum Guide" [2], but has been further adapted to fit the needs of Giraf. This chapter describes what the role of Scrum Master in Giraf entails.

4.1 Responsibilities

As a Scrum Master in Giraf you have a range of different responsibilities to tend to. The most important of these responsibilities are listed below:

- Ensure adherence to Scrum methods.
- Arrange and conduct meetings.
- Schedule dates for Sprint start and end.
- Evolve the Scrum methods to fit the environment.
- Collaborate with the Product Owner on Sprint goals and backlog.
- Keep the development team satisfied and busy.

Ensure adherence to Scrum methods

What is the purpose of using Scrum, if you do not follow and understand its practices and methods? As the acting Scrum Master you must help the development team understand the chosen development methods, and make sure that the methods are stuck to. Examples of what is required by the team: Be present at meetings, add related items to the backlog and work towards Sprint goals. Some of the following listed responsibilities further help the team in adhering to Scrum methods.

Arrange and conduct meetings

If the team must adhere to the development methods of Giraf, then it is crucial to arrange frequent meetings where the team can discuss the employed methods. It is the Scrum Master's responsibility to arrange these meetings, and they should in most cases also conduct them. Chapter 5 goes into great details of different Scrum meeting types, and how they can be arranged.

Schedule dates for Sprint start and end

Besides scheduling the Scrum meetings, the Scrum Master should also schedule the dates for when the Sprints start and end. It can be surprisingly tough to find appropriate dates since you must consider things like courses, holidays, customer availability and Sprint length. A semester has a fixed length, and you can as such initially schedule preliminary dates. The dates for each Sprint might have to be rescheduled, but avoid doing it too often, as it can interfere with people's plans.

Do Ask the development team if they have any objections to the scheduled Sprint dates.

Evolve the Scrum methods to fit the environment

A significant advantage of Scrum is how the project evolves based on frequent feedback from the customer. The feedback from the client is essential to improving Giraf, but equally important is the improvement of the development methods. These methods are what keeps the development team functioning, and they should evolve to fit the ever-changing development environment. The Sprint Retrospective meeting (Section 5.4) is a great place to collect feedback from the development team. The Sprint Planning meeting described in Section 5.2, is an example of a meeting that has been improved in response to feedback from the development team.

Collaborate with the Product Owner on Sprint goals and the backlog

As mentioned in Section 2.5, having goals keeps the development on track and focuses the development efforts of the team. The goals of each Sprint should be of relevance to the primary goals of the semester. Achieving the goals of the semester is one of the Product Owner's primary objectives, but the Scrum Master should help with this task as it is imperative to keeping the project on track. Section 5.2 discusses how the Product Owner and Scrum Master can collaborate on deciding Sprint goals.

Do Arrange a meeting between Scrum Master and Product Owner before each Sprint Planning, to discuss the goals of the upcoming Sprint.

Keep the development team satisfied and busy

The development team is the core of the project, so it is vital that they are happy and satisfied. The Scrum meetings (Chapter 5) and especially the Sprint Retrospective meeting should encourage feedback from the team. The Scrum Master should place great value on this feedback and ensure that it is taken into account when reiterating on the management methods. Section 2.1 describes a group structure, which allows groups to work on domains they find interesting. Working on exciting domains should to some degree keep the team satisfied. The Scrum Master should try to keep the groups busy with user stories and tasks to prevent inaction. The Weekly Scrum (Section 5.1) can be used to keep track of how far groups has progressed with their assigned user stories; are the groups out of work or perhaps too busy? In both cases, the groups are encouraged to contact the Scrum Master, who may be able to find a group that can finish the extra work before Sprint end.

4.2 Progress Monitoring

Supervision of the development progress is necessary to assess the workload of the development team. Different tools exists to track progress and one of them is built directly into the Phabricator platform. The tool supplied by Phabricator is a burn-down chart generation tool which produces graphical representations of how fast the team is 'burning' through the user stories. An example of correct use of the burn-down chart is shown in Figure 4.1.

Burn-down charts are only useful, if all groups understand how to use them. You can arrange a knowledge sharing meeting, where a person guides the team on the purpose of burn-down charts, and how they should be used. Figure 4.2 is a an example of a burn-down chart, where the group did not periodically mark their user stories and tasks as 'finished', but waited until the last day of the Sprint. Such a burn-down chart can naturally not be used to track the progress incrementally which is the main purpose of burn-down charts.

Burn-down charts should ideally only be a complimentary method of monitoring progress during a Sprint since their use might be faulty.

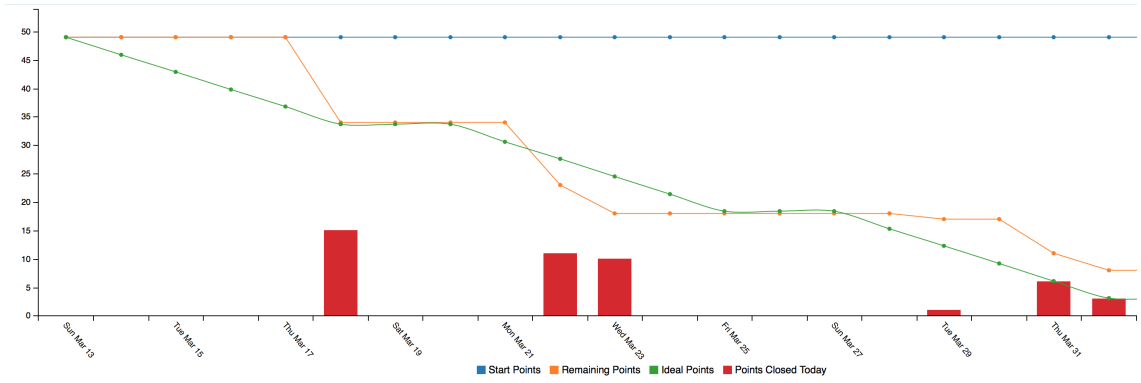

Figure 4.1: Good use of burn-down chart

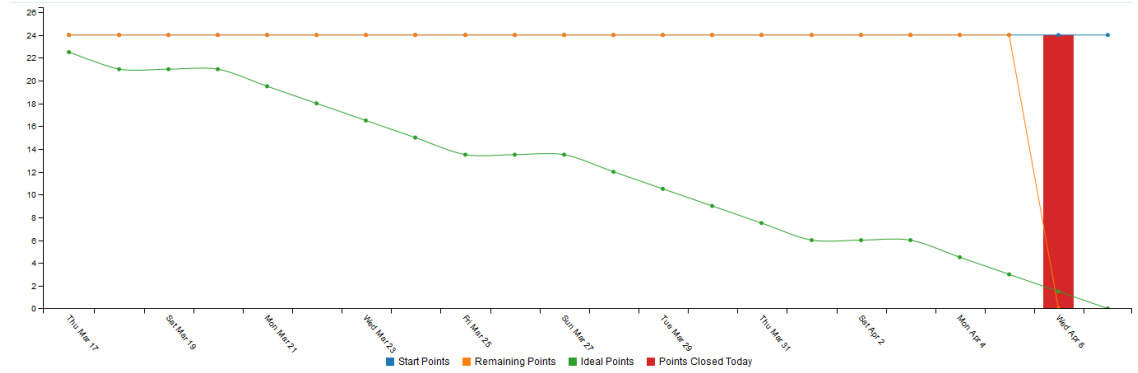

Figure 4.2: Bad use of burn-down chart

Do Instruct the teams on how to properly use their burn-down charts.

The Weekly Scrum meetings can also be used to monitor progress. The meetings are a great way of getting the needed information about progress, since people talk about what they have done during the last week, and if they expect to finish their assigned user stories.

You should, however, not repeatedly ask the groups about their current progress. This repeated questioning will only make the groups annoyed and will potentially lower their efficiency. In some individual cases, it can be necessary to ask about the progress continuously, but it should only be if, for instance, the server is down, or if somebody is working on a crucial part of the system. This rule does, however, not mean that you cannot follow-up on user stories or tasks, just have in mind that you are occupying the valuable development time of the groups.

Avoid Unnecessarily asking groups about their progress outside of Weekly Scrum meetings.

5. Scrum Meetings

Meetings are an important aspect of managing a Scrum team, and can be used for multiple different purposes. In the upcoming sections, we write about a few different types of meetings, that each serves its purpose, but they also serve some common objectives.

The primary objective of a meeting is to keep the development team up-to-date on the latest developments. Everyone on the team should know how the project progresses towards the common goals, and if changes are made to any shared agreements e.g. usability tests have been cancelled or changes are made to scheduled meetings. Some information may be urgent, and instead of waiting for a meeting to announce it, use the agreed upon communication channel (see Section 2.3).

Do Use meetings to remind people of upcoming deadlines.

Meetings should be an open forum, where each group has at least one representative present, and people can ask questions about any domain or decision. It is vital that people have the opportunity to ask questions. People also have the chance to ask for help, since it is very likely that a person at the meeting can help with the problem, or knows of a person who has had the same challenges. Asking for help may even facilitate larger collaborations between groups. Sometimes a discussion between groups is warranted, so before a meeting ends, it is important that the Scrum Masters ask whether any group wants to talk with some other group since people may leave the room quickly after the meeting ends.

As the Scrum Masters, you must schedule each meeting. It is a good idea to schedule and announce temporary dates for the important meetings like Sprint Planning and Review early in the semester. Final meeting dates should be on a per-Sprint-basis, meaning that you should only plan meetings for one Sprint at a time. The scheduled meetings should be announced to the team by Sprint start, possibly at the Sprint Planning meeting. The greatest challenge of scheduling meetings is that they have to fit among the courses people attend. This may cause some Sprints to be longer and meetings to be scheduled on a Friday afternoon. If scheduling conflicts and anomalies appear, find a temporary date, and ask the team if they can accept it.

Do Schedule meetings to fit bus schedules e.g. 09:15 instead of 09:00.

Even though we highly recommend arranging frequent meetings, it is important to avoid spending too much of the development time on meetings. Experience tells us that each meeting becomes shorter as the semester progresses, since people know the structure of the meeting and the Scrum Masters have experience with conducting meetings. We have had great experience with combining the Sprint Review (Section 5.3) and the Sprint Retrospective (Section 5.4) meetings to save even more time and keep up attendance. Figure 5.1 shows an example of how the Sprint meetings can be scheduled.

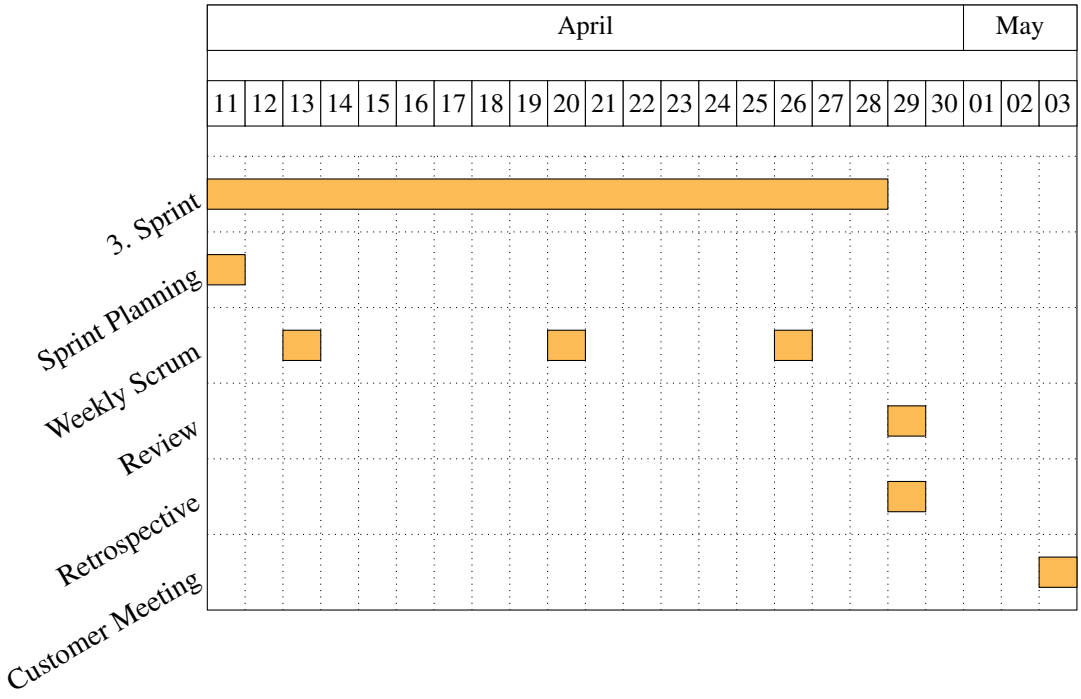

Figure 5.1: Gantt chart showing possible meeting dates during a Sprint

Avoid Spending too much time at meetings.

The role of the Scrum Masters at meetings is primarily to moderate the dialogue and make sure that subjects are kept relevant to all groups and the agenda. Secondarily, the Scrum Masters also need to take meeting notes. For this reason at least two of the Scrum Masters are in attendance at each meeting, but at most meetings it is recommended to be at least three attendees, to comfortably fulfil the above requirements.

The meeting notes should be made readily available for everyone on the team, so people who were not attending can see the highlights. The Scrum Masters should also make an agenda available for each meeting. The agendas usually follow the same structure, but groups are encouraged to add additional subjects to it. In Appendix A you can find a collection agenda template used in Giraf 2016. In Appendix B you can find a list of items to remember when you are arranging meetings.

Do Try to publish the agenda no later than one day before the meeting.

When working together in a large team, where multiple people need to accomplish the same set of actions, it can be necessary to write detailed guides, to save time and ensure that people do the things correctly. In some cases, where the actions involve the whole team, it is recommended to arrange a knowledge sharing meeting. An example of such a meeting is a tour of *Phabricator*, where the meeting conductor presents *Phabricator* and its purpose in the project. The conductor should also show how to create so called sub-projects, tasks, user stories, etc. A knowledge sharing meeting is a great introduction to a topic, but it is still recommended to have written guides for the actions presented at the meeting. Knowledge sharing meetings can be scheduled as an extension of any other meeting, just remember to let people know of the extension.

5.1 Weekly Scrum

Weekly Scrum meetings are held to ensure that things are progressing according to plan. The purpose of this meeting is for each group to inform the other groups of their progress, possible problems and also to allow the Scrum Masters to ascertain the status of the project. To keep the meetings as short and concise as possible, a rigid meeting structure is stuck to. This structure consists of four questions that each group must answer:

- What has been done last week?
- What will be done next week?
- Are there any blockades or do you need help?
- Is your work possibly going to block the work of another group?

The Scrum Masters should make sure that the representatives of each group answer each question. After each team has gone through the questions, a plenary debate will start, in which teams can bring up subjects of interest.

> **Do** Have a meeting agenda document available before the meeting, where people can add subjects to discuss.

The Weekly Scrum is inspired by the 'Scrum of Scrums' meeting described in [3], and as the name suggests, this type of meeting is scheduled every week, as it gives the team time to make significant progress, that is worth talking about. The classic 'Daily Scrum' [2] (also used in 'Scrum of Scrums' [3]) works very well for small teams, who can meet daily, but for a multi-project where the students attend different courses and have no fixed office hours, it becomes very challenging to meet daily. The individual groups are free to arrange daily Scrums internally but it is neither required nor enforced to do so.

We recommend that, due to their frequency, the Weekly Scrum are scheduled at a fixed time of the week e.g. every Wednesday at 10:15[1].

> **Do** Arrange a fixed start time and day of the week for Weekly Scrums.

We recommend that every member of the Scrum Master group gets experience with conducting a Scrum meeting. The Weekly Scrum is well-structured and an ideal candidate for a 'rotating conductor' system - new Scrum Master conductor every week, repeat the rotation when every member has done a conduction.

5.2 Sprint Planning

The Sprint Planning meeting is essential to any Scrum team. The meeting is important since it is used to assign user stories to development teams.

The Sprint Planning meeting structure we developed in 2016 is inspired by the instructions outlined in "The Scrum Guide" [2]. We do however modify the method to better fit the development of Giraf. The guide specifies that subtask generation of the selected Sprint user stories should take place during the Sprint Planning meeting. This can be a great process, yielding good design solutions, but it does, however, take a long time which, the team was dissatisfied with. The meeting structure we decided upon specifies that the team generates subtasks after the Sprint Planning meeting. This might seem like a slight departure from the recommended method but time is still allocated at the meeting to discuss solutions and their strategies.

[1] Right after FredagsFranskbrød at Cassiopeia has been a success for us :)

Avoid Spending too much time at Sprint Planning meeting generating subtasks.

In preparation of the Sprint Planning meeting, the Product Owners selects the set of user stories in the backlog, that they want to be worked on in the upcoming Sprint. The chosen user stories should all be related to an overall goal of the Sprint, defined by the Product Owner. As a part of the collaboration between Product Owner and Scrum Master, the Sprint goal and user stories are presented to the Scrum Masters in advance. The Scrum Masters should assist the Product Owner in determining if the selected Sprint user stories fits within the constraints of the Sprint goal. More information about the collaboration between Scrum Master and Product Owner can be found in Section 2.5.
Possible Sprint goals:

- Make *Voice Game* a stand-alone application.
- Finish the development on the *Week Schedule* application.

The development teams prepare for the Sprint Planning meeting by identifying the user stories they would like to work on. If they have a possible solution for a user story , they are asked to describe the solution in the comments section of the user story in Phabricator.

At the meeting, the Scrum Master and Product Owner present the Sprint backlog and the goals for the upcoming Sprint to the representatives at the meeting. Together, the representatives and Scrum Master go through each user story appointed to the Sprint and discuss the possible solutions that might have been added prior to the meeting. The user story is assigned to a group showing interest in it and possible conflicts are handled through further discussion. This process is repeated until all user stories for the Sprint have been appointed to some group, unless there is good reason to postpone the user story to a later Sprint.

Avoid Just assigning user stories to groups. Encourage discussion of possible solutions before assigning it to a group.

This delegation process includes the risk of assigning too few or too many user stories to a group. To counter this risk, time is allocated after the user story delegation, to be used by the representatives to argue internally about whether or not the amount of user stories assigned to them is appropriate. If a group is not satisfied with the assigned user stories, they are encouraged to speak up, so the problem can be solved during the meeting.

After the meeting, each group should generate subtasks for each user story as well as give an estimate of hours the required to complete the user story. Again, if a team realises they have too many or too few hours of work, they should contact the Scrum Masters, who will then help by reassigning user stories as needed.

Do Use Planning Poker to estimate the workload of user stories and tasks. [2]

5.3 Sprint Review

The Sprint Review meeting serves as a means of keeping the development team up-to-date on what changes have been made to Giraf throughout the sprint. The meeting is held after a Sprint end and before the next Sprint Planning meeting. Our interpretation of the Sprint Review meeting is based upon [2].

[2] See more about Planning Poker at https://www.mountaingoatsoftware.com/agile/planning-poker

A Sprint Review usually includes the customers, but as a consequence of using a Scrum of Scrums project structure, it has been decided that the Product Owner arranges a subsequent review with just the customers (also called customer meeting). The developers are discouraged from participating in this secondary review, to ensure a smaller and hopefully more focused meeting. If the developers want to know about something specific from the customer, they should contact the Product Owner, who can then forward the question to the customers.

Do Arrange a secondary review (customer meeting) only for the Product Owner and the customer.

The meeting starts with small informal presentations by each group, about what they have been working on. The groups are encouraged to include a demonstration of the new features on a tablet (video feed of the interactions with the tablet is shown on a larger screen). Some features do not lend themselves to a visual demonstration e.g. server migration, REST API, encryption, etc. Instead of showing these features on a tablet, the groups are encouraged to draw figures on the blackboard, or make a very short slide show ($\sim$ 2 slides) presentation. The short visual presentation helps the rest of the team to comprehend the amount of work that has been done. The team can ask questions about the design choices of the new features, and the presenter can argue why it was done. This discussion may generate awareness of problems in an application, which may be turned into new tasks or user stories for the backlog.

The Sprint Review is also used to evaluate the user stories and tasks completed during the Sprint. The user stories and tasks assigned to a Sprint are expected to be finished in the same Sprint. The Weekly Scrum is used to monitor the status of tasks and make sure, that they are completed during the sprint. However, there may occur situations that block the completion of a user story or task. In these cases, the groups are encouraged to report back to the Scrum Masters, who will find another group to take over the user story or task. If a user story or task is still incomplete by Sprint end, the team who could not finish it should use the opportunity of the meeting to talk about why they were unable to finish it. If needed, the user story or task can then be re-prioritised.

The Scrum Master group usually arranges and conducts the meetings, but since the Sprint Review deals with the Sprint goals and backlog, the Product Owner group might be a better candidate for the role. We only have experience with the Product Owner presenting the Sprint goals, but it may ease the workload of the Scrum Master if the Product Owner were in charge of the Sprint Review.

Do Consider having the Product Owner arrange and conduct the Sprint Review.

5.4 Sprint Retrospective

The Sprint Retrospective meeting is where the development team discusses the working method and finds improvements to it. The meeting is held in the days following a Sprint end, before the next Sprint Planning meeting. The meeting is placed there since people can reflect upon the just completed Sprint and use these reflections in the upcoming sprint. The Review and Retrospective meetings are both located in the same time span. You might want to combine them into a larger meeting, to avoid having too many different meetings during this time with the Retrospective meeting following the Review meeting.

Do Consider combining the Sprint Review and Sprint Retrospective meetings into one meeting.

In Scrum you should always try to reiterate and improve upon the development method , which

is why we deem the Sprint Retrospective meeting important. Compared to the other types of meetings, the Sprint Retrospective meeting does not discuss topics related to specific tasks in the backlog. This meeting is instead used to discuss the project at a higher level of abstraction, like project structure, development method, working environment and communication. The meeting is structured to be more informal than the other meetings. This informality reflected by a more relaxed attitude of the Scrum Master meeting conductor and by having free coffee or tea. The informality is an attempt at making people more open to reflection and debate.

An important item of discussion, during the meeting, is the *start, stop, keep* topic. This discussion item is used to generate feedback from the development team. The group representatives at the meeting (including the Scrum Masters) are split into groups of 3-4 people, in which they can talk about the things the Giraf team should *start doing*, *stop doing* or *keep doing*. We have found that the best discussion usually happens when the groups consist of members from different development groups.

> **Do** Make sure that discussion groups consists of members from different development groups.

The groups are encouraged to write at least three items of the *start, stop, keep* format onto post-it notes, which are then collected on the blackboard for everyone to see, just like in figure 5.2.

Examples of *start, stop, keep* feedback:

- *Keep* having short Weekly Scrums.
- *Start* writing Giraf wiki guides.
- *Stop* doing poor two-minute code reviews.

The meeting facilitator from the Scrum Master group should walk between the discussion groups and facilitate discussion if needed. This facilitation of the discussion can be kick started by asking questions like: "How do you think collaborations between development teams are going?", "Do you wish to change anything about Sprint Planning?", etc. These types of questions can hopefully start a discussion and generate feedback.

> **Do** Have the meeting conductor kick start discussion groups with questions.

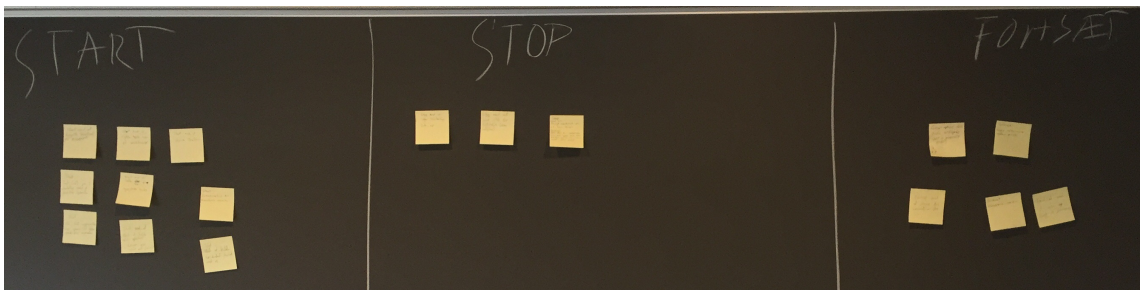

Figure 5.2: Example of the start, stop, keep (fortsæt) feedback from a Sprint Retrospective meeting

The feedback generation process is followed by a break in which the Scrum Masters read and discuss the post-it notes. After the break, the Scrum Masters read the feedback aloud, and the representatives are given a chance to explain the feedback further. Many of the changes that we made to the development process in 2016 were because of the feedback from this meeting and some of the tips and pitfalls we have included in this paper are direct feedback from the Sprint Retrospective meetings.

Do Continue doing Sprint Retrospective meetings even when things are going great.

The Scrum Masters identifies at most five critical feedback notes and turns them into *actionable commitments*. These actionable commitments are an important result of the meeting as they are clear and easy to understand directions on what to do in certain situations. The selection of only the critical feedback notes is essential as an excessive amount of actionable commitments would be overwhelming for the team to follow. The actionable commitments must be relevant to the whole team.

Examples of possible actionable commitments:

- Create a wiki guide if more than one person has to do the same multi-step task.
- Code reviews must be marked as 'Needs Changes' if the reviewer has any changes to the code.

The Scrum Masters must ensure that the actionable commitments are followed, by reminding the development teams of the currently active commitments. The team should be reminded during Weekly Scrums, in the shared wiki and perhaps through the shared Giraf communication tool e.g. a bot on a central Slack channel.

The next Sprint Retrospective meeting should include an evaluation of the degree to which the development team has followed each of the actionable commitments. If a commitment has been satisfied, it can be replaced by a new, but if it has not, it should be kept and perhaps reformulated to make its intent clearer.

The use of *start, stop, keep* is inspired by the "Effective Sprint Retrospectives" article seen here [4].

5.5 Semester Evaluation

As a last meeting of the semester, it can be very useful to arrange a semester evaluation meeting. At this meeting, the development team evaluates the degree to which the semester goals have been fulfilled. All members of the development team are encouraged to attend this evaluation since the team should have a mutual understanding of the goals' fulfilment.

One effective way to structure the evaluation meeting is by using a *World Café* format (also called *Knowledge Café*). The World Café format works by dividing people into groups (we recommend no more than ten people per group) that visit each station, where they discuss a topic together. The topics share a common theme and in this case, they are topics relevant to Giraf. Later the conclusions from these discussions are gathered and used to fuel a debate among the development team. Typically a group spends about 10 minutes at each station and then moves on to another. Each station has a table host, who introduces the group to what the previous groups discussed while also facilitating a discussion of the station topic. The structure is based on the following sources: "World Cafe Method" [5] and "How to Run a Knowledge Café" [1].

To make this work in the context of Giraf, and get the most out of the meeting, you can arrange so that each station topic are different semester goals or perhaps the applications you have been working on. At the stations, you can then discuss how well the goal has been accomplished and what future work is left. Examples of station topics used in 2016:

- Picto Reader + Search
- Week Schedule
- Voice Game
- REST + Launcher

Week Schedule

Goal	Future work

Table 5.1: Example of meeting chart arrangement

Each station has a meeting chart sheet, where the table host writes suggestions for future work or comments on the fulfilment of the goals, related to the topic at the station. You can divide the meeting charts into the two columns: Goal and Future Work (like Table 5.1).

Do Have a tablet at each station running the latest version of the topic application (If the topic involves an application).

When people are done visiting the different stations, you should gather all the meeting charts, and have a joint discussion with the development team. The table hosts present the comments from their stations and the team can discuss whether they agree with these comments. Hopefully, you end with a collective agreement on the goals' accomplishment, which people can then write about in their reports.

6. Further Reading

Before you start digging around for exciting reading material you should have in mind that one cannot just blindly apply what you learn. Books and guides talk about a different Scrum environment, often in a company, where the people in the groups works five days a week from 8-16. This environment does not match the Giraf multi-project environment, where people cannot dedicate all the time to the project due to spending time on courses etc. Having said this, we still believe that many Scrum resources modified to fit the Giraf environment can be excellent.

"The Scrum Guide™: The Rules of the Game", by Ken Schwaber and Jeff Sutherland [2], provides a concise overview of what Scrum constitutes. The guide does, however, lack some concrete examples of for instance ways of doing the Weekly Scrum meeting. To find such information/good practises you then have to go to the Internet or ask a Scrum Master near you.

If you are looking for a more general description of agile methods, then we recommend reading Chapter 3 of Ian Sommerville's "Software Engineering 10th Edition" [3].

If you are interested in reading more about the unmodified version of the start, stop, keep Sprint Retrospective meeting, then you are looking for "Effective Sprint Retrospectives", by David Starr (part of Scrum.org) [4].

If you do not feel like reading then Collab.net have made an E-learning Scrum training series which serves as an excellent introduction to Scrum. You can find it here: http://www.collab.net/services/training/agile_e-learning

YouTube also has a myriad of videos concerning Scrum. The user GroetenUitDelft, in particular, has made some funny and very informative videos, which are worth watching https://www.youtube.com/user/GroetenUit

Bibliography

[1] D. Gurteen. (Dec. 2005). How to run a knowledge café, [Online]. Available: http://www.gurteen.com/gurteen/gurteen.nsf/id/run-kcafe (visited on 05/19/2016) (cited on page 31).

[2] K. Schwaber and J. Sutherland. (Jul. 2013). The scrum guide™, The definitive guide to scrum: The rules of the game, [Online]. Available: http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf (visited on 02/25/2016) (cited on pages 21, 27, 28, 33).

[3] I. Sommerville, *Software engineering*, 10th edition. Boston, Mass. u.a: Pearson, Apr. 2015, ISBN: 1292096136. [Online]. Available: http://iansommerville.com/software-engineering-book/ (cited on pages 27, 33).

[4] D. Starr. (Jul. 2012). Effective sprint retrospectives, [Online]. Available: https://msdn.microsoft.com/en-us/library/jj620912%28v=vs.120%29.aspx#bkmk_making (visited on 03/03/2016) (cited on pages 31, 33).

[5] TheWorldCafe. (Jul. 2015). World cafe method, [Online]. Available: http://www.theworldcafe.com/key-concepts-resources/world-cafe-method/ (visited on 05/19/2016) (cited on page 31).

[6] E. Woodward, S. Surdek, and M. Ganis, *A Practical Guide to Distributed Scrum (IBM Press)*. IBM Press, 2010, ISBN: 0137041136. [Online]. Available: http://www.amazon.com/Practical-Guide-Distributed-Scrum-Press/dp/0137041136%3FSubscriptionId%3D0JYN1NVW651KCA56C102%26tag%3Dtechkie-20%26linkCode%3Dxm2%26camp%3D2025%26creative%3D165953%26creativeASIN%3D0137041136 (cited on page 13).

Index

A. Meeting Agenda Templates

This appendix collects agenda templates for the following meetings: Weekly Scrum, Sprint Planning, Sprint Review and Sprint Retrospective. The templates are based on the meeting agendas from Giraf 2016, and may contain information particular to that semester e.g. groups and responsibility areas.

A.1 Weekly Scrum

Agenda for Weekly Scrum dd-mm-yyyy

- Weekly Scrum
  - Group 611 Server + Database
    - What has been completed last week?
    - What will be completed in the next week?
    - Are there any blockades or do you need help?
    - Is your work possibly going to block another group's work?
  - Group 612 Social + Google Analytics
  - Group 613 Scrum Masters
  - Group 614 PO + Security
  - Group 615 Unit and Integration Tests
  - Group 616 Server + Database
  - Group 617 Graphics + Sound
  - Group 618 Documentation + Wiki
  - Group 619 Usability Tests
- Important dates
- Other (People can add items to this part)

Note that the four questions below group 611 are repeated for each group, but have been left out for convenience.

A.2 Sprint Planning

Agenda for Sprint Planning dd-mm-yyyy

- Introduction
  - Sprint Goal (PO)
- Review Sprint user stories
  - Discuss possible solutions
  - Assign user stories to groups

- Assessment of assigned user stories
- Important dates
- Other (People can add items to this part)

A.3 Sprint Review and Retrospective

The Giraf Scrum Masters of 2016 chose to combine the Sprint Review and Sprint Retrospective meetings into one meeting which is why the agenda is also a combination of both meetings here.

Agenda for Sprint Review and Retrospective dd-mm-yyyy

- Sprint Review
  - Showcase of completed work during the Sprint
  - Open tasks for the Sprint
- Break
- Sprint Retrospective
  - Actionable commitments for the Sprint
  - Start, stop, keep
    - Split into small discussion groups
    - Discuss feedback for the project in the groups
    - Collect feedback on the blackboard
    - Short break (Scrum Masters get an overview of the feedback)
    - Review and discussion of feedback
- Important dates
- Other (People can add items to this part)

B. Meeting Checklist

Arranging meetings involve many small tasks. This appendix is a collection of things to remember when arranging meetings. Some items may only be relevant to specific meetings.

Before meeting:

- Book a suitable room for the meeting
- Add meeting to a shared calendar
- Bring post-it notes (Sprint Retrospective)
- Remind people of the meeting. Perhaps using the chosen communication tool (Using Slack and Google Calendar you can have automatic reminders)
- Write an agenda and share it with the team so that they can add items to it
- Write a summary of the meeting (A person from the Scrum Master group can do it)

After meeting:

- Manage feedback. Make sure that you evaluate the feedback people have (Especially important after the Sprint Retrospective, where a lot of feedback is received)
- Create *actionable commitments* from feedback (Sprint Retrospective)

C. Tools

Listed below are some of the tools and technologies which have been used in the development of Giraf (2016):

- Android Studio https://developer.android.com/studio/index.html
- Arcanist https://github.com/phacility/arcanist
- Artifactory https://www.jfrog.com/open-source/ **Giraf:** http://jenkins.giraf.cs.aau.dk:8080/artifactory/
- Doodle http://doodle.com
- Docker http://www.docker.com/
- Git https://git-scm.com/
- Gogs https://gogs.io/ - **Giraf:** http://git.giraf.cs.aau.dk/
- Google Drive http://drive.google.com/
- Gradle http://gradle.org/
- Javadoc http://www.oracle.com/technetwork/java/javase/documentation/index-jsp-135444.html - **Giraf:** http://git.giraf.cs.aau.dk:8080/
- Jenkins https://jenkins.io/ - **Giraf:** http://jenkins.giraf.cs.aau.dk/
- MirrorGo https://www.wondershare.com/android-mirror/
- Phabricator http://phabricator.org/ - **Giraf:** http://web.giraf.cs.aau.dk/
- Planning Poker https://www.mountaingoatsoftware.com/agile/planning-poker
- Slack https://slack.com/ - **Giraf:** http://giraf.slack.com/
- WildFly http://wildfly.org/ - **Giraf:** http://web.giraf.cs.aau.dk:8080/

D. License

THE WORK (AS DEFINED BELOW) IS PROVIDED UNDER THE TERMS OF THIS CREATIVE COMMONS PUBLIC LICENSE ("CCPL" OR "LICENSE"). THE WORK IS PROTECTED BY COPYRIGHT AND/OR OTHER APPLICABLE LAW. ANY USE OF THE WORK OTHER THAN AS AUTHORIZED UNDER THIS LICENSE OR COPYRIGHT LAW IS PROHIBITED.

BY EXERCISING ANY RIGHTS TO THE WORK PROVIDED HERE, YOU ACCEPT AND AGREE TO BE BOUND BY THE TERMS OF THIS LICENSE. TO THE EXTENT THIS LICENSE MAY BE CONSIDERED TO BE A CONTRACT, THE LICENSOR GRANTS YOU THE RIGHTS CONTAINED HERE IN CONSIDERATION OF YOUR ACCEPTANCE OF SUCH TERMS AND CONDITIONS.

1. Definitions

   "Adaptation" means a work based upon the Work, or upon the Work and other pre-existing works, such as a translation, adaptation, derivative work, arrangement of music or other alterations of a literary or artistic work, or phonogram or performance and includes cinematographic adaptations or any other form in which the Work may be recast, transformed, or adapted including in any form recognizably derived from the original, except that a work that constitutes a Collection will not be considered an Adaptation for the purpose of this License. For the avoidance of doubt, where the Work is a musical work, performance or phonogram, the synchronization of the Work in timed-relation with a moving image ("synching") will be considered an Adaptation for the purpose of this License.

   "Collection" means a collection of literary or artistic works, such as encyclopedias and anthologies, or performances, phonograms or broadcasts, or other works or subject matter other than works listed in Section 1(g) below, which, by reason of the selection and arrangement of their contents, constitute intellectual creations, in which the Work is included in its entirety in unmodified form along with one or more other contributions, each constituting separate and independent works in themselves, which together are assembled into a collective whole. A work that constitutes a Collection will not be considered an Adaptation (as defined above) for the purposes of this License.

   "Distribute" means to make available to the public the original and copies of the Work or Adaptation, as appropriate, through sale or other transfer of ownership.

   "License Elements" means the following high-level license attributes as selected by Licensor and indicated in the title of this License: Attribution, Noncommercial, ShareAlike.

   "Licensor" means the individual, individuals, entity or entities that offer(s) the Work under the terms of this License.

   "Original Author" means, in the case of a literary or artistic work, the individual, individuals, entity or entities who created the Work or if no individual or entity can be identified, the publisher; and in addition (i) in the case of a performance the actors, singers, musicians,

dancers, and other persons who act, sing, deliver, declaim, play in, interpret or otherwise perform literary or artistic works or expressions of folklore; (ii) in the case of a phonogram the producer being the person or legal entity who first fixes the sounds of a performance or other sounds; and, (iii) in the case of broadcasts, the organization that transmits the broadcast.
"Work" means the literary and/or artistic work offered under the terms of this License including without limitation any production in the literary, scientific and artistic domain, whatever may be the mode or form of its expression including digital form, such as a book, pamphlet and other writing; a lecture, address, sermon or other work of the same nature; a dramatic or dramatico-musical work; a choreographic work or entertainment in dumb show; a musical composition with or without words; a cinematographic work to which are assimilated works expressed by a process analogous to cinematography; a work of drawing, painting, architecture, sculpture, engraving or lithography; a photographic work to which are assimilated works expressed by a process analogous to photography; a work of applied art; an illustration, map, plan, sketch or three-dimensional work relative to geography, topography, architecture or science; a performance; a broadcast; a phonogram; a compilation of data to the extent it is protected as a copyrightable work; or a work performed by a variety or circus performer to the extent it is not otherwise considered a literary or artistic work.
"You" means an individual or entity exercising rights under this License who has not previously violated the terms of this License with respect to the Work, or who has received express permission from the Licensor to exercise rights under this License despite a previous violation. "Publicly Perform" means to perform public recitations of the Work and to communicate to the public those public recitations, by any means or process, including by wire or wireless means or public digital performances; to make available to the public Works in such a way that members of the public may access these Works from a place and at a place individually chosen by them; to perform the Work to the public by any means or process and the communication to the public of the performances of the Work, including by public digital performance; to broadcast and rebroadcast the Work by any means including signs, sounds or images. "Reproduce" means to make copies of the Work by any means including without limitation by sound or visual recordings and the right of fixation and reproducing fixations of the Work, including storage of a protected performance or phonogram in digital form or other electronic medium.

2. Fair Dealing Rights. Nothing in this License is intended to reduce, limit, or restrict any uses free from copyright or rights arising from limitations or exceptions that are provided for in connection with the copyright protection under copyright law or other applicable laws.

3. License Grant. Subject to the terms and conditions of this License, Licensor hereby grants You a worldwide, royalty-free, non-exclusive, perpetual (for the duration of the applicable copyright) license to exercise the rights in the Work as stated below:

    - to Reproduce the Work, to incorporate the Work into one or more Collections, and to Reproduce the Work as incorporated in the Collections;
    - to create and Reproduce Adaptations provided that any such Adaptation, including any translation in any medium, takes reasonable steps to clearly label, demarcate or otherwise identify that changes were made to the original Work. For example, a translation could be marked "The original work was translated from English to Spanish," or a modification could indicate "The original work has been modified.";
    - to Distribute and Publicly Perform the Work including as incorporated in Collections; and,
    - to Distribute and Publicly Perform Adaptations.

The above rights may be exercised in all media and formats whether now known or hereafter devised. The above rights include the right to make such modifications as are technically necessary to exercise the rights in other media and formats. Subject to Section 8(f), all rights not expressly granted by Licensor are hereby reserved, including but not limited to the rights described in Section 4(e).

4. Restrictions. The license granted in Section 3 above is expressly made subject to and limited by the following restrictions:

    - You may Distribute or Publicly Perform the Work only under the terms of this License. You must include a copy of, or the Uniform Resource Identifier (URI) for, this License with every copy of the Work You Distribute or Publicly Perform. You may not offer or impose any terms on the Work that restrict the terms of this License or the ability of the recipient of the Work to exercise the rights granted to that recipient under the terms of the License. You may not sublicense the Work. You must keep intact all notices that refer to this License and to the disclaimer of warranties with every copy of the Work You Distribute or Publicly Perform. When You Distribute or Publicly Perform the Work, You may not impose any effective technological measures on the Work that restrict the ability of a recipient of the Work from You to exercise the rights granted to that recipient under the terms of the License. This Section 4(a) applies to the Work as incorporated in a Collection, but this does not require the Collection apart from the Work itself to be made subject to the terms of this License. If You create a Collection, upon notice from any Licensor You must, to the extent practicable, remove from the Collection any credit as required by Section 4(d), as requested. If You create an Adaptation, upon notice from any Licensor You must, to the extent practicable, remove from the Adaptation any credit as required by Section 4(d), as requested.
    - You may Distribute or Publicly Perform an Adaptation only under: (i) the terms of this License; (ii) a later version of this License with the same License Elements as this License; (iii) a Creative Commons jurisdiction license (either this or a later license version) that contains the same License Elements as this License (e.g., Attribution-NonCommercial-ShareAlike 3.0 US) ("Applicable License"). You must include a copy of, or the URI, for Applicable License with every copy of each Adaptation You Distribute or Publicly Perform. You may not offer or impose any terms on the Adaptation that restrict the terms of the Applicable License or the ability of the recipient of the Adaptation to exercise the rights granted to that recipient under the terms of the Applicable License. You must keep intact all notices that refer to the Applicable License and to the disclaimer of warranties with every copy of the Work as included in the Adaptation You Distribute or Publicly Perform. When You Distribute or Publicly Perform the Adaptation, You may not impose any effective technological measures on the Adaptation that restrict the ability of a recipient of the Adaptation from You to exercise the rights granted to that recipient under the terms of the Applicable License. This Section 4(b) applies to the Adaptation as incorporated in a Collection, but this does not require the Collection apart from the Adaptation itself to be made subject to the terms of the Applicable License.
    - You may not exercise any of the rights granted to You in Section 3 above in any manner that is primarily intended for or directed toward commercial advantage or private monetary compensation. The exchange of the Work for other copyrighted works by means of digital file-sharing or otherwise shall not be considered to be intended for or directed toward commercial advantage or private monetary compensation, provided there is no payment of any monetary compensation in con-nection with the exchange of copyrighted works.

- If You Distribute, or Publicly Perform the Work or any Adaptations or Collections, You must, unless a request has been made pursuant to Section 4(a), keep intact all copyright notices for the Work and provide, reasonable to the medium or means You are utilizing: (i) the name of the Original Author (or pseudonym, if applicable) if supplied, and/or if the Original Author and/or Licensor designate another party or parties (e.g., a sponsor institute, publishing entity, journal) for attribution ("Attribution Parties") in Licensor's copyright notice, terms of service or by other reasonable means, the name of such party or parties; (ii) the title of the Work if supplied; (iii) to the extent reasonably practicable, the URI, if any, that Licensor specifies to be associated with the Work, unless such URI does not refer to the copyright notice or licensing information for the Work; and, (iv) consistent with Section 3(b), in the case of an Adaptation, a credit identifying the use of the Work in the Adaptation (e.g., "French translation of the Work by Original Author," or "Screenplay based on original Work by Original Author"). The credit required by this Section 4(d) may be implemented in any reasonable manner; provided, however, that in the case of a Adaptation or Collection, at a minimum such credit will appear, if a credit for all contributing authors of the Adaptation or Collection appears, then as part of these credits and in a manner at least as prominent as the credits for the other contributing authors. For the avoidance of doubt, You may only use the credit required by this Section for the purpose of attribution in the manner set out above and, by exercising Your rights under this License, You may not implicitly or explicitly assert or imply any connection with, sponsorship or endorsement by the Original Author, Licensor and/or Attribution Parties, as appropriate, of You or Your use of the Work, without the separate, express prior written permission of the Original Author, Licensor and/or Attribution Parties.
- For the avoidance of doubt:
  - Non-waivable Compulsory License Schemes. In those jurisdictions in which the right to collect royalties through any statutory or compulsory licensing scheme cannot be waived, the Licensor reserves the exclusive right to collect such royalties for any exercise by You of the rights granted under this License;
  - Waivable Compulsory License Schemes. In those jurisdictions in which the right to collect royalties through any statutory or compulsory licensing scheme can be waived, the Licensor reserves the exclusive right to collect such royalties for any exercise by You of the rights granted under this License if Your exercise of such rights is for a purpose or use which is otherwise than noncommercial as permitted under Section 4(c) and otherwise waives the right to collect royalties through any statutory or compulsory licensing scheme; and,
  - Voluntary License Schemes. The Licensor reserves the right to collect royalties, whether individually or, in the event that the Licensor is a member of a collecting society that administers voluntary licensing schemes, via that society, from any exercise by You of the rights granted under this License that is for a purpose or use which is otherwise than noncommercial as permitted under Section 4(c).

Except as otherwise agreed in writing by the Licensor or as may be otherwise permitted by applicable law, if You Reproduce, Distribute or Publicly Perform the Work either by itself or as part of any Adaptations or Collections, You must not distort, mutilate, modify or take other derogatory action in relation to the Work which would be prejudicial to the Original Author's honor or reputation. Licensor agrees that in those jurisdictions (e.g. Japan), in which any exercise of the right granted in Section 3(b) of this License (the right to make Adaptations) would be deemed to be a distortion, mutilation, modification or other derogatory action prejudicial to the Original Author's honor and reputation, the Licensor will waive or not

assert, as appropriate, this Section, to the fullest extent permitted by the applicable national law, to enable You to reasonably exercise Your right under Section 3(b) of this License (right to make Adaptations) but not otherwise.

5. Representations, Warranties and Disclaimer
   UNLESS OTHERWISE MUTUALLY AGREED TO BY THE PARTIES IN WRITING AND TO THE FULLEST EXTENT PERMITTED BY APPLICABLE LAW, LICENSOR OFFERS THE WORK AS-IS AND MAKES NO REPRESENTATIONS OR WARRANTIES OF ANY KIND CONCERNING THE WORK, EXPRESS, IMPLIED, STATUTORY OR OTHERWISE, INCLUDING, WITHOUT LIMITATION, WARRANTIES OF TITLE, MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NONINFRINGEMENT, OR THE ABSENCE OF LATENT OR OTHER DEFECTS, ACCURACY, OR THE PRESENCE OF ABSENCE OF ERRORS, WHETHER OR NOT DISCOVERABLE. SOME JURISDICTIONS DO NOT ALLOW THE EXCLUSION OF IMPLIED WARRANTIES, SO THIS EXCLUSION MAY NOT APPLY TO YOU.
6. Limitation on Liability. EXCEPT TO THE EXTENT REQUIRED BY APPLICABLE LAW, IN NO EVENT WILL LICENSOR BE LIABLE TO YOU ON ANY LEGAL THEORY FOR ANY SPECIAL, INCIDENTAL, CONSEQUENTIAL, PUNITIVE OR EXEMPLARY DAMAGES ARISING OUT OF THIS LICENSE OR THE USE OF THE WORK, EVEN IF LICENSOR HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.
7. Termination
   This License and the rights granted hereunder will terminate automatically upon any breach by You of the terms of this License. Individuals or entities who have received Adaptations or Collections from You under this License, however, will not have their licenses terminated provided such individuals or entities remain in full compliance with those licenses. Sections 1, 2, 5, 6, 7, and 8 will survive any termination of this License.
   Subject to the above terms and conditions, the license granted here is perpetual (for the duration of the applicable copyright in the Work). Notwithstanding the above, Licensor reserves the right to release the Work under different license terms or to stop distributing the Work at any time; provided, however that any such election will not serve to withdraw this License (or any other license that has been, or is required to be, granted under the terms of this License), and this License will continue in full force and effect unless terminated as stated above.
8. Miscellaneous
   Each time You Distribute or Publicly Perform the Work or a Collection, the Licensor offers to the recipient a license to the Work on the same terms and conditions as the license granted to You under this License.
   Each time You Distribute or Publicly Perform an Adaptation, Licensor offers to the recipient a license to the original Work on the same terms and conditions as the license granted to You under this License.
   If any provision of this License is invalid or unenforceable under applicable law, it shall not affect the validity or enforceability of the remainder of the terms of this License, and without further action by the parties to this agreement, such provision shall be reformed to the minimum extent necessary to make such provision valid and enforceable.
   No term or provision of this License shall be deemed waived and no breach consented to unless such waiver or consent shall be in writing and signed by the party to be charged with such waiver or consent.
   This License constitutes the entire agreement between the parties with respect to the Work licensed here. There are no understandings, agreements or representations with respect to the Work not specified here. Licensor shall not be bound by any additional provisions that

may appear in any communication from You. This License may not be modified without the mutual written agreement of the Licensor and You.

The rights granted under, and the subject matter referenced, in this License were drafted utilizing the terminology of the Berne Convention for the Protection of Literary and Artistic Works (as amended on September 28, 1979), the Rome Convention of 1961, the WIPO Copyright Treaty of 1996, the WIPO Performances and Phonograms Treaty of 1996 and the Universal Copyright Convention (as revised on July 24, 1971). These rights and subject matter take effect in the relevant jurisdiction in which the License terms are sought to be enforced according to the corresponding provisions of the implementation of those treaty provisions in the applicable national law. If the standard suite of rights granted under applicable copyright law includes additional rights not granted under this License, such additional rights are deemed to be included in the License; this License is not intended to restrict the license of any rights under applicable law.