Team Organisation

Team R
Members: Ben Turner, Lukas Vascila, Mihail Tachev

As a team of relatively inexperienced software engineers, we felt it would be inappropriate to conduct full scale and total use of the scrum model in our team. Whilst the scrum model features some definite positives, it also has negatives, primarily in the scrum master role. Nobody on our team has the required training for the scrum master and so, as a group, we decided it would be detrimental to have a full time, fully realized scrum master. Instead, we plan to follow the methodology set out by scrum, that is to say, we shall be following the sprint/lull concept as suggested by both the lecture and our timetable and also making use of the regular, fairly fast paced meetings to monitor progress. As we have yet to get started on the sprint section of the course, it's a little too soon to say how we're going to manage it, but the suggestion currently is not to follow the proposed, daily stand-up meetings as it will certainly break the flow of our work. To expand, we aren't required to be in the same building every day, 9 til 5, and so for us to meet up every day would be a significant waste of time (both in travel time and in planning other events around it). However, it is going to be necessary to keep in constant contact during that period, so we intend to make use of the devices and methods as suggested later in this document.

In line with the scrum methodology, we, as a team, aren't really capable of assigning or fulfilling full time roles. To avoid this, we have given one another pseudo-roles. That is to say, a single person shall be in charge of that area, hopefully to have the best understanding of it, so he can lead the others in the work and direct work flow most effectively.

With that in mind, we have decided on several pseudo roles. The chief architect will be leading the design of the code base and will be most involved with decided on implementation design. In this case, that refers predominantly to logical structures and algorithms. Of course, to lighten the load and to reduce risk and overhead, the whole team will be joining in with this section with multiple brain storming sessions being a must.

Another team member will be tasked with overseeing the code as a whole ensuring the correct implementation of the various algorithms and helping the team work towards the goal. In this scenario he will also be tasked with ensuring we're not working with the wrong idea in mind; he, along with others, will be checking to ensure the code base is semantically correct, so as to reduce risk of failure at a later date. Agile development focuses more on iteration of the code base, and as such, this member will have a very important role in keeping everybody focussed.

Many projects also require a graphical user interface (referred to as a GUI from this point onwards), a team member will be tasked with the design of said GUI. To avoid an unnecessary workload, and to ensure the GUI looks and feels the best it can, the whole team will have input towards the final design, as well as possible outside opinion (permission and necessity depending). Most projects require a high degree of readability and interactivity, so it will be paramount that this section of the project is of the highest quality.

The chances of having a project that is fully within the knowledge and understanding of a team is low. As such, we consider it highly important that we understand the project parameters and

how best to implement the solution. With that in mind, a relative expert is a position we consider very important, who will function as the customer for whom we are designing this project. To attempt to implement an effective and high quality solution, we think a member learning about the deepest depths of the subject would be a key part of a successful and high quality delivery. This member will learn as much as he can to provide a full and total understanding of every parameter. Our project cannot succeed without a deep understanding of the game; this role is critical.

The roles mentioned so far have followed the idea laid out above: leading the group through that area of the project but not expected to succeed alone. However, as in many areas of life, there needs to be a man with brown shoes and a clipboard keeping everyone on track. For our project, we have elected to have a manager to keep an eye on everyone's work, to ensure everyone is working at the rate they should be, and to ensure a timely and complete delivery. Whilst not in charge of any one role, it will be the managers job to ensure everything is on time; no missed deadlines and no missed meetings. The member will chase people for work, hound them for progress and hopefully direct the project to the best conclusion it can reach. This role is broader than the other roles, but not as deep, and full of paper work.

Our suggested roles currently are:

Name	**Role**	**Reasoning**
Chief Architect	Radko	Radko doesn't have any particular experience in this area but is eager to learn and also use some of the key concepts, which we learnt in class, in practice.
Graphics Designer	Lukas V.	Lukas has taken up two years of psychology alongside computing science. That combined with experience gained with Java swing make him a good candidate for this role.
Field Expert	Mihail	Keen on board games and is eager to master and go into depth with this ancient game
Lead Programmer	Lukas G.	TODO
Project Manager	Ben	Ben has experience both working in and leading a project of this kind from his previous University. He is an effective communicator and well organised.

To ensure our group works effectively, we have several methods of communication in mind. Our base of communication will be a Facebook group. This will be checked by every member of the team on a regular basis to ensure everybody is up to date with the project. We will use the page for communication that is not time critical. For time critical communication we have shared contact information (phone numbers). We are also planning to make use of Skype for face to face communication during holidays when, due to weather or distance, we are unable to meet in person. These meetings will be arranged in an ad hoc manner. We have also set up a Google Calendar to ensure that members will be reminded of meeting times and of work deadlines. A Google Drive folder has also been set up to facilitate the easy sharing of work and a Github repository will be used to share and monitor code. These are alongside our regular weekly meetings and our weekly meeting with our supervisor. Speaking directly to others is extremely important and will provide valuable insight into working together and being part of a successful team.

A project does not come without risks. As we are all living and working apart from one another, lack of communication could become an issue. We share classes so meeting up with one another should be simple but in the unlikely chance there is an issue, we will first attempt to contact them on Facebook (to attempt to keep it casual), followed by University email and telephone when it becomes more serious. Our worse case scenario is to carry on without that person, noting the absence and refusal to communicate in our final submission. If, for whatever reason, a member

refuses to work, we shall first attempt to resolve the issue internal to the group. If that fails, we shall have no choice but to escalate the issue to a supervisor so it is officially noted and then we will continue our work as best we can without that member. In the unlikely case someone has to leave the group, the rest of the group will be able to step in and fill the role that has been left, as everyone will be kept abreast of the whole project and all aspects of its development.

There is a risk that a member will attempt to delete, or accidentally delete, one of the repositories and our work along with it. To attempt to combat that, regular back ups will be made. In the case of Github, there is an option for recovery available if one contracts a site admin quickly. Github will also provide version control for our code which will help prevent unnecessary damage to the code.

In addition, we have also decided to institute a majority vote system. If there is a major decision that needs to be made and there is disagreement over the outcome, the decision will be put to the vote amongst the team. As a team of five members, this will always guarantee an outcome and the whole team has agreed to honour this arrangement. We are also intending to make use of paired programming when testing to ensure issues are found and corrected quickly and simply.