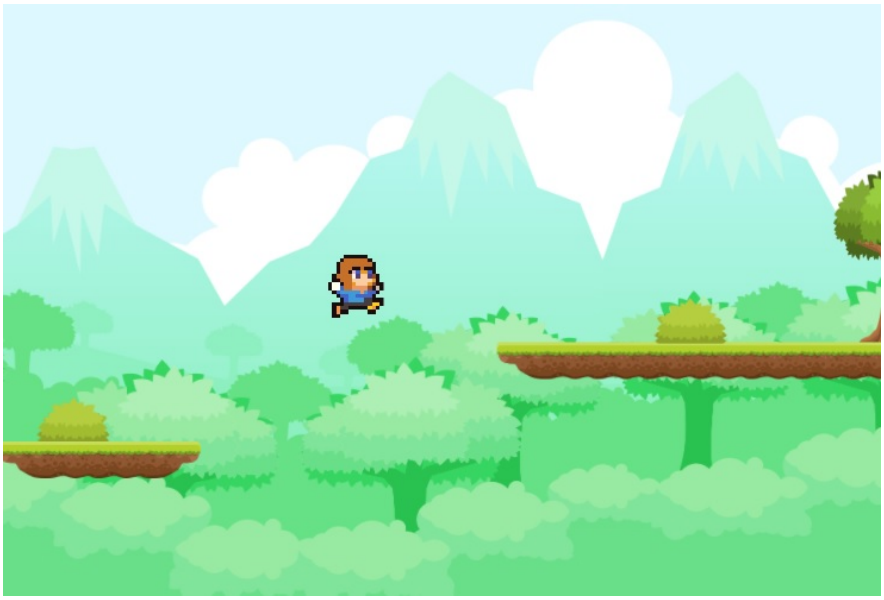
Hoppa Ellie

Post Mortem Report



Software Engineering Project - DAT255
Chalmers University of Technology

Group 257
Hannes Häggander
Dženan Bažd arević
Denise Jing
Dong Yingdong
Max Modig
Su Yu-Hsuan

1. Introduction

1.1 Background

Since all of us who have collaborated on this project live around and are used to going to classes and exercises at Chalmers Campus Johanneberg, we were overwhelmed by a feeling of restlessness when we, in the beginning of the course DAT255, were forced to commute to Campus Lindholmen when summoned there for a workshop. There was simply not enough posts or pictures of substance in our social media feeds to keep us entertained for the entirety of the commute, and we found that the 20 minute bus ride seemed unbelievably long. It was with a great sense of pity for those who are forced to sit, twiddling their thumbs, for a similar amount of time twice every day that we exited the vehicle. There was a collective feeling within the group that something would have to be done to speed up time for the poor souls enduring public transportation every day. That feeling spawned the vision for Hoppa Ellie.

1.2 Vision

We believe that by the end of the project we will have a game, during which the player has a good time, that can fill the void for those who commute and need something to make time go by faster. It is important that the design of the game allows developers to add new levels and objects easily as they seem fit, so that users can get new content. By having this approach we can extend the lifetime of the game for players who have played the game in its previous versions.

We would like to have the game to be desired by people who play in short bursts as well as players who sit down to play for extended periods of time. By focusing on creating a variety of short levels rather than a few longer ones we could answer to both of those criteria.

We aspire to keep everything in the game simple and to the point. There should be no unnecessarily complex menus and we want the player to get right into the game in the blink of an eye and able to quit whenever the he or she feels like like it.

We aim to have the player feel like he or she is progressing through the game by adding features to keep track of how much the player has achieved in the past and reward them with new exciting levels of increasing challenge. Once the player has done the tutorial and the easy levels we need to add complexity so that there is always something new to keep the player interested.

1.3 About Hoppa Ellie

Hoppa Ellie is an Android game application in the style of classic two-dimensional games, such as *Super Mario* and newer games like *Ori and the Blind Forest*. The player assumes the role of *Ellie*, a fictional character whose goal is to advance through levels and avoid obstacles on the way. Every level is completed by reaching the flag at the end which instantly loads a new level, allowing a smooth and uninterrupted experience. The further the player advances, the more difficult the levels get.

Hoppa Ellie was developed by six students at Chalmers University of Technology during the spring semester of 2015. The development of the application took place as a part of a course that introduced the students to working together on a software engineering project. We, the students, were given a lot of freedom with regard to what to develop and how to develop it, and thus the first of many tasks during the project was to come up with and agree on a clear vision for what the finished application would look like. This process and subsequent decisions and processes are given account of in the Method section of this report. Every decision taken during the development of the application, no matter how small it may have seemed at the time, helped shape the application and contributed to the end result of the project, which is presented in the Result section of the report. A more in-depth analysis of the methods, results, things that could have been done better, lessons learnt and more is elaborated on in the Discussion section. Finally, the Conclusions section, which is the last part of the report, briefly summarizes what lessons we bring with us from this project in the future.

2. Method

In order to be able to overcome a task like the development of an Android application in a team of six people with very varying levels of experience in the subject, and from different backgrounds, it is essential to have an idea of how the different aspects and challenges are going to be dealt with. Described below are the most important tools, methods and processes that were applied during the development of Hoppa Ellie.

2.1 Means of Communication

Since most of us had never met before the start of the project, and don't see each other in our day-to-day life, one important thing we had to do before starting the project was to establish a way for us to be able to communicate smoothly in between meetings. Having an easy way to discuss the work was essential to keep track of the process and coordinate the group members' efforts. Since everybody had a Facebook account and most had used it before in similar situations during other courses, setting up a Facebook page with a corresponding group chat was a natural choice in this matter.

2.2 Means of Documentation

For documenting purposes in the project, we decided to use Google Docs, both during and after the actual game development. It has benefits such as allowing multiple people to write simultaneously, automatic synchronization and access from all devices. These features were invaluable to us as a group since many of us had very unsynchronized schedules, and we did not know how often we would be able to sit down and work together on the project.

During the bulk of the project, the Google Docs folder was mainly used to take quick notes and store them somewhere every group member could access them at all times. This was important since not everyone could attend every meeting, and also a good way to keep track of the process for future reference when we predicted would be spending the majority of the time writing documentation. As we approached the end of the project, the most important tool used by the group transitioned from being Unity3D and GitHub to Google Docs. It was ultimately an absolutely vital tool during the writing of this report, when we still had a hard time summoning everyone for a physical meeting.

2.3 Choice of project

Apart from setting up a channel of communication, one of the things that was done at the first meeting after putting together the group was to try to come up with an idea of what to implement. This was a few weeks before the actual start of the project which meant that we had plenty of time to make our minds up about what we wanted to develop once we came back after the Easter holidays. We brainstormed for a few minutes at this first meeting, but it was clear that no final decision was going to be made there and then, and making such a hasty decision seemed unwise anyway, so we made it a task

for every group member to come up with a few ideas for an application during the holidays. These ideas were added continuously over the next few weeks in a document on the group's Facebook page. Every idea had a description, a few user stories and a short assessment of what kind of programming knowledge would be needed to implement it.

When all ideas had been presented, a primary voting process was conducted over Facebook. No final decision was made here, but everybody had a chance to voice their opinion about their preferred idea to implement. The next time we met in real life, unfortunately not with every group member present, a final voting was carried out. The group's wishes were not unisonal, but a majority wanted to develop a simple ski game, so that was finally chosen as our project.

The initial idea of the game was that the player should take the role of an adventurer that travels downhill on a mountain. The adventurer has to avoid traps on the ground by jumping over them. To gather even more ideas on how to develop the ski game we conducted a small benchmarking study in which similar games were tested and briefly evaluated. This resulted in that the ground was chosen to be implemented as a horizontal plane instead of an inclined plane.

2.4 Game Development in Unity3D

One of the group members had previous experience with game development using Unity3D, which is a game engine used by several thousands of game studios and even more amateurs around the world. It is somewhat easy to get familiar with and create a simple game like the one we had in mind for our project, but at the same time experienced developers can use Unity3D to create complex, big budget games. It seemed like a piece of software that could help us greatly with the task that we had before us.

Because of the development experience gap within the group, more experienced members made a base project for the inexperienced to build upon once they had learned the basics. The idea behind the base project was to let everybody get more familiar with the work environment and its settings and the scripting system. We hoped to reduce the learning curve by not diving head-first into the development by starting from scratch, which might have been overwhelming, but instead letting those who had never seen a Unity3D project in the past modify and understand given settings in an already existing project.

After this initial phase of reducing the gap in skill within the group we started implementing more features and levels in the game. This was typically done individually, but also together with other group members in which case two people would be using one computer to try to figure out how to solve a problem related to the project.

A sequential description of the game development process is given in section 2.6 of this report.

2.5 Scrum

Ever since the Lego exercise we tried to follow the scrum methodology but somewhat slower, as we couldn't meet every day for a scrum meeting. Instead, we had one or two meetings every week, where

we talked about what had been done since the previous meeting and what we should do for the next one. It was great for everyone who attended but some did not and we were left short of workforce.

We learned from the Lego exercise that we shouldn't take the more demanding features to start with since it's better to have something done completely rather than just nearly. This fact led us to start off slow and then gradually get a feel for how much we could do in a weeks time and adapt our time depending on our previous progress. If something wasn't quite finished by the end of the sprint we let that team finish their feature before assigning them a new one. This was rarely a problem in our group, but we had some tweaking to do by the end of the project because everyone made their player move at different speeds and got to override others player-objects making their level impossible to complete. We solved this later by having different speeds for every level. It does not differ too much from each other to make the player confused by the change. Often times the change only applied as you move to another set of levels and worked nicely and offered variation when you advanced through the game.

2.6 Sprints

Sprint 1

The first sprint was dedicated to the inexperienced to get up to speed with the base project and the workflow of the game engine. The one who had previous experience started to build a base project for others to start adding to during the start of the second sprint. This would provide a smooth transition from learning to creating.

Sprint 2

The inexperienced members of the group started to understand the development tool, Unity3D, by designing a level. When the levels were roughly finished, we then added features to increase the difficulty. Some of the features were moving platforms and the ability to double jump (i.e. jumping again when already in the air after jumping once).

By adding these features the game got more fun as it made every level different from the others. Previously the player only had to focus on moving towards the end goal, but as complexity of the game was increased, the player had to put more thought into Ellie's moves.

Sprint 3

We started the sprint by having a meeting. During the meeting we planned what to do next, what we had done previously and overall progress. This is recommended when following the scrum methodology. We were to merge all levels and introduce a menu system where the player could choose what level to play. To achieve this we had to merge all of our levels into the master branch and configure the levels so that the player settings worked for all levels. We defined settings for all levels, making all levels completable. Once the merge was completed our main objective was to have all new features to worked as intended and then implement them to all of our levels.

Sprint 4

By the fourth sprint we wanted to finish up all the work we had done previously and polish them to perfection. If there were any bugs left we would iron them out in order to be able to present our game for the fifth sprint. We made a menu that activates when the player presses the buttons on the Android device, accessing buttons to go to the level select screen or the main menu to choose another level.

Sprint 5

The fifth and final sprint was dedicated to finishing up everything and starting to prepare the presentation and the text documents.

3. Result

While it was quite clear at an early stage in the project roughly what tools and practises were going to be used during the development of Hoppa Ellie, we did not know for sure how these would interact. The following segments briefly describe how the combination of them turned out, and what the product of this process was.

3.1 The Development Process

Everyone had their opinion about what features should be in the game. Our solution was to divide the work, meaning that everyone, alone or with another member of the group, creates a level and adds what they want in it. It worked fine at first, but eventually we were faced with some problems by the time we were going to do the integration. The problems were caused by some features impacting other features. That is to say, a new feature might work fine at one level while it produces some bugs in other levels. Moreover, when it comes to the programming, everyone has their own programming style and ideas. A solution to this problem would be to communicate more with each other and decide which parts should be done by whom. By doing this not only we could had reduce the unnecessary and redundant works, but also everyone could get feedback and suggestions on how to improve their feature. Although the progress was very slow for the first weeks we did end up with an unpolished version of the game Hoppa Ellie.

3.2 Beta Version of Hoppa Ellie

We ended up finishing our beta version of Hoppa Ellie which includes basic functionality like jumping, game-over when falling off a platform, a pause menu, a goal for every level and more. However, some features did not turn out as we had planned originally. For example, we were going to build a ski game rather than a jumping game, but it still works well and perfect now. There were some bugs which made the game rough and unfriendly that we finally managed to fix at the end of the project. Some features are implemented but not polished to meet quality standards that we set for ourselves. An example of this is the menu, which is very “snappy” and quick right now. There’s no transition between scenes or when a level loads. We would like to have something that indicates how well you did during a level upon completing it, without interrupting the tempo of the gameplay.

Near the end of the project we introduced a scoring system. This was very good because while it doesn’t affect the actual gameplay much, it makes every level which has the scoring system a lot more complex. That is to say, there is now exactly one optimal path to take through those levels, namely the one during which the player collects the maximum amount of coins and reaches the goal, and a lot of suboptimal paths. In the levels that do not have a scoring system, it does not matter how the player reaches the goal, as long as it is reached, and thus those levels have only one path.

4. Discussion

This chapter of the report is intended to give deeper insight into some of the choices that were made during the development of Hoppa Ellie, and a critical analysis of the main ingredients in the development process. Where problems have arised, reflections on what could have been done differently to achieve a better result are presented.

4.1 Reasoning Behind the Choice to Develop a Game in Unity3D

One of the main underlying advantages of developing a game was that it was thought to be a project that would fit well with the scrum methodology that the course encourages and that we had taken a fancy for during the Lego workshop. The expectation was that the smoothness of dividing the tasks within the group and continuously adding small things just like what we've experienced in the Lego case would translate well into such an incremental project as game development.

The result of a game development process is non-binary in the sense that the game is never really totally complete or incomplete. What we mean by that is that even though the game might be incomplete at a certain point in time in the sense that it does not fulfill all of the criteria that were set for it initially, it might still be *somewhat* complete, playable and maybe even enjoyable thanks to all the small features that have been added up to this point. And by the same philosophy, even though the initial criteria *are* met, and it could be considered finished, it can always be finished *and then some*, by adding additional features or improvements to the existing game. Of course, this does not only apply to game development, but neither does it apply to all possible projects that were being considered.

This suitability for incremental development of a game was thought of as a big positive since we did not know for sure how long time implementing the planned features would take, and we would rather have a simple but yet usable product at the end than taking on a task that eventually turns out to be too extensive to conquer in the given timeframe.

4.2 Evaluation of the Use of Unity3D

The decision to develop a game of the type that we envisioned came with the prerequisite to use a game engine like Unity3D, since the task we had at our hands would be far too extensive to handle otherwise, especially in the given timeframe. One of our group members had used Unity3D before and thought it was a suitable tool for the project. Having someone with experience in use of the software was obviously vital, and provided some sort of feeling of security for the rest of the group.

However, in hindsight, the introduction of a new piece of software of such magnitude as Unity3D in a group of people with very varying levels of computer experience and programming skills could be seen as having been questionable. While the software is easy to use in the sense that it does not take a lot of programming experience to produce interesting results with it, it does have an endless amount of functions, settings, buttons, menus, options and more, which makes for a threshold for achieving

effective and intelligent use. This means that once one knows his or her way around the software, things that seem impossible to do for the less experienced can be done very smoothly, leading to an uneven distribution of responsibilities in such a diverse group of people. This problem was overcome to some degree by letting less experienced people work together on some things, but still remained. While every group member did learn a lot about using Unity3D during the course of the project, choosing a simpler, more familiar development environment would probably have allowed more room for the actual development process, which could have been preferable during such a short project.

4.3 Evaluation of the Communication

We came to the realization some time into the project that Swedish youth use Facebook with a higher frequency than their Chinese and Taiwanese counterparts, which meant that some members of the group had to make a more conscious effort to keep up with the group chat while it was already an integrated part in others everyday life. That small anecdote probably taught us important lessons about the importance of taking the cultural background of every group member into account when working together on a project. It is surely something that we will bring with us into our future more professional undertakings. If we were to do the project again we would have to have a better way of communicating due to some group members never checking facebook in time for meetings or other vital information.

We could also have had set time slots for meetings to discuss progress and what to do next. We had previously said that we would have a meeting every Wednesday before the meeting with our supervisor. This did not work out very well because even though we had a set meeting with the supervisor we were never a full group for some reason. The Facebook page was regularly updated with information from the meetings to keep everybody up-to-date but everyone did not always notice this.

4.4 Evaluation of the Use of GitHub

The fact that we had very different levels of experience with using GitHub, and also different habitual ways of using it, within the group, was another obstacle that we had to overcome during the project. Some members of the group had never heard of GitHub before the course started, while some had higher expertise, but had used it in ways different from one another. This meant that some people had to compromise their regular practices, all the while trying to take into account others' lack of experience, while some had to try to learn on the fly and keep up. Needless to say, this was easier said than done, and something we originally underestimated. At the start of the project we did not address the use of GitHub and the potential issues that could arise with it explicitly, and assumed that it would work out smoothly, which was quite naive in hindsight.

Instead, we started out letting everyone use it as they wished and to the best of their abilities, which turned out quite chaotic after a while. We did not have a structured way of using branches which led to conflicts between files when people unknowingly were working on the same things. Some branches in the repository were left empty, unused or duplicated. The problems were obviously augmented further by the fact that some group members had no previous experience at all with Git. On top of that they were also just starting to learn how to use Unity3D at the same time, which was an overwhelming

combination. One approach that was used to ease the sense of confusion was to simply explore the Unity3D software in itself without committing changes to GitHub, and leaving the organizing of the repository to the group members who knew their way around it better. This obviously led to a slightly skewed division of responsibility, but it was seen as a necessary step on the way to get things in order. Once everyone was somewhat familiar with using Unity3D, we had also figured out more methodical ways to using GitHub.

Even though we eventually worked out our problems with GitHub and everybody learned to handle it smoothly towards the end of the project, the road there was definitely a bumpy one. The correct way to handle the tool was far from intuitive for some people on the team, leading to the repository being a mess or not being used at all for quite some time. The effect of this was mitigated somewhat by working together when we had the chance to, but overall the process definitely left a lot to be desired. The development of Hoppa Ellie would very likely have been a much smoother sail if the way GitHub was going to be used was thoroughly discussed at the start of the project. For example firm guidelines for the purpose of multiple branches, what to put in what branch, when to merge etc. would probably have facilitated the process vastly.

4.5 Evaluation of Teamwork

There were few people who knew each other at the start of the project, partly because we are a international group with members from different parts of the world. Half of our group is from Sweden, two people are from Taiwan, and one is from China. Because of this diversity we had to speak English at all times to understand one another. Although we all think of ourselves as fluent in the English language, we still encountered misunderstandings within our communication, and were somehow bounded when talking another language than your mother tongue, especially when trying to explain the technicalities of producing a product in a workspace where some members have had no previous experience. To ease the problem we gathered and had a group meeting before discussing our project with our supervisor. During the group meetings every member updated each other on their assigned work progress. If any problems, technical or group related, had arised, we would address them and try to solve them as a group, similar to the sprint update meetings described in section 2.5 of this report.

Everyone in the group took the role of a developer to some extent. Some influenced the resulting application more than others but in the end all of us did something that added to the technical aspect of the project. Hannes assumed the role of project leader, planned meetings with the group and kept in touch with the supervisor during development. However, the fact that we almost never had everyone attending our meetings made dividing workload difficult. We had to give assignments to people who were not present and hope that they were fine with doing it. The members who attended the meeting would discuss features we would like to have in the game and then we dealt out the assignments to be done by the next sprint.

5. Conclusion

Even though the game we developed is not all that different from what we originally imagined, the development process has had its share of difficulties. For some of us, it was the first time working with such a diverse group of people. It turns out that working with people that you don't know at all at the start of the project, and having to speak another language than your native one, can be both a demanding and a rewarding experience in very different ways than working on a project with a group of friends from your class.

It was the first time using Unity3D for most of us. Some of us had to familiarise ourselves with GitHub for the first time, too. In hindsight, the introduction of one of those pieces of software would probably have been enough for such a short project. Introducing both at the same time was an overwhelming experience for parts of the group, which led to more time being spent on getting to know the software than the techniques of working together.

The most essential lesson we learnt was probably that of the importance of good communication in a project like this. When you don't know the people you are working with, keeping in touch is not as intuitive, which requires more an effort from everybody. If we were to do a similar project again, we would have to rethink our ways to communicate so that it suits everyone equally well, and have stricter policies with regards to times for meetings and working together.