Requirements

Group 11 - Mandroid Studio

Team members:
Shivam Verma
Martin Halvor Korstveit
Kjetil Vaagen
Dag Erik Gjørvad
Kristian Huse
Thomas Skarshaug

Chosen COTS (Components and technical Constraints):
Android devices
Android Studio
Google Play Games Services

Primary quality attribute - Modifiability
Secondary quality attribute - Usability

Table of content

1 Introduction

1.1 Description of the project

The educational goal of the project is to “Learn to design, evaluate, implement and test a software architecture through game development.”. We are free to choose what kind of game we want to make, but there will be some requirements;

1. It should be a multiplayer game, focusing on the specified quality attributes.
2. The project must utilise architecture and design patterns in the game architecture.

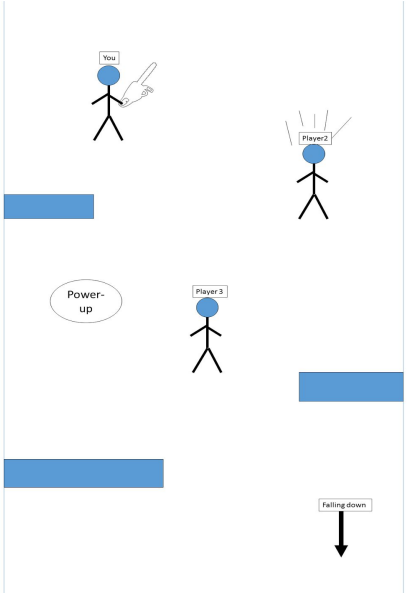
As well as these requirements we have to consider quality attributes about our product. All groups have to focus on modifiability in the design and implementation, but we also have to choose at least one secondary attribute from the following;

- Testability
- Availability
- Usability

In the first phase of the project we will create and document requirements and architecture of the game. This includes agreeing on the project requirements and our concept. All required fields and base layout is given.

1.2 Description of the game concept

We want to make a game based on one or more players moving in free-fall to avoid obstacles in the path with special focus on modifiability and usability. It should be a multiplayer platform, where you can challenge and play with friends. The players should be able to use power-ups and collide with each other to enhance the level of competition among the players. Below is a concept draft of the game.



1.3 Structure of the document

In this document, we will go over our game concept as well as functional requirements, choice of technology and constraints and quality requirements:

- Introduction
- Functional requirements
- Quality requirement
- COTS Components and Technical Constraints
- Issues
- Changes
- References

2 Functional Requirements

Definitions

Functional requirements is supposed to define the core functions of our system that will accomplish the best experience. We have considered components that are crucial for the system, as well as additional attributes that will make the game more enjoyable. This is reflected in our prioritization of the different requirements.

Priorities are listed as:
1 - Must have
2 - Would significantly improve product
3 - A nice addition to the final product
FA: Functional requirements for mobile application

Functional Requirements

ID	Requirement	Comments	Priority
FA1	Network	Must have available network connection	1
FA2	P2P structure	Must be able to connect to Google play game services	1
FA3	Handle touch input	The application must be able to take input from the user through touch control	1
FA3.1	Control player	Through touch input, the system can control a character.	1

FA3.2	Interact with environment	Through touch input it must be possible to interact with the environment.	3
FA4	Create game	It must be possible to create a new game-lobby where game settings can be applied	1
FA4.1	Invite to match	The application should be able to put players into the same game based on player ID	1
FA4.2	Autofill lobby	The system should fill available spots automatically if desired	3
FA4.3	Multiple players	2-4 players should be able to join the same game	2
FA5	Sound	The application should be able to play sound	3
FA6	Settings	The application should offer support for editing certain values connected to sound, graphics and such.	2

Table 1: Functional requirements - Application

3 Quality Requirements

Definitions

Quality requirements are supposed to define the capabilities of the system, specified to a certain scenario. In addition it displays the various parts of the system being affected, how it is affected and how it responds to said scenarios.

- Source of stimulus: The trigger of the scenario.
- Stimulus: What is done to trigger the scenario.
- Artefact: The part of the system being affected.
- Environment: The state of the system.
- Response: How the system reacts to the stimulus.
- Response measure: How the reaction is measured.

3.1 Modifiability

Modifiability to the system defines the ease of adding or extending functionality to the system, performed by developers. Its response measure is generally the time it takes to add said functionality.

M1 Add new levels to the playable game

The development team should be able to extend the set of playable levels
In the game within 30 minutes of cumulative time, for an already developed level.

Source of stimulus	Developer
Stimulus	Add a level to the game
Artefact	The application
Environment	Run-time
Response	Deployed level available for selection in application
Response measure	30 minutes

M2 Add new playable characters

The development team should be able to extend the set of playable characters
In the game within 20 minutes of cumulative time for an already developed character.

Source of stimulus	Developer
Stimulus	Add a character to the game
Artefact	The application
Environment	Run-time
Response	Deployed character available for selection in application
Response measure	20 minutes.

M3 Add new level obstacles

The development team should be able to extend the set of obstacles occurring during the playthrough of a level, for an already developed obstacle.

Source of stimulus	Developer
Stimulus	Add level obstacles to the game
Artefact	The application
Environment	Run-time
Response	Deployed obstacles occurring in the application
Response measure	30 minutes

M4 Add new abilities/gadgets

The development team should be able to extend the set of abilities and gadgets that a character can use in the game, for already developed abilities and gadgets within 20 minutes.

Source of stimulus	Developer
Stimulus	Add abilities and gadgets to the game
Artefact	The application
Environment	Run-time
Response	Deployed abilities and gadgets are available in the application
Response measure	20 minutes

3.2 Usability

Usability delves into the user's interaction with the system; how difficult a task is to perform, and whether or not the wanted result is achieved.
The measured response can be either in time to accomplish a task, or amount of errors occurring during the process.

U1 Start a game with friends

It should be intuitive for the end user to start a game with friends.

Source of stimulus	End user
Stimulus	User starting a game with friends
Artefact	The user interface
Environment	Run-time
Response	The user sets up a game with friends
Response measure	The user fulfills the scenario within 2 minutes.

U2 Configuring the settings options

It should be intuitive for the end user to configure the settings options of the application.

Source of stimulus	End user
Stimulus	User configures the settings options of the application
Artefact	The user interface
Environment	Run-time
Response	The user configures the settings options of the application through known slider functionality.
Response measure	User tests the functionality with no errors.

U3 Understanding the mechanics of the game

It should be intuitive for the user to understand the mechanics of the game

Source of stimulus	End user
Stimulus	User is able to play the game in the way it is intended
Artefact	The user interface
Environment	Run-time
Response	The user plays the game as intended
Response measure	User learns the basic game mechanics within 2 games.

3.3 Performance

Performance to the system describes the requirements of how the system should run. Such as a minimum frame-rate, low load times and equal competitive base.

Performance is not one of our software quality attributes, but we have evaluated these two requirements to be of such a high importance that we wanted to add them to our quality requirements.

P1 Compatibility with older devices

The system should accommodate devices of different system performance, in such a way that it will not impact the user's game experience.

Source of stimulus	System
Stimulus	In-game-process
Artefact	The application
Environment	Run-time
Response	The system work for older devices
Response measure	Game work for android API 15 or newer + between 4'' and 5'' with 1080p

P2 Loading of resources

Whenever the system changes between menus and in-game, the system should remove unneeded resources from memory, and load required resources.

Source of stimulus	System
Stimulus	User navigates to/from menu.
Artefact	The application
Environment	Load-time
Response	The system loads properly without error
Response measure	No errors due to leakage

4 COTS Components and Technical Constraints

For this project our group has chosen to use MVC as the architectural pattern in our application. By using the Android API-15, there are some limitations in regards to the chosen architecture. There can only be one Activity active at any time, with one connected view, requiring the developers to load the various activities one at a time. Changing views (XML), such as navigating through the menus, requires the activity to either inflate a new view, or change Activity all-together. Any shared models during Activity must therefore exist as a globally available Singleton, or stored on the device for it to be reached by other Activities.

In addition, as is common with Android View-classes, is that it is necessary to implement listeners corresponding to a specific view, knowing “where exactly” a user input has been registered. As such it will be necessary to make adjustments to the view-listener-relation to differentiate Controller and View in regards to MVC.

Android device: The application will be made for Android devices, which all offers varying screen resolutions and hardware specifications.

Android API: When creating an Android-application, there are several API-versions available. Each of these offers various functionalities not available in previous versions, and some methods may have become deprecated. In addition not all devices support later versions, and as such it is important that we select a version supported by the majority of devices, being able to reach as many users as possible.

Github: A code-sharing platform for teams, syncing projects via network. Comes integrated in Android Studio. Any used libraries must be added as a dependency, such that all team members will gain the correct version.

Java, *Android Studio*: The app made through Android Studio, Google’s official IDE for android development and Java as its programming language.
Screen resolution: Due to various screen resolutions on different device, we must make drawable resources in different sizes in regards to DPI.

Memory management: Most cellular devices offers a miniscule amount of memory, as such it is important to keep track of when and where resources are loaded, and recycle them accordingly. Android offers automatic loading of resources according to device information, such as HDPI, MDPI etc. These resources however must existing within the folder structure of /resources.

Google Play Games Services: A real-time multiplayer API to connect multiple players into a single game session, and transfer data messages between connected players. It manages network connections to create and maintain a real-time multiplayer room. Also provides a player

selection user interface to invite players to join a room and sends room invitations and updates to players.

5 Issues

We had no issues that is suitable for this section.

6 Changes

Date	Change	Reason for change
16.03.17	Deleted FA2.1	It was a bit unclear if it was a requirement for the system or the user. The content was also covered by FA2
16.03.17	Changed time for response measure of modifiability attributes M1 - M4	Was set to one workday before for development and extending the code. Changed to extending the code for already developed resources and functionality.
16.03.17	Removed U1	Removed U1, as it was not really a requirement of the application.
16.03.17	Added quality requirements for performance.	We mentioned in Architectural description that performance was a concern for the end-user, and as such added requirements for it.
16.03.17	Added intro to all sections	Missing introductions.
24.03.17	P1 and P2 measurement changed	The type of measurement was vague and difficult to test.
24.03.17	P1 changed from “Consistent frame rate” to “Compatibility with older devices”	Gives a more clear overview of the intent of the requirement.
11.04.17	Removed FA5, FA5.1 and FA5.2.	Changed to local matchmaking.
12.04.17	Removed U3 and changed U4 to U3	Changed because we don’t have any friends list since we only have local matchmaking
12.04.17	Changed FA6 to FA5 and FA7 to FA6	Removed FA5, FA5.1 and FA5.2

7 References

- Len Bass, Paul Clements, Rick Kazman, “Software Architecture in Practice – Third edition”, Addison Wesley, September 2012
- eToturials, “Usability Tactics”; web: http://etutorials.org/Programming/Software+architecture+in+practice,+second+edition/Part+Two+Creating+an+Architecture/Chapter+5.+Achieving+Qualities/5.7+Usability+Tactics/ Accessed 26.02.2017
- Android Studio, “Projects overview”; web: https://developer.android.com/studio/projects/index.html/ Accessed 23.02.2017
- Google Play Game Services, “COTS P2P"; web: https://developers.google.com/games/services/common/concepts/realtimeMultiplayer/ Accessed 27.02.2017