**NTNU – Trondheim**
Norwegian University of
Science and Technology

TDT4240 - Software Architecture

# Architectural Description Document

FOODFEUD

*Group A6*
*Android:*

Kjetil Aune
Annie Aasen
Mikal Bjerga
Nikola Radenkovic
Jonathan Brusch Nielsen
Trapnes

Primary focus attribute:
Modifiability

Secondary focus attribute:
Testability

February 26, 2015

# Contents

# 1 Introduction

This document describes the architecture of our "TANK" artillery strategy game for Android developed by Annie Aasen, Mikal Bjerga, Nikola Radenkovic, Jonathan Brusch Nielsen Trapnes and Kjetil Aune. The game takes inspiration from the Worms series. It's a turn-based multiplayer game where the objective is to hit and destroy the enemy tank with different types of ammo. When the round is over, players can visit the store where they can buy new weapons and tank upgrades. We have decided to use different types of food as ammo.

# 2 Architectural drivers

[1]

## 2.1 Functional requirements

- Playable as an offline multiplayer game

## 2.2 Non-functional requirements

- Support rapid design changes

# 3 Stakeholders and Concerns

| Stakeholder | Concerns |
|---|---|
| Users (Anyone running the program) | Is the game easy and intuitive to use?<br>Is the game fun to play? |
| Developers (Group A6) | Will it be possible to develop and test the application in 9 weeks?<br>Is the game easily modifiable?<br>Is our program easy to test? |
| Evaluators (ATAM-groups and course staff) | Does the application meet all the functional requirements?<br>Does the application meet all non-functional requirements?<br>Is the implementation easy to understand (e. g. intuitive variable names and explained through comments) and well-documented?<br>Is the documentation complete and straightforward? |

# 4 Viewpoints

We have chosen to use a logic view, development view and process view, as explained in *The "4+1" View Model of Software Architecture* [2]. We have opted not to use the physical view and scenarios from the same paper, as we have deemed them to be unnecessary. The physical view is not needed as we will only use a single device, running a single process, which does not require any documentation to understand. Using the scenario view was discussed, but abandoned, due to the simplicity of our game. If we choose to modify it into a more complex game, e. g. with multiple game modes or different ways of playing the game (e.g online), a scenario view would have to be added to the documentation.

# 5 Architectural Tactics

This chapter should be a look back at the entire report and summarizing the problem, the solution and the obtained results.

## 5.1 Evaluation of the Assignment

You can include comments about the assignment itself here. While this part is not obligatory and not graded, it is valuable feedback to the course staff that can be used to improve the exercises in the future.

# 6 Architectural and Design Patterns

In this chapter, you should discuss the results you have obtained from your implementation. These can be correctness results, i.e whether the implementation behaved as expected, or numerical results that express runtime or energy measurements.

# 7 View

In this chapter, you should discuss the results you have obtained from your implementation. These can be correctness results, i.e whether the implementation behaved as expected, or numerical results that express runtime or energy measurements.

## 7.1 Logic View

## 7.2 Process View

## 7.3 Developement View

# 8 Consistency Among Views

In this chapter, you should discuss the results you have obtained from your implementation. These can be correctness results, i.e whether the implementation behaved as expected, or numerical results that express runtime or energy measurements.

# 9 Architectural Rationale

In this chapter, you should discuss the results you have obtained from your implementation. These can be correctness results, i.e whether the implementation behaved as expected, or numerical results that express runtime or energy measurements.

# 10 Issues

In this chapter, you should discuss the results you have obtained from your implementation. These can be correctness results, i.e whether the implementation behaved as expected, or numerical results that express runtime or energy measurements.

# 11 Changes

In this chapter, you should discuss the results you have obtained from your implementation. These can be correctness results, i.e whether the implementation behaved as expected, or numerical results that express runtime or energy measurements.

# Bibliography

[1] D. Adams. *The Hitchhiker's Guide to the Galaxy*. San Val, 1995.

[2] Krutchen. *The "4+1" View Model of Software Architecture"*. 1995.