Product Planning

The Cave of Caerbannog

By
Team MIGI2
Taico Aerts, taerts, 4345797
Chiel Bruin, cbruin, 4368436
Bram Crielaard, bcrielaard, 4371755
Wytze Elhorst, welhorst, 4306228
Robin van der Wal, robinvanderwal, 4142918

Context project 2015-2016

FACULTY OF ELECTRICAL ENGINEERING, MATHEMATICS AND COMPUTER SCIENCE.

Instructors:	prof. dr. A. Hanjalic and dr. A. Bacchelli
Context coordinator:	dr. ir. R. Bidarra.
Teaching assistants:	S. van den Oever and J. van Schagen

May 2016

Table of contents

1. Introduction

The purpose of this document is to specify the requirements for our game and to put all these in a schedule. To do this, we will start off in chapter 2 with a high level product backlog, which will list all the requirements of our game grouped by importance as well as a roadmap of when we want to have these requirements implemented.

Chapter 3 is the product backlog which starts off with user stories, describing users in scenarios and what should happen in those scenarios, followed by a release plan which is an overview of the different releases of the game and which features it will have at these releases.

Finally, chapter 4 discusses our definition of done. This part, as its name suggests, describes when we deem something as done. This part is further divided in three parts describing the definition of done for each of these: backlog items, sprints and releases.

2. Product

2.1. High level product backlog

This section gives a high-level overview of all the features we would like our game to have. We have split up this overview in **basic features** and **advanced features**. We regard the basic features as requirements, and the advanced features as “preferable to be included”. When a feature has “a certain amount” or “a specific value”, without specifying the actual value, then this value is still to be decided.

2.1.1. Basic features

- **General**
  - The game is played by two teams, the elves and the dwarves, and one player with an Oculus Rift, the knight of Camelot.
  - When the program is started, it will detect if the Oculus Rift is connected, and show a message to the user if it is not.
  - When the program is started and the Oculus Rift is connected:
    - The knight of Camelot is initially put in a virtual room in which they can select multiple actions, like starting the game, changing settings and starting a tutorial.
    - The beamer will initially display QR codes for the elves and dwarves to join the game on their mobile devices.
  - The game has a time limit. When the time runs out, the dwarves win the game.
- **The virtual environment**
  - The virtual environment consists of multiple rooms.
  - The virtual environment has at least a starting room and a treasure room.
  - Every room in the virtual environment will initially be reachable from the starting room. In other words, all rooms will be connected.
  - The virtual environment contains items and obstacles. These include (at least) bombs, keys, doors, gates and landmines.
  - The virtual environment is presented in different ways to the different players.
    - The knight of Camelot sees the virtual environment in 3D on their Oculus Rift.
    - The elves only see already visited parts of the virtual environment in 2D, on their mobile devices.
    - The dwarves see the entire virtual environment in 2D on their mobile devices.

        - A large 2D overview of the map is projected on the floor with a beamer. This overview only shows already visited parts of the virtual environment. The elves and dwarves can both use this overview.
- **Items**
    - Bombs
        - Bombs can be picked up and can be put down again by the knight of Camelot (in the virtual environment).
        - When a bomb is put down, it can't be picked up any more and it will explode after a certain amount of time.
    - Keys
        - Keys can be picked up by the knight of Camelot (in the virtual environment).
        - Each key has a specific color, and can be used by the knight of Camelot to open doors of corresponding color.
- **Obstacles**
    - Landmines
        - A landmine will explode when the knight of Camelot or enemies get within a specific distance (in the virtual environment).
    - Doors
        - Each door has a specific color.
        - A door can only be opened by the knight of Camelot, if they have a key of the corresponding color.
    - Gates
        - Gates block the knight of Camelot from reaching a specific location.
        - Gates can be opened by the elves.
- **Enemies**
    - Enemies can be spawned by the dwarves
    - Enemies move towards to knight of Camelot and bait, generally prioritizing bait over the knight of Camelot.
    - Enemies die when they are within the explosion range of a bomb or landmine.
- **The knight of Camelot**
    - They see the virtual environment in 3D on their Oculus Rift, and are able to look around in it from a first person view.
    - They are able to move through the virtual environment by using a controller.
    - They are able to pick up items in the virtual environment, when they are standing close to them.
    - They can open doors in the virtual environment, by using keys that have been picked up.
    - They can place a bomb in the virtual environment, when they have picked one up.
    - They win the game when they reach the treasure room in the virtual environment.
    - They have a certain amount of "health", which is decreased by explosions that hit them, by contact with enemies and possibly other things. When the health hits zero, the knight of Camelot dies and the dwarves win.
- **The elves**
    - They can connect their mobile devices to the game by scanning a QR code.
    - They see the virtual environment in 2D, on their mobile devices and on the beamer. They can only see rooms that have already been visited by the knight of Camelot.
    - They are able to see specific items and obstacles in their view of the virtual environment. This list includes at least: keys, bombs, landmines, enemies and gates
    - They can interact with the virtual environment, by specifying actions on their mobile devices. Actions include (at least):

        - Placing/throwing bait in specific places, to distract the enemies and lure them away from the knight of Camelot.
        - Opening gates for the knight of Camelot.
- **The dwarves**
    - The dwarves win the game when either of the following occurs:
        - The knight of Camelot is killed in the virtual environment
        - The time limit runs out
    - They can connect their mobile devices to the game by scanning a QR code.
    - They see the entire virtual environment in 2D, on their mobile devices.
    - They can interact with the virtual environment, by specifying actions on their mobile devices. Actions include (at least):
        - The ability to place landmines
        - The ability to place bombs
        - The ability to spawn enemies
    - They must not be able to place obstructions in such a way that permanently makes it impossible for the knight of Camelot to reach the treasure room.

2.1.2. Advanced features

- The game includes more objects that the elves can interact with. Possible objects are:
    - Moving platforms, which can be moved and/or activated by the elves and sabotaged by the dwarves.
- The game includes more objects that the Oculus player can interact with in the virtual environment. Possible objects are:
    - Light switches which can be activated and deactivated. Light switches provide the elves with more vision over specific aspects of the game.
    - Boxes/rocks which can be moved and climbed to reach higher areas of the environment.
- Keys are color coded. A key of a specific color will only work on doors of that same color.
- The game should have a way to change the difficulty by changing the time limit and the amount of rooms a level has.
- The game has a soundtrack, which will play for the duration of the game.
- The game has ambient sounds.
- The level generation can be seeded, and users should be able to enter a seed before starting the game.
- An elf can only have one gate opened at a time. To open two gates at the same time, both elves need to work together.

2.2. Roadmap

We have divided the sprints over different phases. Below is an overview of what we plan to do in each phase, and an overview of each sprint, the corresponding phase and the important items of that sprint.

[Concept]
We determine our game concept and create the requirements.

[Documentation]
We write documents about the game design.

[Implementation]
We implement the features of our game.

[Bugs and Balancing]
We focus on fixing bugs and increasing the fun factor of our game.

[Finalization and Presentation]

We finalize the game for the final release and create the trailer. No new features are added, but bugs and balancing is still done when necessary.

[Concept] Week 1
- Create five game concepts

[Concept] Sprint 1
- Choose the definite concept, and work out the details and requirements.
- Familiarize with jMonkey.

[Documentation] Sprint 2
- Create Game Design Document, Product Planning and Product Vision
- Implement basic environment in jMonkey

[Implementation] Sprint 3
- Code framework is created
- Player can move in environment, pick up items and interact with objects
- Elves and dwarves can connect to game
- Keys, boms and doors are implemented

[Implementation] Sprint 4
- Elves and dwarves can interact with the maze
- Remaining basic features are implemented
- Playtesting

[Implementation] Sprint 5
- Changes to features based on playtesting
- Advanced features
- Beta is released

[Bugs and Balancing] Sprint 6
- Balance game features
- Release Beta

[Finalization and Presentation] Sprint 7
- Create script for trailer

[Finalization and Presentation] Sprint 8
- Trailer presentation
- Final report draft

[Finalization and Presentation] Sprint 9
- Final release
- Final report
- Presentation

3. Product backlog

3.1. User stories of features

As the knight of Camelot, When the program is started, And the Oculus Rift is connected, Then I see the menu and I can interact with it.	As the knight of Camelot, When the game is started, Then I can see the virtual environment, And I can look and walk around in it.
As the knight of Camelot, When I'm near an item, Then I can pick that item up.	As the knight of Camelot, When I enter the treasure room, Then the elves and I win the game.

As the knight of Camelot, When I have a key of a specific color, And I interact with a door of the same color, Then the door opens.	
As an elf, I can see explored parts of the maze in 2D on my mobile device.	As a dwarf, I can see the entire maze in 2D on my mobile device.
As an elf, I can see obstacles on my map that are invisible to the knight of Camelot.	As a dwarf, I can see obstacles on my map that ar invisible to the knight of Camelot.
As an elf, When I select a gate on my mobile device, Then I can open that gate.	As a dwarf, When I select a tile on my mobile device, Then I can place a bomb or landmine.
As an elf, When I select a tile on my mobile device, Then I can place bait on that tile.	As a dwarf, When I select a tile on my mobile device, Then I can spawn enemies on that tile.
As an elf, When the knight enters the treasure room, Then I win the game	As a dwarf, When the time limit runs out, Then I win the game.
As an enemy, When there is bait on the map, Then I go to the bait.	As a bomb, When I have been placed down, Then I explode after a set amount of time.
As an enemy, When I see the knight, Then I move towards them.	As a landmine, When the knight or an enemy gets in range, Then I explode and the knight or enemy is damaged.

3.2. User stories of defects

At the moment, there are no defects applicable.

3.3. User stories of technical improvements

At the moment, there are no technical improvements applicable.

3.4. User stories of know-how acquisition

jMonkey As a developer, I want to know how jMonkey works, So that I can effectively write code with its framework.	**Playtesting** As a developer, I want to know how players experience our game, So that I can balance and improve the game concepts.

3.5. Initial release plan

The initial release plan is heavily based on the sprints and the roadmap in Chapter 2.2. There is a release every week, which addresses another milestone.

Sprint 0	First game concept ideas are created.
Sprint 1	The game concept is chosen and requirements are created.
Sprint 2	A basic environment in jMonkey is created.
Sprint 3	The knight can move in the maze and interact with objects. Elves and dwarves can connect to the game and see the map.
Sprint 4	The game is now fully playable.
Sprint 5	All basic features are implemented.
Sprint 6	Most bugs are fixed, and the game is balanced to increase enjoyment.
Sprint 7	The final work on the game is done and the trailer of the game will be developed.
Sprint 8	The trailer of the game will be released.
Sprint 9	The game is released and presented.

4. Definition of Done

4.1. Backlog items

A backlog item is done, when the pull request(s) corresponding to the item comply with the following requirements.

- Code has to be tested as thoroughly as possible. Line coverage, as measured by cobertura, has to be at least 80%. If this task is impossible, an explanation must be given as to why this is the case.
- All methods and classes must have a javadoc comment explaining their functionality.
- The code must result in a succeeding build on the continuous integration system, Travis. This implies that all junit tests in the code must pass.
- The code may not contain warnings generated by checkstyle, FindBugs and PMD. If this task is impossible, an explanation must be given as to why this is the case.
- Every pull request must be reviewed by at least two people, excluding the user who created the pull request. The people that review the pull request will leave comments to indicate if the pull request is ready to be accepted, or if there are issues that need to be resolved first.
- The implementation has been checked to fulfill the defined backlog item.

4.2. Sprints

A sprint is done when:

- All user story features from the sprint have been closed. If a user story is not finished an explanation has to be given.
- All the documents on the corresponding sprint backlog have been completed.
- A new release has been created.

4.3. Releases

A release is done when:

- All features that should be present in that release have been tested to be working.
- All relevant documentation has been added to the release.

5. Glossary

Checkstyle
A static analysis tool which determines if the code complies with the style rules agreed upon by a development team.

FindBugs
A static analysis tool which detects possible bugs in java code.

PMD
A static analysis tool which uses a rule-set to determine if the code is erroneous.

Pull request
A request to add or change code and deliverables.

Seed
A number or string to initialize a pseudorandom number generator.