Product Planning

VH-Danshal

Table of Contents

1 Introduction

Artificial Intelligence (AI) is becoming more prevalent in products around the world. One of its uses is to simplify tasks for us, for instance using AI in self-driving cars. However not every project involving AI is about simplifying everyday tasks. An example of that is in the gaming industry, where AI is used to immerse the player in the game world. So why would the gaming industry be so interesting for AI?

This is definitely not the first time this question has been asked. Since in an article in 2015 a level in the game *Super Mario World,* an action platformer game, had been finished by an AI *(Souppouris, 2015)*. It is an amazing accomplishment, but the game is definitely not representative of the real world. There is however a field in gaming that is specialized to help solve real world problems, Serious Gaming. Tygron is a serious game about urban planning and which we have been designated to create an AI for a specified case in the game.

In Tygron people can create their real world cases in which there might be conflict and the stakeholders can try to make compromises with each other to resolve this conflict. A stakeholder is defined as a role in the game that can be defined by the user that has its own goals to accomplish. A conflict is defined as the situation where two different stakeholders have clashing goals.

After the case has been recreated in the game, the different stakeholders can negotiate with each other. Depending on these negotiations a stakeholder can choose an action to perform. The impact of this action is being calculated with the help of the set up indicators to keep track of the goals for each stakeholder. An indicator is defined as how well a stakeholder is performing on a specified goal. If a certain action has been done this can impact other stakeholders and they will be informed about these actions in the game. So ultimately the goal for the stakeholders will be to resolve the conflict between each other, while having visual feedback on what will happen.

For this project the specified case will take place in the TU Delft area and involve the municipality, TU Delft, DUWO (student housing), public housing and business facilities as stakeholders. All these stakeholders have different goals to achieve. Each group working on this project will pick one of these stakeholders and try to build the AI that achieves most or all of its goals, while negotiating with the other stakeholders.

In the remainder of this document the following will be explained: First of all in section 2 the product will be discussed with all the features we will implement for the product as well as a roadmap containing our major release schedule. Secondly in section 3 the features of the product will be discussed in more detail in the form of user stories. Finally in section 4 we will explain our Definition of Done, in other words when we consider a task, sprint or release as finished.

2 Product

The final product that will be delivered is the GOAL agent filling the role of one of the five stakeholder as well as the adapted GOAL to the Tygron connector. This GOAL agent will be able to negotiate with other agents in order to fulfill his own specified indicators. The indicators have been created independently for each of the different stakeholder roles. To be able to achieve this the given connector will need to be extended to accept actions that are not in the environment yet. This connector creates a connection between the GOAL programming language and the Tygron game, which makes it possible for us to run GOAL agents in the game. In this chapter the high-level product backlog will be explained via the MoSCoW model. Also the roadmap is displayed showing the schedule for the rest of the project.

2.1 High-level product backlog

The product backlog is prioritized with the MoSCoW model. This model is a way to categorize the different features. The model differentiates the features in four categories: must haves, should haves, could haves and won't haves. The must haves are the requirements that need to be implemented for the product to work. The should haves are the requirements that the product likes to have, but are not necessary for a working product. The could haves are the requirements that can be added if there is still time left. The won't haves are the requirements that will not be implemented, but would be interesting for a future project. In the following sections we will sum up all our features categorized in these categories.

2.1.1 Must haves

The following features are necessary for the product to function:

- Create the game(case) with the indicators
- Agent performs actions that benefit his indicators
- Agent negotiates with other stakeholders (deals)
- Connector (GOAL can work efficiently through connector with Tygron)

2.1.2 Should haves

The product should have the following features:

- Able to retrieve information about the environment using the Tygron queries.
- Develop different strategies to get high indicators

2.1.3 Could haves

The product could have the following features:

- Agent uses a strategy that helps give all the involved stakeholders good indicators
- An interface for the user to easily add an agent to a session
- Implement the agent so it works in more general cases

2.1.4 Won't haves

The product won't have the following features:

- Add machine learning functionality to the agent
- Be able to use any action in the environment

- Be able to fulfill any indicator
- A configurable high level interface

2.2 Roadmap

The roadmap is planned for quarter four for from april 18th until june 17th. The first week is used as an introduction to the project and will be considered as week 0 in the roadmap. The roadmap is based on SCRUM, where every sprint/week the product is improved with new backlog items.

Week	Tasks
0	Introduction to the product
1	Brainstorm about interesting cases where stakeholders have a lot of conflict with each other Choose of the brainstormed cases a case and a stakeholder the group wants to implement
2	Create the chosen case in the game
3	Agent performs a couple of the already built in actions in the environment
4	Strategy for fulfilling the indicators of the stakeholder that is being implemented
5	Being able to negotiate deals with other agents about the actions they allow each other to do
6	Strategy for fulfilling indicators with other agents by negotiating deals that can be beneficial or disadvantageous
7	Additional features we want to implement
8	Bugfixing

3 Product backlog

The product backlog is the prioritized list of features, containing short descriptions of the functionality we want to implement in the form of user stories. In these user stories a feature is explained as what someone in a specified role would want from the product to do. This gives a general look at what the agent is supposed to do.
In this chapter the different user stories will be explained as well as an initial release plan.

3.1 User stories of features

Must haves

As a GOAL stakeholder agent I want to be able to connect to the Tygron environment so that I can be recognized by the server as the right stakeholder. (Week 3)

As a GOAL stakeholder agent I want to be able to perform actions in the environment so that I am able to accomplish my goals. (Week 3)

As a GOAL stakeholder agent I want to be able to fulfill my goals so that I complete my role in the game. (Week 3)

As a Contextgroup I want to write indicators the GOAL stakeholder agent needs to fulfill so that the agent will do useful actions in order to get a higher score for these indicators.(Week 2)

As a Contextgroup we want to extend functionality of the Tygron connector to fit the needs of our GOAL stakeholder agent so that the agent is able to perform other actions than those already built into the current Tygron connector. (Week 4)

As a GOAL stakeholder agent I want to be able to negotiate a deal with other GOAL stakeholder agents so that me and the other agent is able to get a higher indicator score after performing the accepted actions of each other. (Week 5)

Should haves

As a GOAL stakeholder agent I want to be able to retrieve information about the environment through the Tygron queries so that I am able to consider what the optimal action to be performed is. (Week 4)

As a GOAL stakeholder agent I want to be able to allow other agents only actions that can benefit or won’t be disadvantageous towards me so that I can get high indicators. (Week 6)

Could haves

As a GOAL stakeholder agent when another agent requests information of me, I want to be able to respond so that the other agent knows what my goals are and can decide what to do with that information. (Week 7)

Won’t haves
As a GOAL agent I want to be able to learn from past encounters of other agents with machine learning so that I am able to make better decisions every new encounter with another agent.

3.2 User stories of know-how acquisition

Must haves
As a GOAL programmer I want to understand how to use the new GOAL version so that I am able to program the GOAL agent. (Week 2)

As a Java programmer I want to understand how the Tygron connector works so that I am able to extend the current functionality of the connector. (Week 3)

As a programmer I want to understand how the game Tygron works so that I can create a new conflicting case in the game. (Week 1)

3.3 Initial release plan

The initial release plan is heavily based on the roadmap of section 2.2.

Week	Milestone
1	Conflicting case for the game chosen
2	Create the chosen case in the game
3	Agent performs a couple of the already built in actions in the environment
4	Strategy for fulfilling the indicators of the stakeholder that is being implemented
5	Being able to negotiate deals with other agents about the actions they allow each other to do
6	Strategy for fulfilling indicators with other agents by negotiating deals that can be beneficial or disadvantageous
7	Additional features we want to implement
8	Final release

4 Definition of Done

In this section we will discuss about when we can consider a task as done. This is necessary in order to establish ground rules regarding when someone can do another task when they are done with the one they were assigned to for example. We will now explain when we define a backlog item, sprint and release is done.

A backlog item is considered to be done when it is implemented as code and functional. Also it needs to be tested with a test coverage of at least seventy five percent and pass all these tests.The reason for this percentage of test coverage is the assessment rubrics that define more than seventy five percent to be covered by the tests *(Contextproject TU Delft, 2016)*. Furthermore the code must be commented and styled correctly according to *Checkstyle*. Also the implemented code must not fail the continuous integration build. After it passed all of the above criteria, it also has to be set up as a pull request and be peer reviewed by at least two other people. When it passes all of these requirements, the backlog item is considered to be done.

A sprint is considered to be done when none of the tests fail. Also the continuous integration build should pass. Finally the entire product should be able to function in the form of a demo.

A release is considered to be done when the continuous integration build passes and it passes all the tests. Also all the code is commented and styled correctly. Furthermore it needs to have implemented at least all of the must haves and should haves. The agent needs to work in the Tygron game as the stakeholder it is implemented as. Finally the agent needs to be able to play in the Tygron game with all the other agents and fulfill his goals given the means to achieve his goals.

5 Glossary

Artificial Intelligence	Intelligence exhibited by machine or software
Definition of Done	Definition of when a backlog item, sprint or release is considered to be finished
MoSCoW	Model in which features for the product are categorized into must haves, should haves, could haves and won’t haves
Product backlog	List of features that can be implemented for the product
Roadmap	Plan with the goals to achieve
User stories	Small stories that describe an interaction from a specific role with the product

6 References

1. Contextproject TU Delft. (2016). Rubrics (Code Evolution Quality). Retrieved from https://docs.google.com/spreadsheets/d/1S4MMFTxIK71wYVZB8eMcwko8v3yObnt64jSuw1pbMIc/pubhtml#
2. Souppouris, A. (2015). Artificial Intelligence learns Mario level in just 34 attempts. Retrieved from http://www.engadget.com/2015/06/17/super-mario-world-self-learning-ai/