Product Planning

GROUP: Gamygdala-Integration:
B.L.L. Kreynen
M. Spanoghe
R.A.N. Starre
Yannick Verhoog
Joost Wooning

May 11, 2015

Contents

1 Introduction

For our context project we work in a large group that is subdivided into different smaller group. In total we have 4 groups of 5 students that work together. Together the whole group will make a proof of concept by creating a game in the Tygron engine[4],creating an interface between the Tygron engine and GOAL and creating both a plug-in and an integration of the Gamygdala emotional engine in GOAL.The specific task of our subgroup is to provide the group that will create an agent with an integration of Gamygdala. It is not our task to implement the engine itself, since it is already present, but it is our job to integrate it in GOAL in a way that programmers, like the other groups, can make use of this when creating virtual humans.
First of all this document will describe the product's requirements based on the MoSCoW method. More information on this method is found in the glossary. Secondly, a road map is specified where you have an overview of the sprint weeks and the tasks that will be worked on each week. Furthermore a list of user stories is given. These are numbered so that in the road map and in the release plan it is very clear which tasks and user stories are linked together. Finally there is a section on the Definition of Done. This involves all the specific agreements on when a certain task is completed.

2 Product

In this section you can find the specific tasks our product should fulfill.

2.1 High-level product backlog

By using the MoSCoW method we can subdivide the high level specifications into the following. The subsections decrease in importance or priority. This means that for example the MUST-haves are way more important to implement than the COULD-haves. This method is very handy and more information can be found in the glossary or references.

Must haves

- An agent in GOAL must have an equivalent in Gamygdala.
- The agent's goals in GOAL should also be present in Gamygdala for that agent. This means that if you define a goal for an agent in GOAL that the Gamygdala engine must also know of this. New goals must be communicated to Gamygdala.
- It must be possible to define how good or bad a belief is for certain goals.
- It must be possible to define relations between agents in GOAL.

- It must be possible to have these relations also present in Gamygdala. Again, it must be the case that when a relation is created in GOAL, the Gamygdala engine knows about this relation. Otherwise no emotions between two agents can be calculated.
- It must be possible to retrieve the emotional state of an agent in the same way as his beliefs.
- It must be able to setup an initial emotion of an agent.

Should haves

- It should be possible to set the gain to a specific value.
- It should be possible to set the decay to a specific function.
- The Gamygdala goals and its properties should be possible to change. This also includes the relations regarding beliefs and goals.
- It should be possible to display the emotional states in the Simple IDE[5] provided with GOAL for debugging.

Could haves

- It could be possible to set a custom decay function.
- It could be possible to change relations during runtime.
- It could be possible for an agent to reason about other agents emotional bases.

Will not haves

- Eclipse plug-in

2.2 Road map

In this section we will list for each week what we will be doing. At the end of each week the user stories that will be completed in that week are listed.

Week 1

The first week we mainly have seminars about the project and how things will go. In this week we will setup the group and make sure everybody gets ready to work on the project. Setting up communication between group members for example.

Week 2

During this week we will implement 2 very small and easy games in Javascript. The games will work with Gamygdala. We create this games so that each member can get a better understanding of Gamygdala and how it reasons.

Week 3

This week the focus will be on setting up the repositories. Furthermore there will be a lot of research on where and how to integrate Gamygdala. Also the first steps towards the product will be implemented. The Gamygdala port will be instantiated in GOAL and a parser for the emotional configuration will be completed. This research is handy so that we do not implement wherever we can but really know what to do and where.(1,2)

Week 4

This week the focus will be on continuing the implementation of the instantiation and getting the parsed configuration files working in Gamygdala. Furthermore this week goals in GOAL will be created in Gamygdala as well. All of this needs a lot of testing. We aim for this planning so that we have a working version of the Gamygdala instantiation as soon as possible to that we can test it very thoroughly both this week and the next weeks.(1,2,6)

Week 5

This week the focus will be on updating the Gaygdala database by appraising beliefs of the agents. This means that when agents get percepts these percepts should be checked against the rules in the Gamygdala engine. They could change the emotion of an agent. We will do this before we try to retrieve the emotional database because we think this is more important and this should be tested early. During this week we will also create the emotional database of an agent by getting its information form Gamygdala.(1,2)

Week 6

This week the focus will be on retrieving the emotional state of an agent by getting information out of the emotional database. This needs to be done so that the other group, and other users that might use our product can get the emotional state of their agents. If they can do this, they can thus implement logical rules based on these emotions. Also the setting of gain and decay functions will be finished and tested during this week. We will work on it earlier but by this week it should also be done. This week our first release should be ready to present the other group with a working version of the Gamygdala integration.(4)

Week 7

This week we will mainly focus on implemented an easier way to debug with emotions. This includes creating a debug tool so that programmers can check what is going wrong and what is happening with the emotions at a given moment in time. We will try to extend the Simple IDE [5] so that it includes the emotional database. After we released the first version we aim for the ease of use of the implemented specifications. Therefore we will create the debug tool for the other group.(3)

Week 8

This week we will focus on updating relations on the fly. This can be handy for programmers. Also we have time left to fix tasks that are not finished but are very important. (7)

Week 9

This week we will also fix, test and finish incomplete tasks. Also this week we will finish the extra documentation on how to use the integration. It will be worked on continuously but this week these documents will be completed.(8,9)

Week 10

During this week the presentations will be made and all things related to it will be completed.

3 Product backlog

In this section you can find an overview of the backlog and release plan.

3.1 User

We define a user as a programmer that wants to use GOAL to develop an AI identity. While doing this, he wants to involve emotions and develop his identity in such a way that he can use emotions to trigger actions, but also actions in the environment (via percepts) that change his emotions would be very handy for the user. The user has a good understanding of the programming language GOAL. He has a very small understanding of Gamygdala.

3.2 User stories of features

1

As a programmer I define a relation between a belief and a goal in the emotion configuration. When I run the agent, each cycle the emotions is updated according to the defined relations.

2

As a programmer if I define a relation between to agents in the configuration, when I run this configuration, Gamygdala is aware of this relation and updates the emotion of both agents accordingly.

3

As a programmer when I run an emotional agent with the Simple IDE[5] I can to inspect its emotional database.

4

As a programmer when I create an agent, I can retrieve information from its emotions (emotional database) and use this information to implement logical rules.

5

As a programmer if I launch an agent with the Simple IDE [5] when I press the pause button for debugging, the emotional database is also paused and possible to inspect.

6

As a programmer I want to set a gain and a decay function and when this configuration is run, these should also be set in Gamygdala.

7

As a programmer I want to alter relations after their definitions for both testing purposes and logical rules.

3.3 User stories of know-how acquisition

8

I want to be able to get a better understanding of Gamygdala by reading through the documentation so that more complex things could be implemented.

9

As a programmer I want to be able to get a better understanding of the integration of Gamygdala in GOAL by reading documentations or papers. This so that I can reason about it and create

3.4 Release plan

3.4.1 Milestone 1 - week 6

For week six we need to have a working version of basic integration so that the other group creating the virtual human is able to use this and experiment. If we don't give them any version to work with, it will be too late for them to use the integration of the Gamygdala engine in GOAL. Minimal features are the features we planned until week 6. For this we will implement user stories 1,2,4 and 6

3.4.2 Milestone2 - end

At the end we need to finish our product. The real minimal features are the must haves we have defined earlier. The bare minimum is thus that the features of milestone 1 work perfectly and are tested thoroughly.For this we will implement user stories 3,5,7,8 and 9.

4 Definition of Done

We consider a coding part as done if:

- test coverage of at least 75%
- documentation of new implemented things must be present
- at least two other colleagues agreed on the implementation and also consider it as done.

We consider a sprint done when we delivered a demo of what we have at the end of the sprint week. The sprint reflection needs to be filled in and also the estimation of the efforts has to be present. At the final stage of a sprint and with regard to the sprint reflection, the next sprint plan needs to be filled in. After this is done, the sprint for that week is completed.

We consider a release done when it fulfills all the coding specifications mentioned above and when it successfully agrees with the user stories of that release.

We consider the project as done when both releases are finished and we have at least the MUST-have specifications implemented. Of course for the project the coding specifications mentioned above need to be complete.

References

[1] GOAL programming language http://ii.tudelft.nl/trac/goal

[2] Gamygdala emotion engine http://ii.tudelft.nl/~joostb/gamygdala/index.html

[3] MoScoW method http://en.wikipedia.org/wiki/MoSCoW_method

[4] Tygron engine http://www.tygron.com

[5] Simple IDE https://github.com/goalhub/simpleIDE