Protoss Master Race

TREMOLIERES Lou PHILIPPE Telo LOPEZ Toan

GELAS Enzo BERGER Jonas

March 28, 2025

1 Introduction

1.1 Architecture overview

Our bot is playing as the Protoss race. To determine what each unit should do, we designed a Utility System that create a list of tasks and associate some score to each of them. The program then select the highest scores in order to assign the tasks to the units.
The logic behind both creating the tasks and completing them is handled by Behaviour Trees. It allows the program to reason about the data displayed on the Blackboard and to manage multiple scenarios.

1.2 Different Responsibilities

To be able to scale our program easily, we split it into different parts that handle the different parts of the decision scheme described earlier. As mentioned earlier, we then have a blackboard that allows each parts to centralize their informations.
First, we have Task Emitters that add Tasks to the list (based on Behaviour Trees) when needed. Then an Idle Manager handles all the task that are to be done continuously (such as gathering resources), for which the Task system was a bit too much. Finally, based on the previously computed rewards for each Task, we associate as much Tasks as possible to the units, before running their Behaviour Trees.

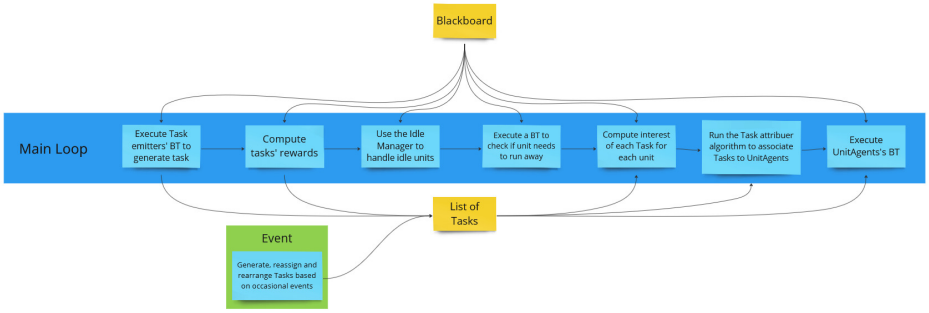

Figure 1: High-level system architecture

2 Behaviour Trees

Here are the schematics of the different Behaviour Trees we used.

2.1 Task Emitters

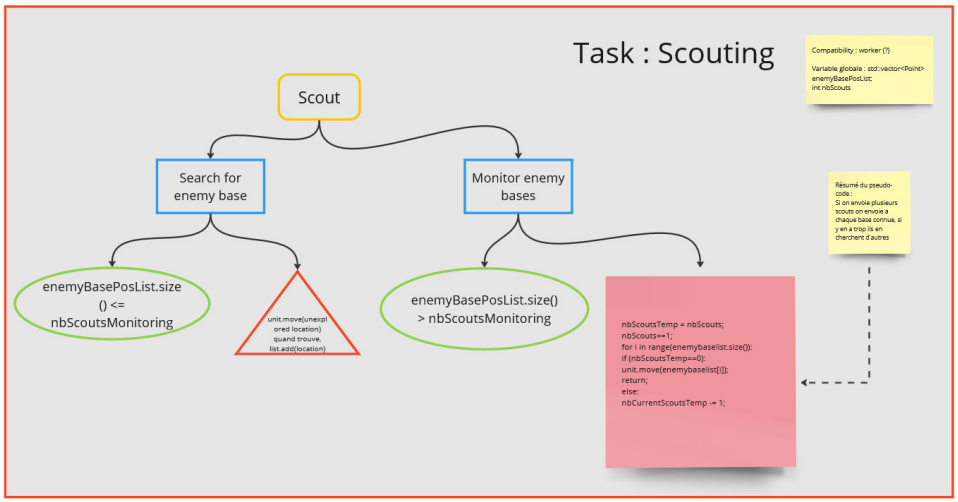

Figure 2: Scouting Tasks Emitter's Behaviour Tree

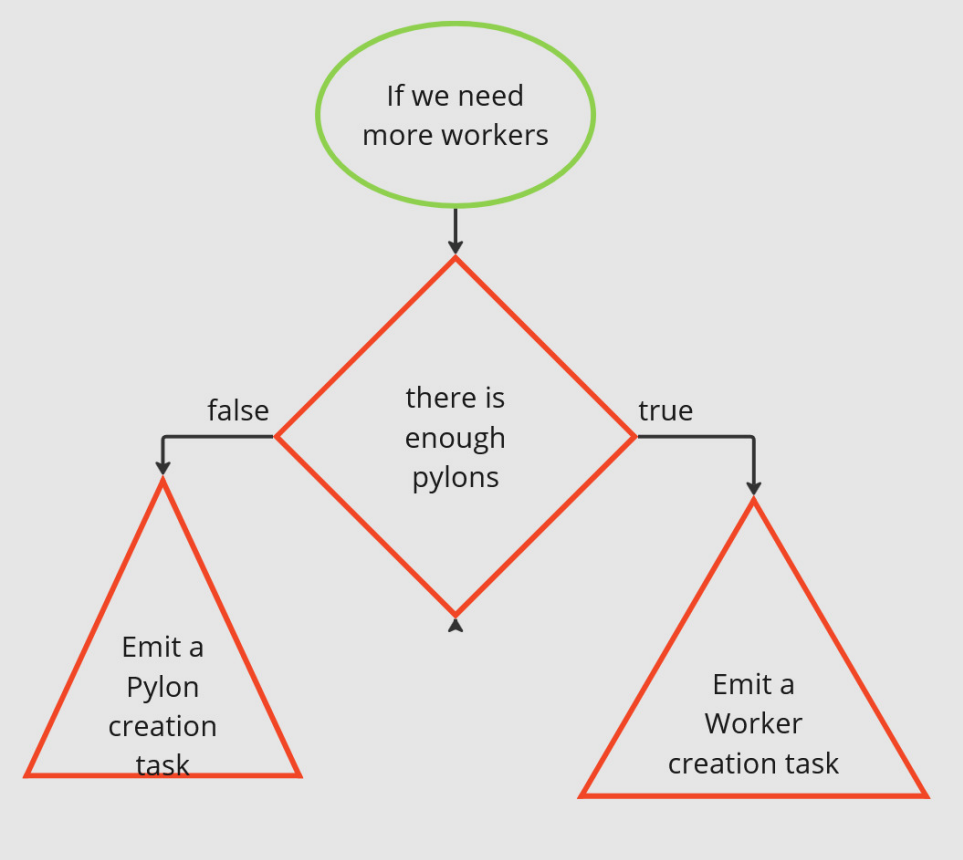

Figure 3: Worker Production Tasks Emitter's Behaviour Tree

2.2 Workers

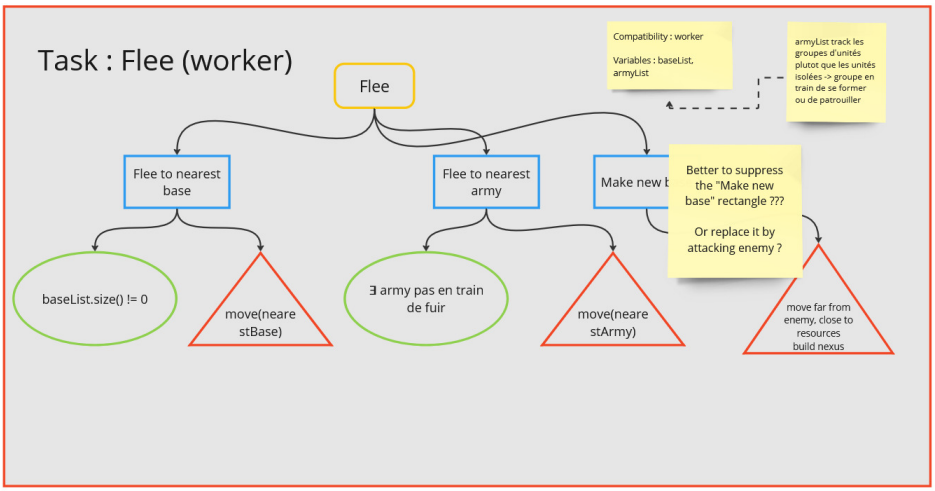

Figure 4: Behaviour tree describing how workers flee from enemies

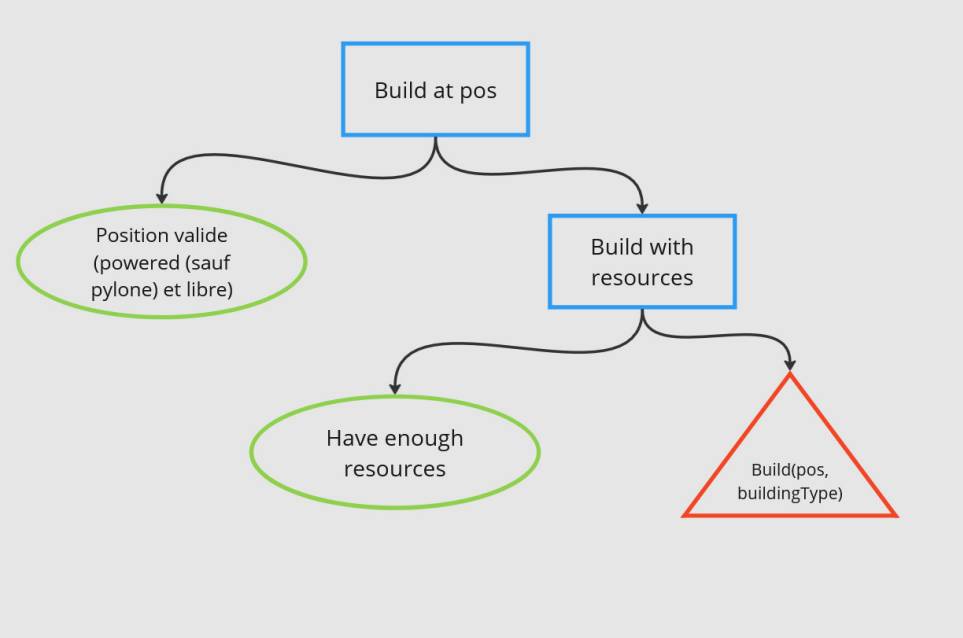

Figure 5: Behaviour tree describing how workers build

2.3 Soldiers

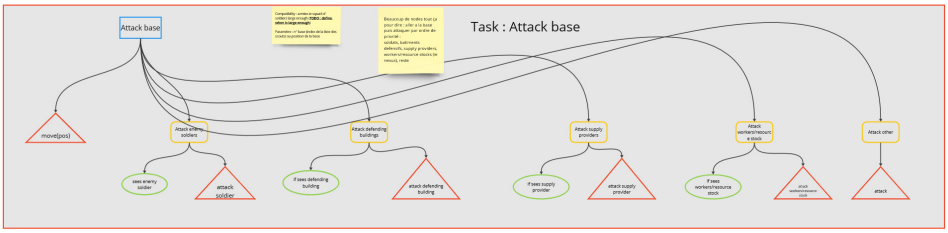

Figure 6: Behaviour tree describing how soldiers attack the enemy's base

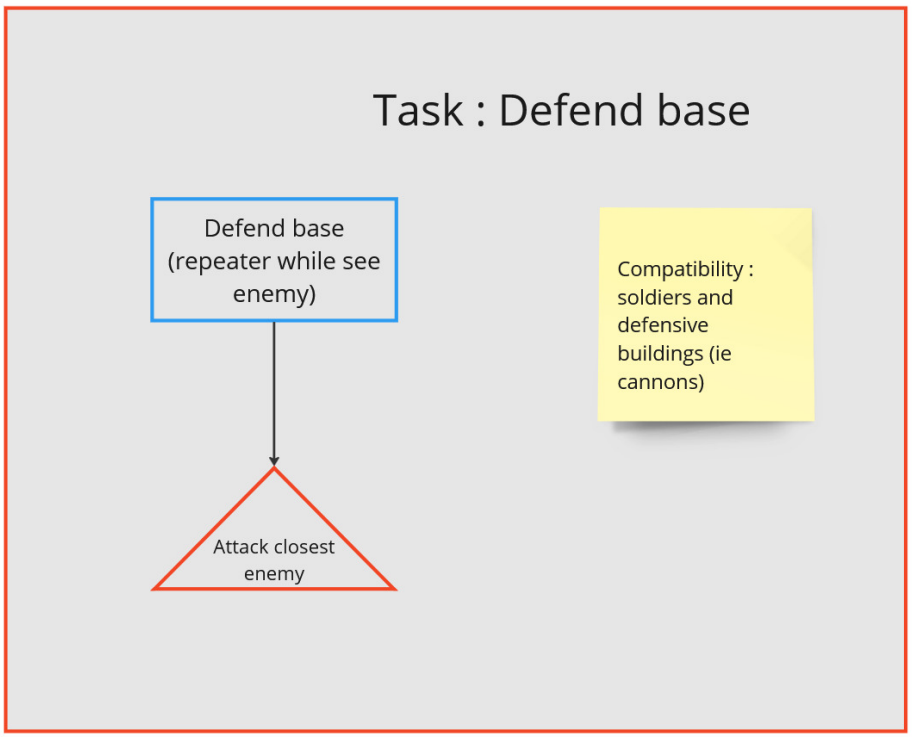

Figure 7: Behaviour tree describing how soldiers defend their base

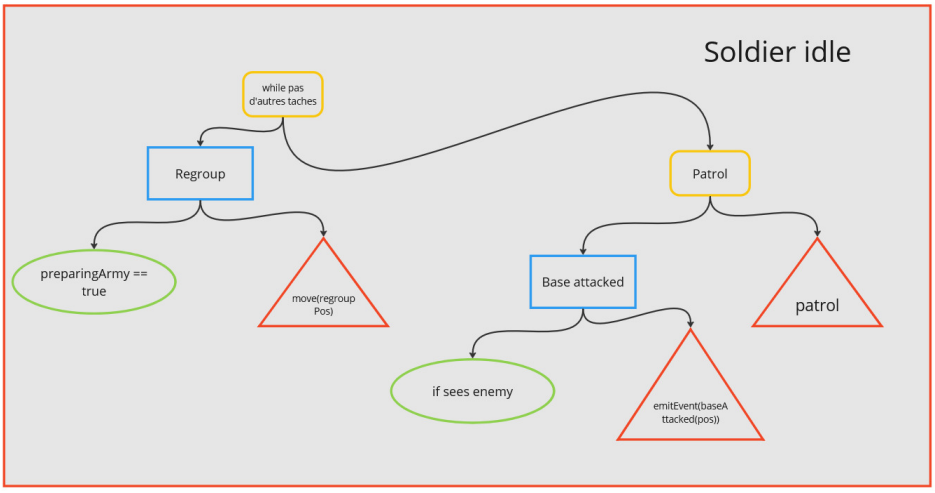

Figure 8: Behaviour tree describing what soldiers when idling

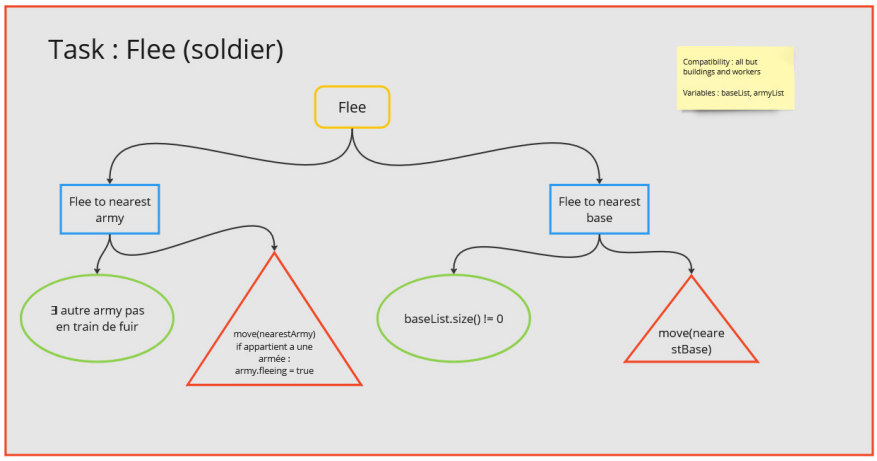

Figure 9: Behaviour tree describing how soldiers flee from enemies

2.4 Build Order

We also made a Task Emitter to emit the tasks related to build order, which is not based on a Behaviour Tree as it is just a sequence of tasks. It also checks for missing elements of the build order in order to rebuild them if needed.