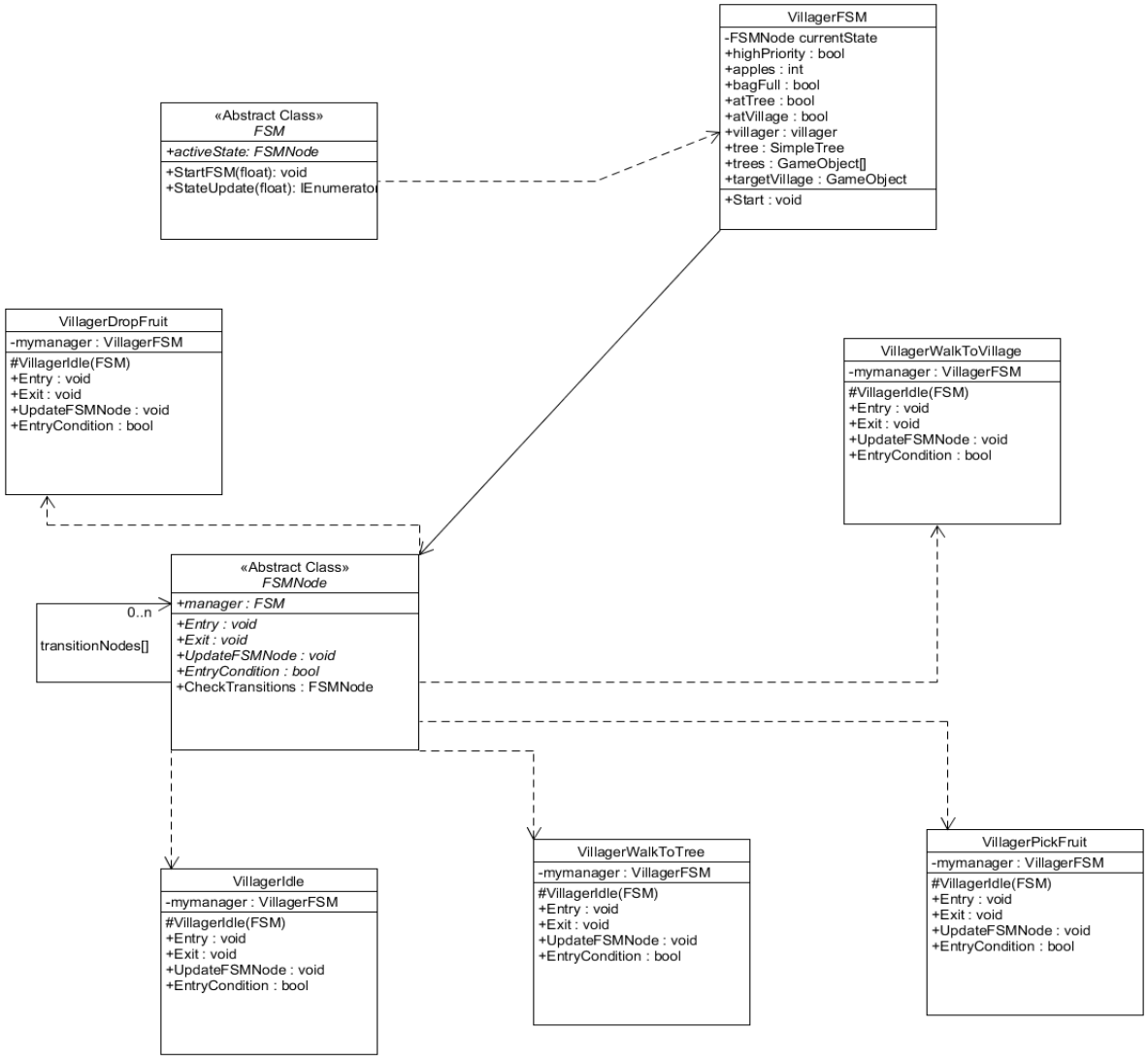
Student Name: Mohammed Hamada Gasmallah Student ID: 10042860

Assignment 2 Part 4

CISC 486

1. **Create a class diagram of your FSM framework. Show the key classes, and any public attributes and operations. I suggest using the UMLet UML diagram editor to create your class diagrams. Your diagram should show just the framework, not the game itself using the framework. Provide brief English-language text to explain the important elements of the class diagram.**



The important parts of this class diagram are the transitionNodes shown as 0…N inside of the abstract FSMNode class. It is important to note that those are only ever instantiated when requested. This way we can avoid infinite recursion.

2. We have arbitrarily assigned priorities to your two NPCs. Describe a more interesting priority assignment algorithm that could be implemented in your level-of-detail scheduler. Your description should clearly explain the criteria used to determine the priority assigned to FSM's. No implementation is required for this question.

A more interesting priority assignment algorithm could be based on both how close the NPC is and how much the player can see the NPC. This would have to be a weighted sort of function. Most likely emphasizing distance the closer a person is to the NPC and emphasizing sight the further away an NPC is. Such a function takes into consideration the senses that affect the immersion. Some probability functions that puts more weight into the appropriate area would make the level of detail more interesting. Above this, there should be level based level of detail (i.e. all creatures in the same level as the player should have a high level of detail). Creatures of different levels or beyond loading screens, or past obstacles should be on a lower level of detail.

3.Based on your experience implementing a finite state machine in this assignment, explain the difficulties inherent in implementing a graphical editor for finite state machines. Specifically consider what information is provided about FSM's in code that would be challenging to specify in a graphical editor.

The difficulties in implementing a graphical editor for finite state machines is particular in that the transitions can become extremely messy and very difficult to debug as well as linking together Entry(), Update() and Exit() can become exceptionally difficult without introducing some form of code. There will need to be some form of code to help flag the transition conditions and to specify easily how to proceed with the state. The graphical editor will simply have to include transitions and perhaps requirements to fill.