System features - Agent Controller

# Introduction

Being able to control any of the agents inside the engine, is what the purpose of this engine is. This Section will cover how we thought the design of an agent controller should look like. To get an overview of the classes used for the ##AgentController##, look at fig. [DOMAINModelAgentController]



[Img note: This sequence diagram shows the process of an AP taking control of agent through the ##AgentManager##, and commanding it through the ##AgentController##]

# Concept

The engine is designed to support the ability to be adapted for all APL**[FOOTNOTE: Agent Programming Language]** types, this means that the engine itself does not support all APL but instead provides a framework for quickly design of an interface between the engine and the APL.

There are two classes that one must use in order to properly design the interface:

**The ##AgentManager##** has the duty of speaking directly with the agent language it attempts to interface with, its job is to arrange it so that all agents the APL wishes to take control of is done so through it, it should do so by spawning ##AgentControllers## to all newly controlled agents. The ##AgentManager## is in that sense much akin to an Abstract Factory which according to the design pattern requires that an abstract class has a method generates a certain type of what object but not exactly which object. The idea is of course that if you have an ##GoalAgentManager## then the controller it constructs would be ##GoalAgentController##. By making it an abstract method we ensure at compile time that the engine framework is properly used which is very good for the user.

**The ##AgentController##** is link between a single agent and the AP**[FOOTNOTE: Agent Program]**, this means its job is to take all commands directed to it and transform them into actions understood by the engine, and apply them to the agent that it controls.

To simplify the ##AgentController## design we provide a method on it called ##PerformAction##, this method purpose is to make it very easy for executing actions on the agent it controls. Basically when the ##PerformAction## is called the ##AgentController## queues the action given through the method. Then puts the ##AgentController’s## thread to sleep, once the action has been executed by the engine. The ##AgentController## is woken up and returns from the ##performAction## method. All percepts received by the ##AgentController## during this time is stored on the ##AgentController## and can be easily accessed by the actual AgentController.

The process of an AP taking control of an agent can be seen on fig. **APConnectingToAndControllingAC,** basically the AP calls the ##AgentManager## to locate the agent it wishes to assume control of. The agent is located through a string which is unique to it and ensures only one agent is taken. When the ##AgentManager## finds the given agent it will immediately generate a new ##AgentController##, the AP will not gain access to the agent but instead it will gain access to the ##AgentController##. Now that the AP possesses the #AgentController## it will have the ability to send the ##AgentController## commands, these commands might not be understood by the engine if the APL is foreign enough to the engine’s own language and as such it is the duty of the AgentController to convert these commands into actual actions which the engine can understand.

## Example

As this project is about working with goal in particular we have created an extension for the AgentController designed specially to work with goal, therefor any example shown here would be incomplete compared to the EIS/Goal implementation we have made. As such go to Section: GOAL/EIS Extension for a proper example of how to use the AgentController.

# Summary

The agent controller is designed very lightweight the reason for this is we do not wish to impose any restrictions that might limit an APL which we know nothing about, as such the ##AgentController## is more akin to a convention or a design pattern for how interfacing with agents should occur. It provides the skeleton of how a link might be designed but does not impose any restriction of how should link should be setup.