CS7056 – Lab 1 – Finite State Machines

Task 1: Basic FSM in C#

See Buckland, pp.48-50.

Build a basic FSM implementation in C# where State and Agent are both implemented as abstract base classes in C#.

The State class should include a method Execute() that takes an Agent object as a parameter and switches state for that agent.

The Agent class should include a method Update() that invokes Execute() for the agent's current state and a ChangeState() method that allows that state object to change the agent's state.

You will want to implement a couple of sample agents too (Bob and Elsa) but you can do that at any point during the lab that you feel is suitable.

Task 2: Generics

See Buckland, p.62. Turn your FSM implementation into a C# generic to allow better reusability.

Unity tutorial: http://unity3d.com/learn/tutorials/modules/intermediate/scripting/generics

Task 3: The StateMachine Class

See Buckland, pp.64-65.

Fix up your design such that most of the code from the Agent class gets moved to a new class StateMachine that encapsulates all the code that has to do with state machines. The idea is that each agent object will own its own StateMachine object.

The only method you should retain in Agent should be one method Update() that invokes the Update() method for that agent's StateMachine object.

Task 4: State Blips

See Buckland, p.63. Add code to your StateMachine class to support global state and state blips.

Task 5: Messaging

See Buckland, pp.67-82. Add messaging capabilities to your FSM implementation.