S1: Design of a new Agent: Wizard Doctor

**Design Description:**

The Witch\_doctor class shares considerable similarity with Warrior class in such it has the ability to counter-attack once being attacked. Therefore, the Witch\_doctor class inherits publicly from Warrior base class and calls Warrior::update and Warrior::describe in support of his “**Attacking**” behaviors. In addition, one can treat Witch\_doctor as a “**Healing**” Warrior, and such feature is easily supported by adding a Boolean variable “**bool healing**” to support the addition of such new state. Our Warrior class is fitted with a “**print\_attack\_word()**” function that is overridden by Witch\_doctor class for outputting his proper message while under attack without the need to rewrite Warrior::update function for such purpose.

Some new functionality has been added to the Agent base class. The addition of **Agent::gain\_health(int curing\_strength)** provides the ability for all agents to be able to be cured. Moreover, in order to determine if the Agent is in “**Attacking**” state, **bool is\_attacking()** function has been added as a “Fat Interface” approach whose default return value is false, and has been overridden in Warrior hierarchy to return the proper value. Such approach is adopted to avoid the need of exposing Warrior’s internal implementation. Finally, in order for Witch\_doctor to know the current health of the Agents, the function “**int get\_health()**” has been added in Agent base class. In Agent\_factory class, the changes are minimal to support the addition of the new Agent.

Few modifications has been done in Model and Controller. First, in order to get the Agent pointer who is within the Witch\_doctor’s range, whose state is “**Not Attacking**”, and whose health is weakest, function “**get\_weakest\_agent\_in\_range**” has been added for such purpose. In Controller class, the addition of a new command “**heal**” is done with little effort for giving user command over the new Witch\_doctor Agent type.