**­­Team Vis Architecture**

**Client GUI *implements* ActionListener**

**Timer gameTime();**

**New Visualizer(gameTime);**

**actionPerformed {**

**if(**beginning of turn OR user toggles view)

*either* **UpdateFull()**

*or* **UpdateMini()** //*depending on full screen/small*

**GlobalCoordinatePlane**

// coordinate system to be

//utilized throughout

final double X\_MIN, Y\_MIN, X\_MAX, Y\_MAX

ArrayList<Event> currentEvents;

void addEvent(Event e)

{

//will add an event to the //globe (in other words, add //to currentEvents)

}

**Visualizer implements ActionListener**

//*should be the ONLY class client*

*//ever has to instantiate*

*//will handle data distribution to //parser, Model, and Visualize*

**Queue events;**

**boolean fullScreenMode**

**Timer gameTime**

**void updateFull()**

*{ //parse simData*

*//fullScreenMode = true;*

*// events = EventQueue.buildQueue (parsed data)*

*// updateModel();*

}

**void updateMini()**

{

*//parse*

*// fullScreenMode = false*

*// events = EventQueue.buildQueue (parsed data)*

*// updateModel();*

}

**void actionPerformed()**

**{** //repaint();

**}**

STARVATIONEVASION.SIMVIS**.CONTROLLER**

**Event Interface**

**Enum Mode //** *MINI or FULL*

//relevant data to event

**EventQueue**

**Queue events;**

getEvents(parsedData);

buildQueue(parsedData);

//build a queue of Events //for Model to process and //Visual to render

**SimParser**

*Data type to store simulation information , not sure what that will look like yet*

**infoDataType parse()**

{

//get data from Client

//return parsed data

}