The classes will be separated into three modules representing their layer in our architecture (3 layer architecture).

-we will not bother with getters and setters, we assume that they will be defined in our classes

-only talk about the key modules

|  |
| --- |
| **LocalNode <<Interface>>** |
| **Input:** None  **Output:** None  **Layer:** Data |
| **Description:**  As an interface, this module will be an abstract class that gives explanations properties of many objects that will interface our system. It gives them the backbone of many classes within our system, however does not tell exactly how to implement the methods/attributes as this is an interface class. The objects implementing this interface are typically each one that is not a link (which is simply a connection between two objects that have implemented this interface) such as an image class, or a variable class. |
| **Attributes:**  xPos::int - Keeps track of the node object’s x position on the coordinate map  yPos::int - Keeps track of the node object’s y position on the coordinate map  xSize::int - The size of the object in terms of its width x  ySize::int - The size of the object in terms of its height y  zOrder::int - The order of the object within the framework. The z axis, if you will, making the  object able to go on top of one another based on its zOrder.  label:String - The static label displayed beneath whichever object this is |
| **Methods:**  pre: n is not null  post: none  error: to be determined  IsValidParent(LocalNode n)::Bool - Checks if the node is a valid parent of node n (ie. an agent is a valid parent for images). True if it is a valid parent, false otherwise. |

|  |
| --- |
| **LocalLink <<Interface>>** |
| **Input:** None  **Output:** None  **Layer:** Data |
| **Description:**  As an interface, this module will be an abstract class that gives explanations properties of many objects that will interface our system. This module is responsible for linking two objects that have LocalNode properties. Henceforth, the classes that will implement this interface are ones that draw connections between two node objects, such as a line or a flow. |
| **Attributes:**  startNode::LocalNode - The node object that initiates the connection (link)  endNode::LocalNode - The node object that receives the link  label:String - The static label which is visually displayed beneath the link  color:: Color - The color of the link/object |
| **Methods:**  pre: none  post: none  error: to be determined  IsValidLink()::bool - Checks if a link between the startNode and endNode is valid to be created |

|  |
| --- |
| **Direction** |
| **Input:** None  **Output:** None  **Layer:** Data |
| **Description:**  A class that implements the LocalNode interface. It is a property that can be used to determine the direction of an object, and thus allowing rotations of the object, back and forth. |
| **Attributes:**  isClockwise::bool - true if the direction is clockwise, false if counter clockwise  isPositive::bool - true if the polarity is positive, false if negative  chargeVisible::bool - true if the polarity is to be displayed, false if not |
| **Methods:**  The methods of this function are precisely getters and setters. The only difference is that the getters check if the state of the object is in the correct form before returning with a bool. |

|  |
| --- |
| **Agent** |
| **Input:** Takes various LocalNodes and holds them within itself (visually, as well)  **Output:** None  **Layer:** Data |
| **Description:**  A class that implements the LocalNode interface. The given class is able to create an “agent” object which can hold other objects relationally (who have implemented LocalNode). |
| **Attributes:**  nodesWithin::List<LocalNode> - a list of LocalNode’s that are currently within the agent object |
| **Methods:**  pre: n is not null  post: a new LocalNode added inside the agent  error: to be determined  AddNode(LocalNode n)::void - when an object is inserted into the agent, it must be checked if implements LocalNode and then added to the nodesWithin list.  pre: n is not null, n exists in the agent  post: the node n is removed from the agent  error: to be determined  RemoveNode(LocalNode n)::void - when an LocalNode is removed from the agent, remove from the nodesWithin list. |

|  |
| --- |
| **LocalCell <<Interface>>** |
| **Input:** None  **Output:** None  **Layer:** Data |
| **Description:** This is the local copy of a shareable object. When it is updated, it updates the corresponding shareable object. |
| **Attributes:** None |
| **Methods:**  pre: none  post: the shareable object is updated to match the local object  error: to be determined  onUpdate()::void - updates the shareable object to match the local object |

|  |
| --- |
| **CollabCell** |
| **Input:** Receives update transfer from a object with LocalCell properties  **Output:** Changes the JSONData of a property  **Layer:** Data |
| **Description:**  A class that is updated by objects with LocalCell properties. |
| **Attributes:**  JSONData::String - the JSON... |
| **Methods:** None |

-----------------------------------------------------

**Software Tools**

**Joint.js**

A Javascript library which specializes in focusing on the interactions and visualizations of graphs and diagrams. Chosen due to its simplicity and ease of use in creating the visual entities as well as their connecting components in our project, thus, speeding up development.

**Google Real Time API**

Google Realtime API allows collaboration as a service for files in Google Drive via the use of operational transforms. The API is a JavaScript library hosted by Google that provides collaborative objects, events, and methods for creating collaborative applications. For example, if one wrote code to manipulate maps, lists, and your own custom JavaScript data model objects. When your code makes a change to the data model on behalf of some user, the data model automatically changes for all the users on the document, hence realtime.

**Selenium**

Chosen as the main testing program of our project. Selenium itself is a web application testing framework that allows you to write tests in many programming languages like Java, C#, Groovy, Perl, PHP, Python and Ruby. The advantages of using Selenium include that members in our group have experience with it. The current consensus is that it will be used for all tests.

**jQuery, Backbone, and Lodash**

Required dependencies of Joint.js. Will help considerably with the interface and framework.