QM Lab Course Project

Requirements Document

Group 3

Angela Stankowski (ans723), Bengin Lee (bel529), Connor Nettleton-Gooding (cwn973), Corey Hickson (crh208), Darvin Zhang (ddz369), James McKay (jlm012), Jordan Wong (jtw289), Mack Mahmoud (mtm464), Matt Hamilton (mch986), Michael Kelly (mlk121), Michael Ruffell (mar492), Mitchell Lau (chl929), Royce Meyer (brm979), Shane Williamsom (saw056)

February 7th, 2016

I. Introduction

II. Background

In January of 2016, our group was commissioned by Nathaniel Osgood and Geoff MacDonnell to design an application, termed by our group as QM Lab, which would serve as a qualitative and collaborative modelling platform for users to create diagrams of various systems.

The need for this project does not come from the lack of modelling tools, but the lack of modelling tools which allow real time collaboration a la Google Docs. Therefore, a major portion of this project will be combining existing technologies and architectures to meet this need.

The schedule for this project is **five** incremental deliverables dated for February 7th, February 24th, March 8th, March 4th, and April 7th of 2016, respectively. We will complete this project with a group of 14 members broken up into a design team, implementation team, testing team, triage team, as well as a build project lead, integration officer, and risk officer.

III. Scope

A. The software must:

- have the ability to support multiple users accessing the system through web browsers and interactively/simultaneously editing/adding components to diagrams,
- To create, save, and load projects
- Tools/elements needed to support creation of UML, flow and stock diagrams, as well as support text
- Adding in one of a small number of shapes and images
- Create variables and links between components
- Basic ability for anyone to create a project and generate an editable/viewable link

B. The software should:

- A login system - with user control administration (such as the creator of a project would have the rights to add and kick certain members in/out of a project - or a set of project (creation of rooms))
- A log of who edited what at certain points - potentially save state/backups
- Power to select a set of the diagram, not only the individual parts (component and the links attached to it)
- Components that contain other components

- Ability to resize and color the elements for a diagram (and the diagram itself)
- Change the font/style of text within components

C. It would be nice if the software could:

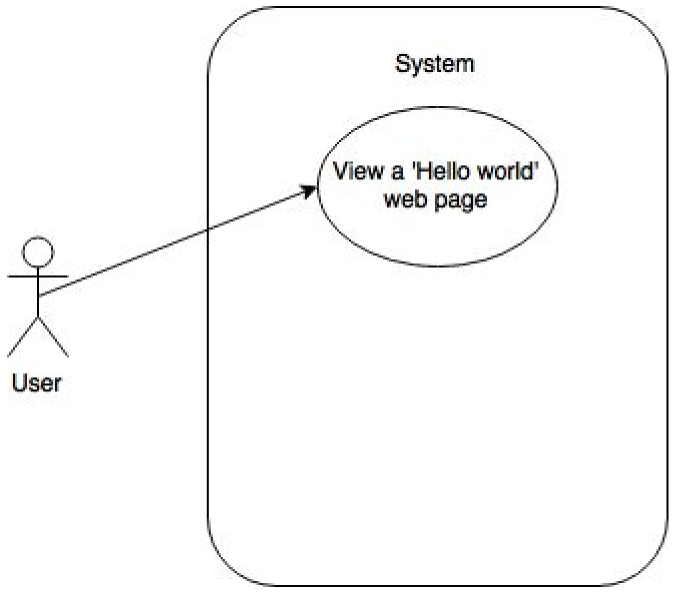
- Customized components such as a user specified picture is added and named to the elements available for a certain project/team
- Customizable overlay for the software
- A chat system - private chat
- General templates to guide users
- Differents styles of components (different looks available for say a UML class diagram)
- Undo function possibly based on history
- A tutorial?
- Support of video maybe a slideshow/add pages to a project?

IV. Actors

User:

V. Use Case Diagrams

For the first incremental deliverable, the use case is rather simple, as diagramed below. The only real requirement for an actor currently is to view a web page. This provides proof of concept that our technologies can connect. The remaining proof of concept come in the form of spike prototypes which will be thrown out later.

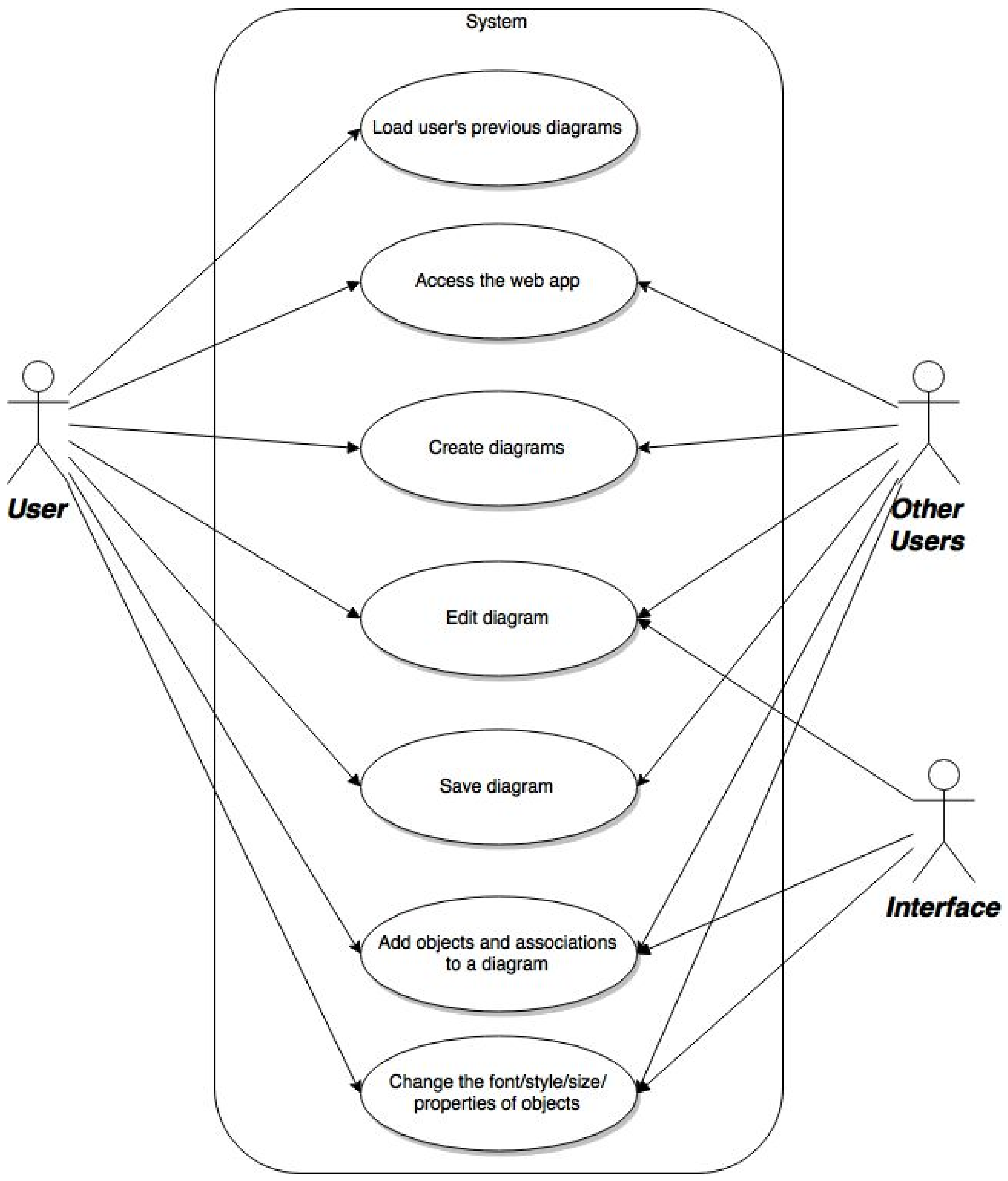


Use case diagram for the first incremental deliverable

For our the future, we have also created a potential use case diagram, below, of what we would like the users to ultimately be able to do.

One of the important distinctions on this diagram is the nuance between a user and other users. Other users should not be able to access and load diagrams of the original user. That is, a user can only access their own diagrams, and not access diagrams for which they do not own.

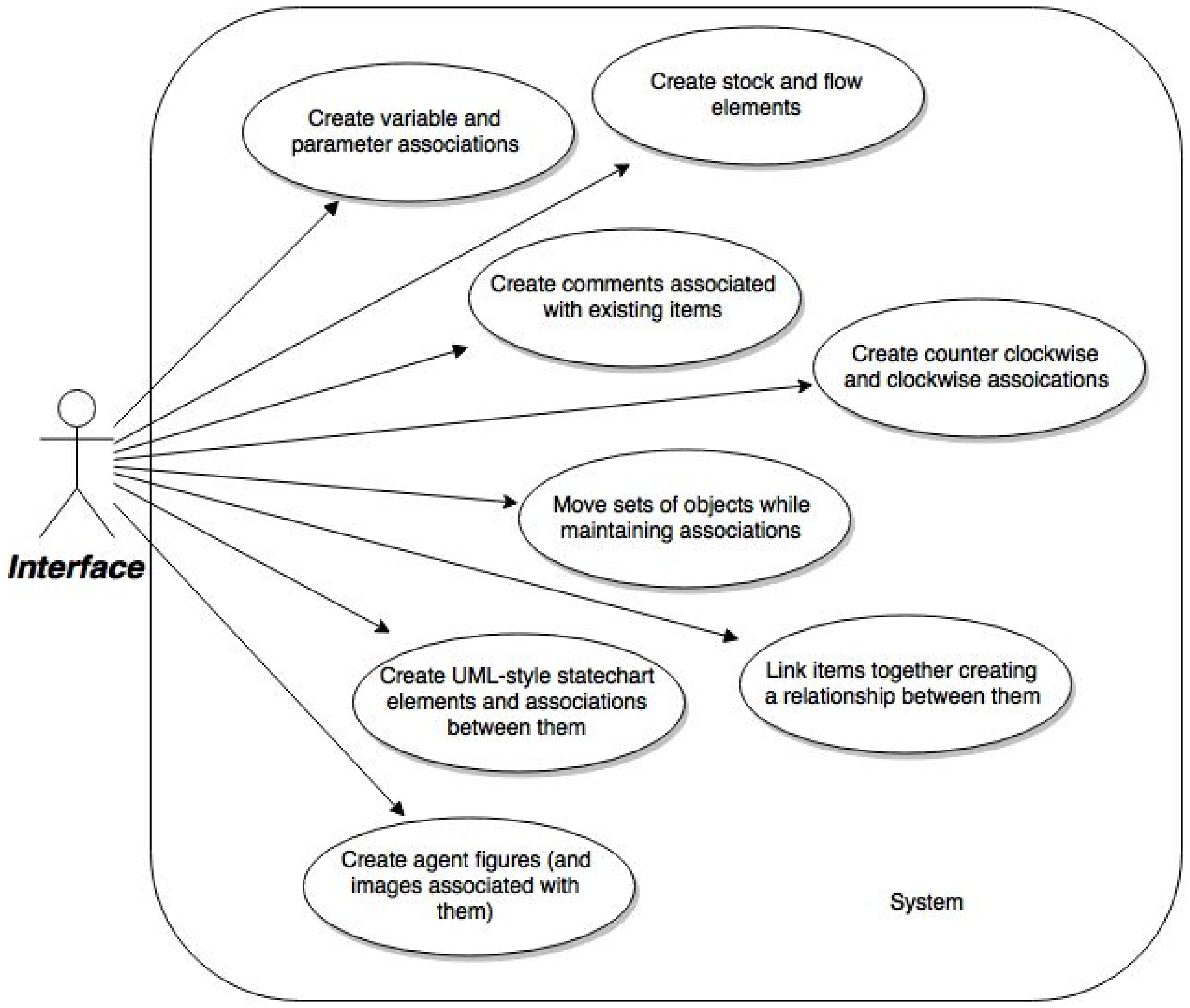
Other than that, we have generalized the interactions which a user would take. Due to the complexity of the actual interface and how it works, this has been saved for another diagram showing off the interface as an actor which interacts with objects.


System
Load user's previous diagrams
Access the web app
Create diagrams
Edit diagram
Save diagram
Add objects and associations to a diagram
Change the font/style/size/ properties of objects
User
Other Users
Interface

Use case describing the actors and interactions looking beyond ID1

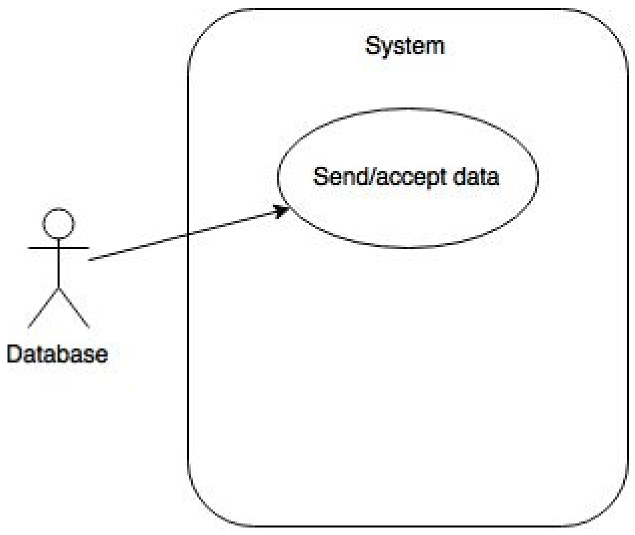
Lastly, we have created a use case for the abstract idea of an interface, seen below. The point of this was to better specify exactly what the interface with the system, so we could broadly say the user was simply adding, editing and creating diagrams.

This use case would similarly extend to the user, as they should be able to create all of the things listed in the use case, except with the interface as an actor, we can specify requirements such as moving while maintaining association between sets of objects. Without the relationships and associations between objects, the project loses a lot of value.

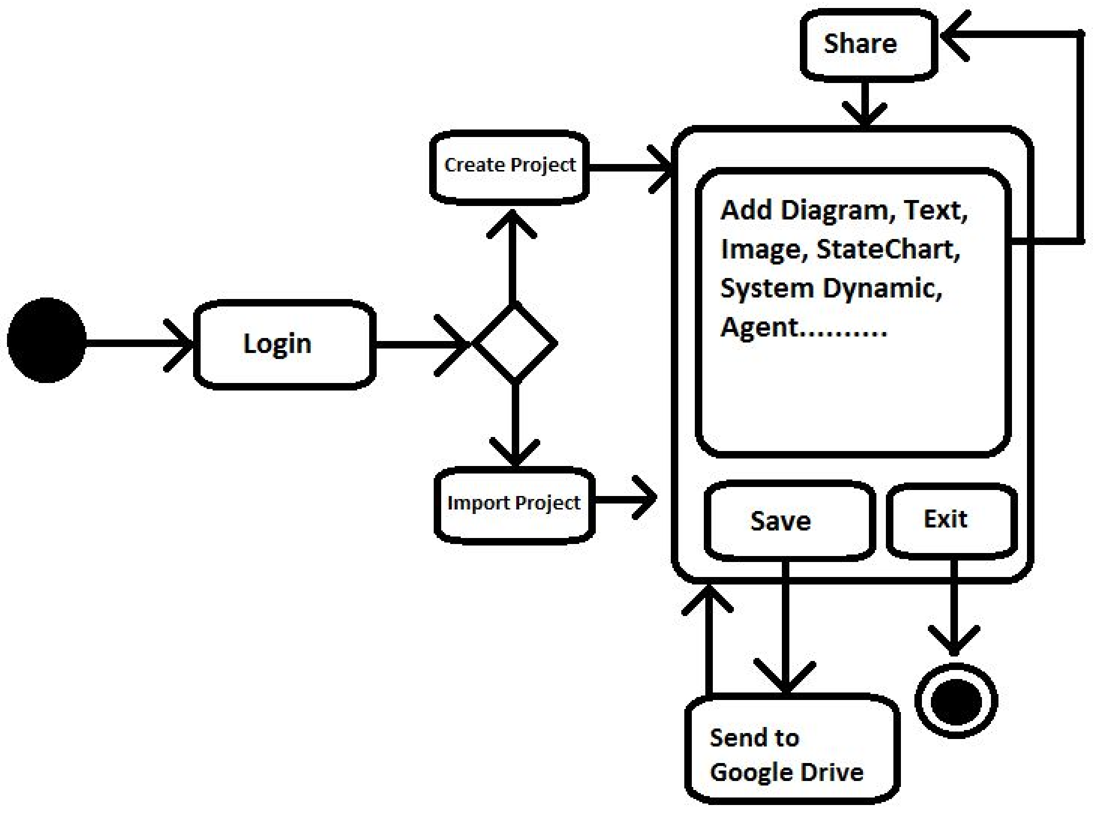

Create variable and parameter associations
Create stock and flow elements
Create comments associated with existing items
Create counter clockwise and clockwise assoications
Move sets of objects while maintaining associations
Create UML-style statechart elements and associations between them
Link items together creating a relationship between them
Create agent figures (and images associated with them)
Interface
System

Use case diagram of an abstract interface actor and how it interacts with the system

Lastly, we have included for the sake of completeness, a use-case diagram with our database as an actor. Currently, the database system is relatively unexplored and not even certain if it is needed, but it could be a potential actor interacting with the system depending on how we use it.



VI. Activity Diagrams


Share
Create Project
Add Diagram, Text, Image, StateChart, System Dynamic, Agent..........
Login
Import Project
Save
Exit
Send to Google Drive

VII. Storyboards

Image to show System Dynamic:

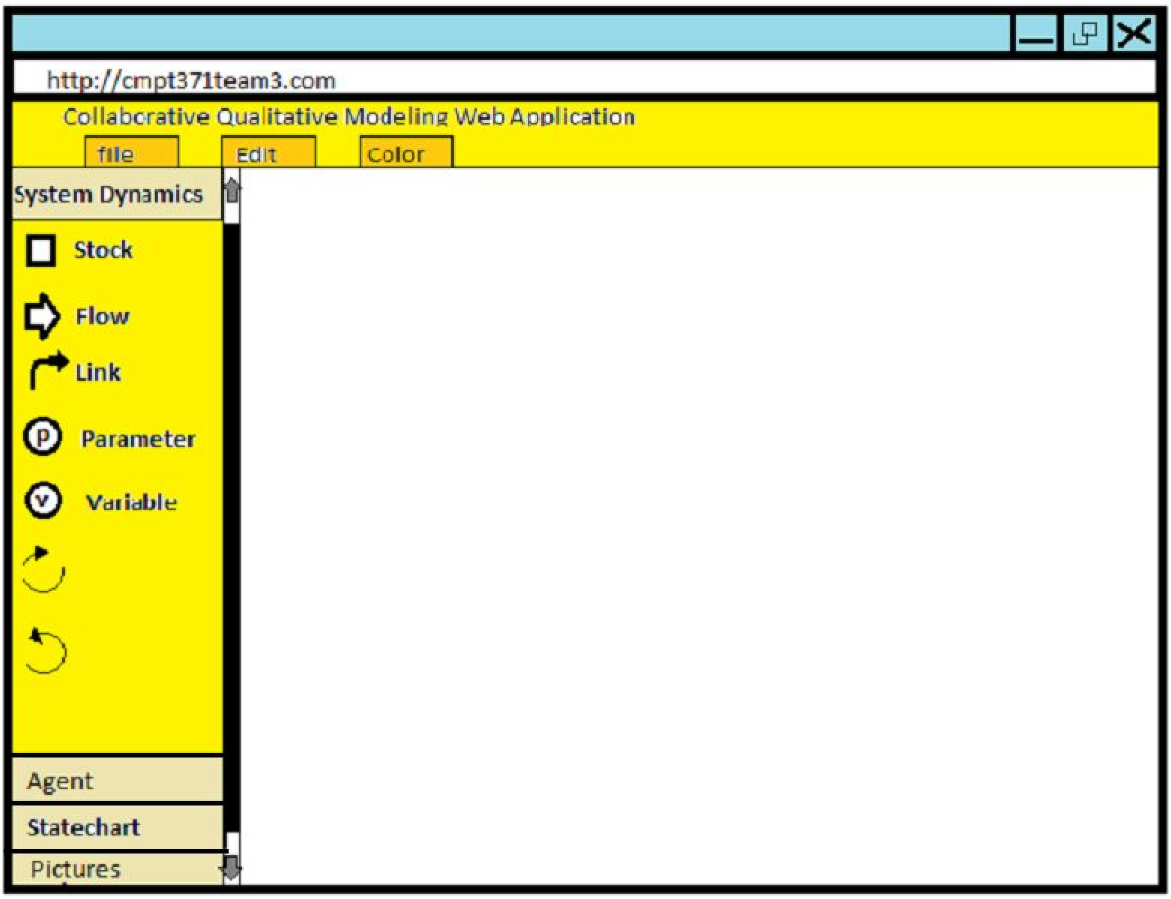
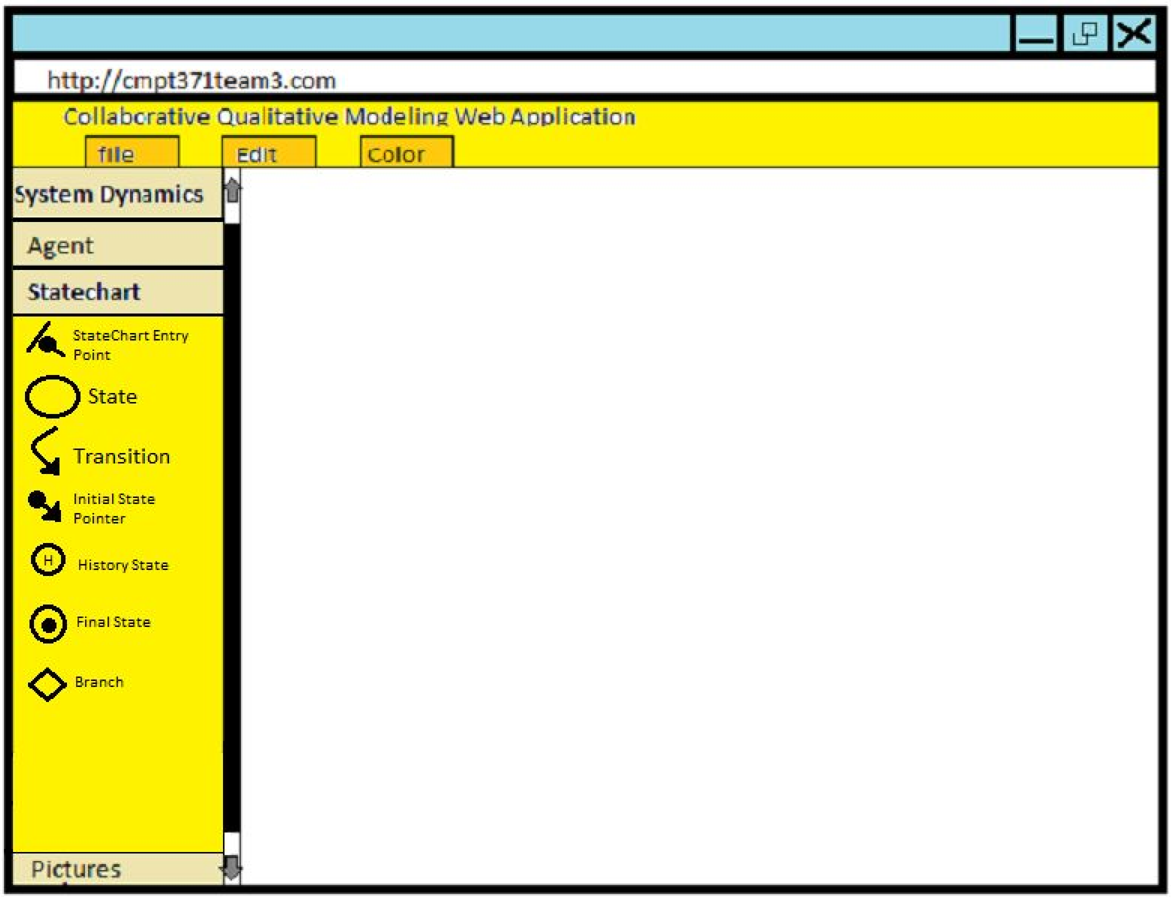


Image to Show StateChart:



VIII. Testability

Whether or not we can test certain requirements.

IX. Conclusion