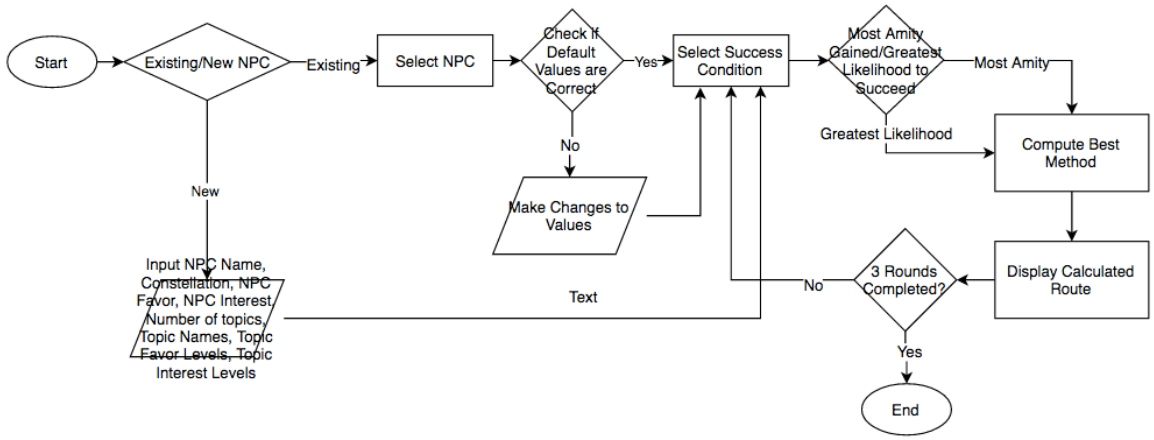
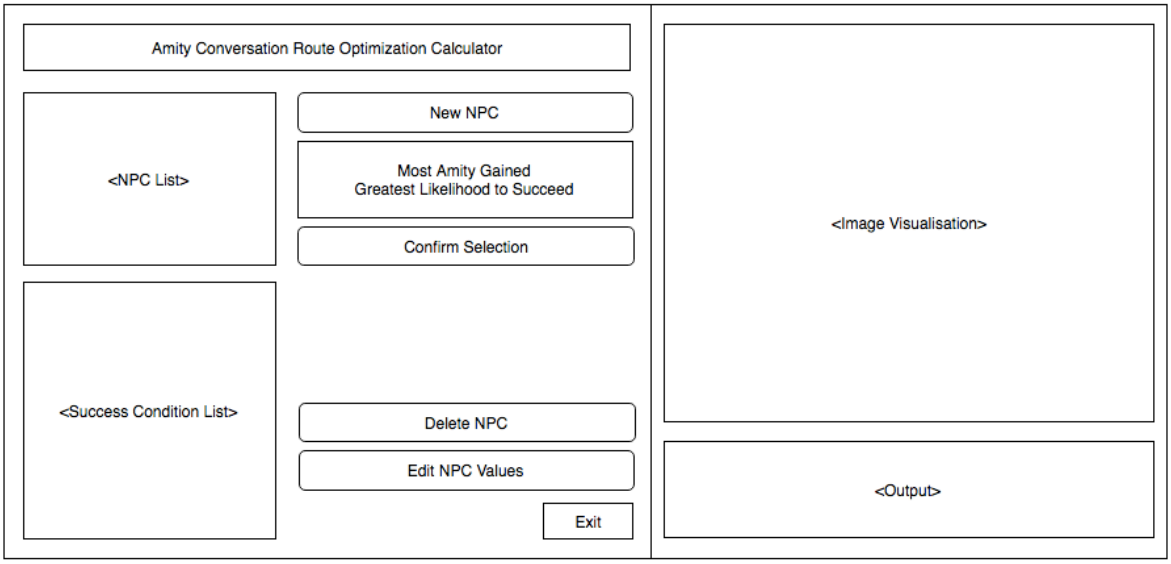
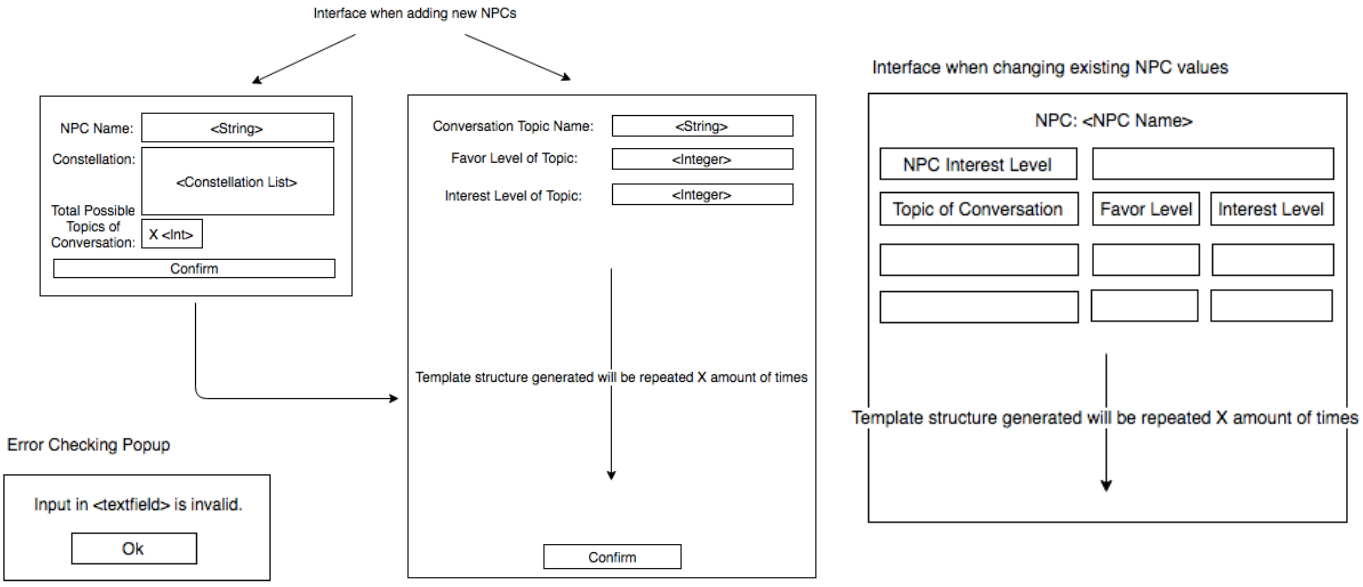
Program Flowchart Structure Outline



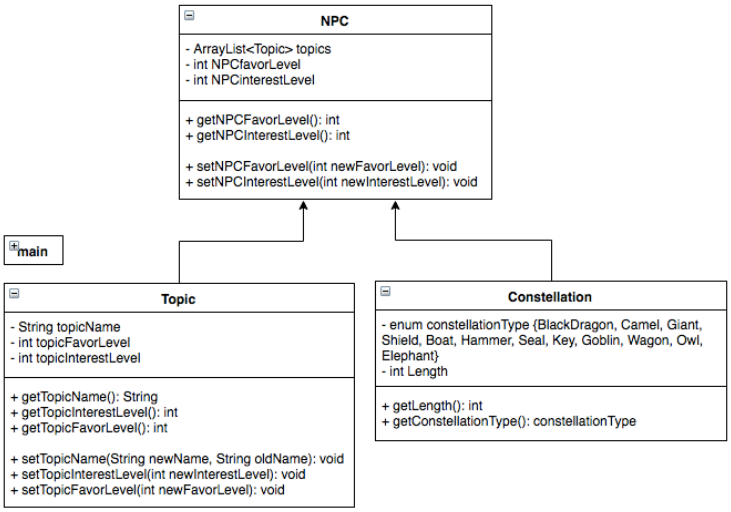
User Interface Design

Main Interface





UML Class Design



Constellations will be fixed and preset, hence no mutator methods are present.

Pseudocode of "Compute Best Method" Calculations for "Most Amity" and "Greatest Likelihood of Success" Criterion

```
get NPC #takes NPC from GUI user selection
int[] interests #NPC topics interestrates array
for int counter from 0 to length(topics) #extracts NPC topics interests and places them in array
            interests[counter] = topics[counter].getTopicFavorLevel
int[] percentageSparkInterest #NPC percentageSparkInterest conversion values from interests[] array
for int counter from 0 to length(interests) #converts percentage of sparking interests and places into above array
            if (topics[counter].getTopicInterestLevel()/NPC.getNPCInterestLevel()) > 1
                        percentageSparkInterest append(1)
            else
                        percentageSparkInterest append(topics[counter].getTopicInterestLevel()/NPC.getNPCInterestLevel())
get successCondition #takes successcondition from GUI user selection
get mostAmity/mostLikely #takes mostAmity/mostLikely from GUI user selection

int[constellation.getlength()] positions #order and number of placements array
int likelihood = 1; #Default value of success, 100%, will change according to best route calculations below

Output bestRoute(successCondition, mostLikely/mostAmity)

String bestRoute(successCondition, mostLikely/mostAmity) #bestRoute computation method
            for int count from 0 to constellation.getlength() #loop adding to the positions of the best topics of conversation
                        if successCondition = (sparkInterest5 || sparkInterest4 || sparkInterest3 || sparkInterest2 || sparkInterest1 || talkfreely)
#initializing values of max/min posmax/posmin depending on success condition
                                    if mostLikely
                                                max = percentageSparkInterest[0]
                                    else if mostAmity
                                                max = NPC.topics[0].getTopicInterestLevel()
                                    posmax = 0
                        else if successCondition = (failsparkInterest5 || failsparkInterest4 || failsparkInterest3 || failsparkInterest2 ||
failsparkInterest1)
                                    if mostLikely
                                                min = percentageSparkInterest[counter]
                                                posmin = 0
                                    if mostAmity
                                                max = NPC.topics[0].getTopicInterestLevel()
                                                posmax = 0
                        for int counter from 0 to length(NPC.topics) #getting highest amity/highest percentageSparkInterest and its positions
                                    if mostAmity
                                                if NPC.topics[counter].getTopicInterestLevel() > max && counter !in positions[]
                                                            max = getTopicInterestLevel[counter]
                                                            posmax = counter
                                    else if mostLikely && successCondition = (sparkInterest5 || sparkInterest4 || sparkInterest3 || sparkInterest2 ||
sparkInterest1 || talkfreely)
                                                if percentageSparkInterest[counter] > max && counter !in positions[]
                                                            max = percentageSparkInterest[counter]
                                                            posmax = counter
                                    else if mostLikely && (failsparkInterest5 || failsparkInterest4 || failsparkInterest3 || failsparkInterest2 ||
failsparkInterest1)
                                                if percentageSparkInterest[counter] < min && counter !in positions[]
                                                            min = percentageSparkInterest[counter]
                                                            postmin = counter
                        if ((sparkInterest5 || sparkInterest4 || sparkInterest3 || sparkInterest2 || sparkInterest1 || talkfreely) && mostLikely) ||
((sparkInterest5 || sparkInterest4 || sparkInterest3 || sparkInterest2 || sparkInterest1 || failsparkInterest5 || failsparkInterest4 ||
failsparkInterest3 || failsparkInterest2 || failsparkInterest1 || talkfreely) && mostAmity) #placing positions of selected values into array
                                    positions[count] = posmax
                        else if ((failsparkInterest5 || failsparkInterest4 || failsparkInterest3 || failsparkInterest2 || failsparkInterest1) &&
mostLikely)
                                    positions[count] = posmin
                       if successCondition = (failsparkInterest5 || failsparkInterest4 || failsparkInterest3 || failsparkInterest2 || failsparkInterest1)
                                    likelihood *= (1-percentageSparkInterest[posmax]) #computing new likelihood of success with the new value in the
order
                        else
                                    likelihood *= percentageSparkInterest[posmax]
            String output = "Route: "
            for int count from 0 to constellation.length()
                        output += topics[positions[count]].getTopicName() + "-"
            output += "Likelihood " + likelihood.toString()
            return output #Output returned as a string
```

Testing Plan

Action test	*Method of Testing/Result*
Test if program interface opens correctly	Double click on program icon, a window with the UI appears
Check if the interface for adding a NPC works	Click on button "New NPC", the correct window interface appears
User input presence check, prompting error popup	User does not choose a NPC from the list, clicks confirm selection, error popup appears, calculation not performed
Too many inputs check	User selects multiple NPCs and/or multiple success conditions, error popup appears, calculation not performed
Test if program computes the correct optimal route	Self calculate optimal route, compare with program generated route
Test if UI generates correct number of topic inputs when user adds a new NPC	Enter the values of 1-10 for "Total possible topics of conversation" when adding a new NPC, the correct number of repeated topic inputs appear
Check that the "Delete NPC" button works correctly	Select a/multiple NPCs from the list, click delete button, NPC is removed from the list and background data
Check that the image visualization displays the correct constellation	Once the user selects a constellation, it appears on the image visualization pane, check that it is the correct image by comparing it with the one in-game
Check if the algorithm for adding a NPC works	Add a NPC, and perform calculations of optimal route, compare it with output value
Check that the tick boxes for "Most Amity Gained" and "Greatest Likelihood to Succeed" cause the correct outputs to appear	Select NPC, select success condition, select MAG and/or GLS, click confirm, if both ticked 2 outputs should be displayed, if one is ticked manually calculate that the correct optimal route is outputted.
Check that the exit button works	Click exit, all open windows should close.

(Word Count: 285)