Design Decisions for the Development of a GUI to Assist the C2KA TOOL[1]

By: Dylan Leveille, Idir Zerrouk

May 14:

- Read about the Haskell language.
- Read PHD thesis (ON THE MODELLING, ANALYSIS, AND MITIGATION OF DISTRIBUTED COVERT CHANNELS) and research article (Mitigating covert channels based on analysis of the potential for communication).

May 15:

- Continued readings.
- Math was deemed complicated; explanation of the basics were given by the professor which greatly aided the understanding of the GUI we were supposed to develop.

May 16:

- Commenced development of the main window for the GUI using tkinter (made for python).
- Page to add stimuli was created, which included a button to remove an entry box and another button to add a stimuli. We decided the entry boxes would stack on top of each other.
- Page to add an agent and its behaviour was created. An entry box was made for the agent name, and another entry box underneath it was created to enter the agent's behaviour. We used a histogram function that we developed earlier this year in the Fall term[2] to extract the agent behaviours from the string typed in th entry box.
- To link the pages created, we created a global variable pageNum which keeps track of the page number throughout the program. The pageNum variable is used with the previous and next page buttons, decreasing by 1 when previous page is clicked, and increasing by 1 when next page is clicked. The pageNum variable is global since button functions cannot take parameters.
- To store entry boxes, list were used to hold the entry boxes, and a dictionary was subsequently created when the next page button was clicked to store the entry box values in their respective dictionaries (one for the stimuli, another for the behaviours).
- Buttons were packed in the bottom of the GUI in layers since we were not able to have them stay in the bottom corners using this button placement geometry.

May 17:

- We attempted to implement a scrollbar for the stimuli page, unfortunately, it was difficult to do this. Therefore, we left the scrollbar out of the GUI.

---

[1] Tool created by Dr. Jason Jaskolka

[2] Course: SYSC 1005

- We chose to design a table for the circle and lambda tables; this proved difficult, since the tables had to be created using the grid geometry, but our program used the pack geometry (and cannot be combined together). Therefore, we created the table within a frame, and used the pack geometry on that frame to trick tkinter.
- We learned on this day that data structures must only be declared global once to be used in every function, but other types must be declared globally in each function.
- Like the stimuli and behaviour pages, the data entries and labels of the table were saved in a data structure. We chose to use a dictionary to save the boxes and use tuples as the keys to represent the coordinates of each entry box in the tables.

May 18:

- Learned that python does not like it when you change a counter variable used in a for loop within the for loop. Therefore, to delete rows and columns that were no longer in use in the tables, we decided to use a recursive function which keeps track of the number of rows to delete and of the current row it is checking. When a row or column was marked for deletion, it is shifted to the bottom of the row/column. This way, once the function has returned, we apply another function to iteratively delete the number of rows/columns returned by the functions.
- To modularize the code, a function was created to handle all things related to modifying the table (fix_grids()).

May 22:

- Worked on functions for the tables to add rows/columns if stimuli or behaviours were added. This was done by adding a row/column at the bottom of the table and then switching the row/columns to math the order defined by the user.
- Order of table labels was alphabetical at first, but we decided to implement the tables' order with the user's specified order instead.
- Now that the table had been mostly functional, we created a create_text() function which generated the text file from the data gathered throughout the program. We were able to find the correct amount of whitespace to use to align the '=' sign using python's built-in max() function.

May 23:

- The tables were fully functional (or so we thought).
- We got rid of the set in the histogram to collect the behaviors, and went with a list instead, since in a set, there is no specified order in which the items are sorted, which omits the user's order in the process.
- We found a source code online which implemented the scrollbar we wanted (with both horizontal and vertical scroll). This was hard to implement, since the scroll length wouldn't adjust when a new stimuli was added to the page.

May 24:

- Scrollbar was finally implemented. To allow it to resize itself when a new stimulus was added, the frame had to be forgotten and re-created every time. This is very inefficient, but is the only way we could get it to work well. The process to make it work relied on trial and error.
- The window size was chosen to be set to a specific size, and not resizeable.
- Started to implement concerte behaviours with agents (to be used in the text file).

May 25:

- Started to implement error handling with pop-ups in the table page.
- We decided to use a dictionary to hold table values rather than to extract them from the dictionary containing the tables' entry boxes (more efficient this way for later functions).
- We decided that the pop-up window for the errors would hide the main window until the 'return' button is clicked (to help convey the message).
- Added a more engaging UI for the concerte behaviours (similar to the stimuli page).
- The 'X' button could not be implement with a close window command on an Apple computer. Therefore, we decided to remove the File Menu bar, hence forcing the user to close the window using the 'return' button.

May 28:

- Got familiar with GitHub.
- Finished pop-ups for the tables.
- Decided that empty (blank) entry boxes in the stimuli page would be ignored by the dictionary.
- Pop-ups not resizeable (like the main window).
- Lambda wouldn't display properly when the project was downloaded from one computer to another.
- A pop-up was created for when no behaviour was entered.

May 29:

- To support scrollbar feature on tables, the tables (similar to the stimuli entry boxes) must be generated every time the user returns to the table page.
- There was a problem with the table; turns out we forgot to keep track of the tables' new size everytime fix_grids() was called, resulting in errors. This was fixed and the tables became fully functional.
- Fill lambda table with neutral stimulus button was added.
- We tried very hard to put fraktur in the entry boxes for the fillN() function (to look pretty), but it turns out that tkinter entry boxes cannot be inserted 'odd' characters.

May 30:

- We tried to have the behaviours automatically be uppercased, but this was too difficult to implement (since a new case function had to be created for each agent the user may need). Therefore, we went with an approach which uppercases the behaviours in the dictionary once entered.
- Implemented a better parsing which validates every entry entered in the entire program (everytime the next button is pressed).
- The buttons were successively packed in the bottom corners (by hard coding their width). Although this is not the best approach, since the main window cannot be resized, it is an acceptable solution.
- We decided that stimTitle should be in the scrollbox, with the entries, to make the UI look better on that page. Unfortunately, this means it must be recreated every time the addStim button is called () in order to be placed in the new scrollbox.

May 31:

- The code has been modularized (took all day to do). It is now easier to find what you're looking for in the code.

June 1:

- Designed a button to add a specified number of stimuli in the add stim page.

June 4:

- Learned that, in python, variables of any kind declared in the global frame are global. Therefore, no need to declare them as global variables in a function.
- Tried to make stimuli-related functions more efficient by making a list which holds the words from the entries.
- To delete the scrolling frame for the stims, we decided to use a global variable; this makes the program more efficient since there is no need to hide a packed frame behind the scenes.
- Specified number of stims button nearly finished.
- Found out you can modify something by reference if passed in a list.

June 5:

- Implemented delete CBS function for the concrete behaviours.
- SpecifyStim button fully functional.
- Starting to implement a file explorer to have the user save the text file wherever he/she wants.
- Implemented a preview of the text file before having it saved on the computer.

June 6:

- The file explorer is now functional with the save button.
- Highlight invalid stimuli.

- Completed concrete behaviours and scrollbar to go with it.

June 7:

- Heavily documented code for doxygen.

June 8:

- Finished documenting code.
- Tried creating an executable.
- Made the entry stim warning more user friendly by waring the user of an incorrect entry before having the user confirm they want to delete the stims.
- Added parsing for concrete behaviours.

June 11:

- Handling errors for CBS. No entry will give a pop-up.
- Working on a 'Read-Me' file for GitHub.
- Making CBS appear in text file.

June 12:

- Better (more versatile) code for CBS. The create_text() function now splits by line, and then by word, to make sure that the concrete behaviours are always properly displayed.
- Modified the function to get behaviours (better parsing).
- Behaviours now grouped based on their similarities.

June 13:

- Better parsing for agent behaviour entry (corrects for right amount of whitespace).
- Better looking UI. We pictures of arrows to indicate the next/previous page buttons. We also used an image of a plus and and x to respectively add and remove stimuli. Furthermore, the ttk module was imported to make better looking widgets.
- Made pop-ups better (no more override to remove task bar). Instead, pop-ups are now similar to pop-ups in Windows.

June 14:

- Better create text function.
- Stimuli scrollbar much better (works with a delete button next to each entry).
- More efficient to delete specific stimuli boxes, since we decided to simply delete at an index (this automatically shifts the boxes in the list).
- The red colour is now less alarming.
- Created a button to fill circle table with the behaviour present in each row.

June 18:

- Multiple agents calls for more complicated UI (still friendly).

- Changed to a scrolling area for multiple agent entry.
- Decided to use a different layout if the number of agents is greater than one on page 3.

June 19:

- Had to change all the data structures to even bigger forms so as to hold the same information but for multiple agents.

June 20:

- Made multiple agents work with a single agent.

June 21:

- Worked on handling errors for multiple agents.
- Tables are only recreated if any modifications were made.

June 25:

- Had multiple bugs related to the multiple agents page. Most of the bugs were fixed by assigning the dictionaries to their updated versions (for each agent).

June 26:

- Made the create text function work for single and multiple agents.
- We chose to simply rename and reconfigure the edit scrolling area (for multiple agents) to create the save text page. Therefore, there is no need to create another scrolling area.

June 27:

- Made pop-ups wider to completely show the text in the message.
- Added a pop-up error if two agents of the same name were entered. Duplicates are highlighted red.
- Made it possible to switch two agents' positions.
- Implemented errors if the user tries to return to a previous page without closing any of the pop-ups.

June 28:

- The window sizes are no longer hardcoded, but relative to the screen's resolution, making the program compatible with different screen resolutions.

June 29:

- Added a blue circle to indicate if the user has previously edited an agent (for the multiple agents page). This way, the user doesn't need to remember which agent he/she modified.

July 3:

- Made the program a bit more efficient by recreating the concrete behaviours page/tab only if modifications (new behaviours) have been made to the agent(s).

July 3:

- Made it so that if one agent was multiple, but now single, the entered data does not get removed from memory. Therefore, the tables and CBS page keep the data that was entered for the multiple agents.
- Added better looking labels and a scrollbar in the multiple agents edit page in case the agent names get too long.

July 4:

- Made it so that if a multiple agent becomes single, its data isn't forgotten in memory (and vice versa).
- Made some buttons fit more comfortably in the window.

July 5:

- Fixed switching agent bug found in code when going from single agent to multiple agents.
- Finished dependency diagram using Lucidchart.
- Packed buttons better in the program.

July 6:

- Made resolution dependencies better by testing it out for different resolutions.
- Fixed yet another bug when going from multiple agents to single, then back to multiple (by adding a statement in the if statement).
- Shared a link for the lucidchart diagram.
- Added a better packing order for the buttons/scrolling areas.
- Added the design decisions in the GitHub.