SmartSPEC GUI Manual

Andrew Chio

Version 1.2

Preface

Contents

Version Updates

Version	Changes
1.2	• Fixed instructions to compile and run via command line.
1.1	• Added instructions to compile and run via command line/IDE. • Cleaned dependencies for SmartSPEC GUI.
1.0	• Original SmartSPEC GUI Manual published.

1 Introduction

This document is a general guide that describes the use of the SmartSPEC GUI to generate smart space scenarios. Our code is available on GitHub [1]. If you use SmartSPEC in your work, please reference the papers in [2, 3].

2 Installation and Running

In this section, we cover the dependencies used by the SmartSPEC GUI, as well as instructions for compiling and running our program.

Dependencies. The SmartSPEC GUI is written in Java and requires the following dependencies:

- Java FX (version ≥ 17) [1]
- json-simple (version $\geq 1.1.1$) [2]
- ini4j (version $\geq 0.5.4$) [3]

We recommend that JavaFX is installed as described in the linked URL, which describes the installation process for using JavaFX with an Integrated Development Environment (IDE), or with command line build tools. The `json-simple` and `ini4j` dependencies should be downloaded as `jar` files.

Compiling and Running using Command Line. To compile the SmartSPEC GUI on a Linux machine, run the following command from the `gui-src` directory:

```
javac \
    -cp "<path-to-json-simple>:<path-to-ini4j>" \
    --module-path <path-to-javafx> \
    --add-modules javafx.controls \
    controller/*.java \
    form/*.java \
    model/*.java \
    modelutils/*.java \
    view/*.java \
    viewutils/*.java \
    application/Main.java
```

For windows users, the classpath (i.e., the `-cp` option) should use a semicolon (`;`) instead of a colon (`:`) for the delimiter. Additionally, the backslash (\) should be used instead of forward slash (`/`) in all filepaths. The multiline separator (\) is used for clarity and can be omitted (or replaced with the equivalent Windows multiline separator).

[1] `https://openjfx.io`
[2] `https://code.google.com/archive/p/json-simple/`
[3] `https://ini4j.sourceforge.net`

Then, to run the SmartSPEC GUI, run the following command:

```
java \
    -cp ".:<path-to-json-simple>:<path-to-ini4j>" \
    --module-path <path-to-javafx> \
    --add-modules javafx.controls \
    application.Main
```

Important 1: Please note the `".:"` in front of the class path!
Important 2: Please note that `application.Main` is used here!

Compiling and Running using an IDE. In general, you must (i) add the jar dependencies to the build path of the project; and (ii) add the arguments `--module-path <path-to-javafx> --add-modules javafx.controls` to the run configurations.

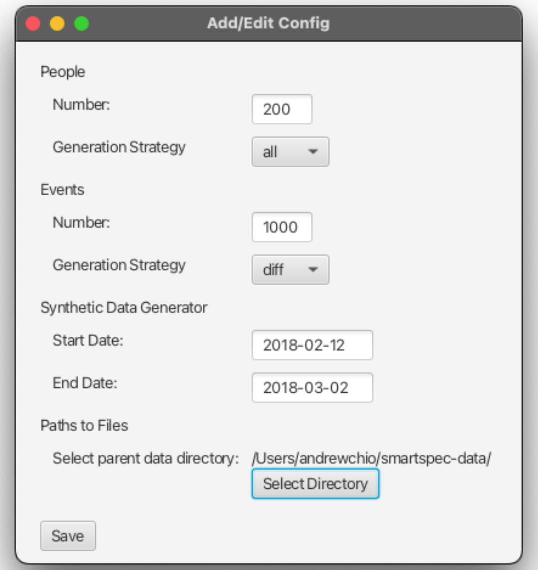
3 QuickStart Tutorial

In this section, we will describe how the SmartSPEC GUI can be used to generate a scenario. The tutorial scenario and its corresponding data can be found in the SmartSPEC GitHub [1]; the source code is located in the `gui-src` directory, while the data is in the `scenario-generation/data/data-gui` directory.

3.1 Setting up the Configuration file

The SmartSPEC GUI first requires that a configuration file is defined or loaded. This configuration file will contain information about how people and events should be automatically generated, the duration that the simulation should run, and the paths to other required input files. To initialize a configuration file, start by going to **Config > Add/Edit Config**. This will create a new window as seen in Fig. 1. Then, configuration options should be entered as listed below:

1. **People and Events**. In the configuration options for people and events, two main entry fields are presented: `Number` and `Generation Strategy`. The `Number` field indicates the number of people / events that should be in the designed scenario, and must be set to a non-negative integer. The `Generation Strategy` field indicates the manner in which people / events will be produced. It must be set to one of "all", "diff", or "none". If the generation strategy is set to "all", then `Number` new people / events will be produced. If the generation strategy is set to "diff", then one person / event from each metaperson / metaevent will first be generated (up to `Number`), followed by additional people / events (up to `Number`). If the generation strategy is set to "none", then manually defined people / events will be used. Note that `Number` should still be set to "0" in this case.



People

Number:	200
Generation Strategy	all

Events

Number:	1000
Generation Strategy	diff

Synthetic Data Generator

Start Date:	2018-02-12
End Date:	2018-03-02

Paths to Files

Select parent data directory:	/Users/andrewchio/smartspec-data/

Figure 1: Input form for Config File.

Tip: It is recommended that the generation strategy for both people and events are set to either "all" or "diff" to reduce manual effort.

2. **Synthetic Data Generator**. In the configuration options for the synthetic data generator, the start / end date should be specified. These indicate the start / end dates of the entire simulation. Both dates must be in the format `YYYY-MM-DD`, and the start date must be before the end date.

3. **Paths to Files**. The last configuration option requires a parent (base) directory in which all other input files should be located. Click the `Select Directory` button and select the directory to store all other input files. The selected directory should be displayed in the config menu window afterwards.

   Tip: It is recommended that the SmartSPEC GUI is run on the same machine in which the SmartSPEC code will be run, since absolute file paths to SmartSPEC input will be recorded in the configuration file.

4. **Saving**. After all configuration options are specified, click the `Save` button to save the configuration file. If this is successful, then the SmartSPEC

GUI should display “Saved config file: `config-filepath`” in the output pane. By default, the name of the configuration file is set to `config.txt`.

5. **Loading**. If a configuration file has already been defined, then it can be loaded by going to **Config** > **Load Config File**, which will open a prompt to select a configuration file. If this is successful, then the SmartSPEC GUI should display “Loaded config file `config-filepath`” in the output pane.

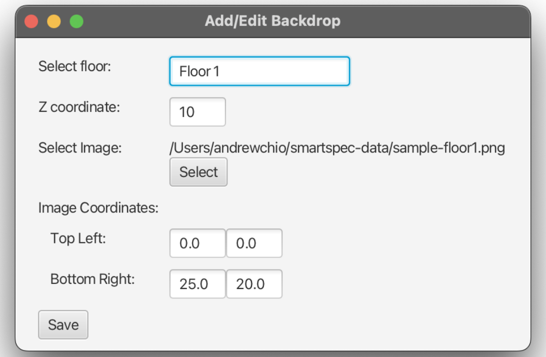
   Important: The base directory specified in the loaded configuration file (under `filepaths/generated-files`) should be the same as the directory where the configuration file exists.

3.2 Setting up Spaces and Backdrops

After setting up the configuration files, the underlying space in which the simulation will take place should be defined. This will consist of two main steps: loading space backdrops and defining a space graph.

3.2.1 Backdrops

Space backdrops are images of the layout / floor plan of the space. They are used to indicate where certain spaces should be located. A new space backdrop can be loaded by going to **Spaces** > **Load Backdrop**. This will create a new window as shown in Fig. 2, and should be filled out as follows:


Add/Edit Backdrop

Select floor:	Floor 1
Z coordinate:	10
Select Image:	/Users/andrewchio/smartspec-data/sample-floor1.png
Image Coordinates:	
Top Left:	0.0 0.0
Bottom Right:	25.0 20.0

Figure 2: Input form for Backdrop File.

1. **Floor**. The floor entry is used to provide a semantic name for the floor to define. It will be displayed in the title of the tab dedicated to the provided

backdrop. The floor name cannot contain parenthesis and must be unique (i.e., duplicate floor names should not be used).

2. **Z-Coordinate**. The z-coordinate represents the "height" of the floor, or the distance (e.g., in meters) above ground level. Similar to the floor name, the z-coordinate should be unique for each backdrop.

   Important: The z-coordinates used for different floors should be on the same "scale" defined as for image coordinates (e.g., in meters; see item 4).

3. **Image**. The image data entry prompts for an image to display as the layout of the floor. Click the `Select` button to select the backdrop image. The selected file name should be displayed in the backdrops form window afterwards.

   Note: The size (i.e., width, height) of the uploaded image must be modified as desired before loading it into the SmartSPEC GUI. Resizing the image in the GUI is not yet supported.

4. **Coordinates**. The image coordinates are used to define the top left / bottom right corners of the image. Conceptually, these numbers should define the physical location of the top left / bottom right corner (e.g., in meters). The xy-coordinate specified for the bottom right corner must be *larger* than the xy-coordinate specified for the top left corner in value (i.e., x-coordinates will increase by going right; y-coordinates will increase by going down).

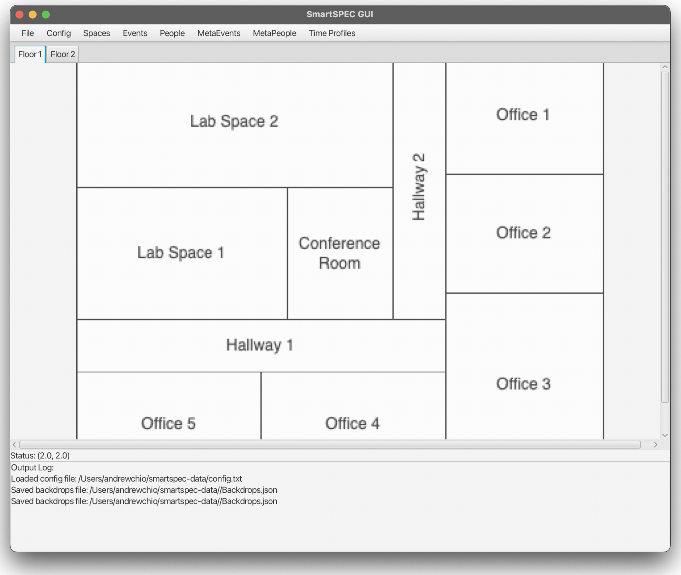
   Important: The coordinates specified here should be on the same "scale" defined as for z-coordinates (e.g., in meters; see item 2).

5. **Saving**. After all data entries are entered, click the "Save" button to save the backdrop. If this is successful, then the SmartSPEC GUI should display "Saved backdrops file: `backdrops-filepath`" in the output pane, and the new backdrop should be displayed in the main window, such as in Fig. 3.

3.2.2 Space Nodes and Edges

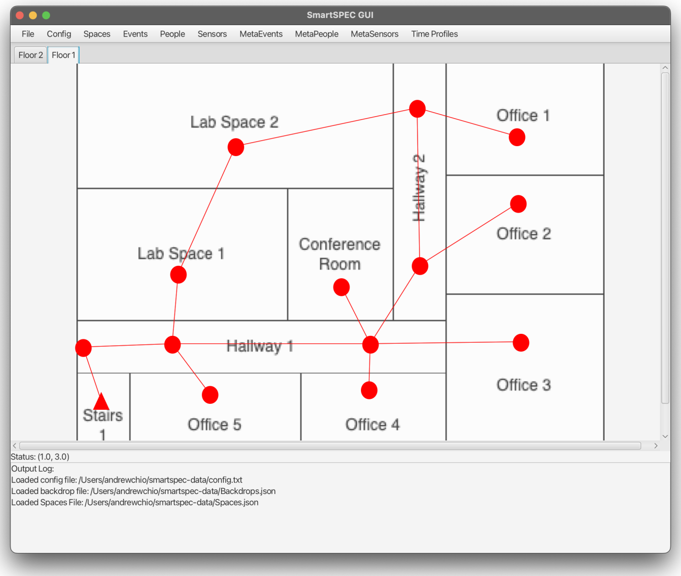
SmartSPEC internally uses a space graph to represent logical spaces and their neighbors. An example of a fully defined space graph can be seen in Fig. 4. In general, this must be done visually on the loaded space backdrop as follows:

1. **Adding a new space node**. To add a new space node, right-click on an appropriate location on the space backdrop, and select the context menu item to add a new space (i.e., **Right-Click** > **Add new space**). This will open a new window with partially-filled data entries for a new space, as seen in Fig. 5. The `description` field is optional and serves as a semantic label for the space; the `capacity` field must be a non-negative integer representing the maximum number of people that can physically be in the room.



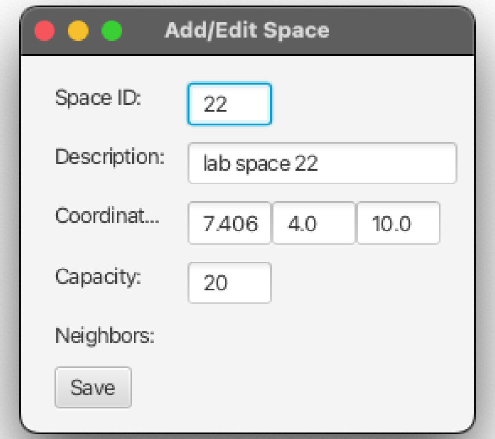
```
Status: (2.0, 2.0)
Output Log:
Loaded config file: /Users/andrewchio/smartspec-data/config.txt
Saved backdrops file: /Users/andrewchio/smartspec-data//Backdrops.json
Saved backdrops file: /Users/andrewchio/smartspec-data//Backdrops.json
```

Figure 3: SmartSPEC GUI after defined backdrop



```
Status: (1.0, 3.0)
Output Log:
Loaded config file: /Users/andrewchio/smartspec-data/config.txt
Loaded backdrop file: /Users/andrewchio/smartspec-data/Backdrops.json
Loaded Spaces File: /Users/andrewchio/smartspec-data/Spaces.json
```

Figure 4: SmartSPEC GUI after defined spaces.



Space ID: 22

Description: lab space 22

Coordinat... 7.406 4.0 10.0

Capacity: 20

Neighbors:

Figure 5: Input form for Space File.

Note 1: After a space is added, hovering over a space node will show a small tooltip with the ID and description of the space.

Note 2: A pink circle will be temporarily drawn on the space backdrop as a placeholder for the location of the new space. This will turn into a red circle after the space is properly initialized.

Note 3: Spaces must all have a unique ID. See item 2 if a previously defined space must be edited.

Important: One of the space nodes defined must be the special "outside" space node from which all people enter / exit the smart space. It is recommended to set the capacity to some large number.

2. **Editing a space node**. If a previously defined space node needs to be edited, then right-click on the node and select the option to `Edit space XX`. This will open another window with the data associated with the selected space.

   Note: The ID field of an edited space cannot be modified; delete the node and create a new one if the ID needs to be changed.

3. **Deleting a space node**. If a previously defined space nodes needs to be deleted, then right-click on the node and select the option to `Delete space XX`. The node and its associated edges should be then removed.

4. **Space Edges on the same floor**. After adding space nodes in the SmartSPEC GUI, space edges (which indicate that a person can travel directly between the two spaces) can be defined by clicking one node, dragging the line that is drawn to another node, and releasing. If the two nodes are not yet neighbors, then adding an edge between them will make them neighbors. Otherwise, if an edge already exists between the two nodes, it will be deleted.

   Tip: It is recommended that all space nodes are defined first, before neighbors are added.

5. **Space Edges between different floors**. The SmartSPEC GUI calls nodes that can be used to access different floors a "z-neighbor". Conceptually, z-neighbor nodes should represent stairs, elevators, etc. To add a z-neighbor, right click on a node and select the option to `Add new z-neighbor`. Then, select the space (on a different floor) to add as a z-neighbor. Similarly, z-neighbors can be removed by right-clicking on a node and selecting the option to `Delete z-neighbor`. The output pane will display appropriate messages if successful.

   Note: A space that has a z-neighbor to a floor above or below it will be denoted with a triangle pointing up or down, respectively. A space that has a z-neighbor to floors above and below it will be denoted with a diamond.

   Important: If there are multiple floors to simulate, at least one node on each floor should be a z-neighbor to another floor in order to be reachable.

6. **Saving**. By default, the spaces JSON file is not saved after each change to the spaces graph. Instead, go to **Spaces** > **Save Spaces File** to write the spaces JSON file. If successful, the output pane should display "Saved spaces file: `spaces-path`".

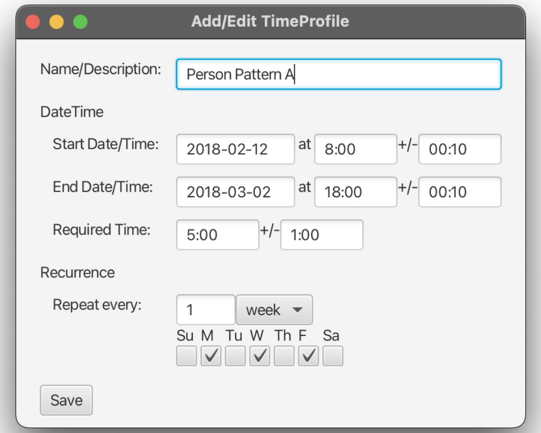
7. **Loading**. If a spaces JSON file has already been defined, then it can be loaded by going to **Spaces** > **Load Spaces File**. This will open a prompt to select a spaces JSON file, and if successful, will display "Loaded spaces file: `spaces-path`".

3.3 Setting up MetaEvents and MetaPeople

After setting up the spaces graph, the metaevents and metapeople for the simulated scenario should be defined. This will consist of three steps: setting up time profiles, declaring both metaevents and metapeople, and then refining metaevents / metapeople.

3.3.1 Setting up Time Profiles

Time profiles are used by both metaevents and metapeople to specify when an event would occur, and when a person enters / exits the smart space, respectively. The general definition for both metaevents and metapeople are the same, as they both indicate periods of time.



Add/Edit TimeProfile

Name/Description: Person Pattern A

DateTime

Start Date/Time: 2018-02-12 at 8:00 +/- 00:10

End Date/Time: 2018-03-02 at 18:00 +/- 00:10

Required Time: 5:00 +/- 1:00

Recurrence

Repeat every: 1 week

Su	M	Tu	W	Th	F	Sa
☐	✓	☐	✓	☐	✓	☐

Figure 6: Input form for Time Profiles.

1. **Defining Time Profiles**. To define a new time profile, click **Time Profiles > Add New Time Profile**, which will open a new window with a time profile form. The `Name / Description` entry is used provide a semantic label for the time profile, and should not contain parenthesis. The `Start Date/Time` and `End Date/Time` data fields are used to indicate the start date/time and end date/time for the time profile. In general, a date should be provided in the first box, the mean time in the second box, and a standard deviation time in the third box. Dates should be in the format "yyyy-MM-dd", while times should be in the format "HH:MM". The entries for required time need two times (a mean time and a standard deviation time), and indicates the amount of time to spend between the given start/end time. The recurrence field is used to indicate the frequency of the time profile. Two recurrent patterns are provided: "day" and "week". If "week" is selected, then the days of the week must also be specified. Fig. 6 shows an example of the data entries for time profiles.

   Note: The SmartSPEC GUI provides a simplified view of time profiles. Currently, it does not support recurrence for months or years.

   Tip: Both time profiles for metaevents and metapeople must be defined in this step. It is recommended that the time profiles for metaevents generally span significantly less time than metapeople. This is since a time profile for an event determines when it occurs in the day, while a time profile for a person determines when they are in the smart space.

2. **Saving**. In general, time profiles are not saved into a file after definition. Instead, go to **Time Profiles > Save Time Profile File** to write a time profile file. The default name for a time profile file is "TimeProfiles.json".

   Note: The format in which time profiles are saved in this file is *not* the same as the format in which time profiles need to be saved for metaevents and metapeople.

3. **Loading**. If a time profile JSON file has already been defined by the SmartSPEC GUI previously, then it can be loaded by going to **Time Profiles > Load Time Profiles File**. This will open a prompt to select a time profiles JSON file. If successful, the output pane will display "Loaded time profiles file: `time-profiles-file`".

   Note: The format in which time profiles are saved in this file is *not* the same as the format in which time profiles need to be saved for metaevents and metapeople.

3.4 Declaring MetaEvents and MetaPeople

Since both metaevents and metapeople rely upon each other, the SmartSPEC GUI requires that users first "declare" all the different metaevents and metapeople for the simulation. In general, this will ease the difficulty in later defining proper metaevents and metapeople. To declare metaevents, go to **MetaEvents > Declare MetaEvents**. Similarly, to declare metapeople, go to **MetaPeople > Declare MetaPeople**.

The declaration form for both metaevents and metapeople are in the same format as seen in Fig. 7, and collects information as follows:

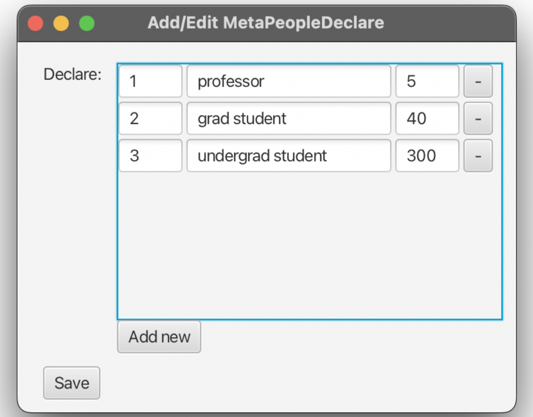
1. **MetaEvent ID / MetaPerson ID**. The leftmost data entry box is for the metaevent ID / metaperson ID. This should be a unique integer with respect to the metamodel.

   Tip: While a metaevent and a metaperson may have the same "id", it is recommended to keep them separate for ease of interpretation.

2. **Description**. The middle data entry box should contain a semantic description of the metaevent / metaperson.

3. **Probability**. The third data entry box should contain the "probability" of each metaevent / metaperson in the simulated space. In particular, the "probability" for a metaevent is the expected number of derived events of the given metaevent type. On the other hand, the "probability" for a metaperson denotes the proportion of the population that is expected to be of given metaperson type.

   Note: The total "probability" across all metaevents / metapeople does *not* need to sum to 1; the SmartSPEC simulator will normalize the input numbers.



1	professor	5
2	grad student	40
3	undergrad student	300

Figure 7: Input Declaration form for MetaPeople.

4. **Adding / Removing entries**. To add a new row, click on the `Add new` button located towards the bottom of the window. On the other hand, to remove a particular row, click on the button "-" to right of the desired row.

5. **Saving**. After declaring all relevant metaevents / metapeople, click "Save" at the bottom of the window to process all entries. The output pane will list all metaevents / metapeople that were properly declared.

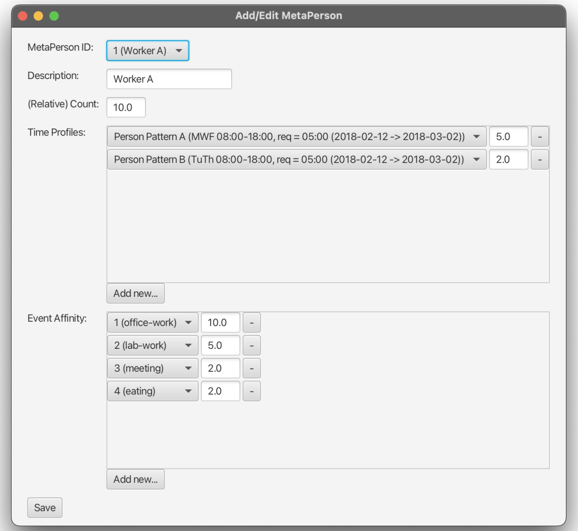
   Important: All relevant metaevents / metapeople should be declared upfront with the correct ID during this step. Currently, we do not support modification of declared entry IDs (after saving).

3.5 Refining MetaEvents and MetaPeople

Once all metaevents and metapeople are declared, both metaevents and metapeople must be individually refined to properly be initialized. To edit a metaevent, go to **MetaEvents** > **Edit MetaEvent...**. Similarly, to edit a metaperson, go to **MetaPerson** > **Edit MetaPerson...**. An example of the form for metapeople and metaevents can be seen in Fig. 8 and Fig. 9.

The form for metaevents should be filled as follows, for each declared metaevent:

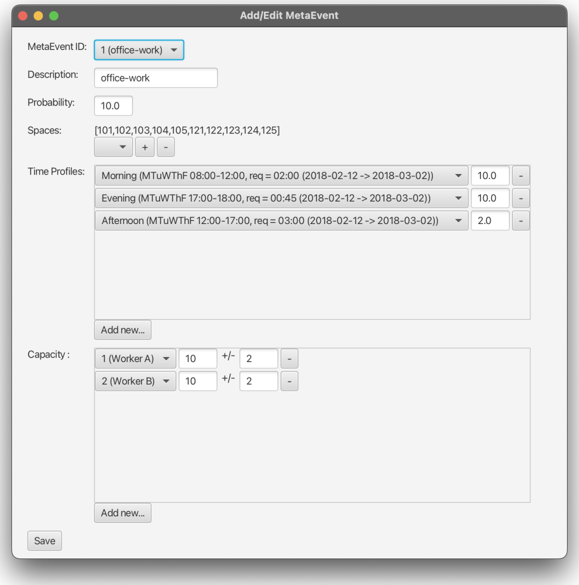
1. **MetaEvent ID, Description, and Probability**. Using the dropdown entry box for the metaevent ID, a particular metaevent to edit should be selected. The description and probability are then filled in according to the data provided to the declaration form.


Add/Edit MetaPerson

MetaPerson ID:	1 (Worker A)
Description:	Worker A
(Relative) Count:	10.0
Time Profiles:	Person Pattern A (MWF 08:00-18:00, req = 05:00 (2018-02-12 -> 2018-03-02)) 5.0 - Person Pattern B (TuTh 08:00-18:00, req = 05:00 (2018-02-12 -> 2018-03-02)) 2.0 - Add new...
Event Affinity:	1 (office-work) 10.0 - 2 (lab-work) 5.0 - 3 (meeting) 2.0 - 4 (eating) 2.0 - Add new...

Save

Figure 8: Input Form for MetaPeople.


Add/Edit MetaEvent

MetaEvent ID:	1 (office-work)
Description:	office-work
Probability:	10.0
Spaces:	[101,102,103,104,105,121,122,123,124,125] + -
Time Profiles:	Morning (MTuWThF 08:00-12:00, req = 02:00 (2018-02-12 -> 2018-03-02)) 10.0 - Evening (MTuWThF 17:00-18:00, req = 00:45 (2018-02-12 -> 2018-03-02)) 10.0 - Afternoon (MTuWThF 12:00-17:00, req = 03:00 (2018-02-12 -> 2018-03-02)) 2.0 - Add new...
Capacity :	1 (Worker A) 10 +/- 2 - 2 (Worker B) 10 +/- 2 - Add new...

Save

Figure 9: Input Form for MetaEvents.

2. **Spaces**. The dropdown menu should be used to select the different types of spaces in which the metaevent could be hosted. The "+" and "-" buttons are used to add / remove the selected space. The list of spaces in which the metaevent can occur are listed next to the data entry boxes.

3. **Time Profiles**. To add time profiles for the metaevent, click `Add new...`, which will create a new row in the time profiles entry box. Here, a profile should be selected, and its relative probability should be entered. A time profile can be removed by clicking the button with "-". Duplicate time profiles should not be entered.

4. **Capacity**. The attendance capacity in the "Capacity" entry box is used to indicate requirements for event attendance. Click on `Add new...` to add a new attendance capacity requirement. Then, use the dropdown menu to select a metaperson. The first text field indicates the average maximum capacity of the given type of metaperson, while the second text field indicates a corresponding standard deviation. Rows can be removed by clicking the "-" button.

5. **Saving**. By default, the metaevents JSON file is not saved after each individual metaevent is edited. Go to **MetaEvents > Save MetaEvents File** to write the metaevents file. The output pane should display "Saved metaevents file: `<metaevents-path>`" if successful.

The form for metapeople should be filled as follows, for each declared metaperson:

1. **MetaPerson ID, Description, and Counts**. Using the dropdown entry box for the metaperson ID, a particular metaperson to edit should be selected. The description and relative count (i.e., proportion of the population) are then filled in according to the data provided to the declaration form.

2. **Time Profiles**. The process to add time profiles to metapeople is similar to that for metaevents. In particular, to add time profiles for the metaperson, click `Add new...`, which will create a new row in the time profiles entry box. Here, a profile should be selected, and its relative probability should be entered. A time profile can be removed by clicking the button with "-". Duplicate time profiles should not be entered.

3. **Event Affinity**. Each metaperson should indicate their affinity towards different metaevent types. To add a new affinity, click `Add new...` for the capacity data entry field, which will create a new affinity data entry row. Use the dropdown to select the metaevent, and enter the corresponding affinity in the text box. In general, the affinity should be a non-negative floating point number (or integer, if desired), that relatively denotes the probability of attending different types of metaevents. Duplicate metaevents should not be listed.

4. **Saving**. By default, the metapeople JSON file is not saved after each individual metaperson is edited. Go to **MetaPeople** > **Save MetaPeople File** to write the metapeople file. The output pane should display “Saved metapeople file: `<metapeople-path>`” if successful.

After all appropriate metapeople and metaevents are defined and saved, the SmartSPEC scenario generation code can be run with the config file as generated by the SmartSPEC GUI.

4 Use and Distribution

5 References

[1] SmartSPEC, `https://github.com/andrewgchio/SmartSPEC` (2022).

[2] A. Chio, D. Jiang, P. Gupta, G. Bouloukakis, R. Yus, S. Mehrotra, N. Venkatasubramanian, SmartSPEC: Customizable smart space datasets via event-driven simulations, in: 2022 IEEE International Conference on Pervasive Computing and Communications (PerCom), IEEE, 2022.

[3] A. Chio, D. Jiang, P. Gupta, G. Bouloukakis, R. Yus, S. Mehrotra, N. Venkatasubramanian, SmartSPEC: A Framework to Generate Customizable, Semantics-based Smart Space Dataset, in: Pervasive and Mobile Computing, Elsevier, 2022.