System Theoretic Process Analysis (Spec-books) Tutorial

This tutorial is meant to assist you in using the ***spec-books*** tools provided in this development environment.

STPA-Sec and the accompanying intent specification is designed to analysis a system for security and safety concerns and provide requirements sufficient to mitigate those concerns.

spec-books is intented to be a jupyter notebook version of the STPA intent specification; designed to document the STPA process and the resulting requirements during in-person "STPA workshops".

Loading spec-books

spec-books is a python library located in the /CODE directory of this repository. It is an ever growing library of helper functions designed to aid in the creation, editing, and analysis of STPA data.

For now, ***spec-books*** only modifies tables (in the form of jupyter tables converted to excel spreadsheets) and a limited set of DRAWIO xml files customized to build STPA related drawings both automatically and manually.

Every ***spec-books*** jupyter notebook should start by loading the ***spec-books*** code library by running the below cell. The below command instantiates the library and all the classes and functions within it. We are importing ***spec-books*** as the name **sb** for this tutorial

[1]:
```
## Run this cell first to import the spec-books library
import os
import sys
code_path = os.path.join(os.path.dirname(os.path.dirname(os.getcwd())),'CODE')
if code_path not in sys.path:
    sys.path.append(code_path)
import spec_books as sb
```

[2]:
```
## Define the excel workbook that contains the spec-books data
WB = 'Tutorial.xlsx'
```

Tables

Because much of STPA revolves around tables, ***spec-books*** has a tables feature. These tables can be created, edited, and modifed using the notebook python interface. The data for the tables is stored in an excel spreadsheet.

Starting with a template

In the CODE directory there is a template excel file called "Template.xlsx". To get started, copy that file to the DATA directory and rename it. For this tutorial, we already did that and named it "Tutorial.xlsx"

Creating a table

spec-books rely on a Settings worksheet and a data worksheet. The Settings worksheet defines the data column headers, the width of the column and any special features about each column. Presently there are only two types of columns; text and enum (for enumeration or dropdown list). To display and edit a table in the notebook you simply invoke stpa_table command. The command takes two arguments:

1. The name of the table
2. The relative path and name of the excel spreadsheet

For example the following command creates a table and assigns it to the variable settings

[3]:
```
settings = sb.stpa_table('Settings',WB)
```

Add Row | Delete Row | Save Table

Filter...

Table Name	Col_Name	columnPosition	width	dataType	allowableValues
Settings	Table Name	1	75	text	
Settings	Col_Name	2	75	text	
Settings	columnPosition	3	75	text	
Settings	width	4	75	text	
Settings	dataType	5	75	enum	[text,enum]
Settings	allowableValues	6	75	text	
Components	ID	1	30	text	E
Components	Parent	2	30	text	
Components	Name	3	300	text	
Components	Order	4	50	text	
Components	Type	5	50	enum	[HC,AC,CP]

The settings table allows you to specify the contents of other tables within the design. Normally this table is not directly used in your notebook, rather it is shown here as an example of how you can edit the settings of the ***spec-books*** tables.

There are some tables already pre-defined and we highly encourage you not to remove these tables; specifically the tables that are used to generate the control diagrams and state diagrams.

In this example the class settings holds both the path to the excel spreadsheet that the data exists and a temporary pandas dataframe of the data as well. You can access the data directly by:

There are two ways to create new tables

1. Edit your excel spreadsheet directly
2. Use the built in "create_new_table" function

Edit your excel spreadsheet directly

Let's start with editing the spreadsheet directly. Let's say you have table "AI_Hazards" in that table you want

- An ID field
- A name field
- A dropdown field with the following options ["Really Bad", "Kinda Bad", "Not so Bad"]

First you need to edit the Settings tab.

At the end of the row add the following

AI_Hazards, ID , 1, text,
AI_Hazards, Name , 2, text,
AI_Hazards, Badness , 3, enum, ["Really Bad", "Kinda Bad", "Not so Bad"]

Let me describe. The first row is your ID field, 1 is the order in which the column will be presented, "text" is the type of field it is. and just means there are no settings for that field type. The same is true for the "Name" field. However, the "Badness" field is a dropdown list. So, you need to provide the different possibilities for the dropdown. This is done by putting the options in quotes, separated by commas, in a bracket.

Now that your settings are entered, create a new tab in the table called "AI_Hazards" the tab name and the Settings/Table Name must match Also, within the "AI_Hazards" Table the column headers must match the Col_Name field in the Settings table. I.e. for AI_Hazards you should have the column names ID, Name, Badness

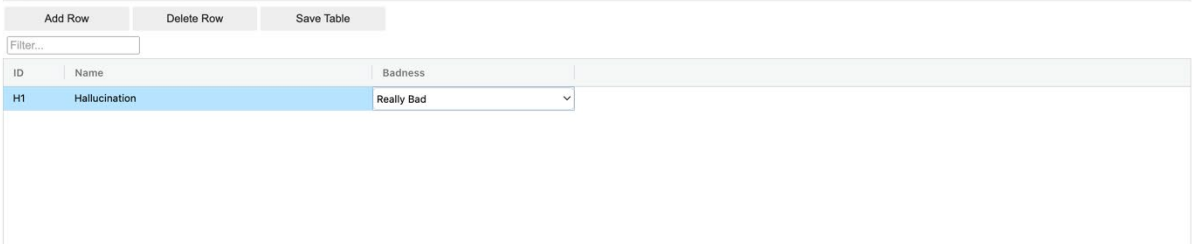
Use the built in create new table function

This is very similar but can be done programatically. However you have to specify a json type file for this to work correctly. Let me illustrate in code below. First you need to create a data json file. The format needs to be exactly like the format above. Here is the example data for the AI_Hazards Settings:

[4]:
```
data = {
    'Table Name': ['AI_Hazards', 'AI_Hazards', 'AI_Hazards'],
    'Col_Name': ['ID', 'Name', 'Badness'],
    'columnPosition': [1, 2, 3],
    'width': [75, 75, 75],
    'dataType': ['text', 'text', 'enum'],
    'allowableValues': [None, None, ["Really Bad", "Kinda Bad", "Not so Bad"]]
}
```

Now that the data has been created you can call the create_new_table function

[12]:
```
AI_Hazards_Table = sb.stpa_table('AI_Hazards',WB)
```



Creating a new Drawing

There are two types of drawings; "Control" and "State". All have the same table format. Each drawing contains two worksheets. A "state" or "control" worksheet and a "Transitions" worksheet

To create a new drawing from scratch simply call the create_new_drawing function like such:

[7]:
```
excel_file_name = WB
drawing_file_name = 'AI_Control_Diagram'
component_sheet_name = 'AI_Components'
transition_sheet_name = 'AI_Transitions'
drawing_type = 'control'
draw = sb.create_new_drawing(excel_file_name,drawing_file_name,component_sheet_name,transition_sheet_name,drawing_type)
```

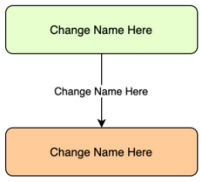
```
Table 'AI_Components' already exists.
Table 'AI_Transitions' already exists.
```

Once the new drawing tables are created, you can launch the drawing application within your notebook. The code to do that is shown below The variables above are used to call the right tables. You will notice the drawing is blank. To add a component click one of the buttons on the top row. HC is human controller, AC is automatic controller, CP is controlled process, and ST is state

Similarly, the bottom row will insert an arrow that is either Feedback (FB), control action (CA), Message (MSG) or State Transition (SM)

Once you have added the components you can hit SAVE (big green save) and the data will be saved to the excel spreadsheet. The little blue "Save & Exit" does not work. Also, any items you add from the toolbar will not be saved to the spreadsheet and may break spec-books.

Try adding two components and a transition arrow. Once you do that, save the table. You should be able to view the data associated with your drawing. See the example below. If you were successful, the drawing should look like this:



Conclusion

The CASTOR and Kharsansky Sat are examples of the use of the spec-books tool. This Tutorial and the Template.xlsx contain pre-defined tables and diagrams needed for STPA analysis.