6SENG001W Reasoning about Programs

Tutorial 1. Exercises using the B Tools Atelier B & ProB

B-Method Tools: Atelier B & ProB

The first lab session is intended to introduce you to the B tools **Atelier B** & **ProB** and to B's *Abstract Machine Notation (AMN)*.

You will be required to type in the *PaperRound* abstract machine into **Atelier B**.

Next you should **syntax & type check** it using **Atelier B** & finally **animate** it using **ProB**.

You should refer to the tool's manuals.

Accessing B Tools

Atelier B & **ProB** are installed on the University's Windows PCs, they are accessible from the **AppsAnywhere** application.

Exercise 1.1

Setting up & using the **Atelier B** tool & creating a *"workspace"*.

1. Start the **Atelier B** tool from the **AppsAnywhere** application.
2. Start by creating a *"workspace"* for all your B specification projects.
    - Using Window's Explorer create a directory/folder in your **H:** *Home Drive* directory to be used as your B specification *"workspace"*.
    - Next, create the "workspace" from the **Atelier B** *"Atelier B > New > Workspace"* menu.
    - Use the *"Browse"* option to select the directory/folder you just created on your **H:** drive.
3. Next set the *"Default Project Directory"* (place where B specification projects are created) to be the *"workspace"* you have just created.
    - Set the "Default Project Directory" from the **Atelier B** *"Atelier B > Preferences"* menu.

- Select the *"Project"* tab, then use the *"Browse"* option to select the same *"workspace"* directory/folder you just created on your **H:** drive.
- Finally, select *"Software Development"*

Exercise 1.2

Create a *"Project"* for your new B specification, i.e. *PaperRound*.

1. Do this from the **Atelier B** *"Atelier B > New > Project"* menu.

   Type in a *"Project Name"*

   Select "Project Type" as *"Software Development"*.

2. Then add components to your B specification, i.e. a *B MACHINE* that makes up the system specification.

   Do this from the **Atelier B** *"Atelier B > New > Component"* menu.

   Type in a new component name, in this case it should be just *PaperRound*, you do not need to type in the *".mch"*.

3. If everything has worked correctly you should see an *orange box with "PaperRound"* in it.
4. To begin typing the *PaperRound* specification in just *"double-click"* the *"orange PaperRound"* box.

Exercise 1.3

Using the **Atelier B** built in editor, to type in the *PaperRound* specification.

You should create it in a file called

`PaperRound.mch`.

You will need to convert the B symbols into their ASCII equivalent, see the first lecture & the online symbols list.

Exercise 1.4

Syntax & Type Checking the specification , using the **Atelier B** tool.

You can either syntax & type check the *PaperRound* specification as you type it in or after you have finished typing it in.

The **Atelier B** tool will type check it automatically immediately after you have saved any changes.

Error messages will be displayed in the "Outline" sub-window & underlined in red in the specification.

Alternatively, you can "force" type checking by either:

- pressing the *blue circular "Tc" button* at the top of the tool's main screen.
- pressing Control-T, i.e. hold down the "Ctrl" (Control) key & at the same time press the "T" key.

Exercise 1.5

Once the *PaperRound* machine has been syntax & type checked & there are **no errors**, you can *animate* it using the **ProB** animator.

***** SEE THE ONLINE - *Tutorial First Step* *****

1. To do this start the **ProB** animator tool from the **AppsAnywhere** application.
2. Then *open* the `PaperRound.mch` file you created in **Atelier B**.

   You can open the `PaperRound.mch` file from the *"File > Open"* menu, by using the *"Browse"* option.

3. If there are **no errors** then you should see:
   - The specification in the top window.
   - In the bottom *"Enabled operations"* window, you should see - **INITIALISATION({})** .
4. To begin the animation *"double-click"* on **INITIALISATION({})** .
5. You can now animate the PaperRound machine by *"double-clicking"* any of the operations that are listed there.

6. Note that **only operations that are enabled for the current state** are listed in this window.
7. Try all of the actions out & see what happens.

Exercise 1.6

Add an enquiry operation

```
ans <-- getsPapers( houseNumber )
```

that checks if the *housenumber* has papers delivered to it.

If it does then it outputs the number 1.

Hint: this operation can be defined by combining:

- the PRE-THEN-END of the *addNewHouse(newHouse)* operation &
- using a results variable like *numbHouses* in the *howManyHouses* operation.

Exercise 1.7

Add an operation

```
cancelPapers( houseNumber )
```

this cancels paper deliveries by deleting house's number from the set of house numbers that are delivered to.

Hints:
(a) Think about what must be true about the house number for it to be deleted. **(Pre-condition)**

(b) What effect should this operation have, i.e. what effect does it have. **(Post-condition)**

(c) This is similar to the *addNewHouse(newHouse)* operation.

Last updated: 16/9/2020
