6SENG001W Reasoning about Programs

Tutorial 6: Functions in B

Introduction

These tutorial exercises refer to the notes for **Lecture 6: Functions**.

In this tutorial you are required to use the two B tools Atelier B & ProB to animate & extend a B machine that represents a *family*.

This family specification illustrates how functions (Lecture 6's topic) can be used in a specification.

The *Family* specification is given in this B machine `Family.mch`.

The specification includes information about a family:

- the *family members*,
- & for each family member their *age* & *sex*.

The specification also has the following operations:

- For someone to have a birthday -- `HadBirthday`.
- For some female to have a baby -- `HadABaby`.
- For someone to die, that is no longer be a member of the family - `PersonDies`.
- An enquiry that tests if there is someone in the family with a particular sex,
  i.e. `Male`, `Female` or `Trans` - `DoesAFamilyMemberHaveThisSex`.

Exercise 6.1

1. Read the **Lecture 6: Functions** notes.
2. Read & familiarise yourself with the **AMN versions of the Functions symbols**.

   See ProB's "Help" menu & the lecture notes.

Exercise 6.2

Create a new B "Project" using Atelier B, then create a new "Component" & then paste the *Family* into it.

Type check it using Atelier B.

Exercise 6.3

Load the Family machine specification into ProB.

Animate it & execute the four operations:

```
HadBirthday

HadABaby

PersonDies

DoesAFamilyMemberHaveThisSex
```

Test both the *successful* & *error* cases of the operations.

After executing each operation **check** how the state has been modified.

Do this by use ProB's "`Eval`" terminal to check the values of the three state variable that make up the state.

Also use the "`Eval`" terminal to test the truth value of the:

- operations's *preconditions* &
- IF statements's *conditions*.

Exercise 6.4

Add a mapping to the Family's state that maps an individual to his or her *unique* Passport number.

Use the `PASSPORT_NUMBER` constant to represent the type of these numbers.

The mapping's *properties* must be defined & it should be initialised as appropriate.

Exercise 6.5

Add a relation *parentOf* to the *Family* state that relates each family member to their children.

So for example, if *Grandad* is *Paul*'s dad then the maplet

```
Grandad |-> Paul
```

would be a member of the *parentOf* relation, i.e.

```
Grandad |-> Paul : parentOf
```

This relation should be initialised as appropriate.

This will be updated when the `HadABaby` operation is successful, then it needs to be updated with the new mother/baby relationship.

```
parentOf := parentOf \/  { mother |-> baby }
```

And when someone in the family dies they need to be removed from the relation, both as a parent (in domain) & as a child (in the range).

```
parentOf :=  ( { person } <<| parentOf ) |>> { person }
```

Exercise 6.6

Add a function *myMum* to the *Family* state that maps a person to their mother, if still alive.

What kind of function should this be?

Consider each of the possible options & think about what the implications are of each one in relation to the family. (Make a note of your conclusions.)

So for example, if *Mary* is *Joe*'s mum then the maplet

```
Joe |-> Mary
```

would be a member of the *myMum* function.

This function should be initialised as appropriate.

This will be updated when the `HadABaby` operation is successful, then it needs to be updated with the new mother/baby relationship.

And when someone in the family dies they need to be removed from the function, both as a child (in domain) & as a parent (in the range).

Exercise 6.7

Add an operation for a family member getting their *first* passport - `FirstPassport`.

This operation inputs a person & a *unique* passport number for the person's new passport.

If the person is not a family member or the passport number has already been assigned to a family member then it should report an error. Add appropriate reporting messages to the `REPORT` set.

Test all of the cases of the `FirstPassport` operation.

Exercise 6.8

Add a similar operation but this one is when a family member is *renewing* their passport - `RenewPassport`.

This operation inputs a person & a new *unique* passport number, provided the person is a family member, already has a passport then their passport number is changed to the new number.

If the person is not a family member or the passport number is already in use by a family member or the person doesn't have a passport then it should report an error.

Test all of the cases of the `RenewPassport` operation.

Exercise 6.9

Add an enquiry `HasChildren` to test if a family member has any children.

This enquiry inputs a person's name & tests if they have any children, if they do it outputs *yes*, if not then it outputs *no* & finally if they are not a family member then it replies *Do not know*.

Test all of the cases of the `HasChildren` operation.

Last updated: 5/11/17
