Formal Specification F28FS2, Lecture 12
Implementing schema in ML

Jamie Gabbay

March 10, 2014

The phone directory spec

Taken from Michael Butler's introductory notes on Z.

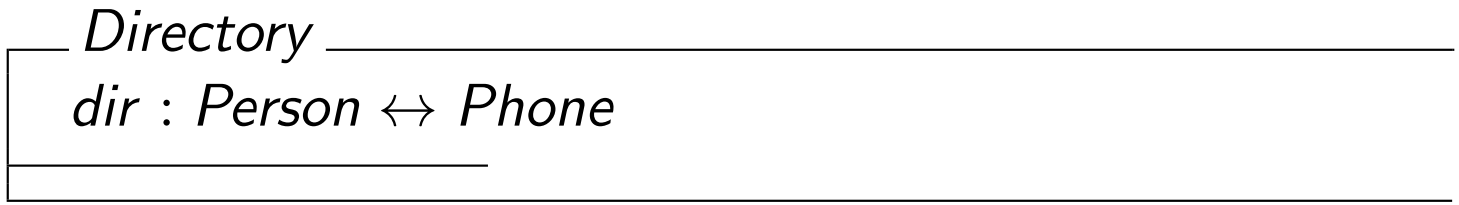
Assume types $[Person, Phone]$.

I don't want to number people, so let's model *Person* by the ML type `string`, and *Phone* by `int`. We could write the following if we want:

```
type Person = string;
type Person = string
type Phone = int;
type Phone = int
```

The directory

The state of the directory:




```
fun Directory (dir:(Person*Phone) list) = true;
val Directory = fn :  (Person * Phone) list -> bool
```

Initialising the directory

Initially the directory is empty:

InitDirectory

$$dir' : Person \leftrightarrow Phone$$

$$dir' = \{\}$$

```
fun initDirectory () = ([]:(Person*Phone) list);
val initDirectory = fn :  unit -> (Person * Phone)
list
```

Adding to the directory

We add a name and number pair to the directory:

AddEntry

$$dir, dir' : Person \leftrightarrow Phone$$
$$name? : Person$$
$$number? : Phone$$

$$dir' = dir \cup \{name? \mapsto number?\}$$

```
fun AddEntry dir (name:Person) (number:Phone) =
(name,number)::dir;
val AddEntry = fn :  (Person * Phone) list -> Person
-> Phone -> (Person * Phone) list
```

Directory number lookup

An operation to get all the numbers associated with a name:

GetNumbers

$\Xi Directory$

$name? : Person$

$numbers! : \mathbb{P} Phone$

$numbers! = \{n : Phone \mid name? \mapsto n \in dir\}$

```
fun GetNumbers ((p,n)::tl) (name:Person) = if
(name=p) then n::(GetNumbers tl name) else
(GetNumbers tl name)
  | GetNumbers [] name = [];
val GetNumbers = fn :  (Person * 'a) list -> Person
-> 'a list
```

I did not bother to return `dir'`, and I did not bother to restrict the polymorphism on `'a`. Does that make me a bad person?

Directory name lookup

An operation to get all the names associated with a number:

GetNames

$\Xi Directory$

$number? : Phone$

$names! : \mathbb{P} Person$

$names! = \{p : Person \mid p \mapsto number? \in dir\}$

```
fun GetNames ((p,n)::tl) (number:Phone) = if
(number=n) then p::(GetNames tl number) else
(GetNames tl number)
  | GetNames [] number = [];
val GetNames = fn :  ('a * Phone) list -> Phone -> 'a
list
```

Remove Entry

An operation to remove an entry from the directory:

RemoveEntry

$\Delta Directory$

$name? : Person$

$number? : Phone$

$dir' = dir \setminus \{name? \mapsto number?\}$

```
fun RemoveEntry (hd::tl) (name:Person) (number:Phone)
= if (hd=(name,number)) then (RemoveEntry tl name
number) else hd::(RemoveEntry tl name number)
  | RemoveEntry [] name number = [];
val RemoveEntry = fn :  (Person * Phone) list ->
Person -> Phone -> (Person * Phone) list
```

Exercises

1. Write a schema to check whether *name*? appears in the phone directory, and return 'true' if it is, and 'false' if it is not (you might like to declare an enumerated type $Bool ::= true \mid false$).
2. Implement it.
3. Write a schema to return the set of people with more than one number.
4. Implement it.

Exercises

Implement the Birthday Book schema from (see Butler's notes; links on the course webpage, search for 'Butler').