Homework IV Sample Solution

We endeavor to organize this change to the telephone database so that changes to Diller's scheme definitions are minimal. Schemes whose definitions are unchanged from Diller are not included here.

We start with the state space which must be enhanced to keep track of the two types of phones, and this is done so as to both ensure no overlap of the types of phones and to retain the same variable as the collection of all phones. The added components of the state space lead to additional appropriate invariants.

PhoneDB

$$
\begin{array}{|l}
\text{members}: \mathbb{P}\ \text{Person} \\
\text{cell, land, telephones}: \text{Person} \pfun \text{Phone} \\
\hline
\text{dom telephones} \subseteq \text{members} \\
\text{telephones} = \text{cell} \cup \text{land} \\
\text{cell} \cap \text{land} = \varnothing
\end{array}
$$

Of course, this change requires that a corresponding change be made in the initial state scheme, namely:

InitPhoneDB

$$
\begin{array}{|l}
\Delta \text{PhoneDB} \\
\hline
\text{members} = \varnothing \\
\text{cell} = \varnothing \\
\text{land} = \varnothing
\end{array}
$$

The new operations for adding a cell or land line are very similar to the AddEntry scheme of Diller, but with post-conditions for the relevant variables.

AddCell

$$
\begin{array}{|l}
\Delta \text{PhoneDB} \\
\text{name?}: \text{Person} \\
\text{newnumber?}: \text{Phone} \\
\hline
\text{name?} \in \text{members} \\
\text{name?} \mapsto \text{newnumber?} \in \text{telephones} \\
\text{cell}' = \text{cell} \cup \{\text{name?} \mapsto \text{newnumber?}\} \\
\text{land}' = \text{land} \\
\text{members}' = \text{members}
\end{array}
$$

Then, as in Diller, a scheme expression is provided for the complete operation.

$$DoAddCell \triangleq AddCell \gg Success \mid NotMember \mid EntryAlreadyExists$$

AddLand

$$\Delta PhoneDB$$
$$name?: Person$$
$$newnumber?: Phone$$

$$name? \notin members$$
$$name? \mapsto newnumber? \notin telephones$$
$$land' = land \oplus \{name? \mapsto newnumber?\}$$
$$cell' = cell$$
$$members' = members$$

And again, the scheme expression.

$$DoAddLand \triangleq AddLand \gg Success \mid NotMember \mid EntryAlreadyExists$$

The FindPhones operation must be revised to return two sets of phones but this requires only a simple change.

FindPhones

$$\Delta PhoneDB$$
$$name?: Person$$
$$cellnumbers!, landnumbers!: \mathbb{P}\ Phone$$

$$name? \in dom\ telephones$$
$$cellnumbers! = cell(\!|\{name?\}|\!)$$
$$landnumbers! = land(\!|\{name?\}|\!)$$

It would be possible to describe the RemoveEntry with greater specificity for this new state space, but since the invariant of the new state space guarantees that an entry will appear for only one of the two types of phones, the RemoveEntry scheme requires no change -- from the telephones relation changes as stated in Diller, it can be deduced that either the cell or land relation must change according to the kind of phone.

Lastly, the one remaining thing is to specify the CountPhones operation. Since the CountPhones operation has no pre-conditions, there are no exceptional cases to consider and we write the DoCountPhones scheme directly.

DoCountPhones

ΞPhoneDB
cellCount!, landCount!: $\mathbb{Z}$
rpt!: Report

---

cellCount! = #(ran cell)
landCount! = #(ran land)
rpt! = "Okay"

This completes the revision of Diller's specification as required. Justification is provided by the behavior of the Miranda animation.

The Miranda prototype for problem 2 is in the class directory.