LECTURE 17: LIBRARY CASE STUDY

Software Engineering

Mike Wooldridge

1 A Library Management System

- In this lecture, we specify a simple library system.
- Operations:
  - check out a book;
  - return a book;
  - add a book to library;
  - remove book from library;
  - get list of books by author or subject area;
  - get list of books checked out by particular borrower;

- All books must either be checked out or available for check out.
- No book may be simultaneously checked out and available.
- There is an upper limit to number of books that may be checked out.

1.1 Types

- We need sets for:
  - all possible books;
  - all possible copies of books;
  - all possible people;
  - all possible authors;
  - all possible subjects;
  - the various reports that may be produced.
- So parachute in:

$$[BOOK, COPY, PERSON, AUTHOR, SUBJECT, REPORT]$$

1.2 State Space

- The state space is describes in several steps. First, a schema containing information relating to books in the library.

ParaLibrary

$$
\begin{array}{|l}
\hline
instance\_of : COPY \pfun BOOK \\
written\_by : BOOK \pfun \mathbb{P}\, AUTHOR \\
about : BOOK \pfun \mathbb{P}\, SUBJECT \\
\hline
\mathrm{dom}\ written\_by \subseteq \mathrm{ran}\ instance\_of \\
\mathrm{dom}\ about \subseteq \mathrm{ran}\ instance\_of \\
\hline
\end{array}
$$

- *instance_of* tells us what book a copy is an instance of;
- the set

  $$\mathrm{ran}\ \mathit{instance\_of}$$

  is the set of all books in the library;
- *written_by* tells us who a book is written by; there may be more than one author, hence the powerset operation; there may be no authors;
- *about* tells us the subjects a book is about; there may be no subjects;
- first invariant tells us that we only know who wrote books in the library;
- second invariant tells us that we only know subjects of books in the library.

- The database part of the schema is as follows:

LibraryDB

$$
\begin{array}{l}
borrower, staff : \mathbb{P}\, PERSON \\
available, out : \mathbb{P}\, COPY \\
borrowed\_by : COPY \pfun PERSON \\
\hline
borrower \cap staff = \emptyset \\
available \cap out = \emptyset \\
\mathrm{dom}\ borrowed\_by = out \\
\mathrm{ran}\ borrowed\_by \subseteq borrower \\
\forall p : borrower \bullet \#borrowed\_by^{\sim}(\!|\{p\}|\!) \\
\quad \leq MaxCopies
\end{array}
$$

- *borrower* is the set of all borrowers known to the system;
- *staff* is the set of all staff known to the system;
- *available* is the set of all available books;
- *out* is the set of borrowed books (i.e., ones that have been checked out);
- *borrowed_by* tells us who borrowed the books out on loan.

- 1st invariant tells us that a person cannot be both a borrower and a staff;
- 2nd invariant tells us that books cannot be both available and checked out;
- 3rd invariant tells us that the only books appear have been borrowed by someone are those that are out;
- 4th invariant tells us that books can only be borrowed by borrowers;
- 5th invariant tells us that a borrower can only have out up to the maximum number of books.

- The library state space is then as follows:

Library

ParaLibrary
LibraryDB

$$\mathrm{dom}\, instance\_of = available \cup out$$

- the only invariant in this schema tells us that the library does not know anything about books which are not in stock.

1.3 The Operations

- We assume initialisation operations; these are trivial.
- First we look at checking out books...
- Inputs: person name ($n?$) and copy ($c?$).

CheckOut

$$
\begin{array}{l}
\Delta Library \\
n? : PERSON \\
c? : COPY \\
\hline
n? \in borrower \\
c? \in available \\
\#borrowed\_by^{\sim}(\!|\{n?\}|\!) \\
\quad < MaxCopies \\
available' = available \setminus \{c?\} \\
out' = out \cup \{c?\} \\
borrowed\_by' = borrowed\_by \cup \\
\quad \{c? \mapsto n?\}
\end{array}
$$

- (Note that $f^{\sim}$ is the inverse of f.)
- 1st precondition is that the person trying to borrow must be a known borrower;
- 2nd precondition is that the book must be available;
- 3rd precondition is that the person trying to borrow must have out fewer than the maximum number of books available;
- the postconditions define the changes made to *available*, *out* and *borrowed_by*.

1.4 Returning a Book

- One input: the copy to be returned.

Return

$\Delta Library$
$c? : COPY$

$c? \in out$

$available' = available \cup \{c?\}$
$out' = out \setminus \{c?\}$
$borrowed\_by = \{c?\} \lhd borrowed\_by$

- precondition states that the book can only be returned if it is out;
- 1st post-condition says that the book is available after the operation;
- 2nd post-condition says that the book is no longer out;
- 3rd post-condition uses domain subtraction to remove the correct record from the *borrowed_by* function.
- For example,

$$borrowed\_by = \{b01 \mapsto mjw, b02 \mapsto en, b03 \mapsto mjw\}$$

$$\{b01\} \lhd borrowed\_by = \{b02 \mapsto en, b03 \mapsto mjw\}$$

1.5 Adding Books to the Library

- There are two cases to consider:
  - where the book is completely new to the library;
  - where the book is another copy of a book that is already in the library.
- We have two schemas to capture these two situations:
  - *AddNewBook;*
  - *AddAnotherCopy*.

AddNewBook

$\Delta Library$
$c? : COPY$
$b? : BOOK$
$a? : \mathbb{P}\, AUTHOR$
$s? : \mathbb{P}\, SUBJECT$

$b? \notin \mathrm{ran}\ instance\_of$
$c? \notin available \cup out$

$available' = available \cup \{c?\}$
$instance\_of' = instance\_of \cup \{c? \mapsto b?\}$
$written\_by' = written\_by \cup \{b? \mapsto a?\}$
$about' = about \cup \{b? \mapsto s?\}$

AddAnotherCopy

$\Delta Library$
$c? : COPY$
$b? : BOOK$

$c? \notin available \cup out$
$b? \in \mathrm{ran}\ instance\_of$

$available' = available \cup \{c?\}$
$instance\_of' = instance\_of \cup \{c? \mapsto b?\}$

1.6 Removing Books

- Removing a books from the library is similarly complicated; once again there are 2 possibilities to consider...
  - removing a book that is the only copy;
  - removing one copy of a book leaving several other copies behind.
- Two schemas:
  - *RemoveOther* to remove one of several copies;
  - *RemoveLast* to remove the last copy.

RemoveOther

$\Delta Library$

$c? : COPY$

$c? \in available$

$\#(instance\_of^{\sim}(\{\{instance\_of(c?)\}\})) > 1$

$available' = available \setminus \{c?\}$

- Note that there is no need to alter any variables in *ParaLibrary*; we only change *available*, to indicate that the book is no longer available.

RemoveLast

$\Delta Library$

$c? : COPY$

$c? \in available$

$\#(instance\_of^{\sim}(\{\!|\{instance\_of(c?)\}|\!\})) = 1$

$available' = available \setminus \{c?\}$

$instance\_of' = \{c?\} \lhd instance\_of$

$written\_by' = \{instance\_of(c?)\} \lhd$
$\quad instance\_of$

$about' = \{instance\_of(c?)\} \lhd about$

1.7 Interrogating the Database

- Two options:
  - search by author;
  - search by subject;
  - find out what copies someone has borrowed.

- *ByAuthor* takes an author name and produces the set of all books that the author appeared in the 'author' list of.

ByAuthor

$$\Xi Library$$
$$a? : AUTHOR$$
$$out! : \mathbb{P}\, BOOK$$
$$out! = \{b : BOOK \mid a? \in written\_by(x)\}$$

- *BySubject* takes a set of subjects and produces a list of all the books which have these subjects in their 'about' list.

BySubject

$$\Xi Library$$
$$s? : \mathbb{P}\, SUBJECT$$
$$out! : \mathbb{P}\, BOOK$$
$$out! = \{b : BOOK \mid s? \subseteq about(b)\}$$

- Finally, finding out who has borrowed what...

BooksBorrowedBy

$$\Xi Library$$
$$n? : PERSON$$
$$out! : \mathbb{P}\, COPY$$

$$n? \in borrower$$
$$out! = borrowed\_by^{\sim}(\!|\{n?\}|\!)$$