Abstract Model Specifications

- Build an abstract model of required software behavior using mathematically defined (perhaps using axioms) types (e.g., sets, relations).

- Define operations by showing effects of that operation on the model.

- Specification includes:
  - Model
  - Invariant properties of model
  - For each operation:
    - name
    - parameters
    - return values
  - Pre and post conditions on the operations
Z (pronounced Zed)

• Z specifications are made up of "schemas"

• A schema is a named, relatively short piece of specification with two parts:
  
  - Above the line: the definition of the data entities
  - Below the line: the definition of invariants that hold on those data entities
Z : Defining the Abstract Model

- **Library**
  - books: \( \mathbf{P} \) BOOK
  - status: BOOK \( \rightarrow \) STATUS
  - \( \text{books} = \text{dom status} \)

- **Declaration** says library has two visible parts of its state:
  - books is a set of BOOKS, which are atomic elements.
  - status is a partial function that maps a BOOK into a STATUS
    (which is another atomic element that can take values In or Out)

- **The invariant** says the set of books is precisely the same as the domain of the function status.
  - Says every book in the Library has exactly one status
  - Two books may have the same status.

**Example of a legal state for Library is:**
- books = \{Principia Mathematica, Safeware\}
- status = (Principia Mathematica \( \rightarrow \) In,
  Safeware \( \rightarrow \) Out)
Z: Defining Operations

Borrow

\[ \mathbf{\triangle Library} \]

\[ \text{book?: BOOK} \]

\[ \text{status} (\text{book?}) = \text{In} \]

\[ \text{status}' = \text{status} \oplus (\text{book?} \mapsto \text{Out}) \]

- \[ \mathbf{\triangle Library} \] declaration says operation modifies state of Library
- \text{book?} is the input
- A prime indicates the value after the operation
- The first invariant defines a pre-condition on the operation, i.e., the book to be borrowed must be currently checked in.
- The second invariant defines the semantics of borrowing, i.e., it overwrites the entry in the status function for the borrowed book.
Example: After a borrow operation, the set of books in the library remains unchanged.

books' = books

books' = \textit{dom} \textit{status}'

= \textit{dom} (\textit{status} \oplus \{\textit{book}? \leftrightarrow \text{Out}\}) \quad \text{[from invariant of Library]}

= \textit{dom} (\textit{status} \cup \{\textit{book}? \leftrightarrow \text{Out}\}) \quad \text{[from post-condition of Borrow]}

= \textit{dom} \textit{status} \cup \textit{dom} (\{\textit{book}? \leftrightarrow \text{Out}\}) \quad \text{Follow from mathematics}

= \textit{book} \cup \textit{book}?

= \textit{book} \quad \text{[true because first invariant of Borrow implies that book? is an element of books]}
Z : Defining Observing Operations

<table>
<thead>
<tr>
<th>Find Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>triples Library</td>
</tr>
<tr>
<td>book? : BOOK</td>
</tr>
<tr>
<td>status! : STATUS</td>
</tr>
<tr>
<td>book? ∈ books</td>
</tr>
<tr>
<td>status! = status (book?)</td>
</tr>
</tbody>
</table>

- triple indicates state is not changed by operation (a lookup op)
- status! is an output variable
- The first invariant says that schema only defined if the book is known. Therefore, function application in second invariant cannot fail.
Z : Another Observing Operation

\[
\begin{align*}
\text{CheckedOut} & \subseteq \text{Library} \\
\text{out!} & = P \text{ BOOK} \\
\text{out!} & = \{b: \text{books} | \text{status}(b) = \text{Out}\}
\end{align*}
\]

- Produces all the books that are currently checked out of library
- Returns a (possibly empty) set of books.
Z : Initialization

- InitLibrary
  - Library
  - books = ∅

- Still need schemas to add and remove books from the library.