Z – Example No. 1

Hans–Jürgen Ficker, Sönke Jarré, Klaus Lüttich, Raymond Scholz

Bremen University, January 16, 1998

1 A class manager's assistant

This example is taken from the book "Software Development with Z" by J.B. Wordsworth (Addison-Wesley 1991). Schemas and other definitions have not being altered.

The class manager's assistant represents a small program, which provides support for a class manager to admit students to a class, record their exercises and discharge them. This specification will exhibit many features of a common Z specification. These are an informal statement of the requirements, the definition of some fundamental sets and a data model presented as a schema. Then the system's behaviour the user might see is presented and related to the operations that the user requires in the interface. Firstly the normal aspect of the operation is presented, secondly the possible errors. Finally the previously described operations are combined into a simple interface.

The informal statement of requirements is as follows: The class manager's assistant is required to keep track of students enrolled on a class, and to record which of them has done the exercises. When a student applies for a class, he will be enrolled on it, unless the class is full. Such a student will be presumed not to have done the exercises. When a student completes the exercises, the fact is to be recorded. Students may leave the class even if they have not done the exercises, but only students who have done the exercises are entitled to a completion certificate.

The following four operations can be derived from these requirements: **Enrol** to enrol a student on a class, issuing a warning if the class is full, or the student is already enrolled. **Test** to record a student as having done the exercises, or warning if the student is not enrolled or has already done the exercises. **Leave** to remove a student from the class with an indication whether he is entitled to a certificate. Only students who have done the exercises are entitled to a certificate. If the student is not enrolled, a warning is given.

1.1 The given sets for the specification

Students are represented by a given set *Student*. The maximum class size is specified by *size* but is not fixed to a certain value. Next we need a type definition for the responses to the operations.

$[Student]$

$| \; size : \mathbb{N}$

$$
\begin{aligned}
Response ::=\ & success \\
& \mid notenrolled \\
& \mid nocert \\
& \mid cert \\
& \mid alreadyenrolled \\
& \mid alreadytested \\
& \mid noroom
\end{aligned}
$$

1.2 The abstract data state

The notion of a class consists of the notion that some students, possibly none are enrolled on it. Some of the enrolled students, possibly none of them have done the exercises, so that they are tested. Not more than *size* students can be enrolled. Students who have done the exercises must have been enrolled.

$$
\begin{array}{|l}
\multicolumn{1}{l}{\textit{Class}} \\
\hline
enrolled, tested : \mathbb{P}\, Student \\
\hline
\# enrolled \leq size \\
tested \subseteq enrolled \\
\hline
\end{array}
$$

1.3 The abstract initial state

In the initial state of the class manager's assistant, no students will be enrolled. Since those who have done the exercises must have been enrolled, it follows that there can be no students who have done the exercises. This is expressed by a theorem.

$$ClassInit \mathrel{\widehat=} [\, Class' \mid enrolled' = \emptyset \,]$$

$$ClassInit \vdash tested' = \emptyset$$

1.4 Enrolling a student

With the enrolment of a student, the class changes. The student *s?* is supplied as input, rising an output response *r!*. To be able to add a student, there must be enough room. The student is added to the set of enrolled students. A newly enrolled student is supposed to not having done the exercises, so the set *tested* is unchanged. The output response is *success*.

$$
\begin{array}{|l}
\multicolumn{1}{l}{\textit{Enrolok}} \\
\hline
\Delta Class \\
s? : Student \\
r! : Response \\
\hline
s? \notin enrolled \\
\# enrolled < size \\
enrolled' = enrolled \cup \{s?\} \\
tested' = tested \\
r! = success \\
\hline
\end{array}
$$

1.5 Testing a student

Testing a student causes a change of the class. The student has been enrolled but is not recorded as having done the exercises. The student is added to the set of tested students.

Testok

$$\Delta Class$$
$$s? : Student$$
$$r! : Response$$

$$s? \in enrolled$$
$$s? \notin tested$$
$$tested' = tested \cup \{s?\}$$
$$enrolled' = enrolled$$
$$r! = success$$

1.6 Discharging a student

To record that a student is leaving the class, the class changes. The student has to be enrolled. The response depends on whether the student has done the exercises. If he has done so, he will be discharged with a certificate, if not, the response will indicate no certificate.

Leaveok

$$\Delta Class$$
$$s? : Student$$
$$r! : Response$$

$$s? \in enrolled$$
$$enrolled' = enrolled \setminus \{s?\}$$
$$((s? \in tested \land tested' = tested \setminus \{s?\} \land r! = cert)$$
$$\lor (s? \notin tested \land tested' = tested \land r! = nocert))$$

1.7 Enquiries

This is an enquiry operation, which does not change the state of the class. The operation should be self–explanatory.

Enquire

$$\Xi Class$$
$$s? : Student$$
$$r! : Response$$

$$((s? \notin enrolled \land r! = notenrolled)$$
$$\lor (s? \in (enrolled \setminus tested) \land r! = alreadyenrolled)$$
$$\lor (s? \in tested \land r! = alreadytested))$$

1.8 Errors

To indicate an error during the usage of the class manager's assistant, the following schemas are defined. None of them changes the state of the class. The usage of these schemas is explained above.

$$
\begin{array}{|l}
\hline
\textit{AlreadyEnrolled} \\
\Xi Class \\
s? : Student \\
r! : Response \\
\hline
s? \in enrolled \\
r! = alreadyenrolled \\
\hline
\end{array}
$$

$$
\begin{array}{|l}
\hline
\textit{NoRoom} \\
\Xi Class \\
r! : Response \\
\hline
\# enrolled = size \\
r! = noroom \\
\hline
\end{array}
$$

$$
\begin{array}{|l}
\hline
\textit{AlreadyTested} \\
\Xi Class \\
s? : Student \\
r! : Response \\
\hline
s? \in tested \\
r! = alreadytested \\
\hline
\end{array}
$$

$$
\begin{array}{|l}
\hline
\textit{NotEnrolled} \\
\Xi Class \\
s? : Student \\
r! : Response \\
\hline
s? \notin enrolled \\
r! = notenrolled \\
\hline
\end{array}
$$

1.9 The user interface

Users of the system have the following operations at their disposal. If the user attempts to enrol a student who is already enrolled or to enrol a student though there is no room for him, adequate responses will be created.

$$Enrol \mathrel{\widehat=} Enrolok \lor NoRoom \lor AlreadyEnrolled$$

$$Test \mathrel{\widehat=} Testok \lor NotEnrolled \lor AlreadyTested$$

$$Leave \mathrel{\widehat=} Leaveok \lor NotEnrolled$$