This example Z specification dates from the occasion when a colleague decided he wanted to try a heart rate monitor (HRM), and organized a group of us to share the cost. The specification addresses the problem of ensuring fair usage of the HRM.

---

section *share* parents *toolkit*

$[Person]$

There is a set of persons, not all of whom are shareholders. (We should really specify a minimum size for this set.) Conversely, all shareholders are persons. (Pets are excluded, or if you prefer are treated as persons.)

$$Time == \mathbb{N}$$

Times are represented by ordinal numbers.

Share

$$key : \mathbb{N}$$
$$last\_used : Time$$

Each share has a key to distinguish it from other shares and records the time at which its bearer last used the HRM.

$$
\begin{array}{|l}
\textit{State} \\
\hline
bearer : Share \pfun Person \\
location : Person
\end{array}
$$

The state comprises a function from the born shares to persons (every share is born by one person), and a location, that is the person currently in possession of the HRM. This does not exclude one person from bearing more than one share, though we might want to reconsider this. It assumes that several people would not want to share a single share.

$$
\begin{array}{|l}
instigator : Person
\end{array}
$$

$$
\begin{array}{|l}
\textit{InitState} \\
\hline
\Delta State \\
\hline
bearer' = \{(\!| \, key == 0, last\_used == 0 \, |\!), instigator)\} \\
location' = instigator
\end{array}
$$

In the initial state, the instigator bears the sole share and has possession of the HRM.

IssueShare

$$\Delta State$$
$$new\_punter? : Person$$

$$bearer' = bearer \oplus$$
$$\quad \{s' : Share \mid s'.key = \#(dom\ bearer) \land$$
$$\quad\quad s'.last\_used = max\{s : dom\ bearer \bullet s.last\_used\} + 1 \bullet$$
$$\quad\quad\quad (s', new\_punter?)\}$$
$$location' = location$$

A new share can be issued to a person, though in practice this will require the consent of the existing shareholders, as their shares will decrease in value as a result. The new share's timestamp is set so as to give precedence to existing shareholders. The HRM remains with whoever already had possession of it.

AcquireShare

$$
\begin{array}{l}
\Delta State \\
punters? : \mathbb{P}\, Person \\
\hline
\forall\, p : punters? \bullet \\
\quad min\{s : dom(bearer \rhd \{location'\}) \bullet s.last\_used\} = \\
\quad min\{s : dom(bearer \rhd \{p\}) \bullet s.last\_used\} \\
\exists\, s : dom(bearer \rhd \{location'\}) \bullet \\
\quad bearer' = bearer \oplus \\
\qquad \{\mu\, s' : Share \mid \\
\qquad\quad s'.key = s.key \land \\
\qquad\quad s'.last\_used = max\{s2 : dom\ bearer \bullet s2.last\_used\} + 1 \bullet \\
\qquad\qquad (s', bearer\ \ s)\}
\end{array}
$$

The Acquire operation defines how a conflict between a set of persons all wanting to use the HRM at the same time is resolved: the person bearing a share that shows that person to have used the HRM least recently acquires possession of the HRM. The timestamp in that share is revised. The case of only one person wishing to use the HRM at a particular time is a special case that is covered by this general operation. The case of the same person holding more than one share that have all been used less recently that those of the other persons is resolved non-deterministically: any one of those shares has its timestamp revised.

TransferShare

$$\Delta State$$
$$vendor?, purchaser? : Person$$
$$share? : Share$$

$$(share?, vendor?) \in bearer$$
$$bearer' = bearer \oplus \{(share?, purchaser?)\}$$
$$location' = \text{if } location = vendor? \text{ then } purchaser? \text{ else } location$$

Shares can be transferred from one person to another, for example when a person leaves York. If that person is currently in possession of the HRM, the HRM is also transferred to the purchaser. Whether the vendor engages in a search for a highest bidder is outside the scope of this specification.

There is no operation to destroy a share; a share can simply fall into disuse.

Postscript: the instigator left, taking the HRM with him, and then refunding the shareholders!