Université Toulouse III – Paul Sabatier
118 route de Narbonne
31062 Toulouse cedex 9

Practical session 3 – Hoare Monitors

Reader-Writer model

The reader-writer model schematizes a situation encountered in the management of shareable files or in the access to databases.

Two families of processes access this information. Readers only want to consult the information. Writers want to modify this information. The readers can therefore access the information in parallel, whereas writers must access it in mutual exclusion.

The behaviors of these processes are therefore as follows:

A reader	A writer

Loop on {
…
Ask to read;
Read;
Signal reading is finished;
…
}
	Loop on {
	…
	Ask to write;
	Write;
	Signal writing is finished;
	…
	}

We propose to write a Hoare monitor managing the access to the common resource according to the policy defined previously and according to the following variants.

The specification of the monitor is as follows:

```
Monitor Reader_Writer {
  void startRead();
  void endRead();
  void startWrite();
  void endWrite();
end Reader_Writer;
```

Variant 1

When no readers are reading, readers and writers have the same priority. On the other hand, as soon as a reader is reading, all other readers requesting it can also read, since readings can be done in parallel, regardless of the number of editors waiting.

When a writer is writing, no other client can access the resource (neither reader nor writer).

When the writer is done writing, it tries to activate a writer over the readers.

Variant 2 – Priority of writers over readers

We want to give priority to writers so that the information available is not obsolete for readers. When a writer requests access to the resource, it should therefore get it as soon as possible. Of course, it is not possible to interrupt ongoing reading or writing. Likewise, it has no right to go before other writers in the queue (who arrived before it and have not yet been accepted). On the other hand, it has priority over the waiting readers.

Variant 3 - More equitable access management

To what erroneous situations can variants 1 and 2 lead?

What solution can be proposed to correct such situations?

In this variant, a writer who finishes writing must give priority access to all the readers waiting at that moment, and not to the next writer as specified in variant 1. On the other hand, readers who arrive later (after this writing request) must respect the rule of priority of writers over readers as expressed in variant 2.

Variant 4 – FIFO management

We now assume that data access is performed according to a FIFO policy; requests are processed in the order of their arrival.

To globally order the readers and writers waiting, all processes will be blocked on the same condition. Indeed, it is not possible to know whether the 3rd writer has arrived before or after the 4th reader when readers and writers are ordered in separate queues. It should be noted that, when waking up, it is not possible (for the "signaller") to distinguish a reader from a writer. Therefore, it may be necessary to wake up a process (reader or writer) and then block it again afterwards if it is impossible to unlock it in the current situation.

Questions :

For each variant :

- Specify the monitor.
- Specify the blocking and waking conditions of a Reader process and a Writer process.
- Deduce the state variables and the “condition” variables used in the monitor.
- Give the “code” (algorithm) of the monitor.