Complement on Monitors

- Definition: A monitor encapsulates
  - shared data,
  - all the operation on the data,
  - and any synchronization required to access the data.
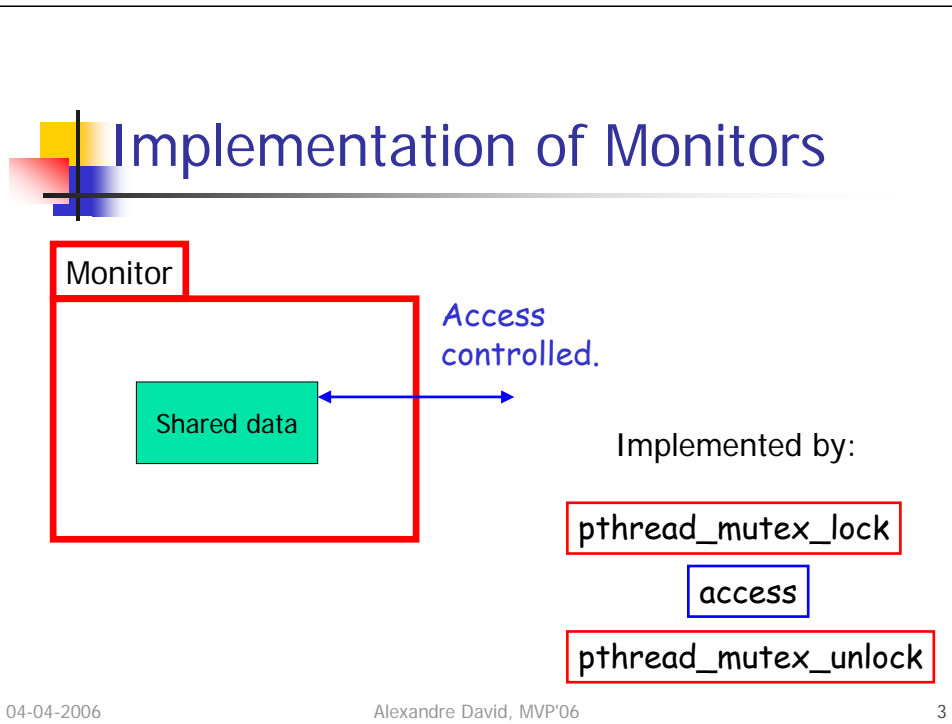- A monitor has separate constructs for condition synchronization.

You should know the concept of monitor. Monitors in Java are implemented by having all the access methods "synchronized" and declaring all data private for a class. It is straight-forward. In C++, you have to use a mutex inside all the methods and encapsulate your data again (and do not cheat with "friend" access functions).

Conditional wait – a condition synchronization – is using a construct with monitors.
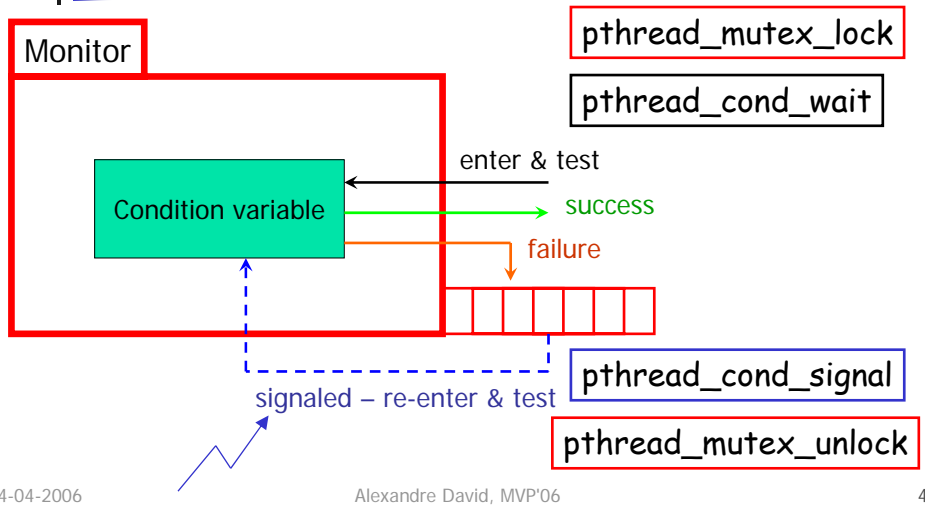
Monitors & Pthreads

- There is no monitor in pthreads.
- We implement the concept and use the construct for condition synchronization.

Implementation of Monitors

Monitor

Shared data

Access
controlled.

Implemented by:

pthread_mutex_lock

access

pthread_mutex_unlock

Make the distinction between the concept and how it is implemented in a particular language with a particular API. Java implements the same concept in a completely different manner.

Implementation of Conditional Wait

Monitor

pthread_mutex_lock

pthread_cond_wait

enter & test

Condition variable

success

failure

pthread_cond_signal

signaled – re-enter & test

pthread_mutex_unlock