AARHUS
UNIVERSITY
DEPARTMENT OF ENGINEERING

Architecture & Design of Embedded Real-Time Systems (TI-AREM)

Concurrency Patterns 1.

BPD Chapter 5. 203-258

Version: 16-2-2015

Agenda

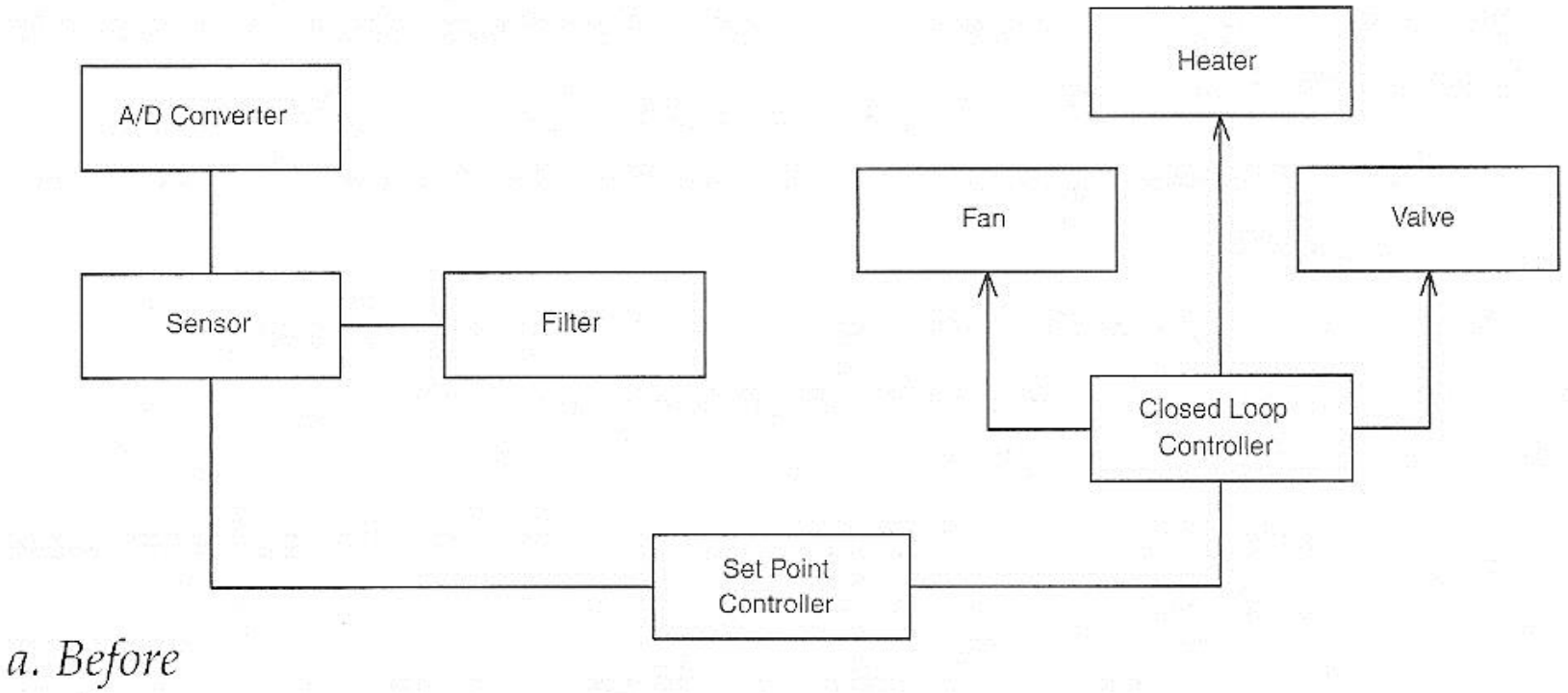
Introduction to concurrency

1. Interrupt Pattern
2. Message Queuing Pattern
3. Guarded Call Pattern
4. Rendezvous Pattern

---

5. Cyclic Executive Pattern
6. Round Robin Pattern
7. Static Priority Pattern
8. Dynamic Priority Pattern

Concurrency Patterns (1)

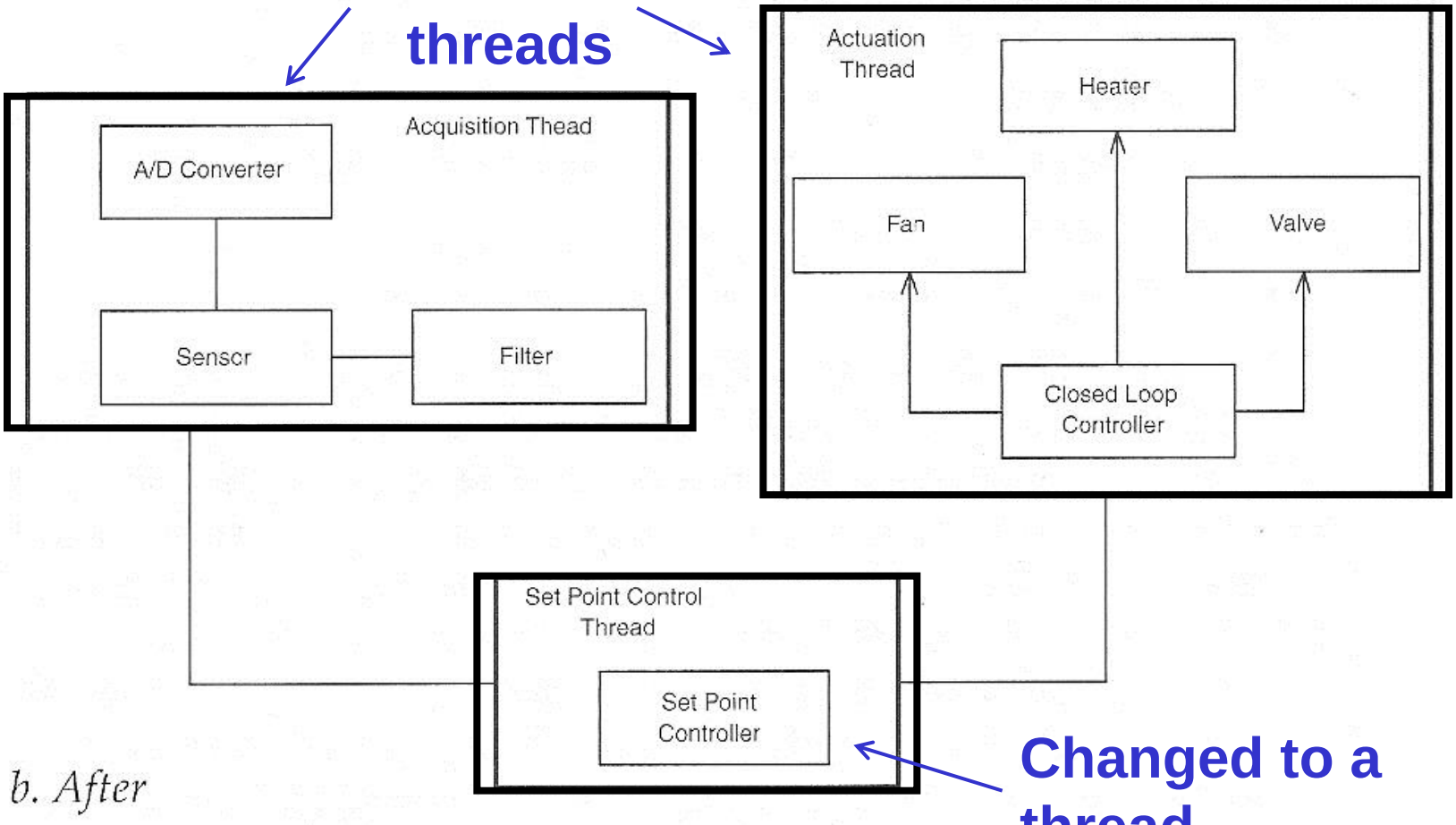


a. Before

Introducing concurrency (bottom up approach)

Concurrency Patterns (2)

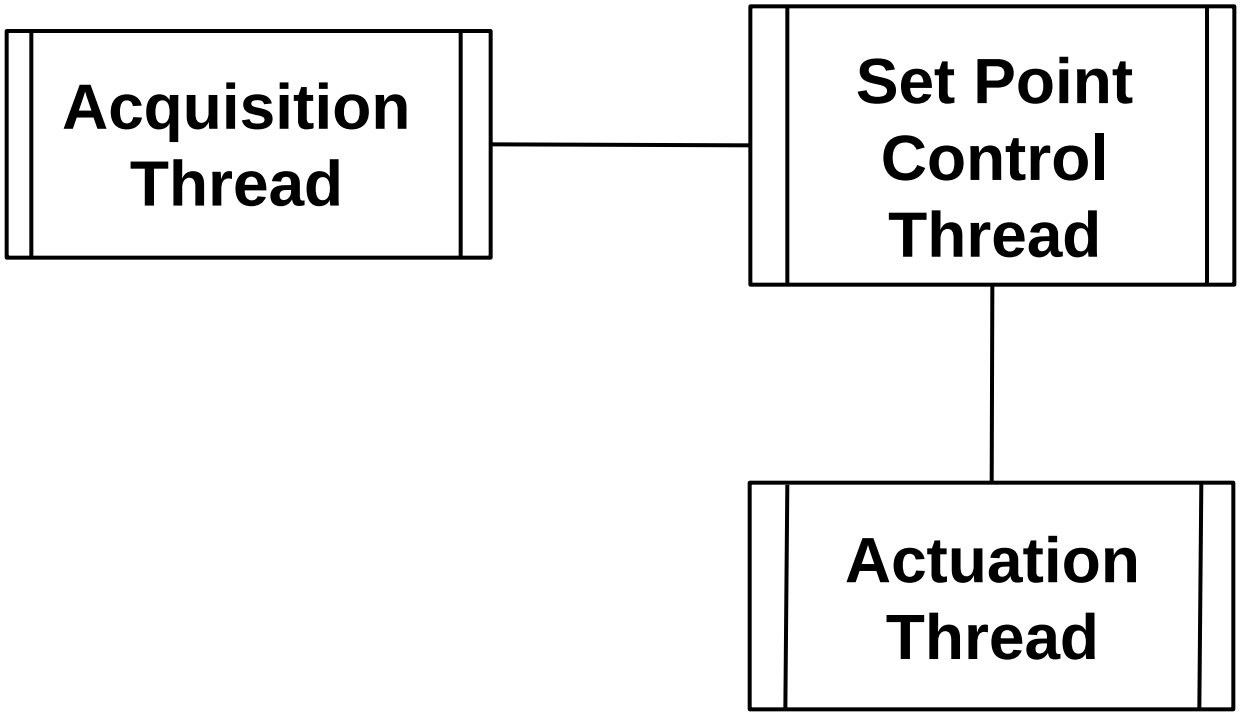
New threads



Changed to a thread

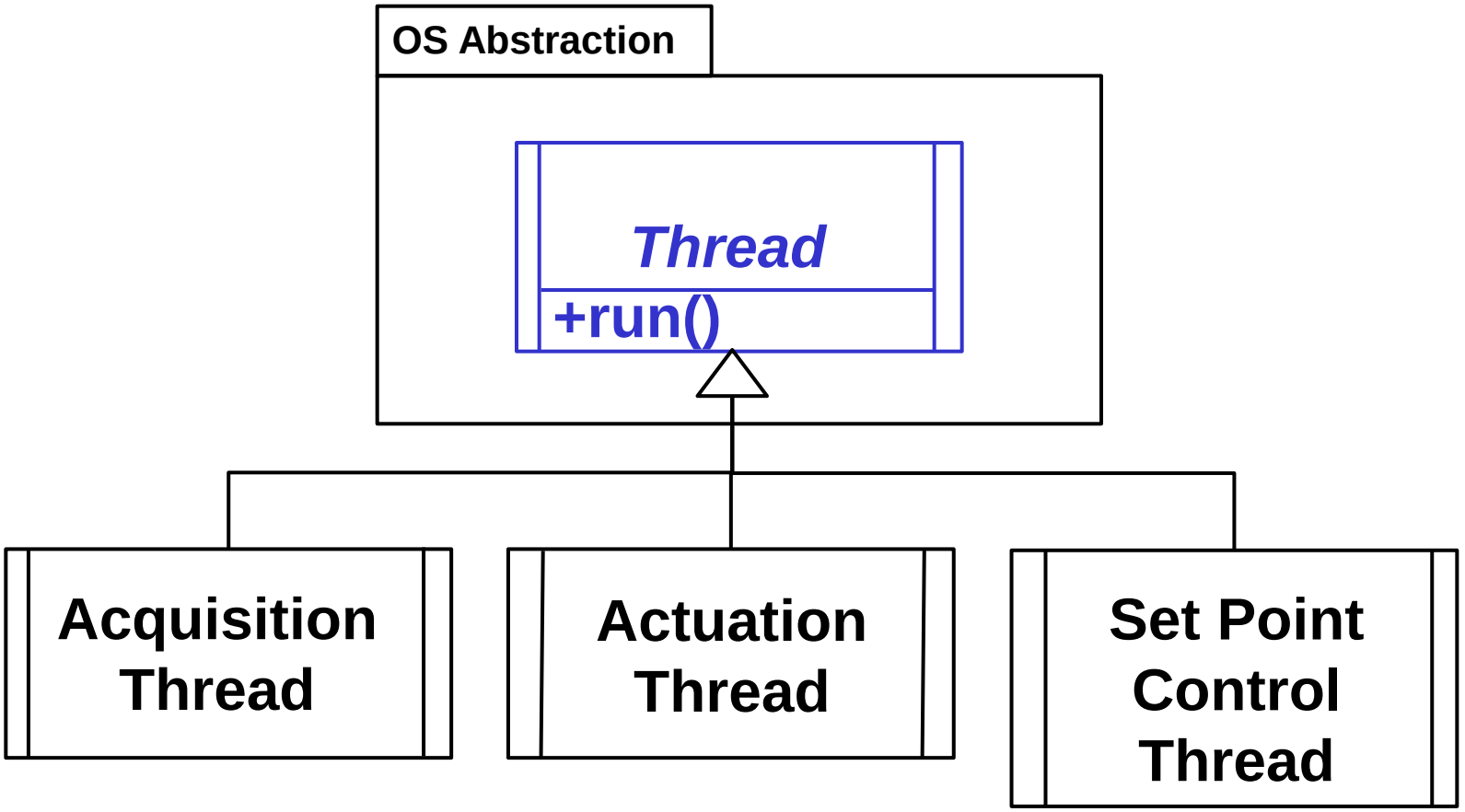
Concurrency design

Concurrency Patterns (3)



Introducing concurrency (top down approach)

Concurrency Patterns (4)

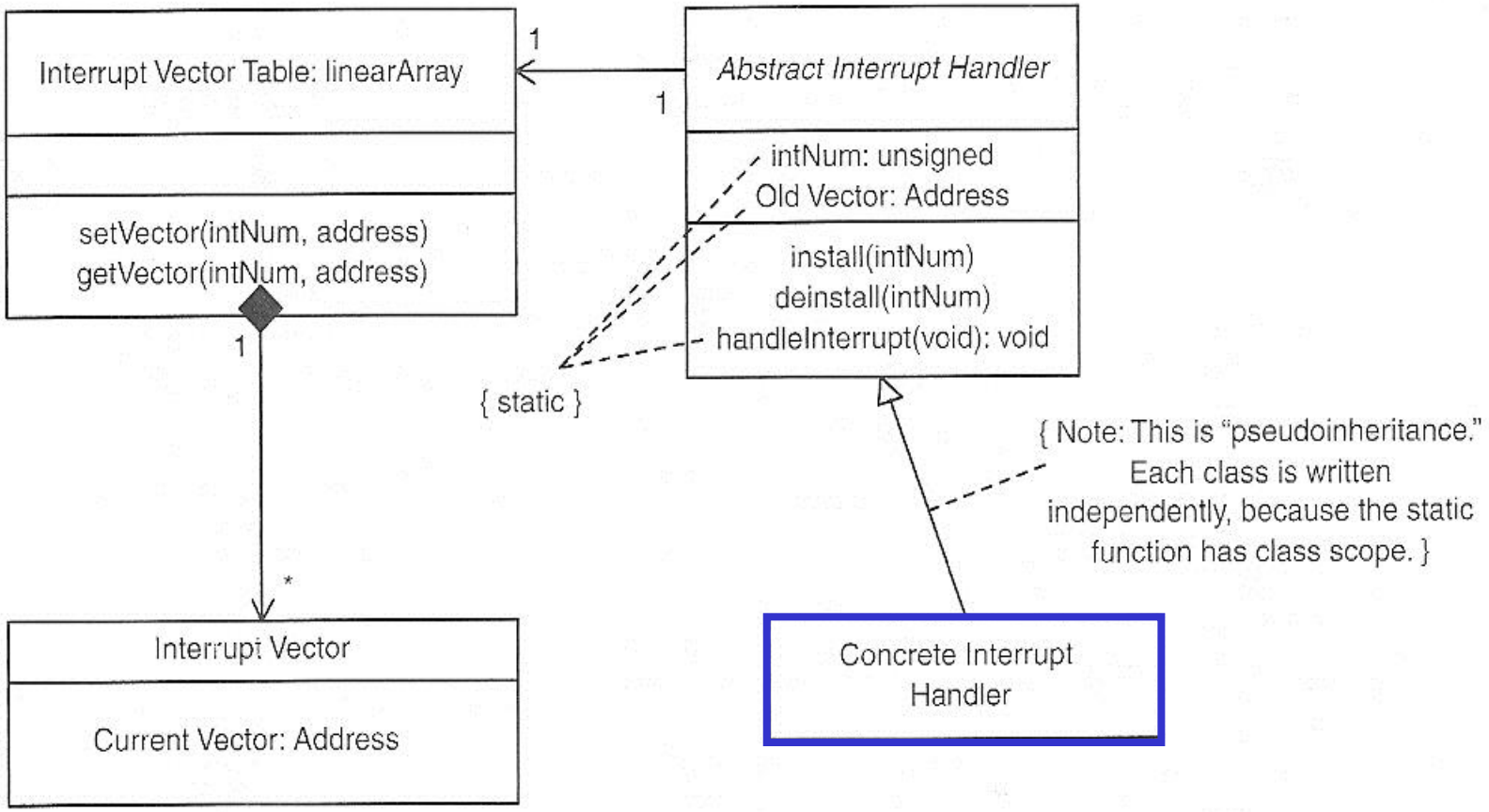


Normal Thread Implementation Pattern

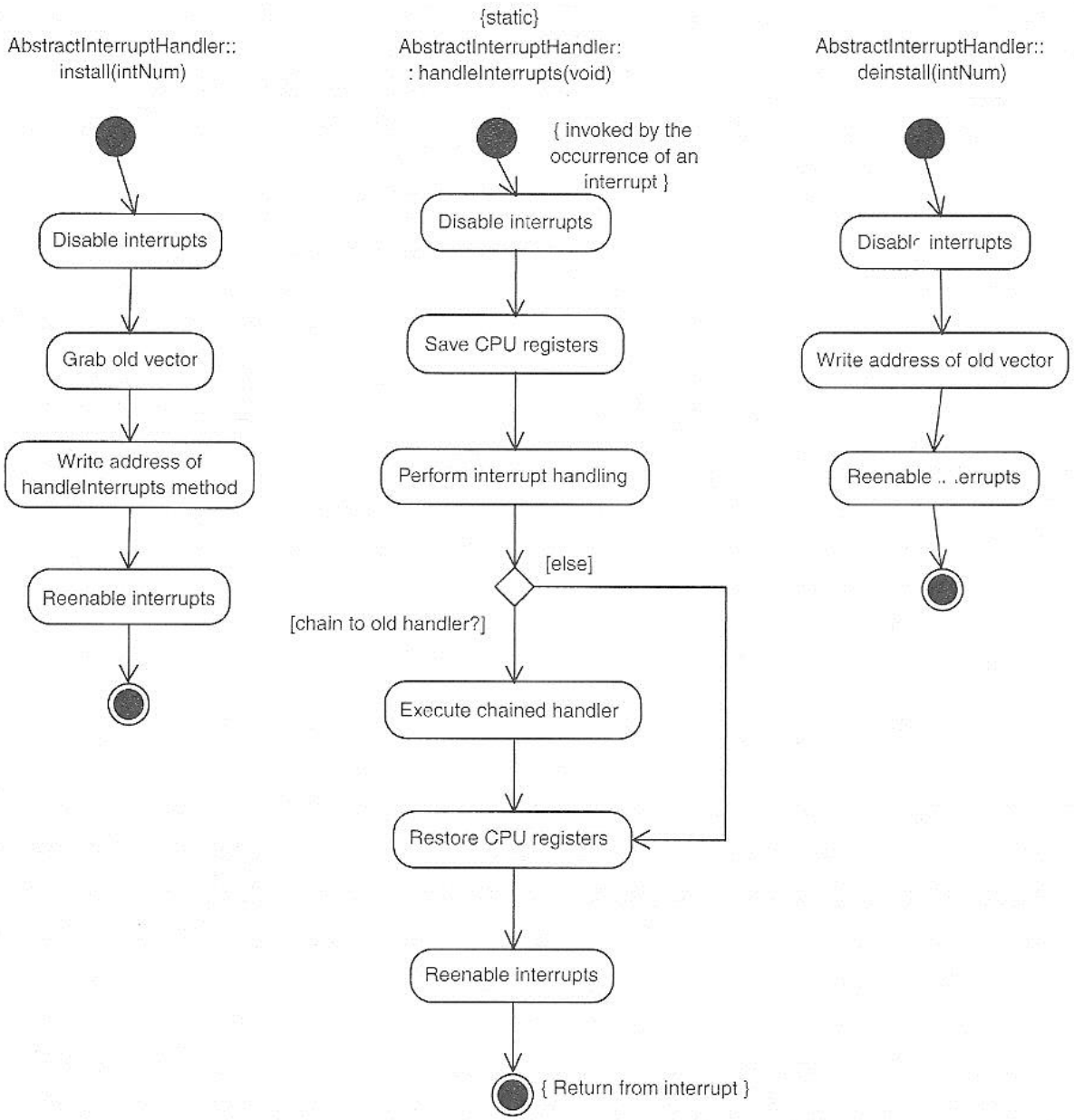
1. Interrupt Pattern

The Interrupt Pattern can be an excellent solution for handling events that must be responded to quickly and efficiently

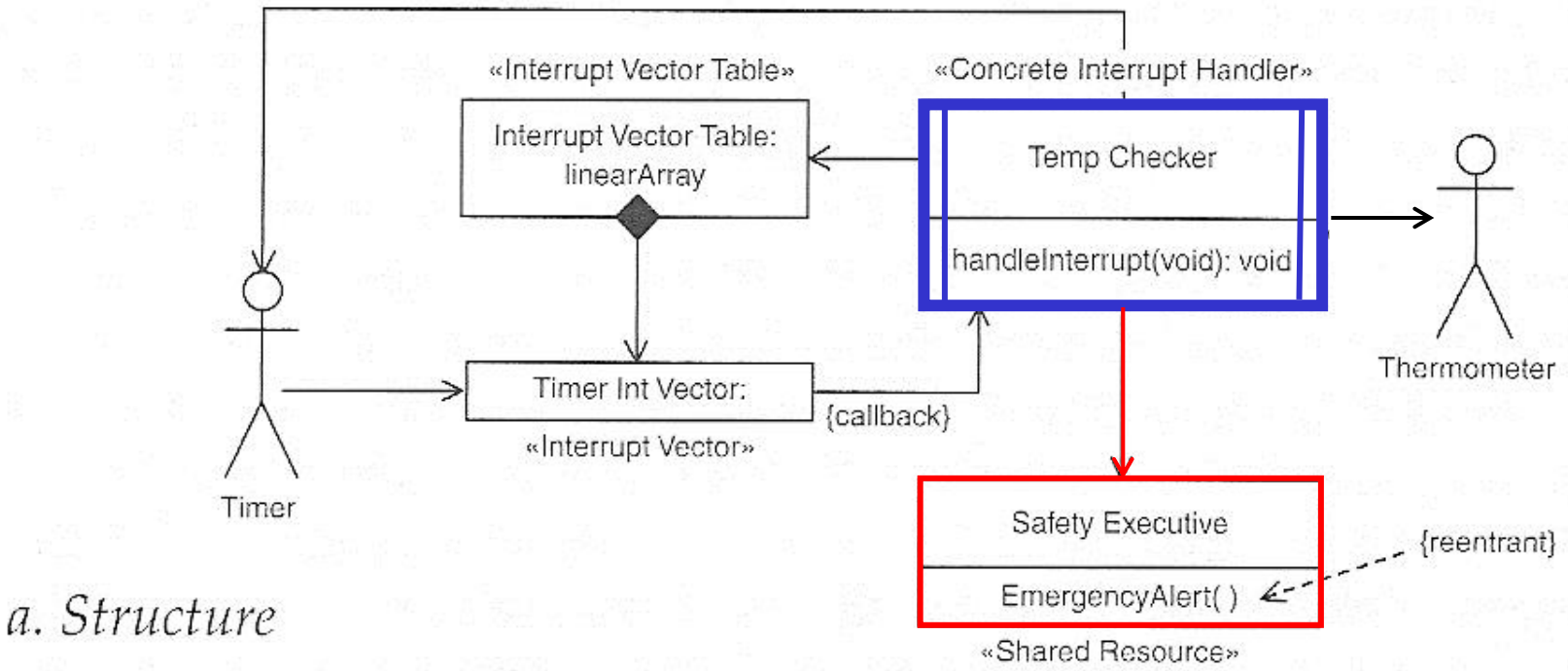
Interrupt Pattern Structure



Interrupt Handling Methods

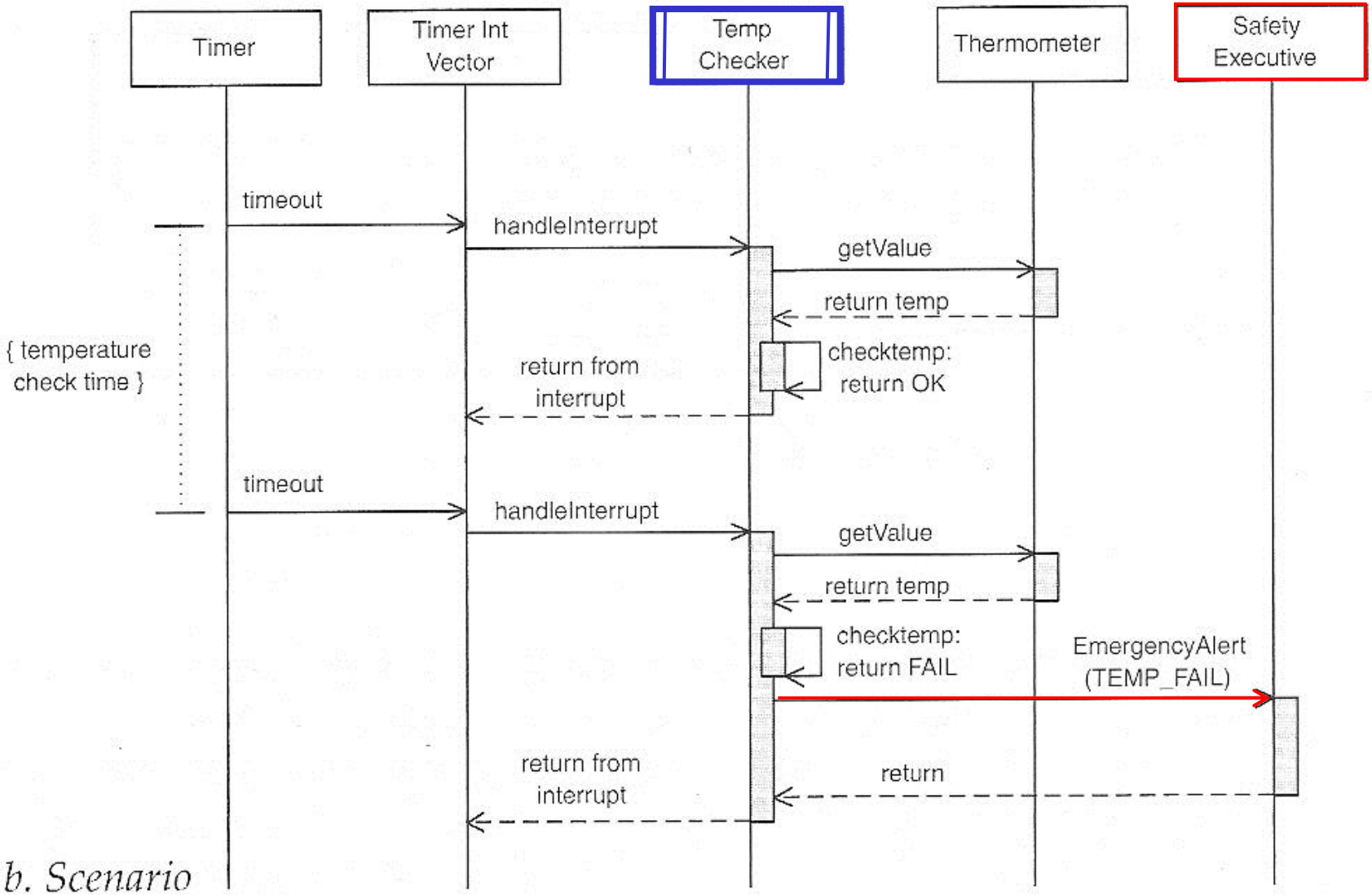


Interrupt Pattern Example (1)



a. Structure

Interrupt Pattern Example (2)

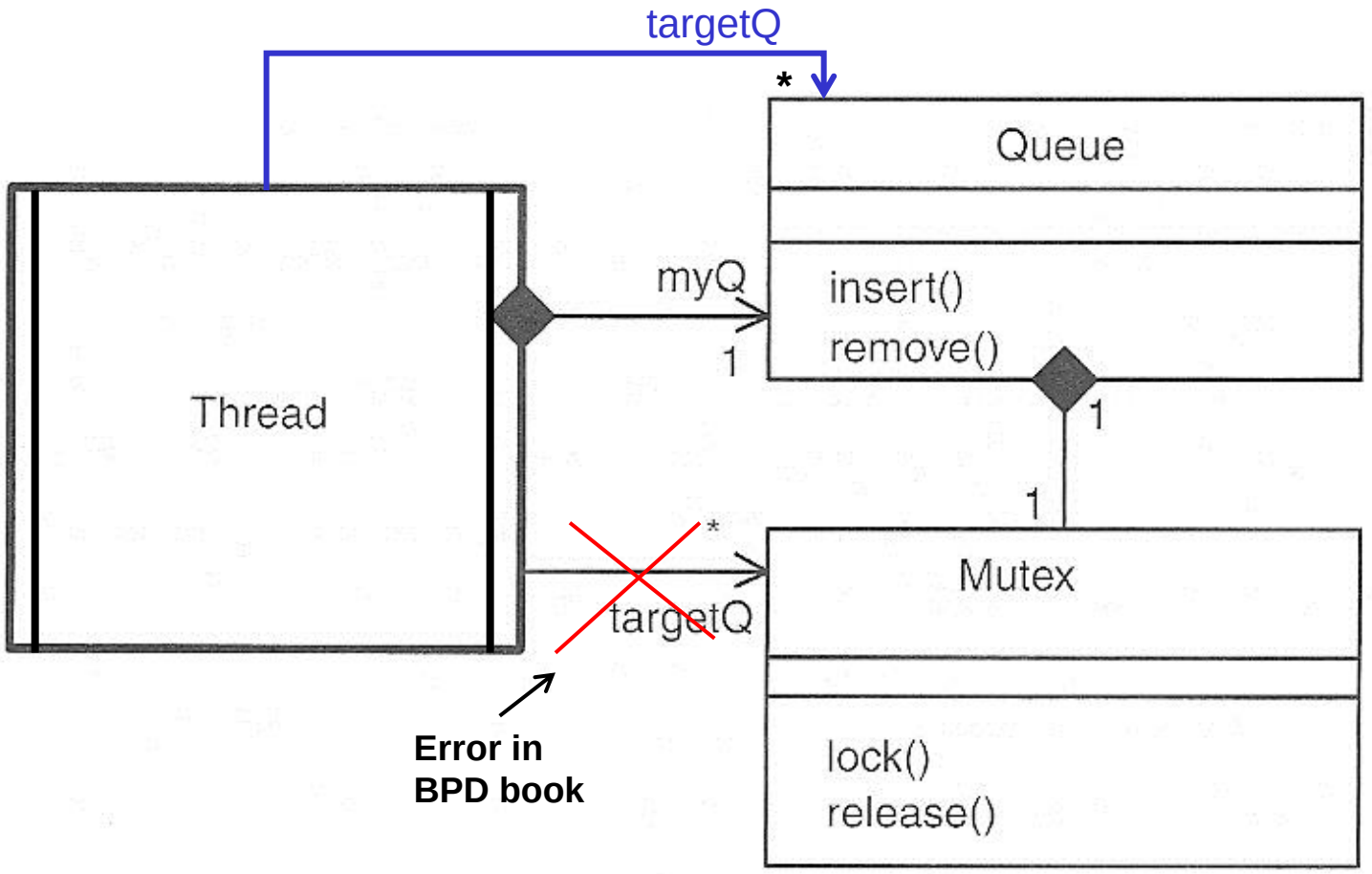


2. Message Queuing Pattern

The Message Queuing Pattern uses **asynchronous communications**, implemented via queued messages, to synchronize and share information among tasks.

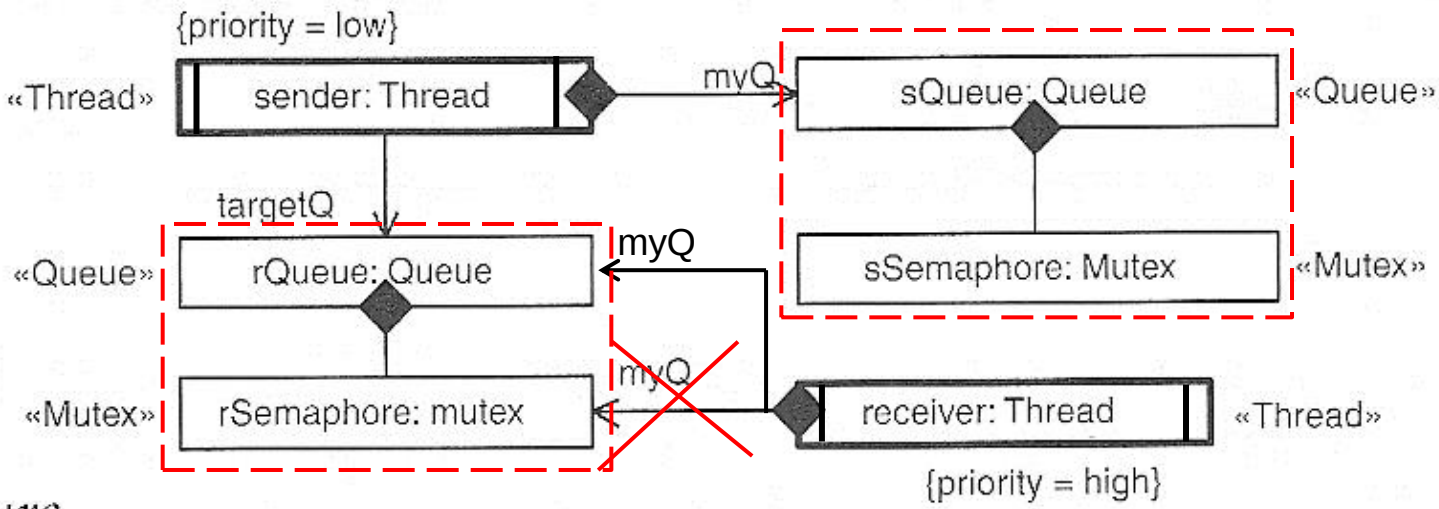
Any information shared among threads is passed by value.

Message Queuing Pattern Structure



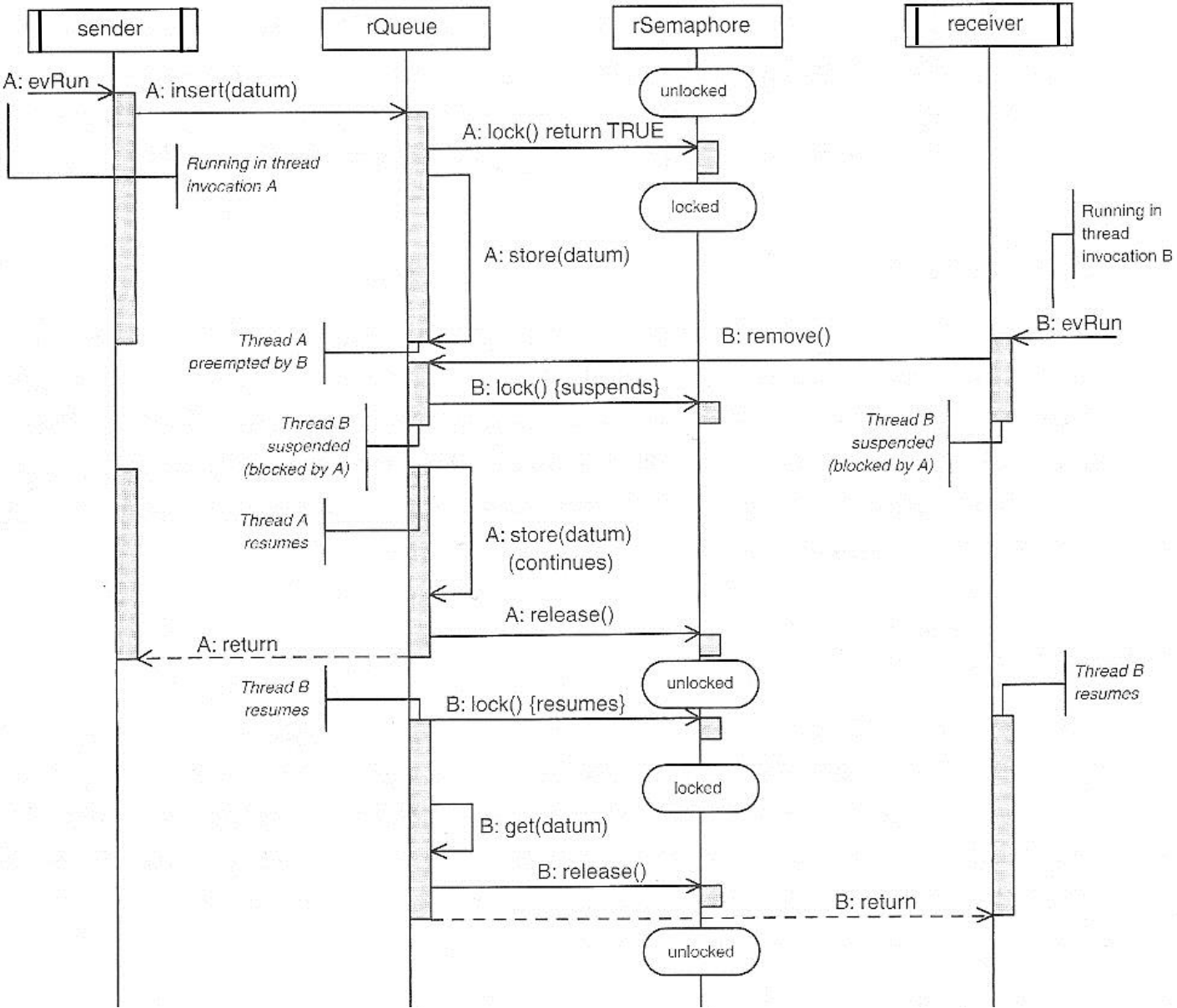
asynchronous communications

Message Queuing Pattern Example (1)



a. Structure

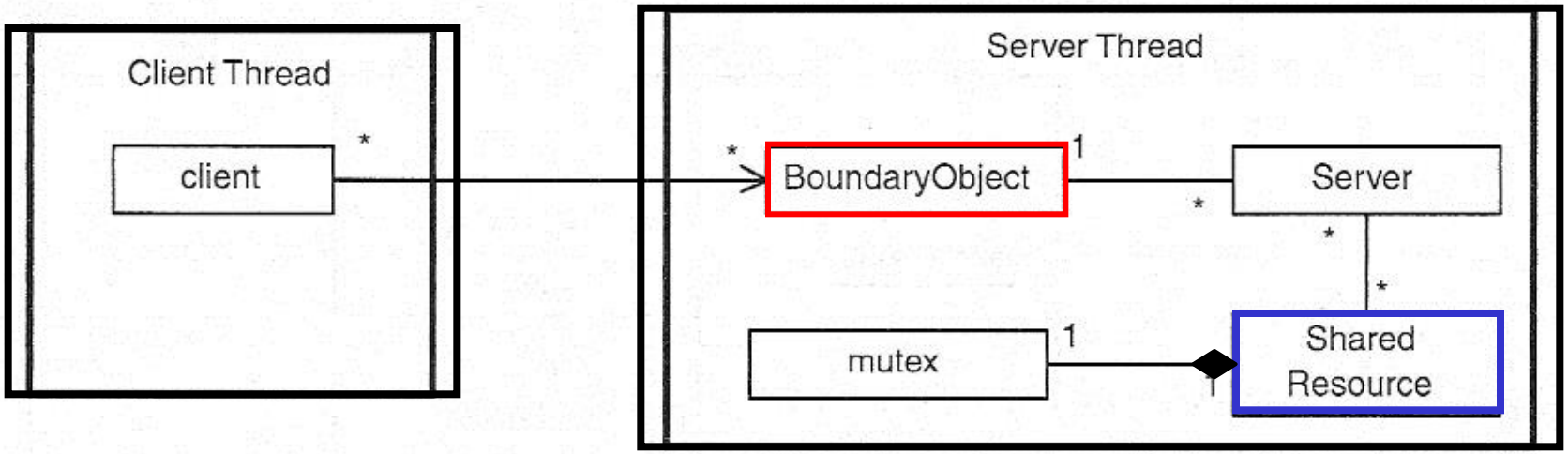
Message Queuing Pattern Example (2)



3. Guarded Call Pattern

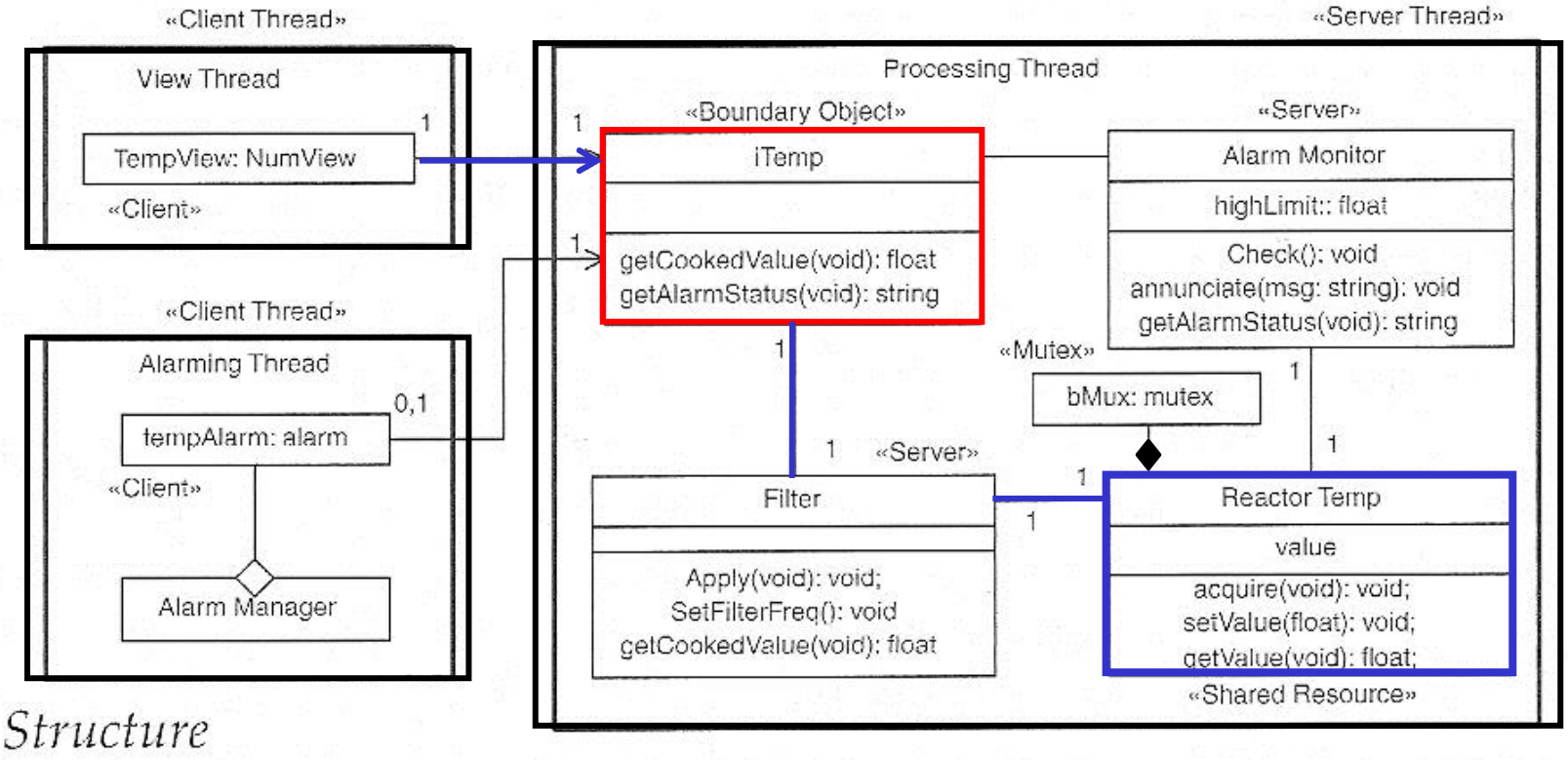
The Guarded Call Pattern allows to **synchronously** invoke a method of an object, nominally running in another thread

Guarded Call Pattern Structure



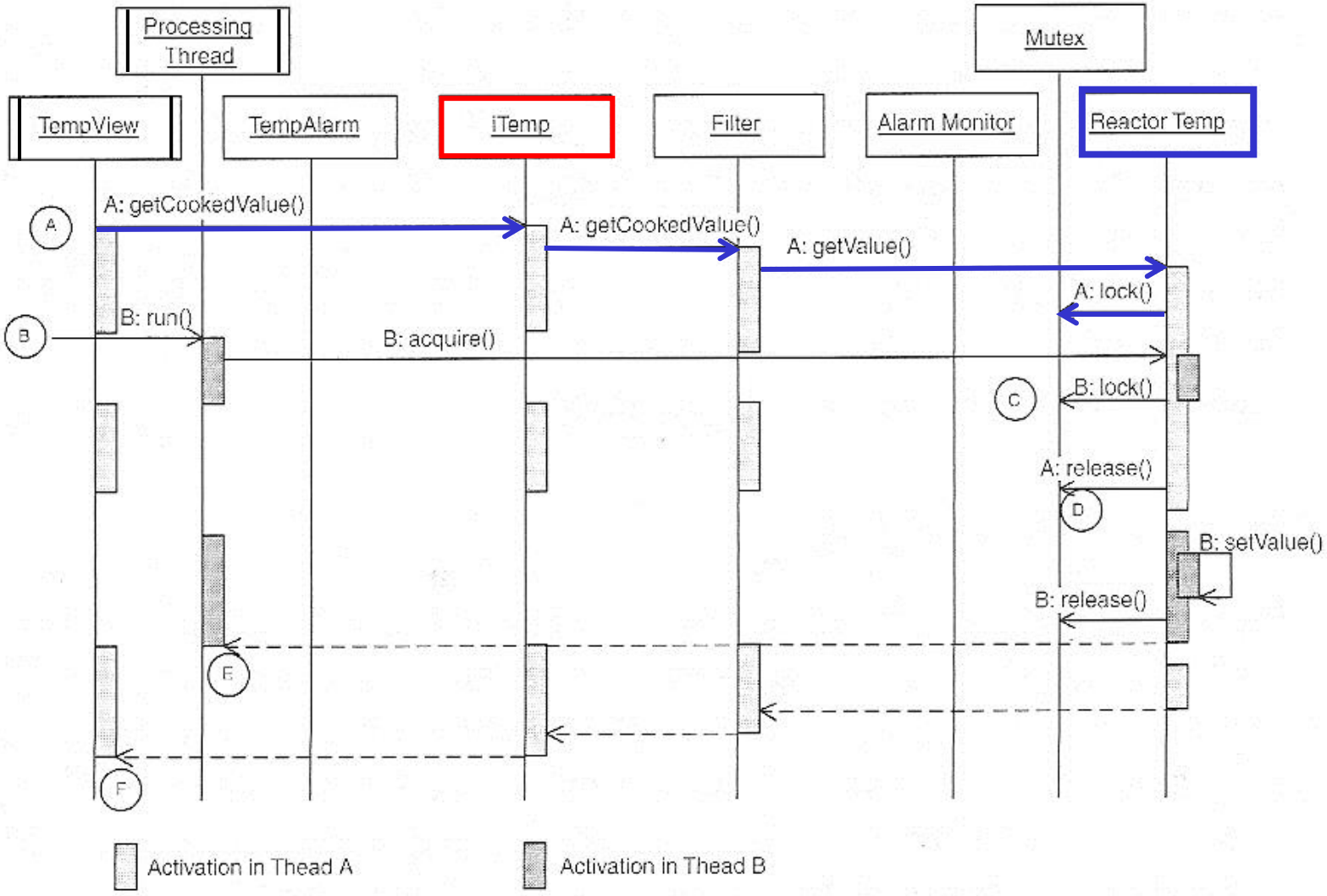
Synchronous communication / call between threads

Guarded Call Pattern Example (1)



a. Structure

Guarded Call Pattern Example (2)

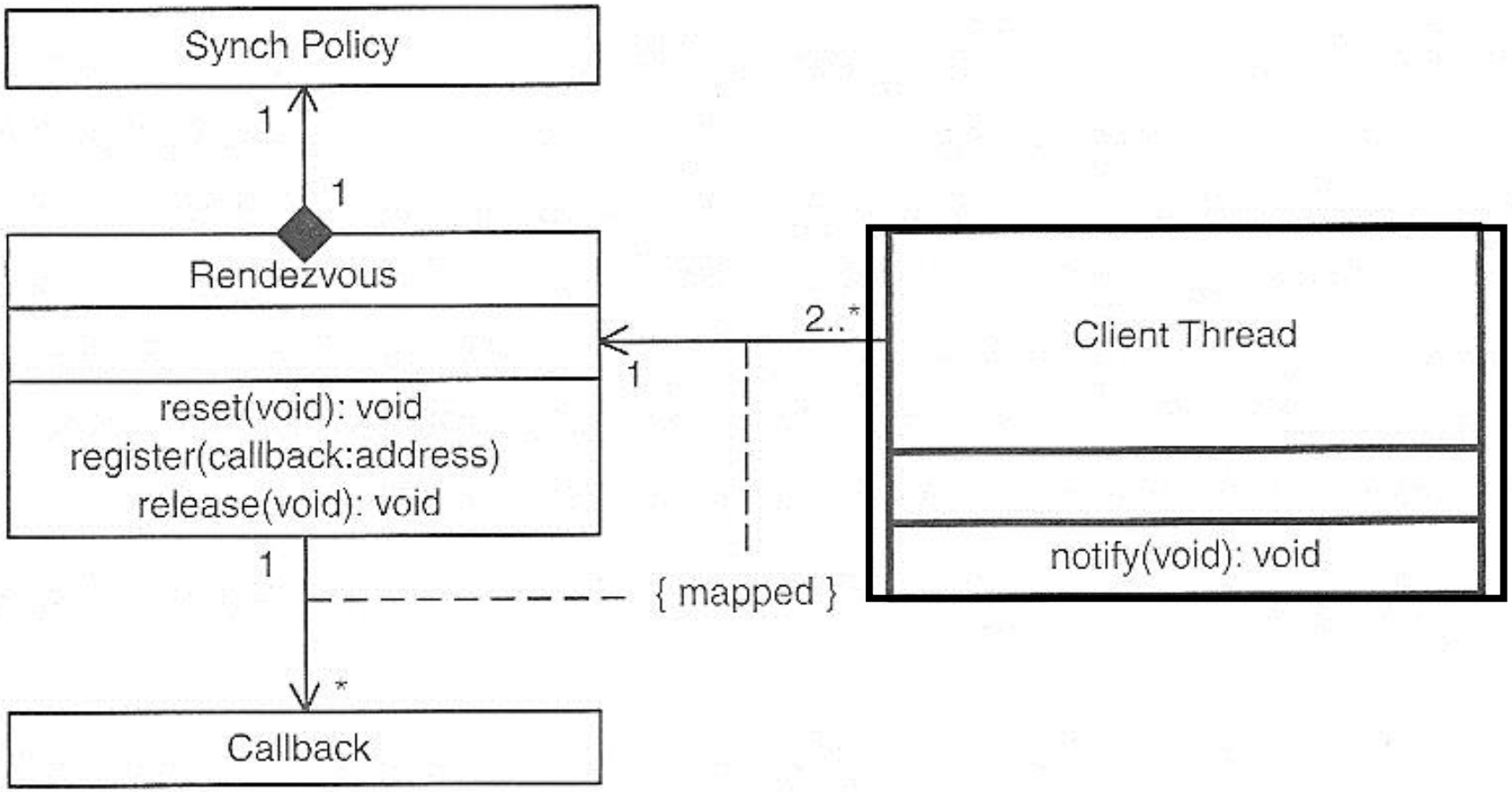


Scenario

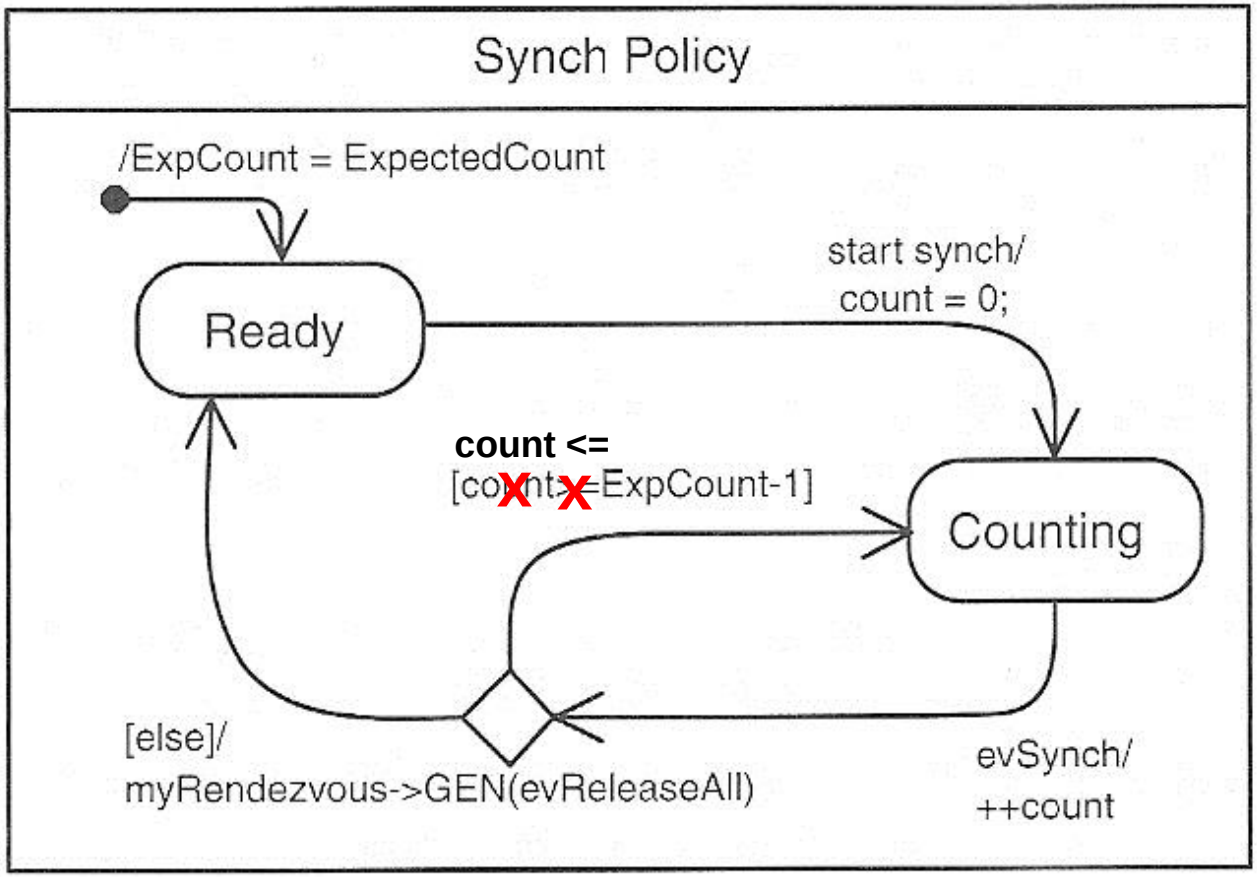
4. Rendezvous Pattern

The Rendezvous Pattern is a simplified form of the Guarded Call pattern used to either synchronize a set of threads or permit data sharing among a set of threads

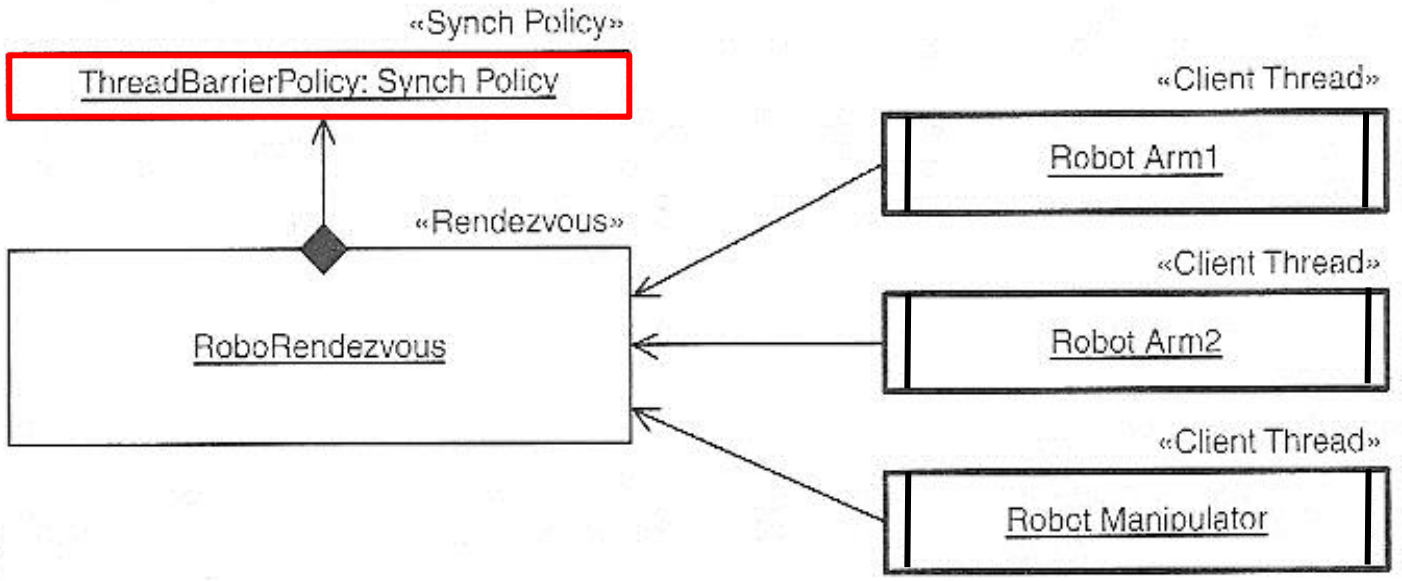
Rendezvous Pattern Structure



Thread Barrier Synch Policy State Diagram

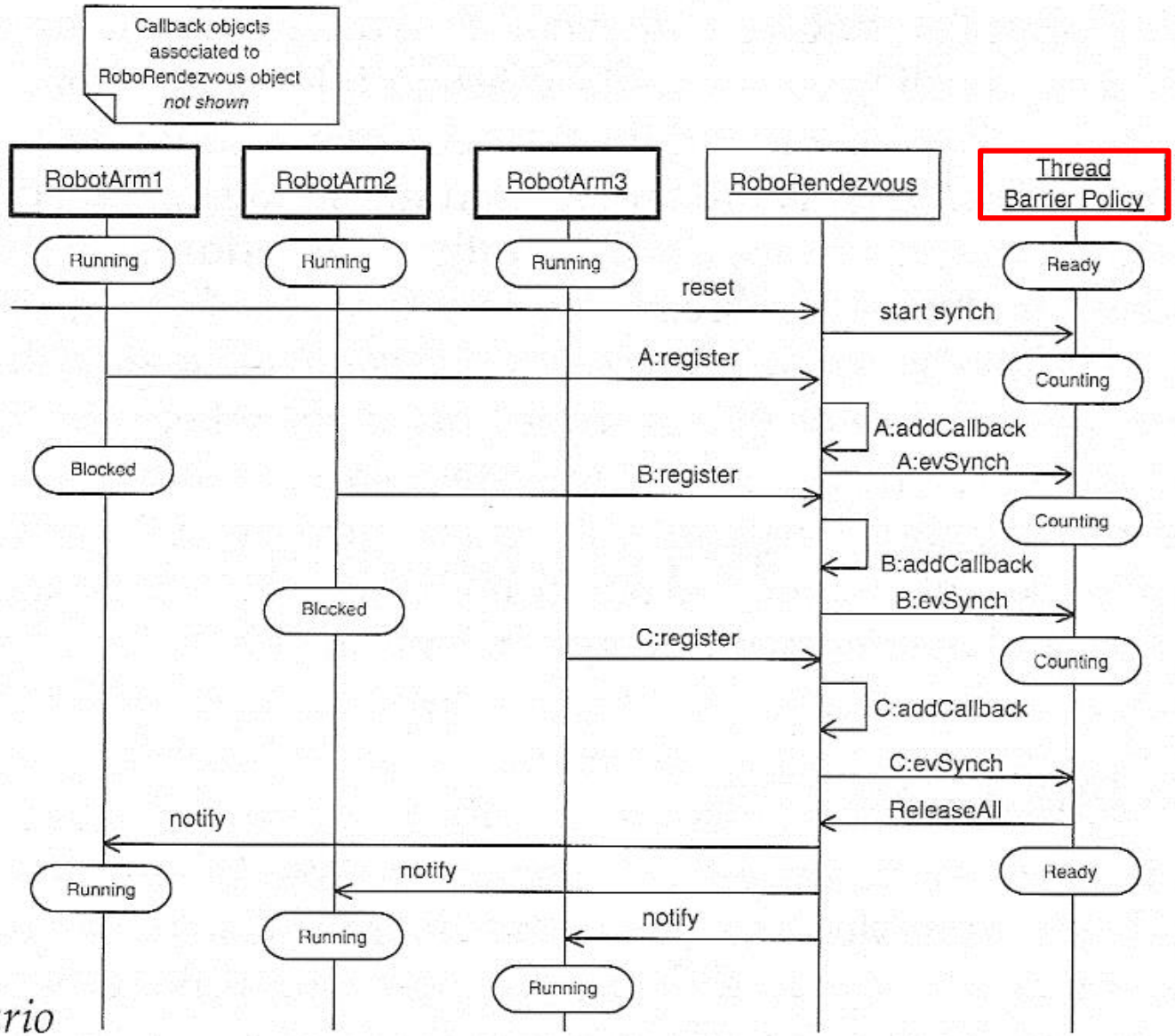


Rendezvous Pattern Example (1)



Structure

Rendezvous Pattern Example (2)



b. Scenario

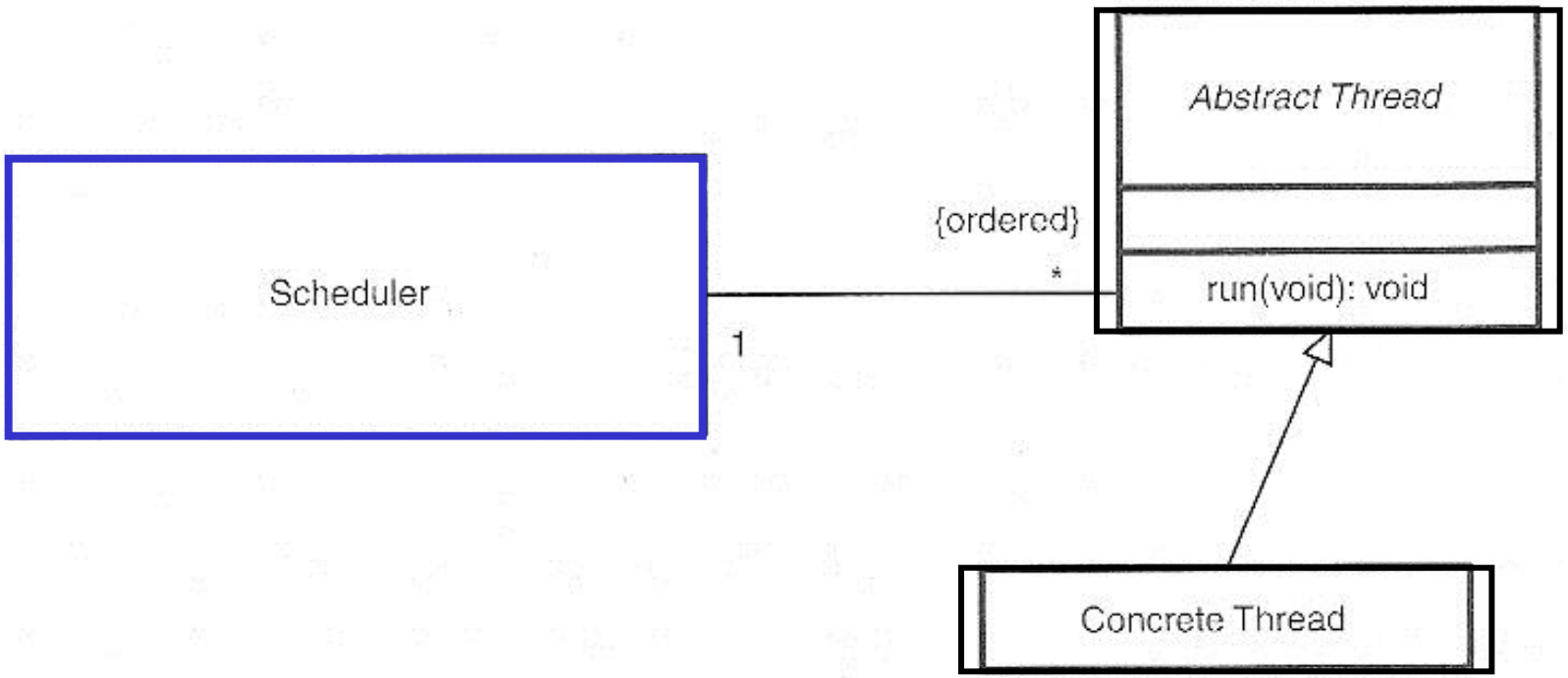
5. Cyclic Executive Pattern

The Cyclic Executive Pattern has the advantage of almost mindless simplicity coupled with extremely predictable behavior.

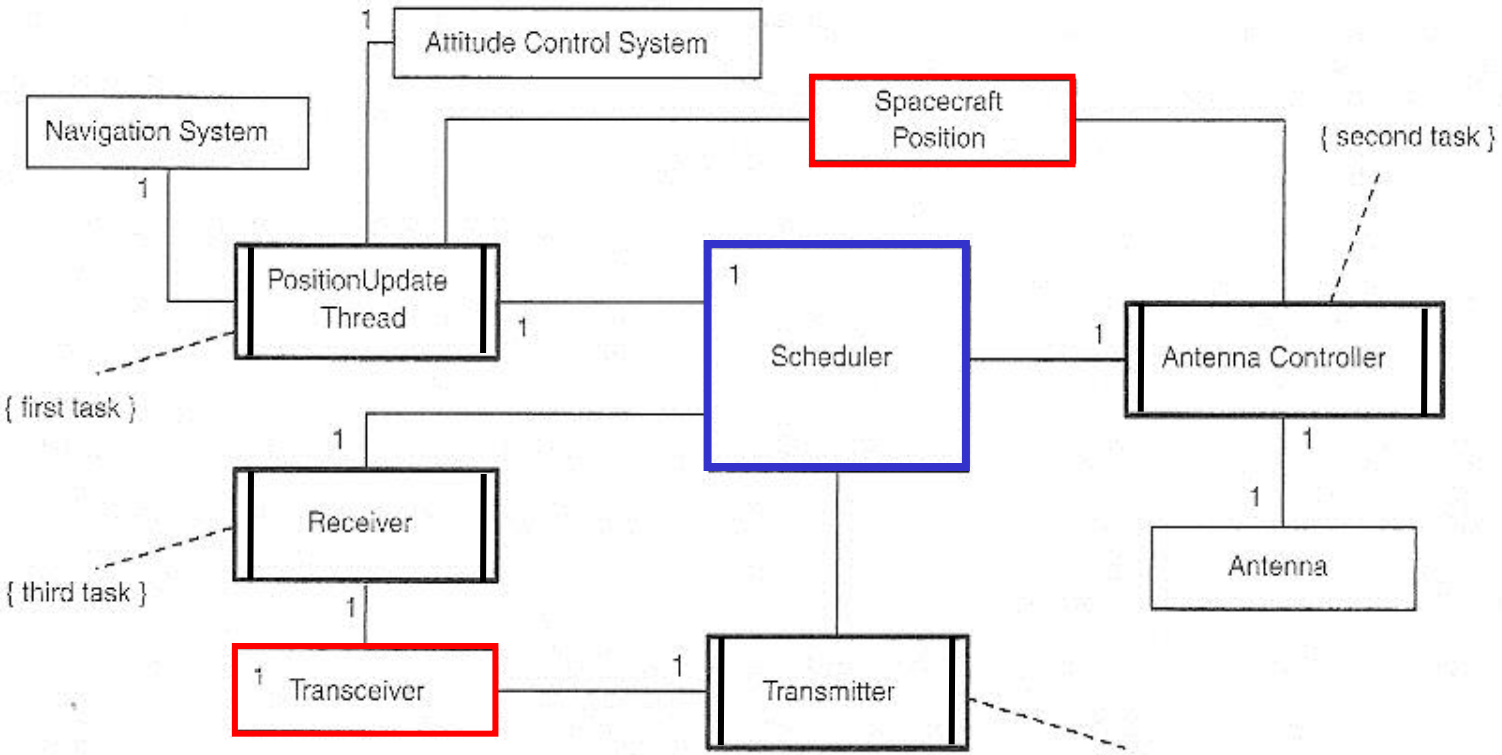
Useful for very small systems or

for systems in which execution predictability is crucial.

Cyclic Executive Pattern Structure

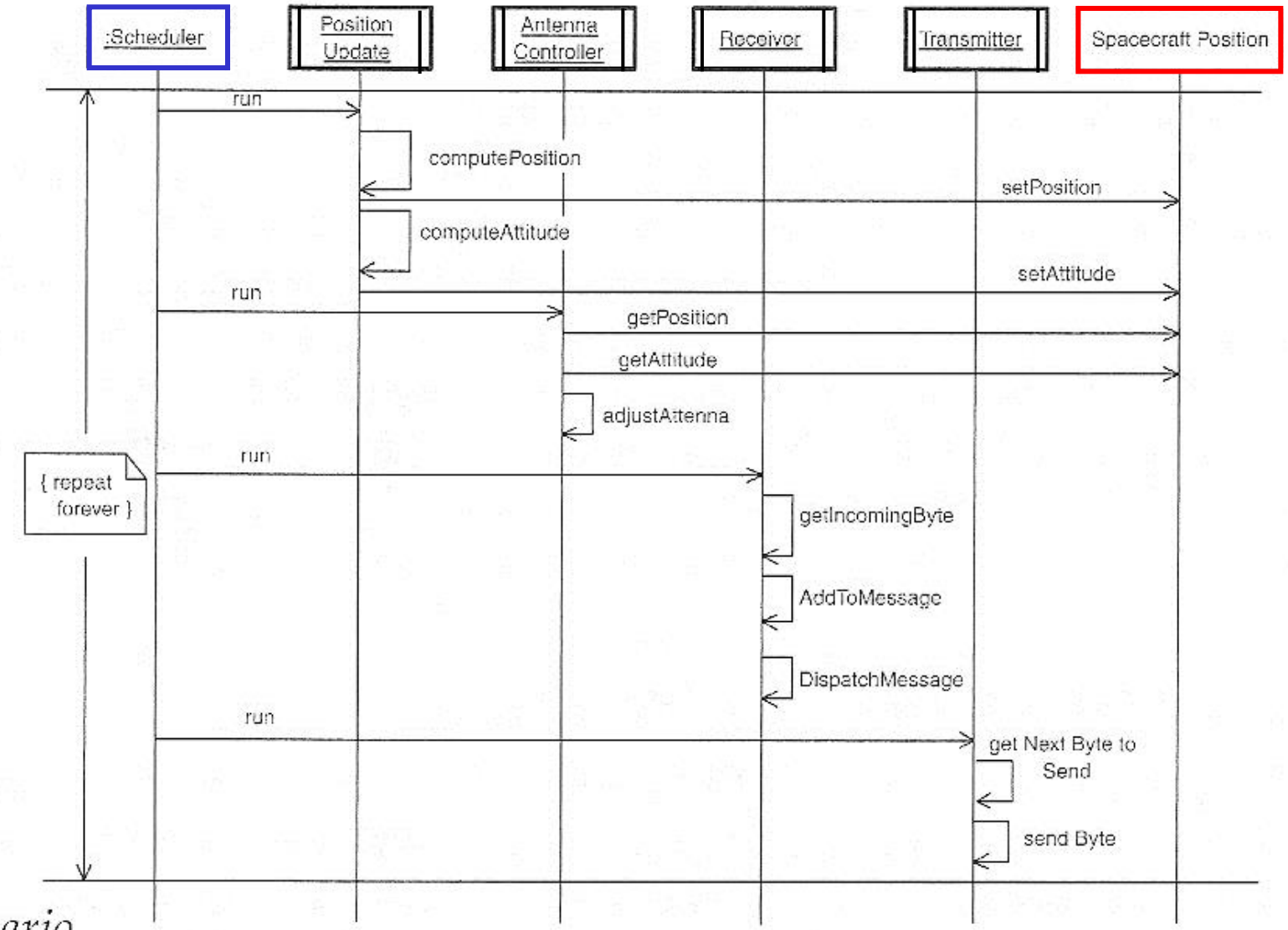


Cyclic Executive Pattern Example (1)



Structure

Cyclic Executive Pattern Example (2)

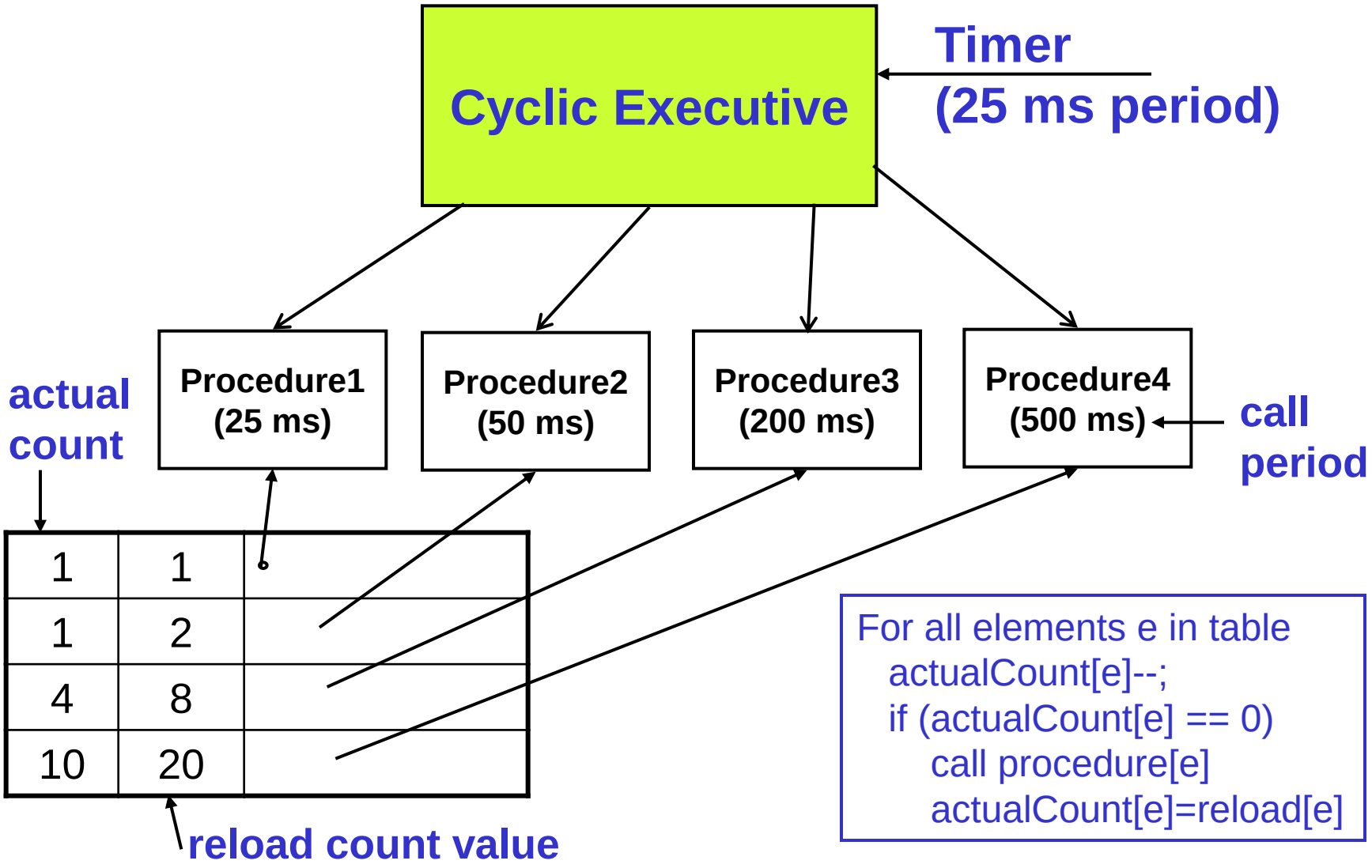


Scenario

Timeline Cyclic Executive (1)

- A Timeline uses a timer to trigger a task (a procedure) every minor cycle
- The system is design as a set of procedures, with strict time requirements
- The procedures are placed in a predefined list covering every minor cycle
- When a minor cycle begins, the timer task calls each procedure in the list
- Concurrency is not used (don’t have to deal with synchronization issues)
- Long procedures must be manually broken to fit frames

Timeline Cyclic Executive (2)



Timeline Cyclic Executive (3)

- Advantage
  - time can be guaranteed
  - simple executive
  - no concurrency problems
- Disadvantage
  - time driven procedural structure
  - very expensive to maintain
- Recommended by FDA standard for safety critical systems
  - (FDA: Federal Drug Administration in USA)

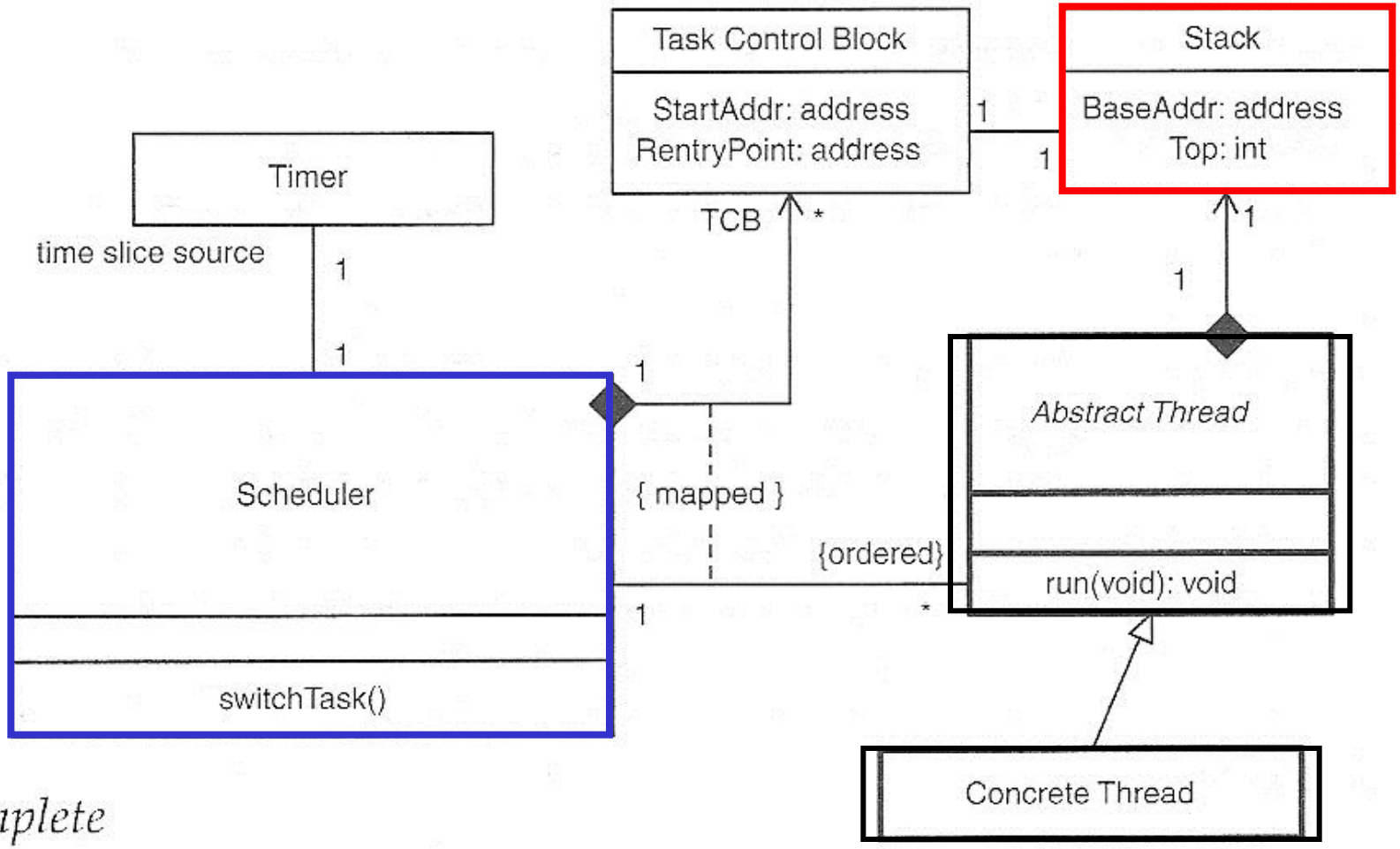
Timeline Cyclic Executive (4)

- **Example: Flight Control Computer**
  - Primary Inputs:
    - Aircraft navigation information, sensor information, equipment status, flight plan information
  - Primary Outputs:
    - Aircraft control commands, recording information, crew display update
  - Number & Types of processors:
    - 2-5 processors, 1 MB RAM
    - 1 task for each computer,
    - no synchronization
  - Typical Time Constraints
    - Most requirements are *Hard Real Time* requirements, typically 20 ms, human interface 100 ms
  - Used in newer Airbus aircrafts and in Boeing 747

6. Round Robin Pattern

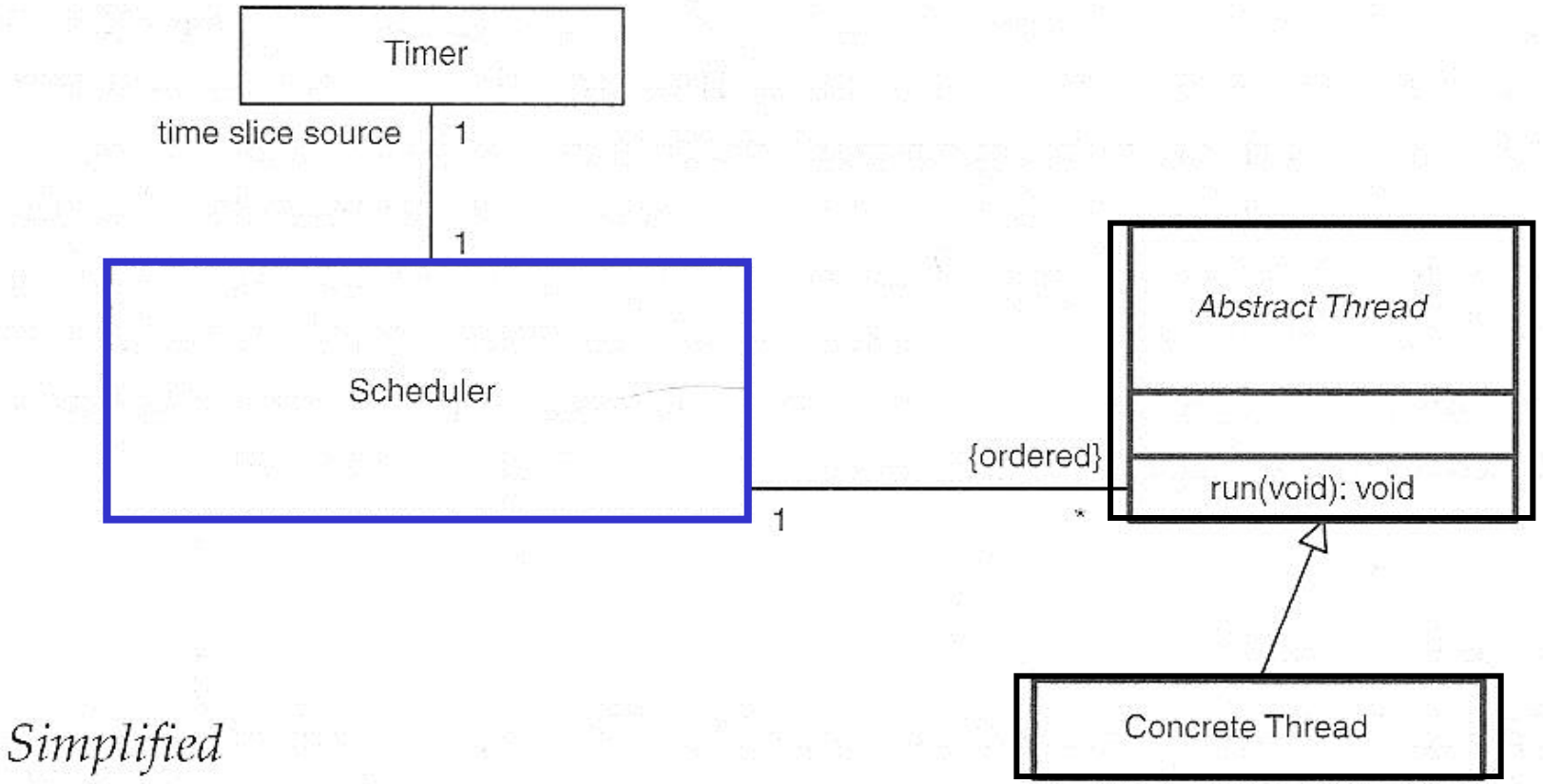
The Round Robin Pattern is a simple variation of the Cyclic Executive Pattern. The difference is that the Scheduler has the ability to preempt running tasks and does so when it receives a tick from its associated timer.

Round Robin Pattern Structure (1)



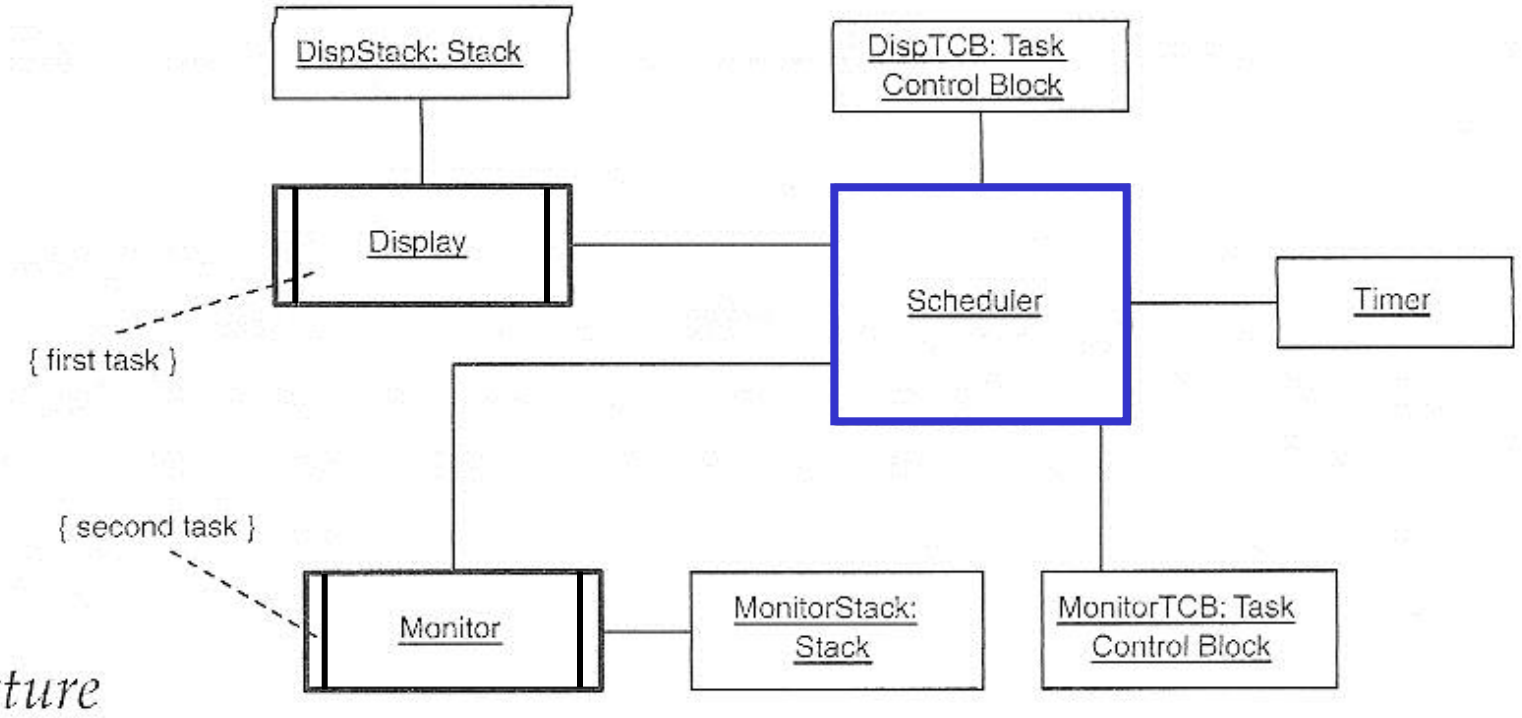
Complete

Round Robin Pattern Structure (2)



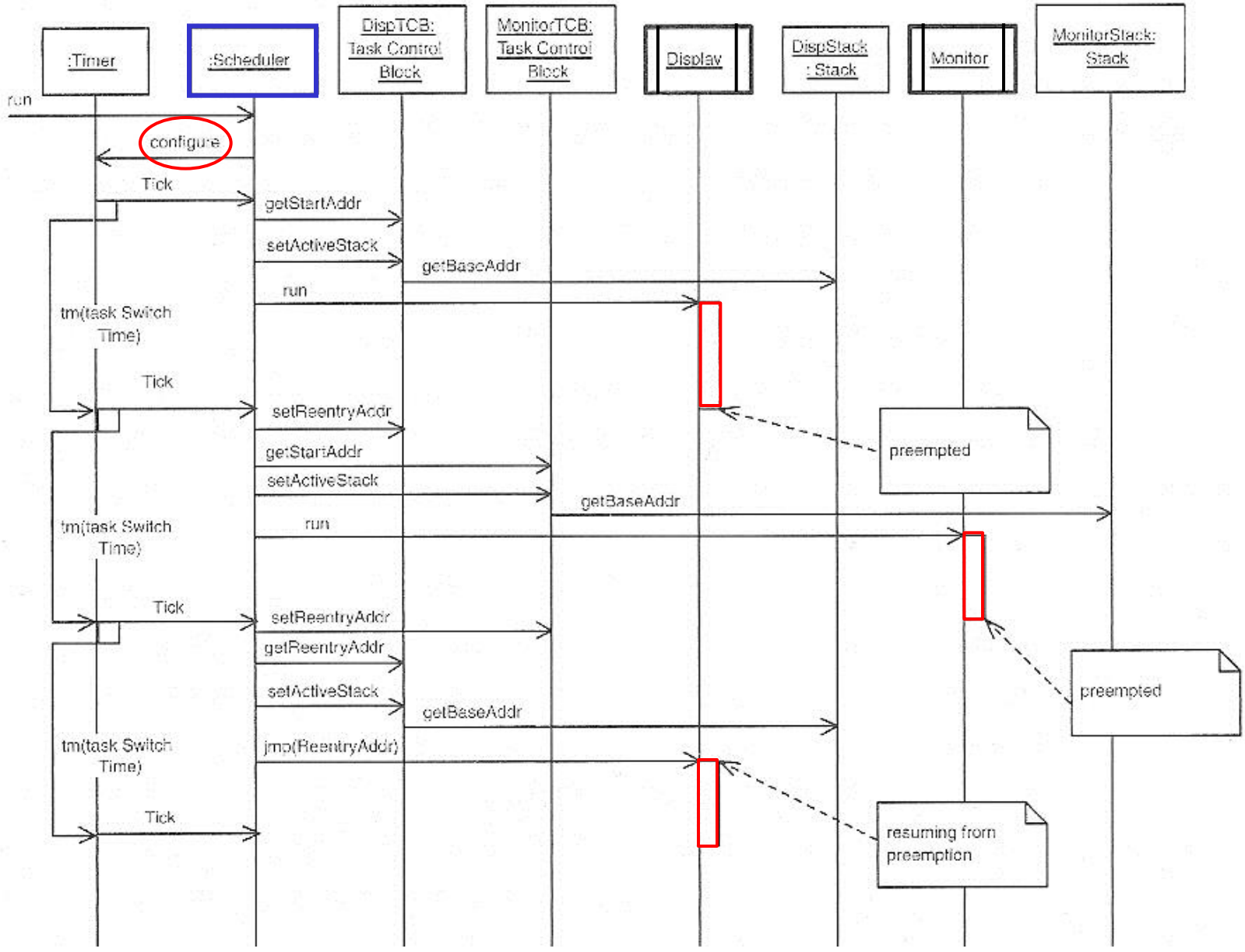
Simplified

Round Robin Pattern Example (1)



a. Structure

Round Robin Pattern Example (2)

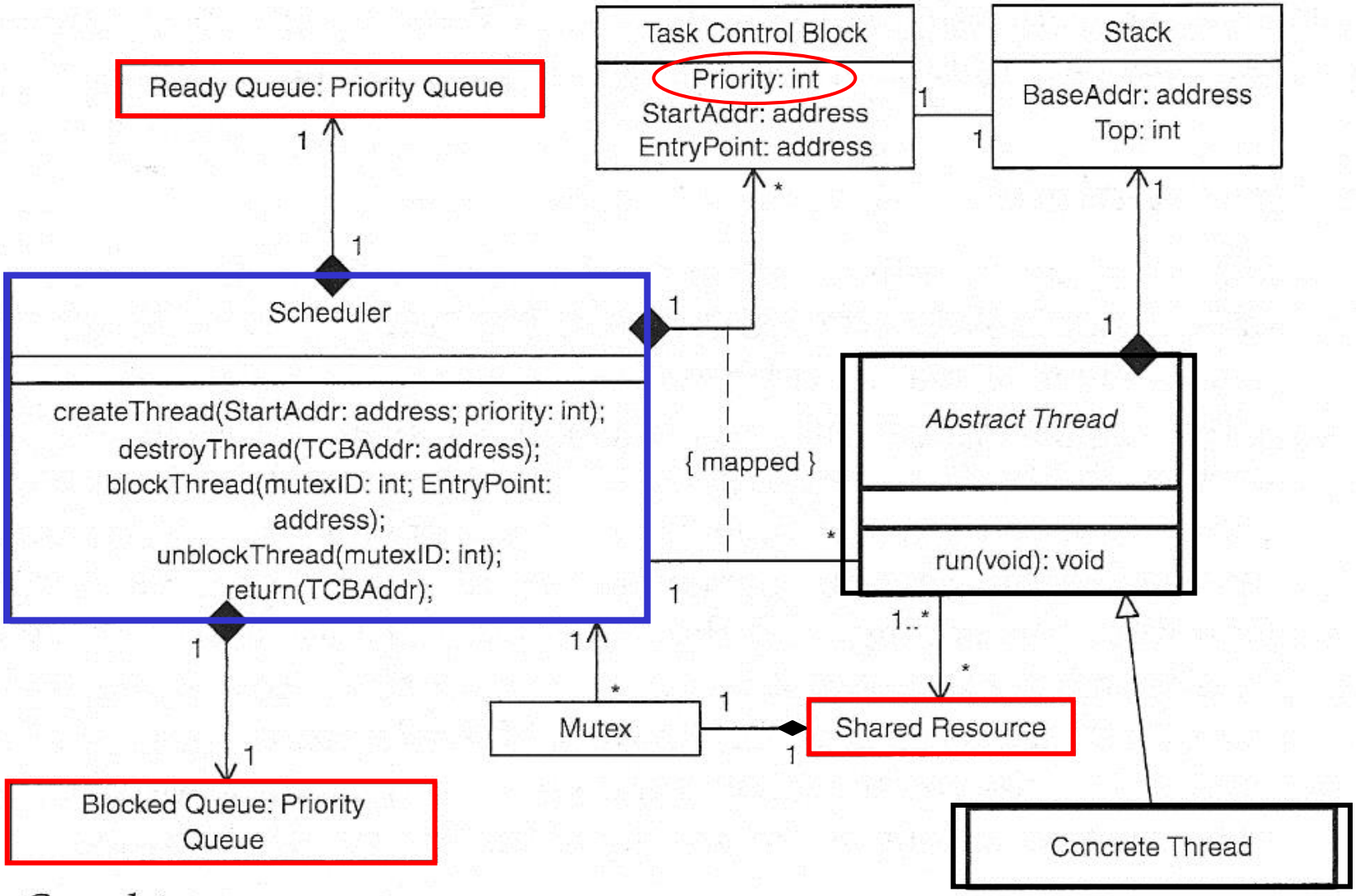


7. Static Priority Pattern

The Static Priority Pattern is the most common approach to scheduling in a real-time system.

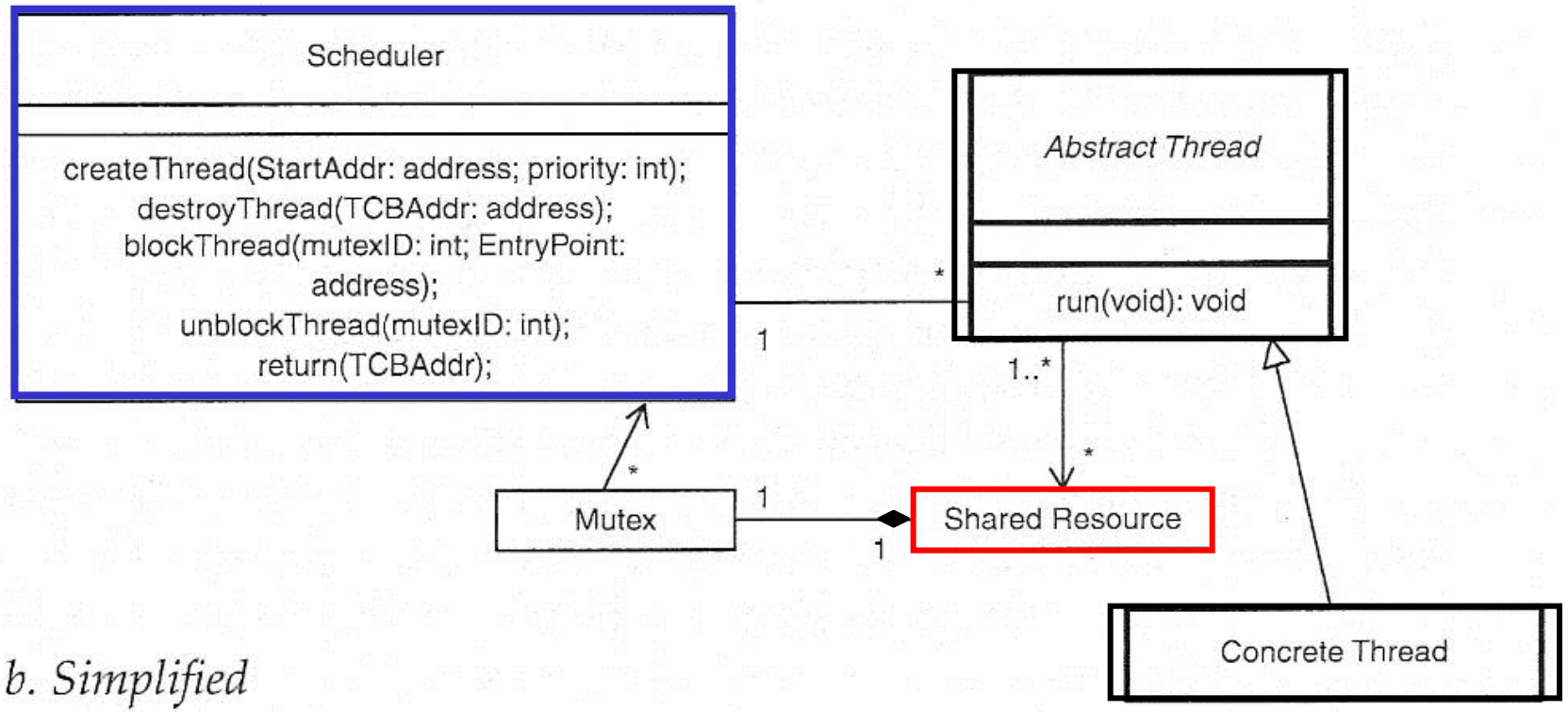
It has the advantage of being simple and scaling fairly well to large numbers of tasks

Static Priority Pattern Structure (1)



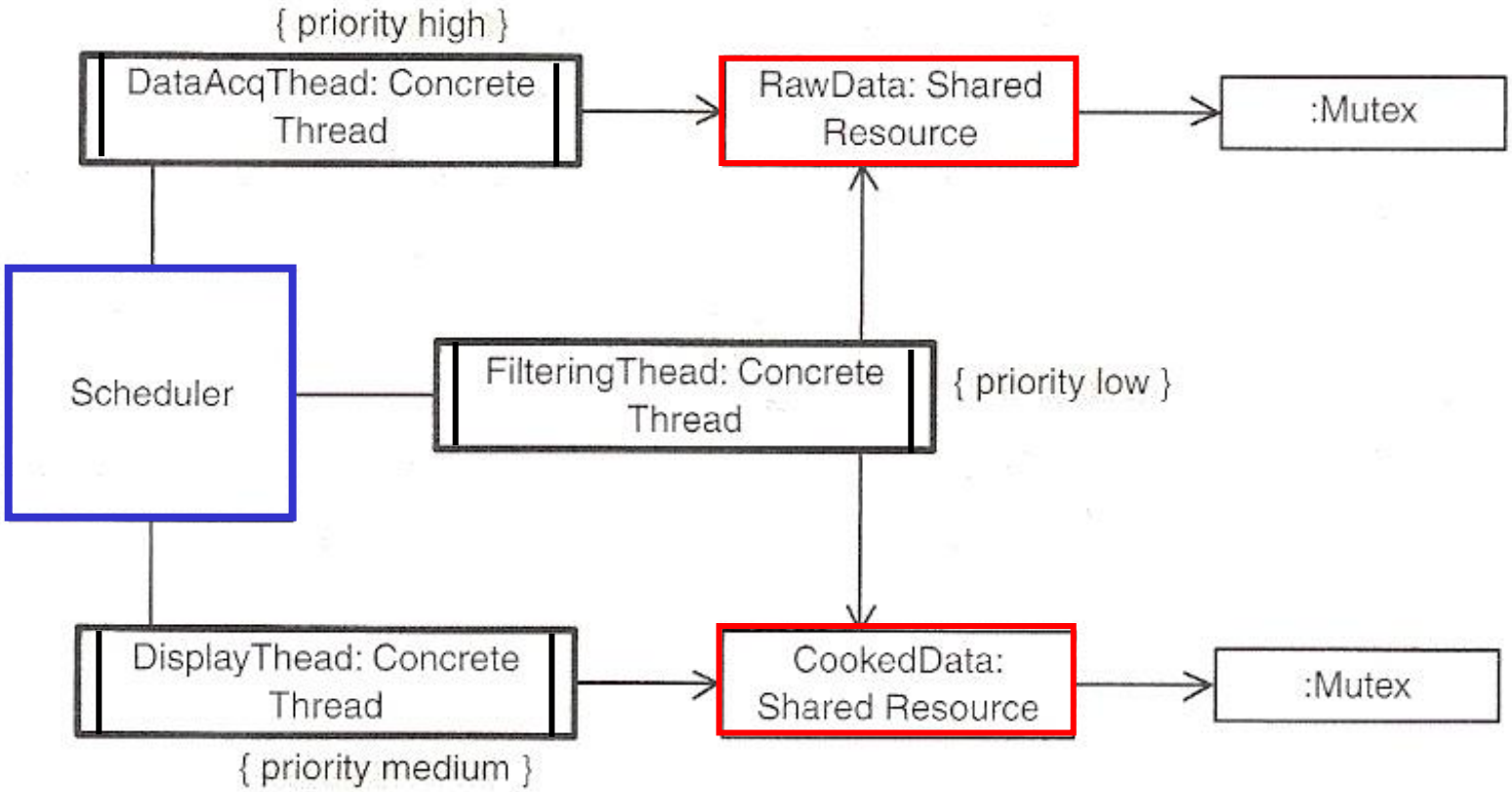
a. Complete

Static Priority Pattern Structure (2)



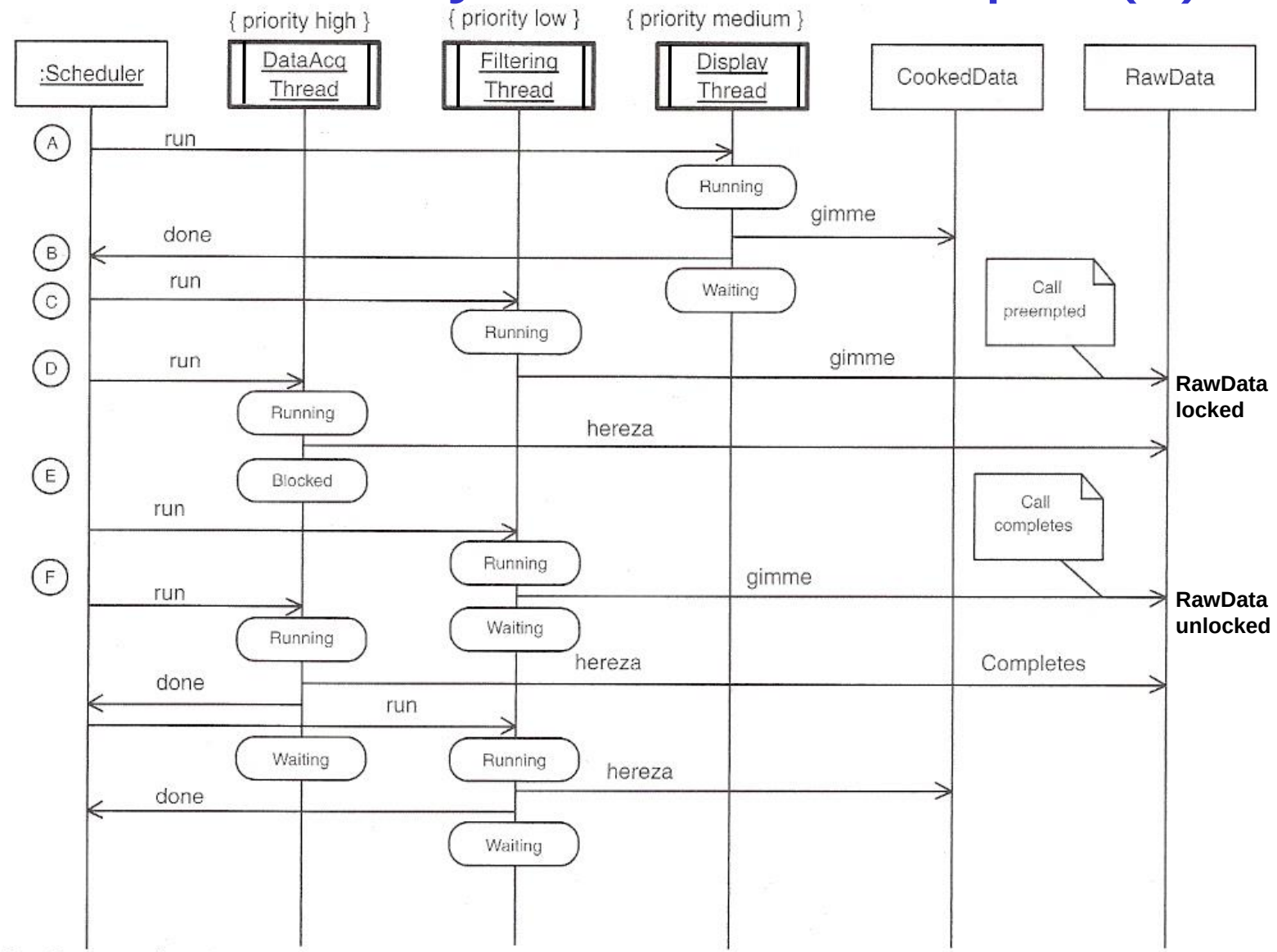
b. Simplified

Static Priority Pattern Example (1)



Structure

Static Priority Pattern Example (2)



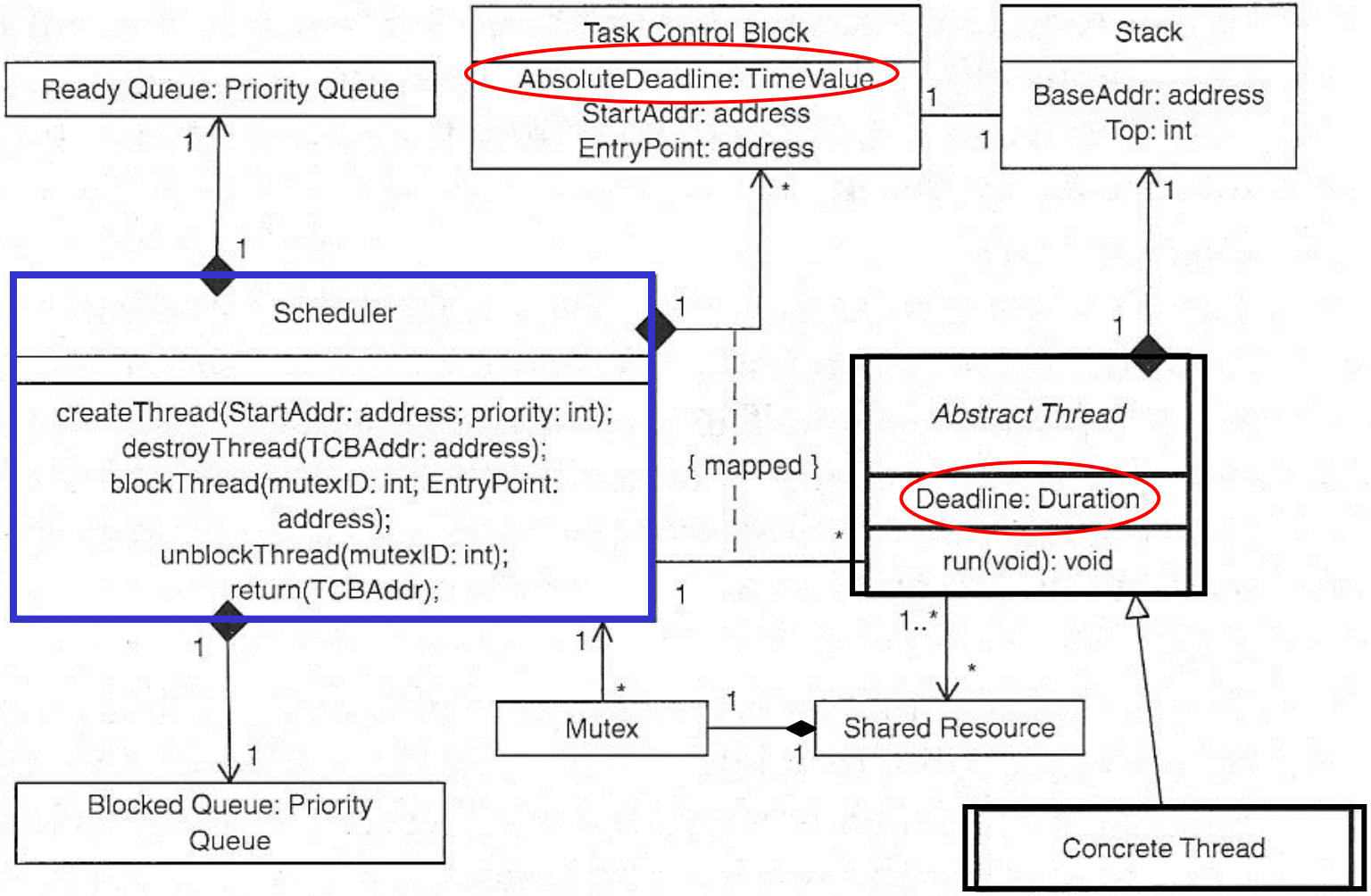
b. Scenario

8. Dynamic Priority Pattern

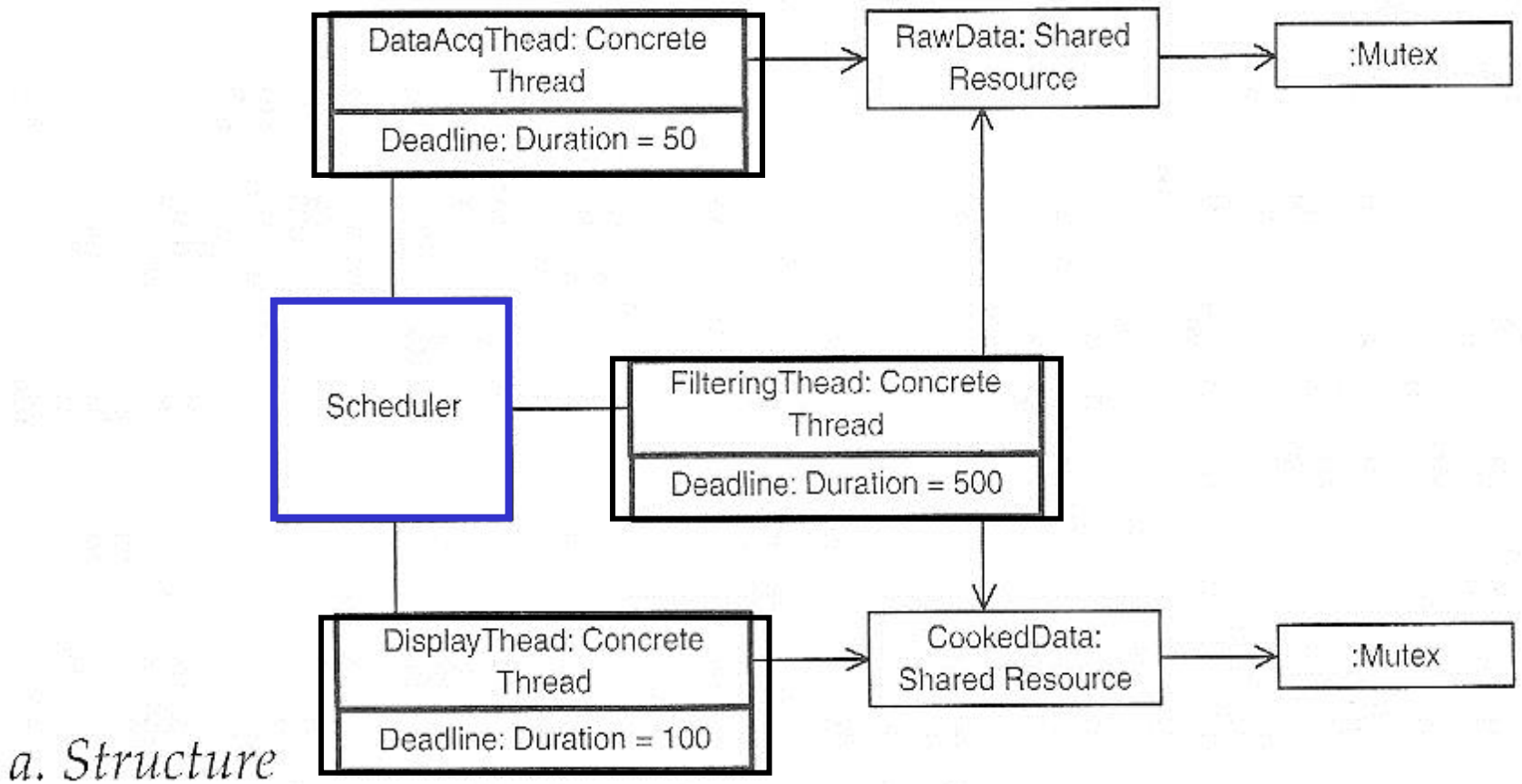
The Dynamic Priority Pattern automatically updates the priority of tasks as they run to reflect changing conditions.

The most common strategy is called Earliest Deadline First.

Dynamic Priority Pattern Structure



Dynamic Priority Pattern Example



Summary

1. Interrupt Pattern
2. Message Queuing Pattern
3. Guarded Call Pattern
4. Rendezvous Pattern
5. Cyclic Executive Pattern
6. Round Robin Pattern
7. Static Priority Pattern
8. Dynamic Priority Pattern