

Java Monitor Objects: Coordination Example Visualization



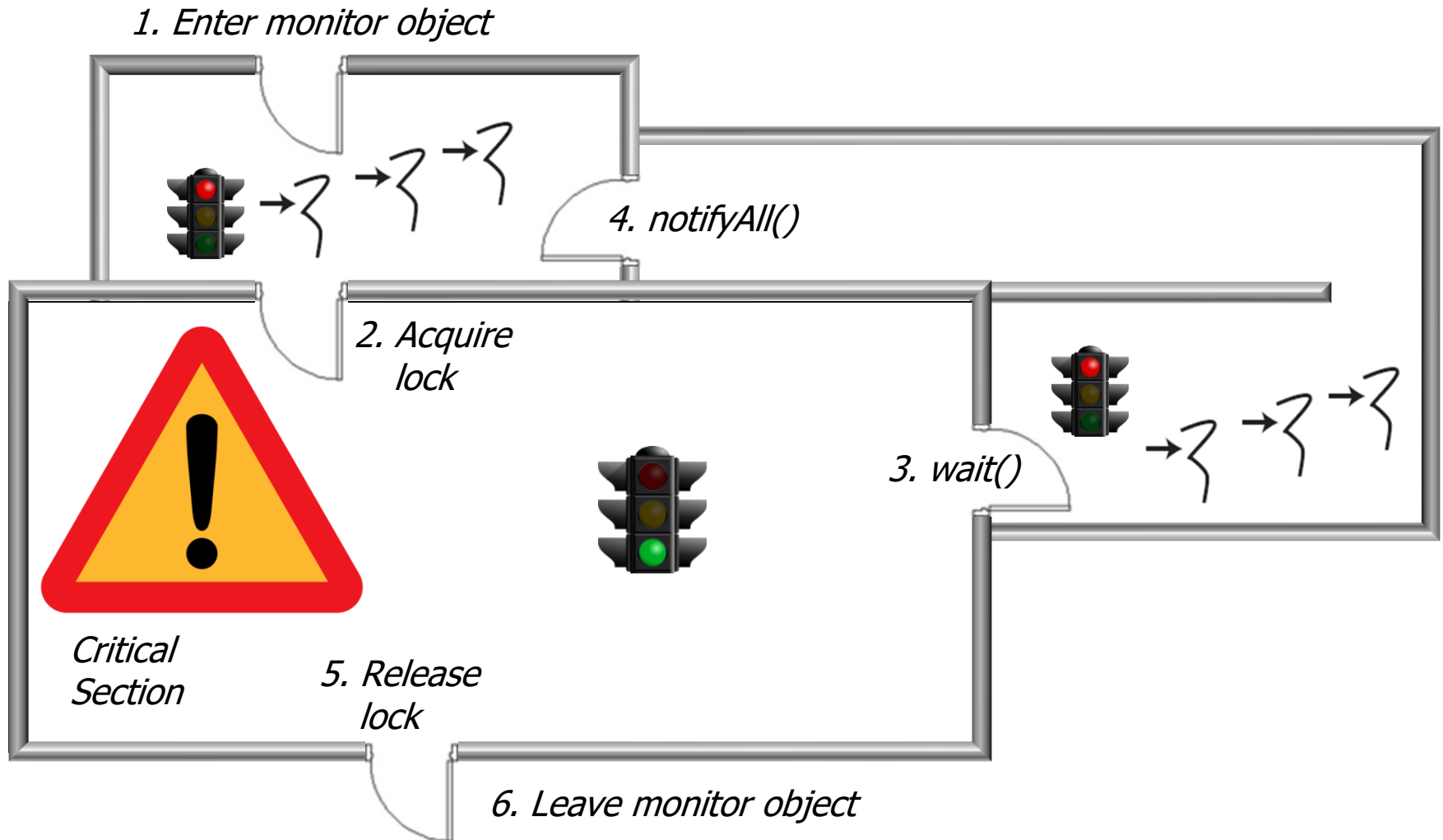
Douglas C. Schmidt
d.schmidt@vanderbilt.edu
www.dre.vanderbilt.edu/~schmidt

**Institute for Software
Integrated Systems
Vanderbilt University
Nashville, Tennessee, USA**



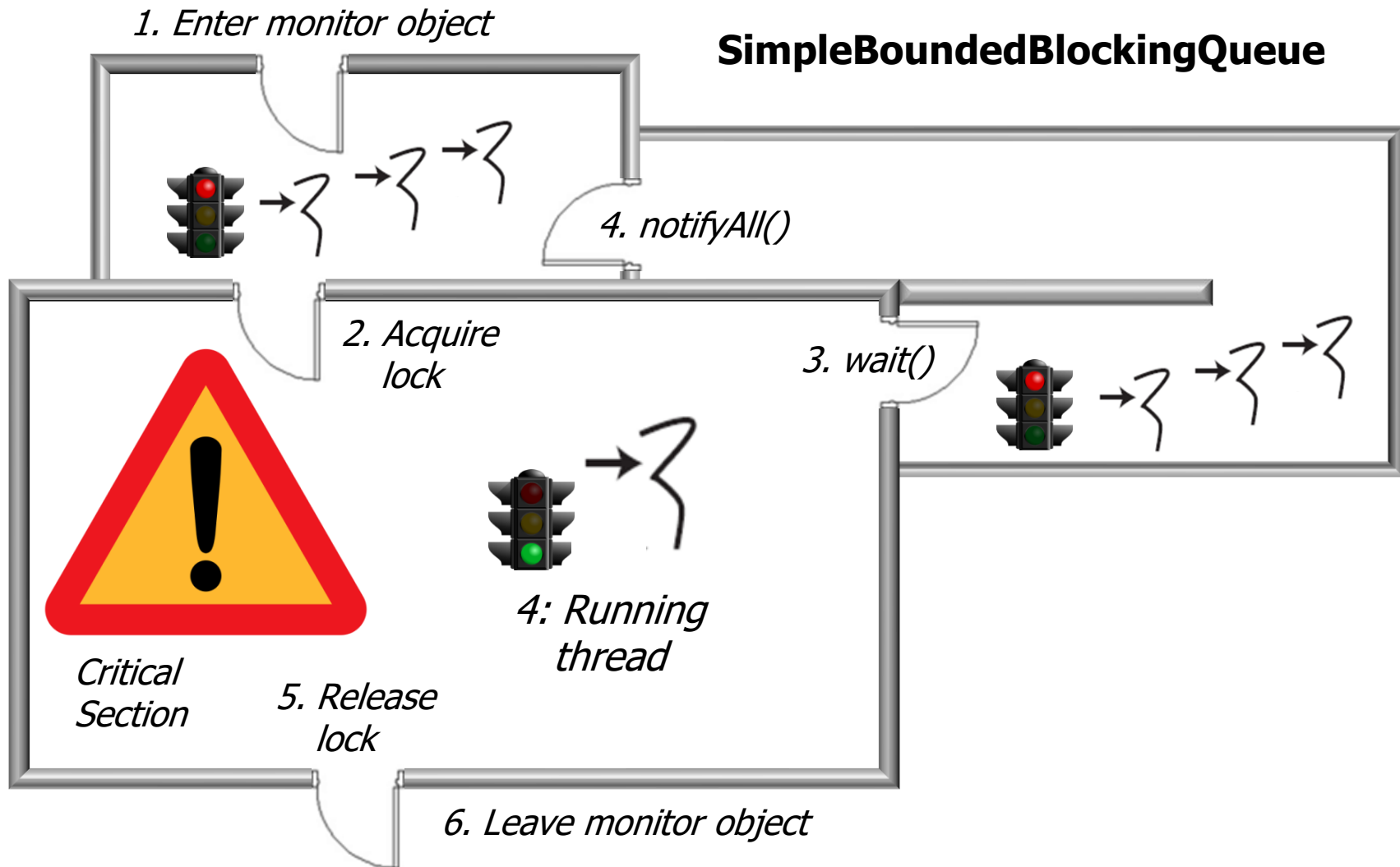
Learning Objectives in this Part of the Lesson

- Learn how to fix a buggy concurrent Java program using Java's wait & notify mechanisms, which provide *coordination*
- Visualize how Java monitor objects can be used to ensure mutual exclusion & coordination between threads running in a concurrent program



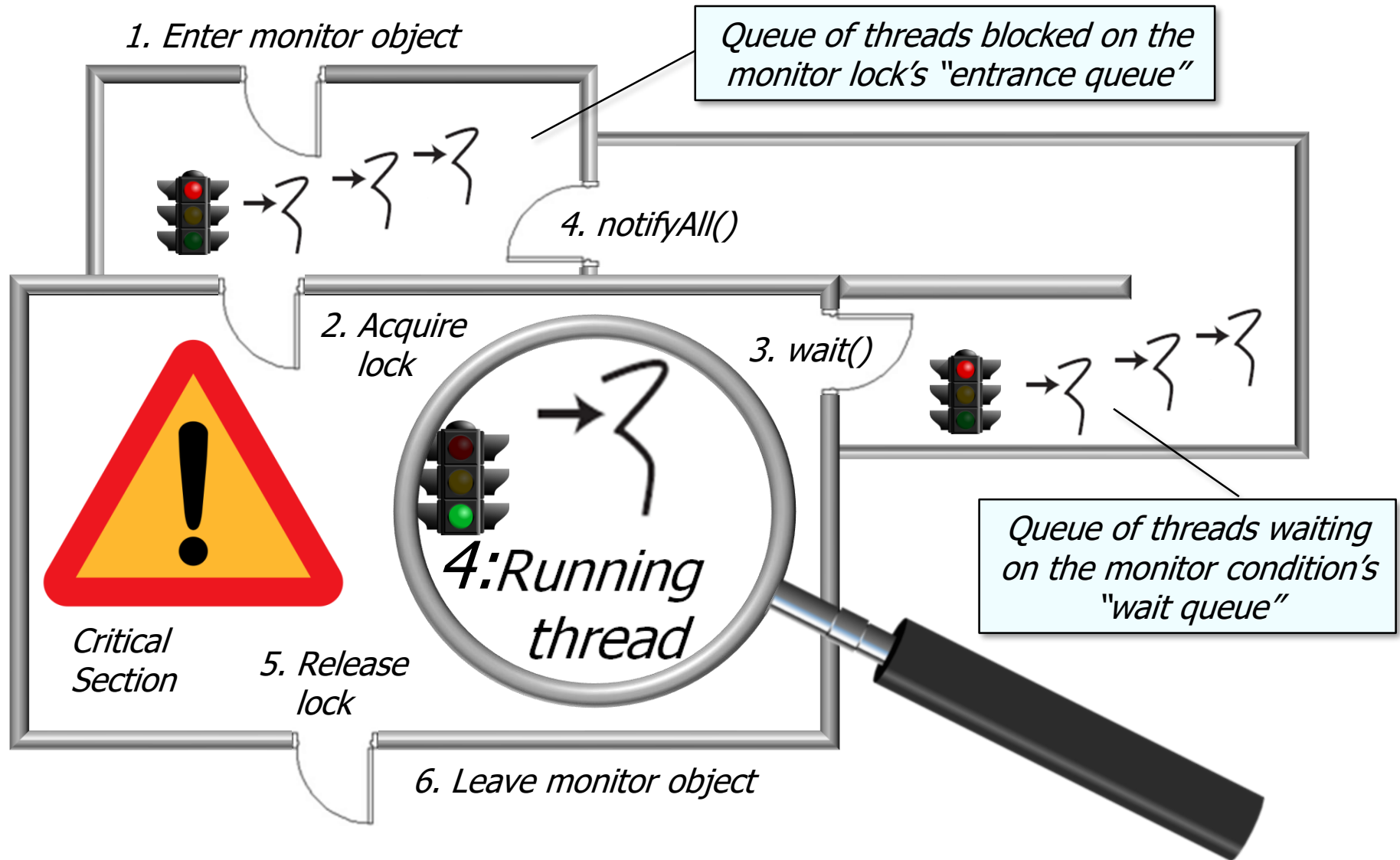
Visual Analysis of the SimpleBlockingBounded Queue Example

Visual Analysis of SimpleBoundedBlockingQueue

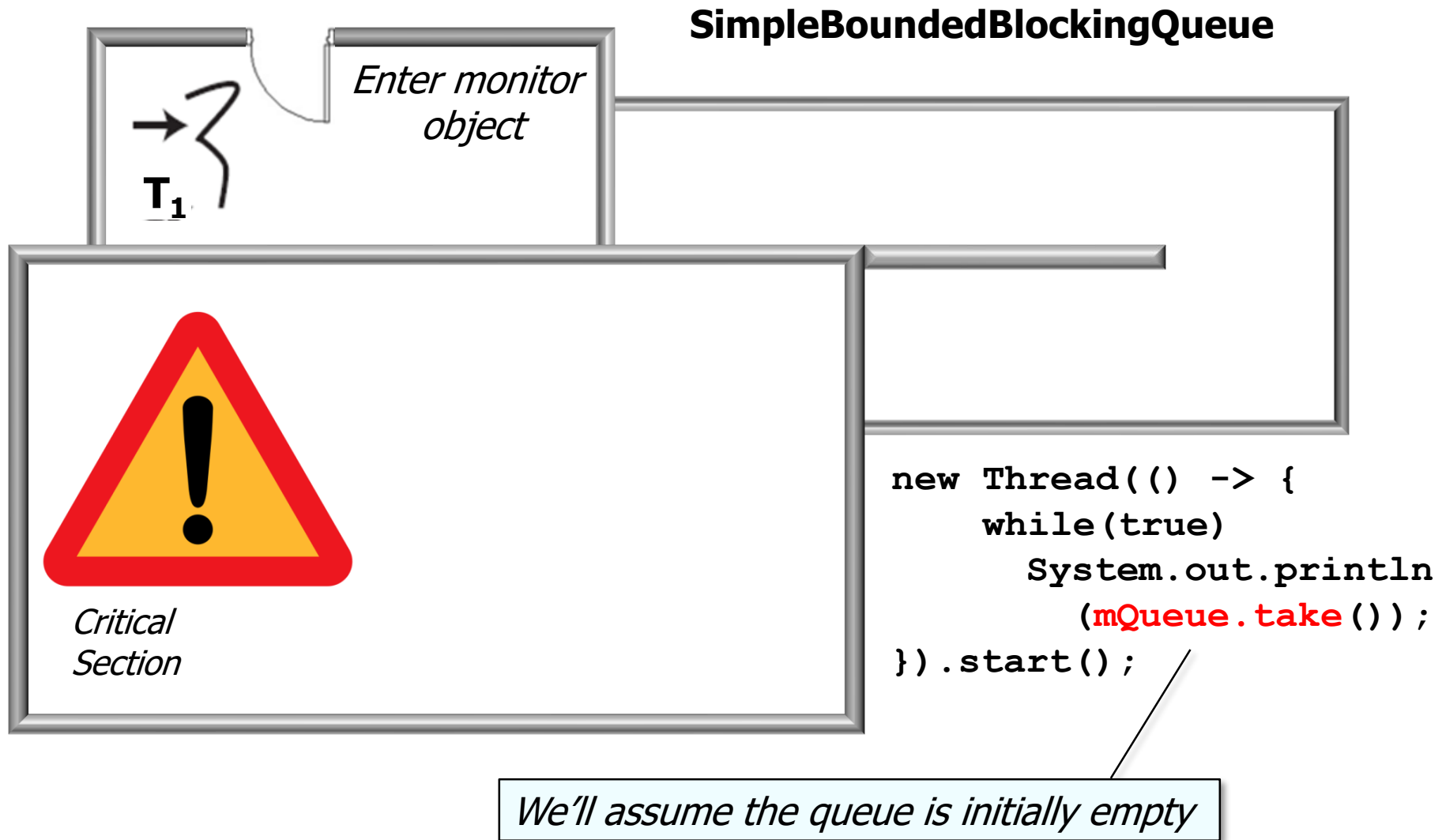


See github.com/douglasraigschmidt/POSA/tree/master/ex/M3/Queues/SimpleBoundedBlockingQueue

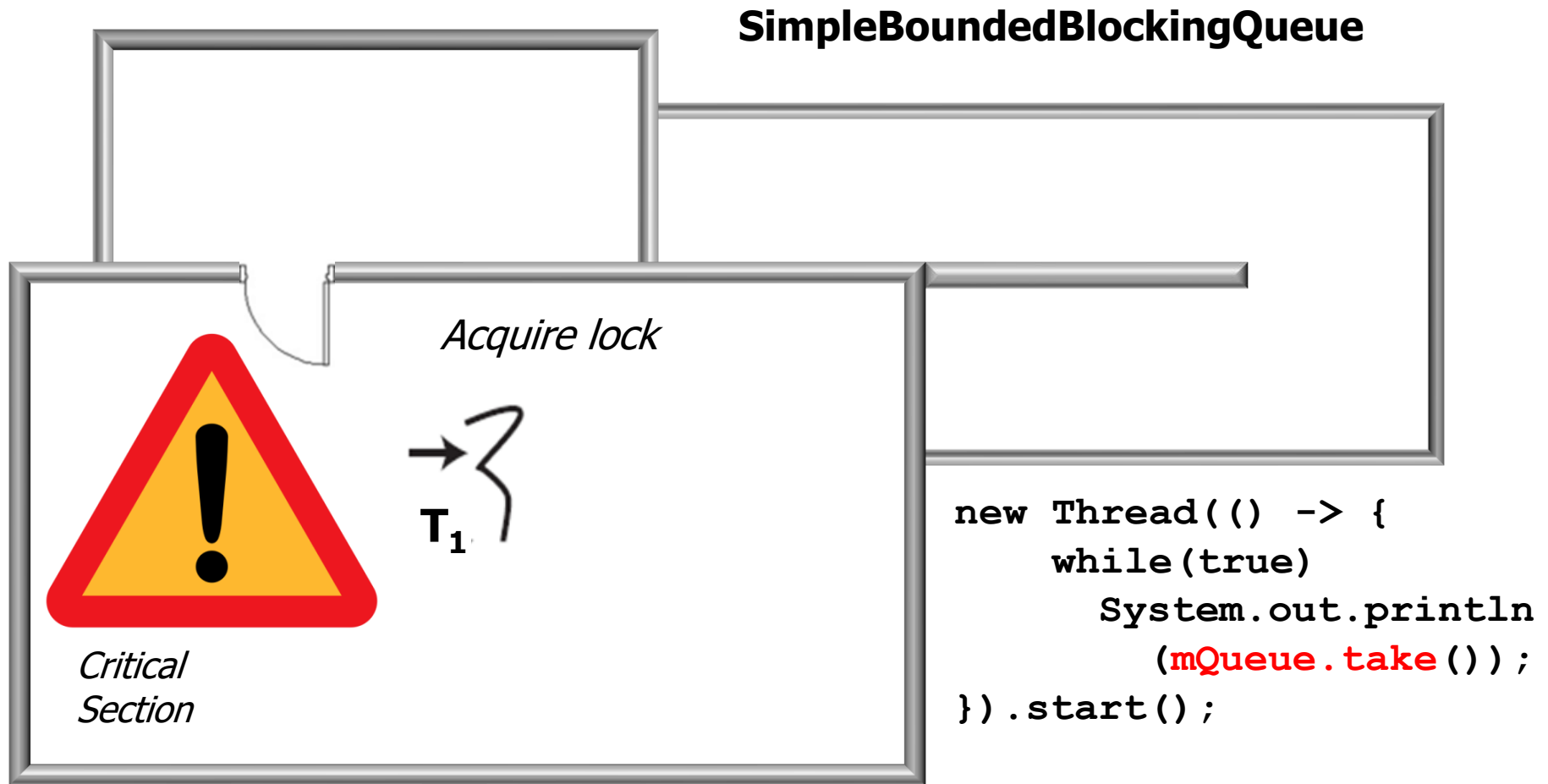
Visual Analysis of SimpleBoundedBlockingQueue



Visual Analysis of SimpleBoundedBlockingQueue



Visual Analysis of SimpleBoundedBlockingQueue



Visual Analysis of SimpleBoundedBlockingQueue

SimpleBoundedBlockingQueue



*Critical
Section*

→ {
T₁

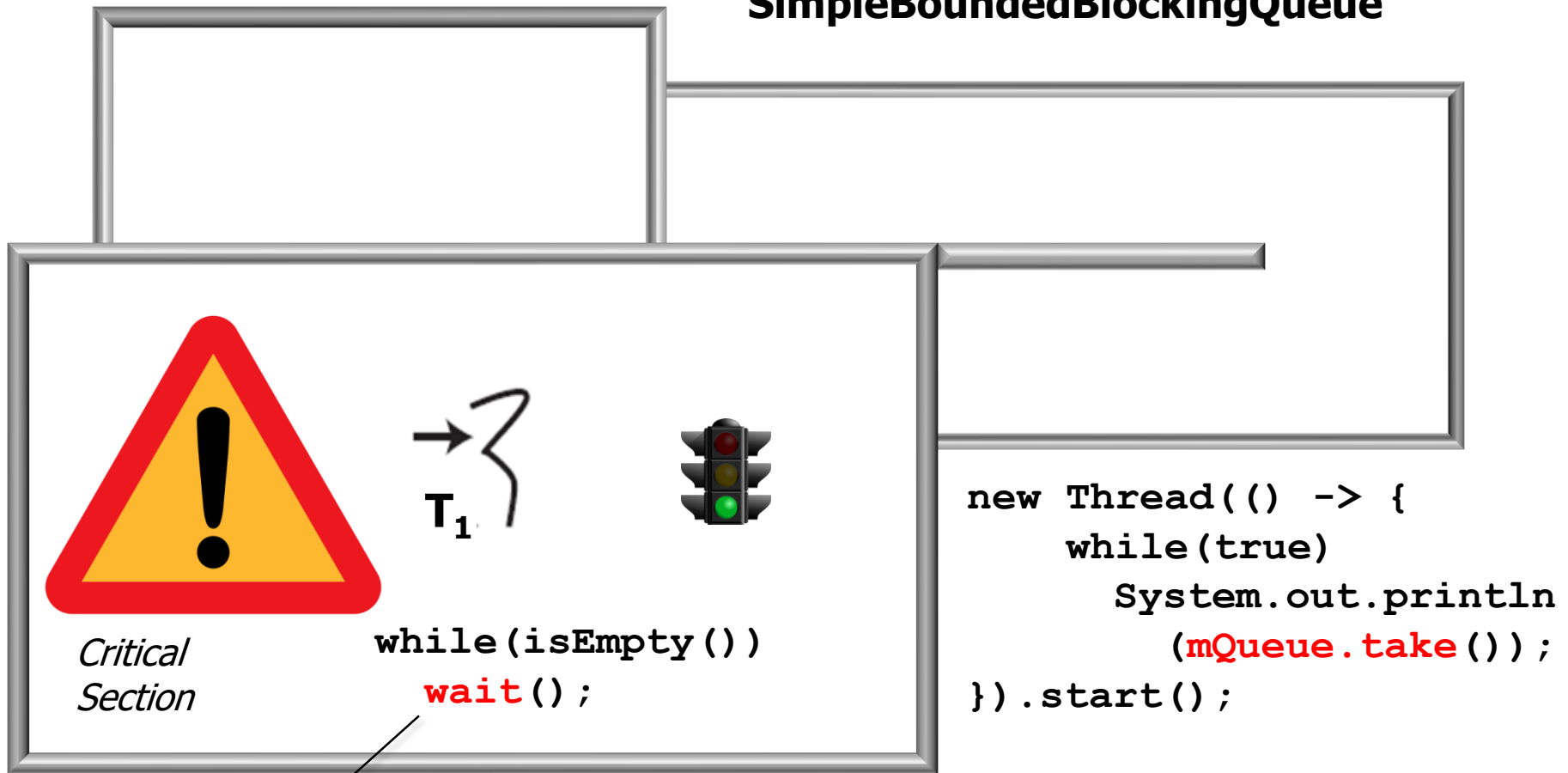


```
while (isEmpty())  
    wait();
```

```
new Thread(() -> {  
    while (true)  
        System.out.println  
            (mQueue.take());  
}).start();
```

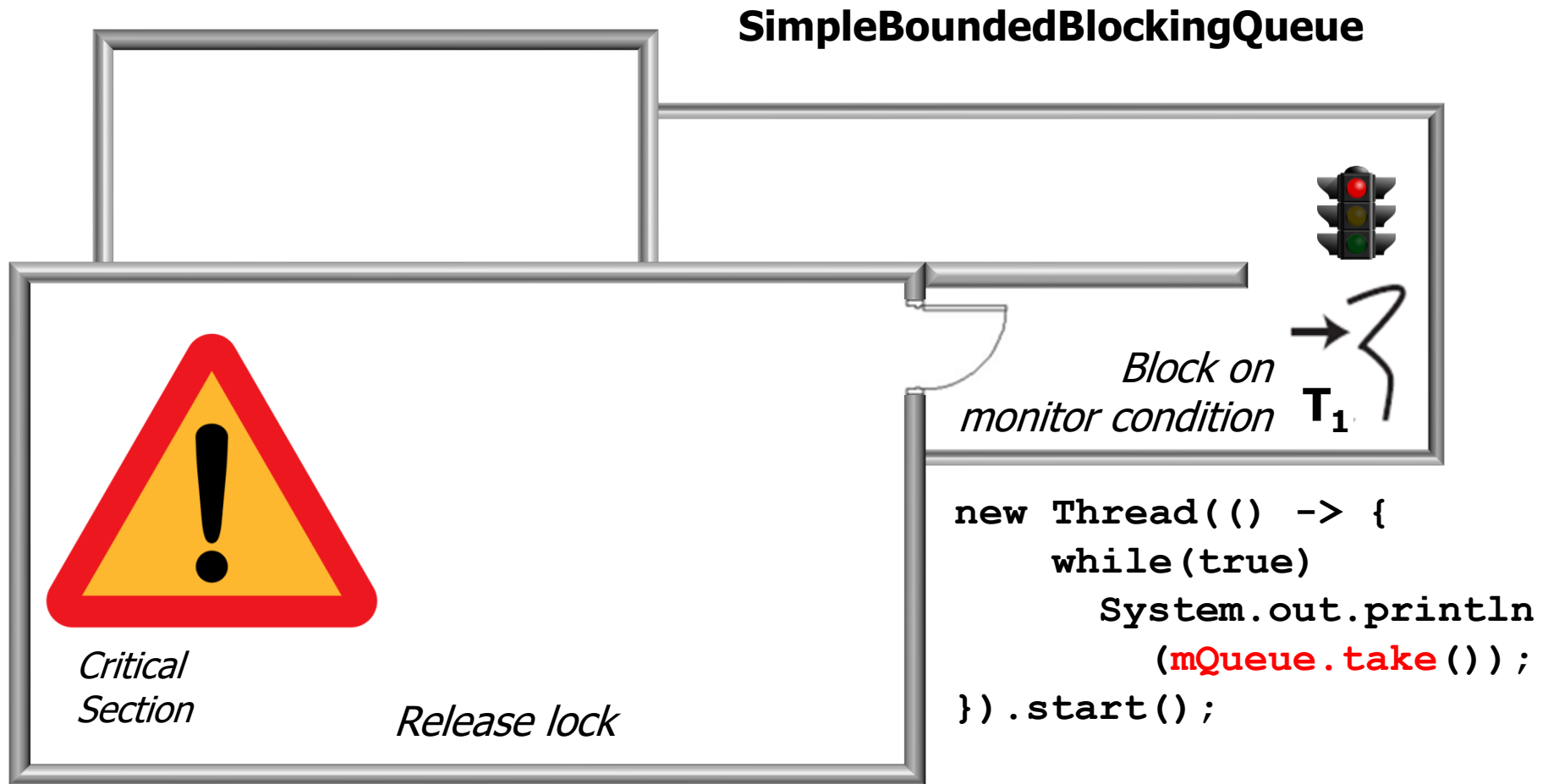
Visual Analysis of SimpleBoundedBlockingQueue

SimpleBoundedBlockingQueue

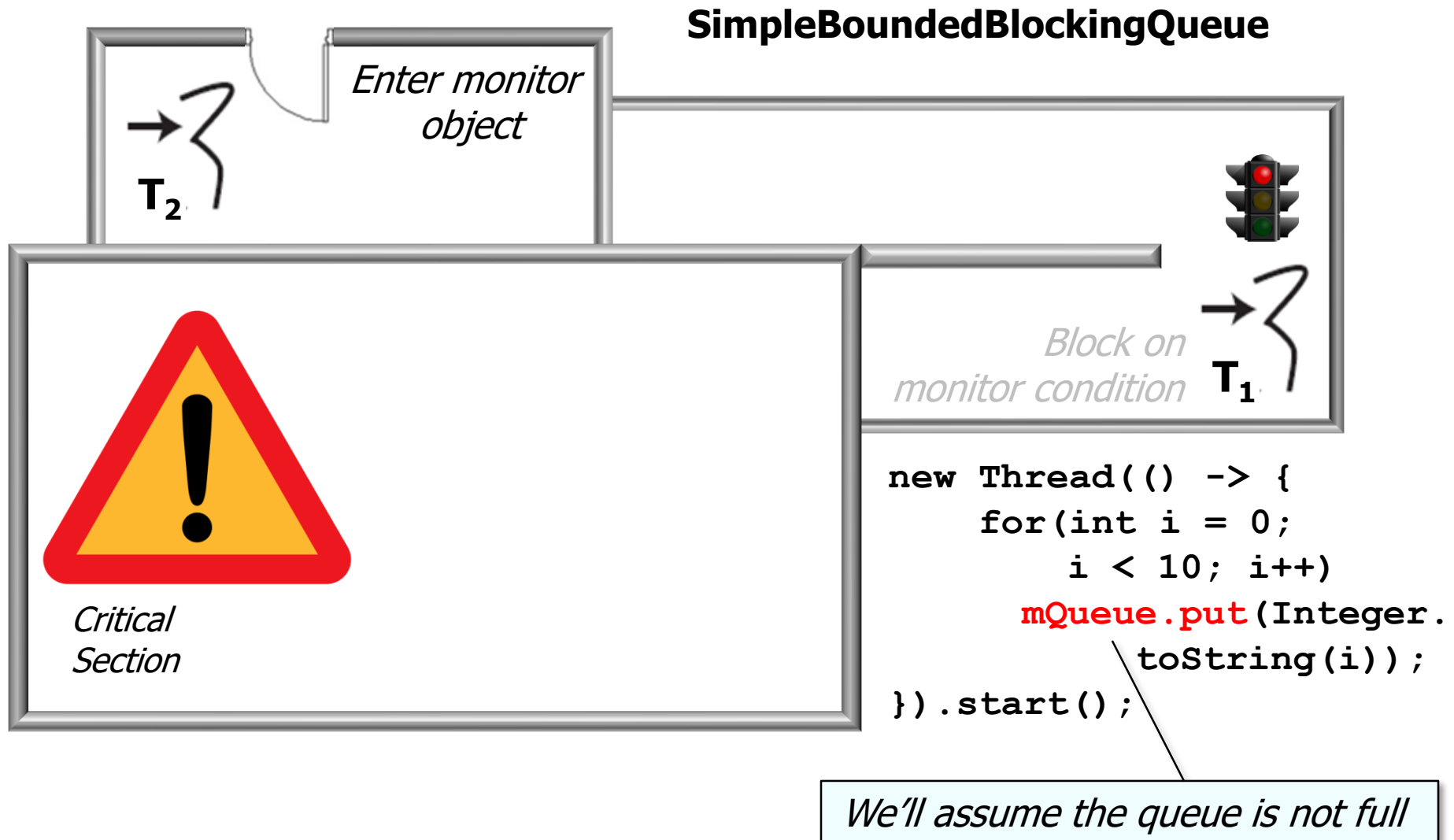


Calling wait() atomically releases the monitor lock & puts the calling thread to sleep

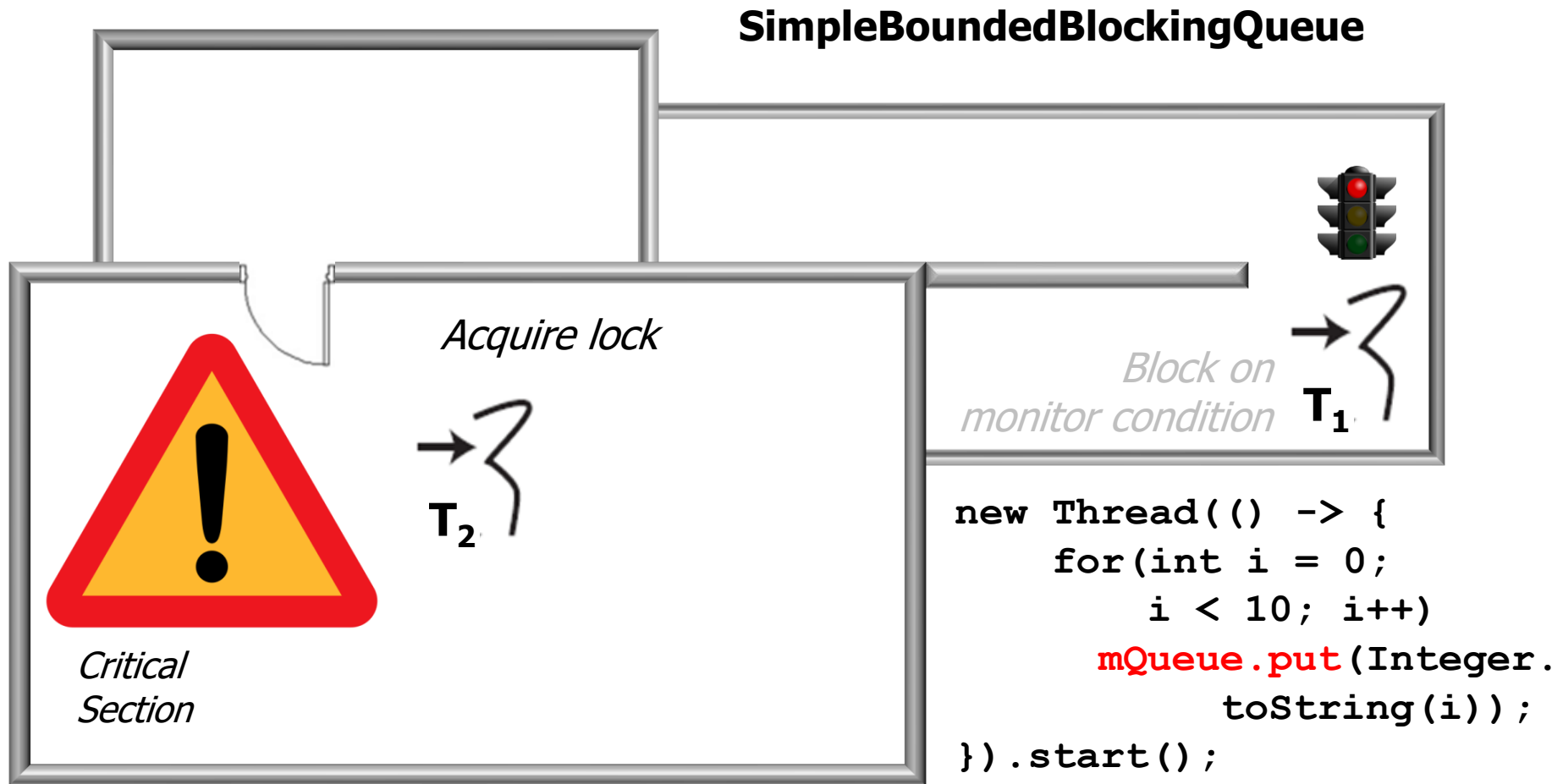
Visual Analysis of SimpleBoundedBlockingQueue



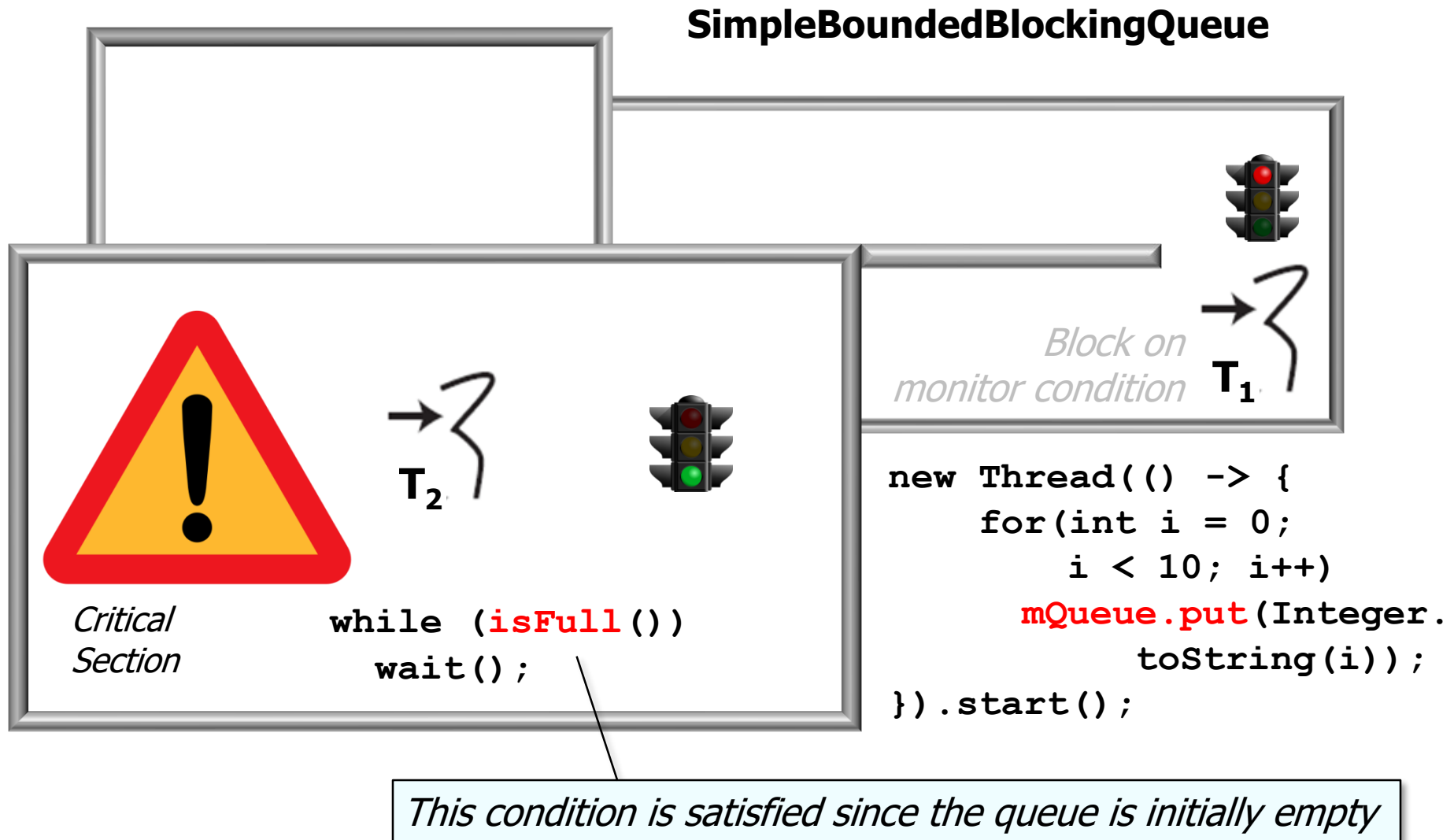
Visual Analysis of SimpleBoundedBlockingQueue



Visual Analysis of SimpleBoundedBlockingQueue

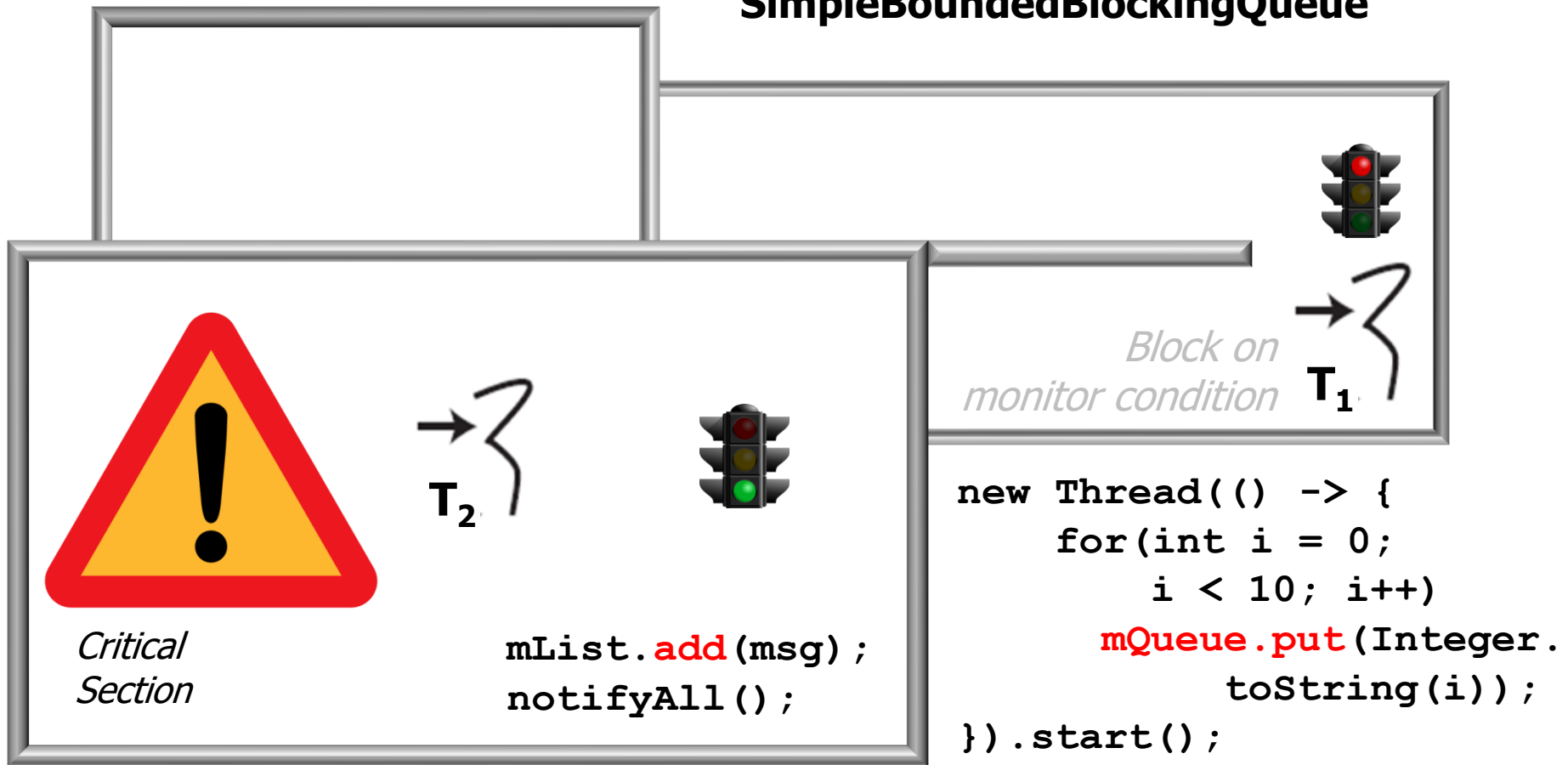


Visual Analysis of SimpleBoundedBlockingQueue

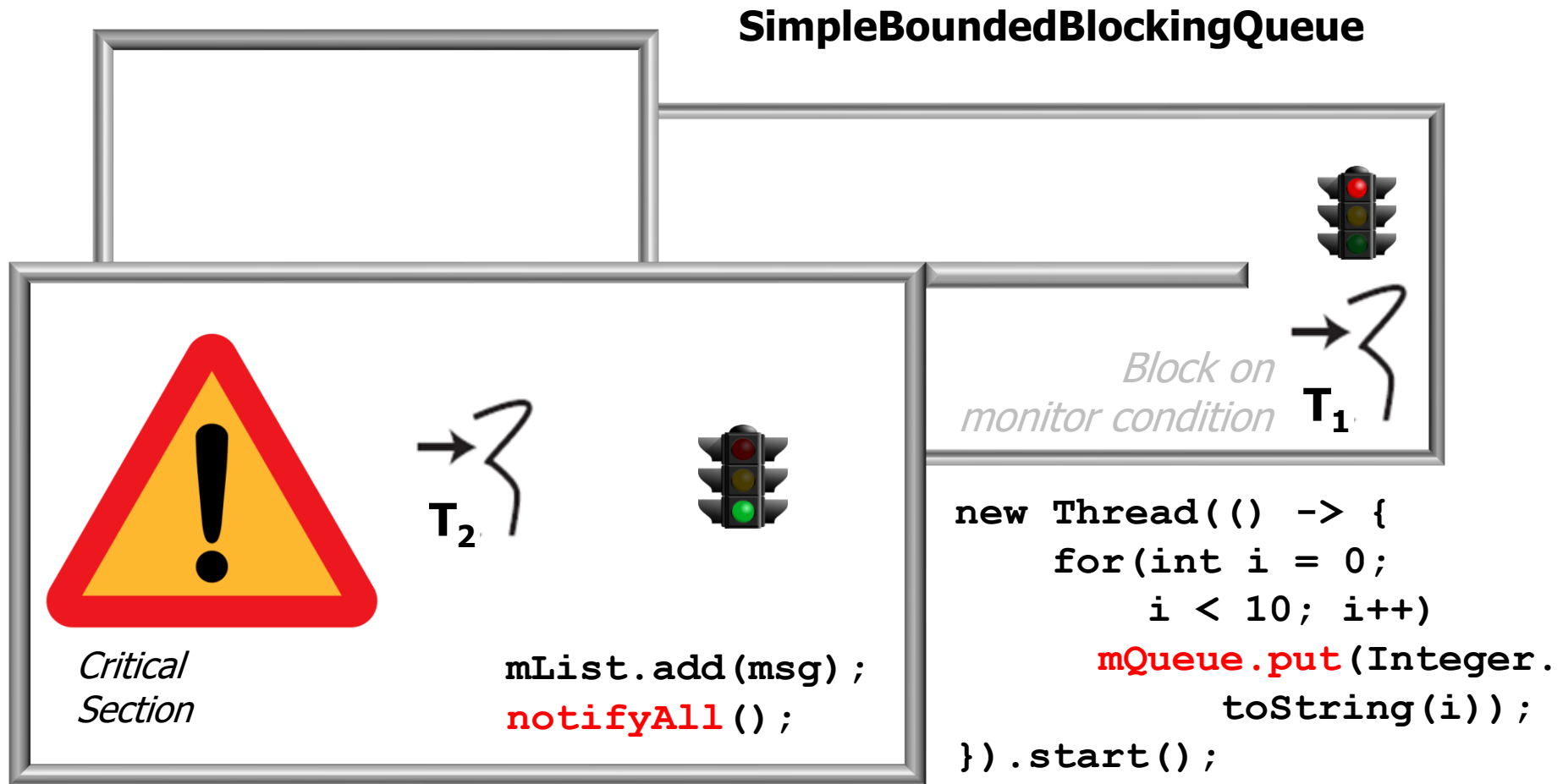


Visual Analysis of SimpleBoundedBlockingQueue

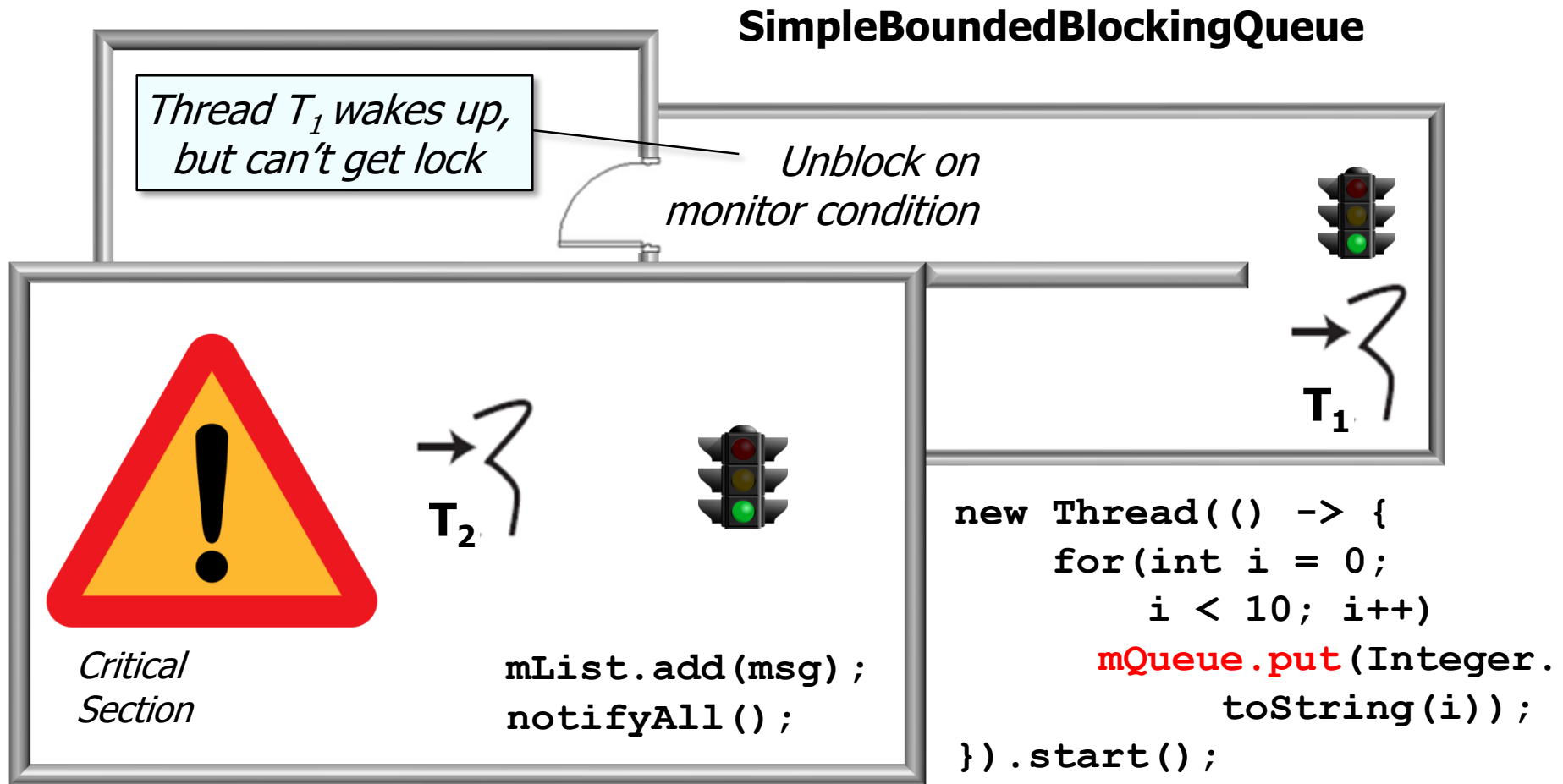
SimpleBoundedBlockingQueue



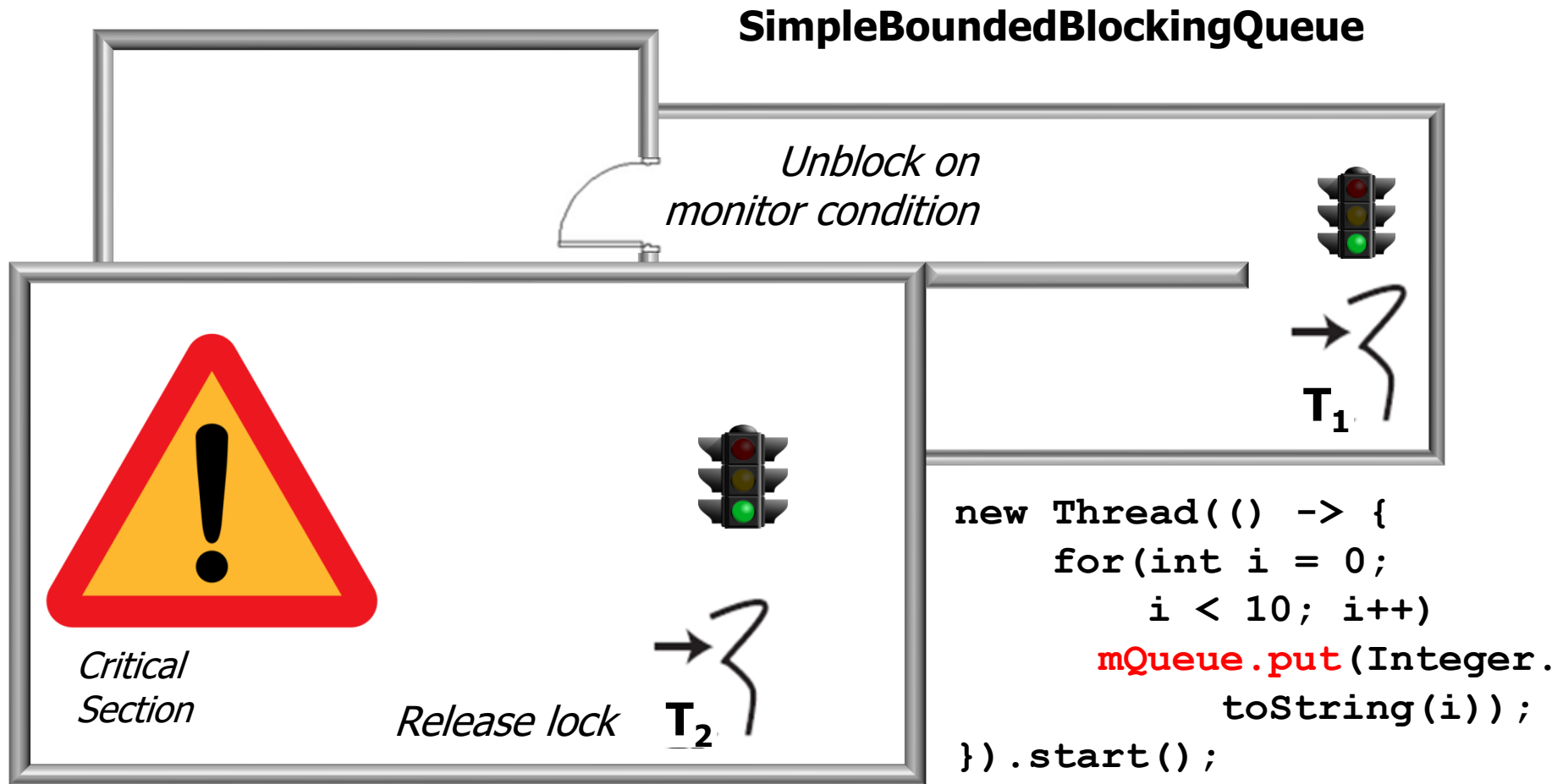
Visual Analysis of SimpleBoundedBlockingQueue



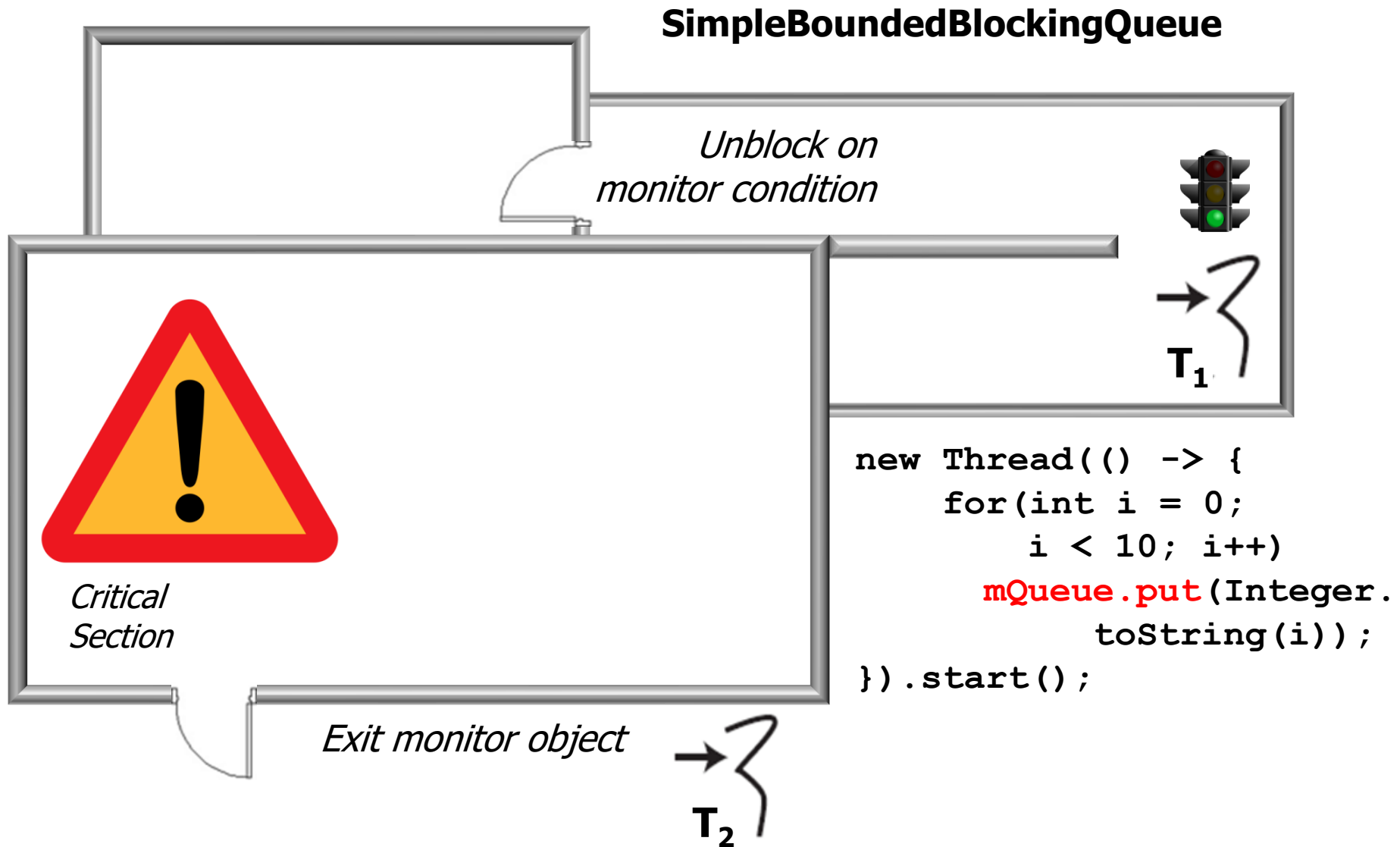
Visual Analysis of SimpleBoundedBlockingQueue



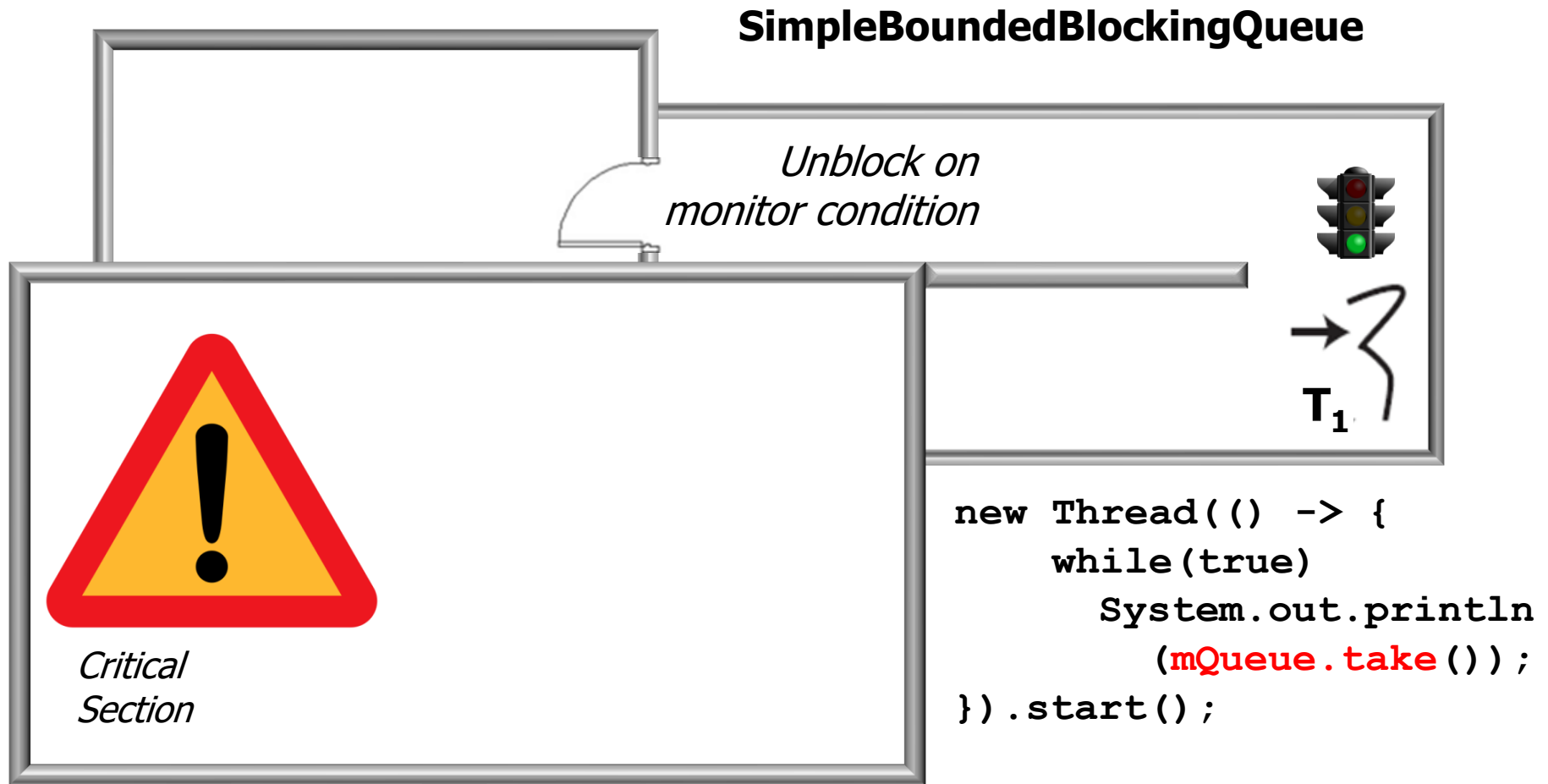
Visual Analysis of SimpleBoundedBlockingQueue



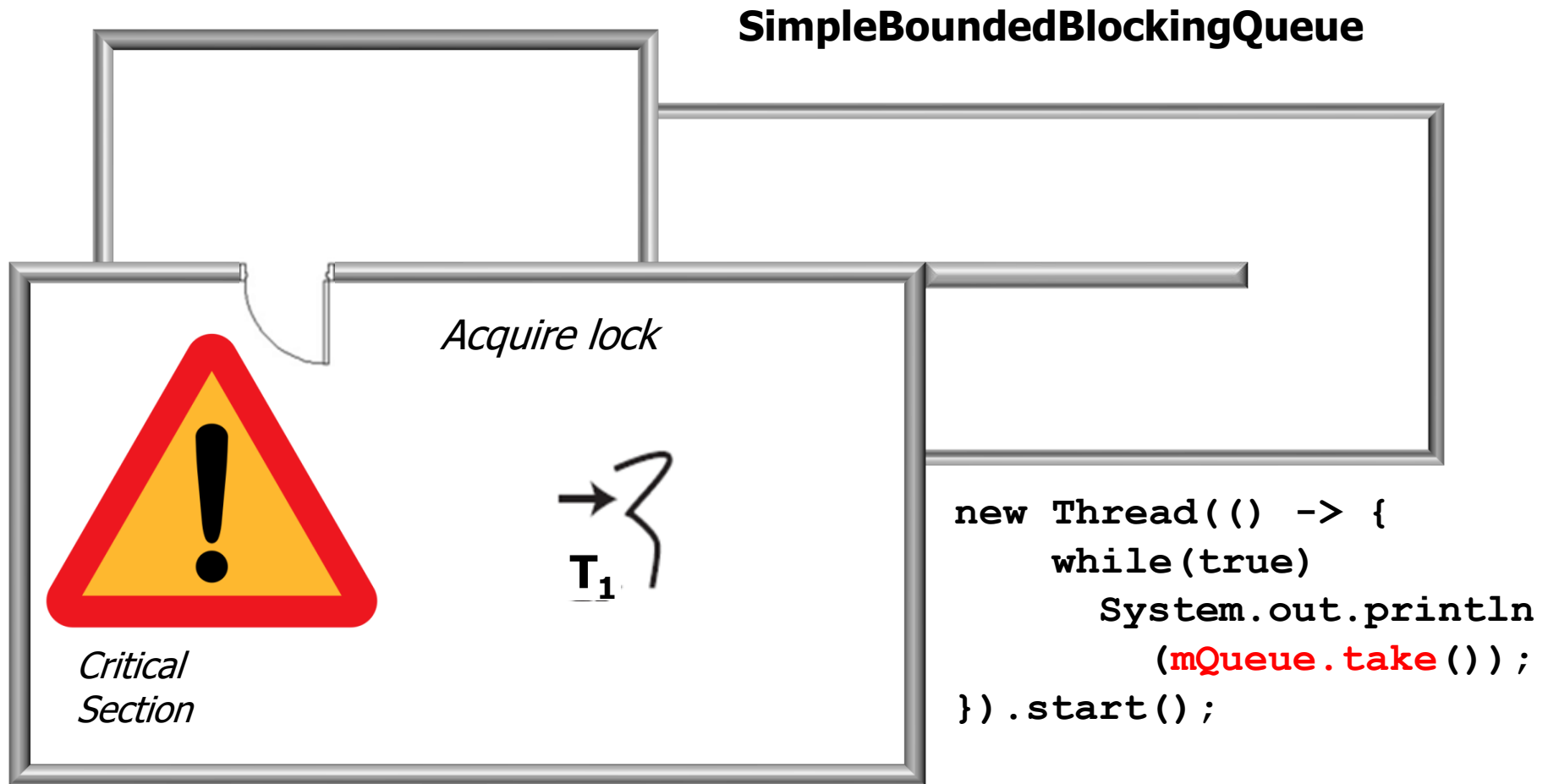
Visual Analysis of SimpleBoundedBlockingQueue



Visual Analysis of SimpleBoundedBlockingQueue



Visual Analysis of SimpleBoundedBlockingQueue



Visual Analysis of SimpleBoundedBlockingQueue

SimpleBoundedBlockingQueue



*Critical
Section*

`while(isEmpty())
wait();`

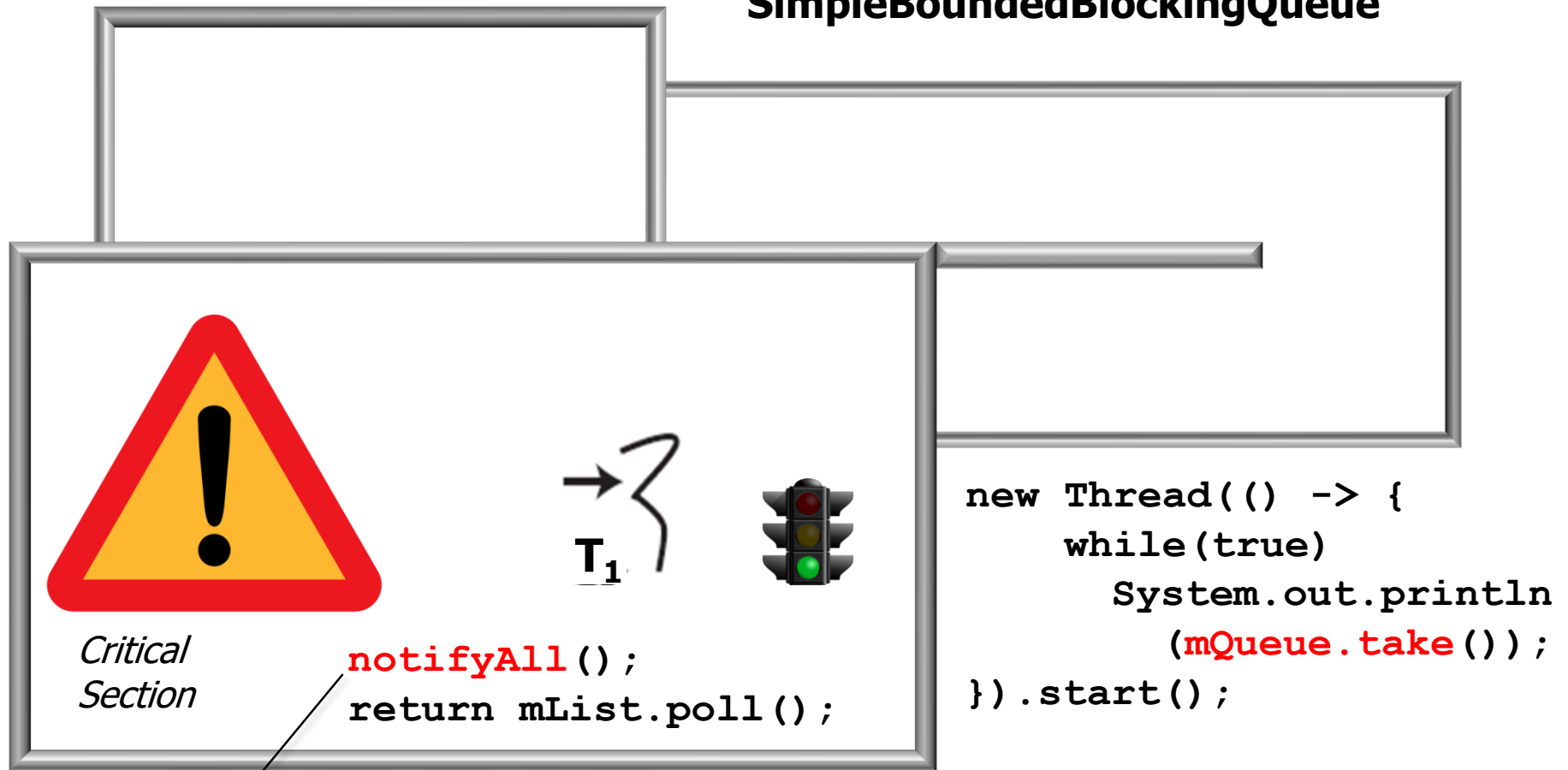
→ {
T₁

```
new Thread(() -> {  
    while(true)  
        System.out.println  
            (mQueue.take());  
}).start();
```

This condition is satisfied since the queue is no longer empty

Visual Analysis of SimpleBoundedBlockingQueue

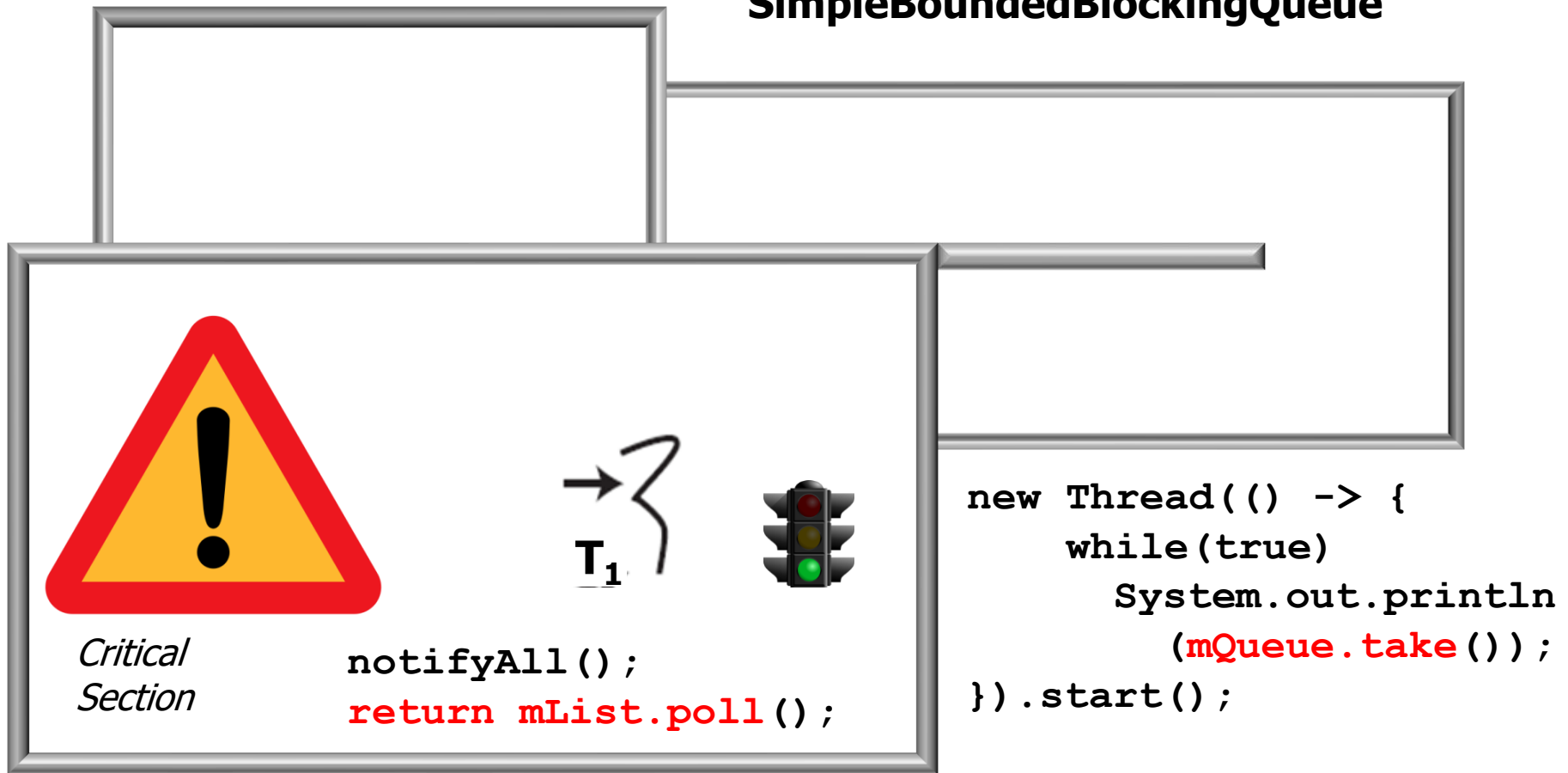
SimpleBoundedBlockingQueue



Calling notifyAll() before removing/returning the front item in the queue is ok since the monitor lock is held & only one method can be in the monitor object

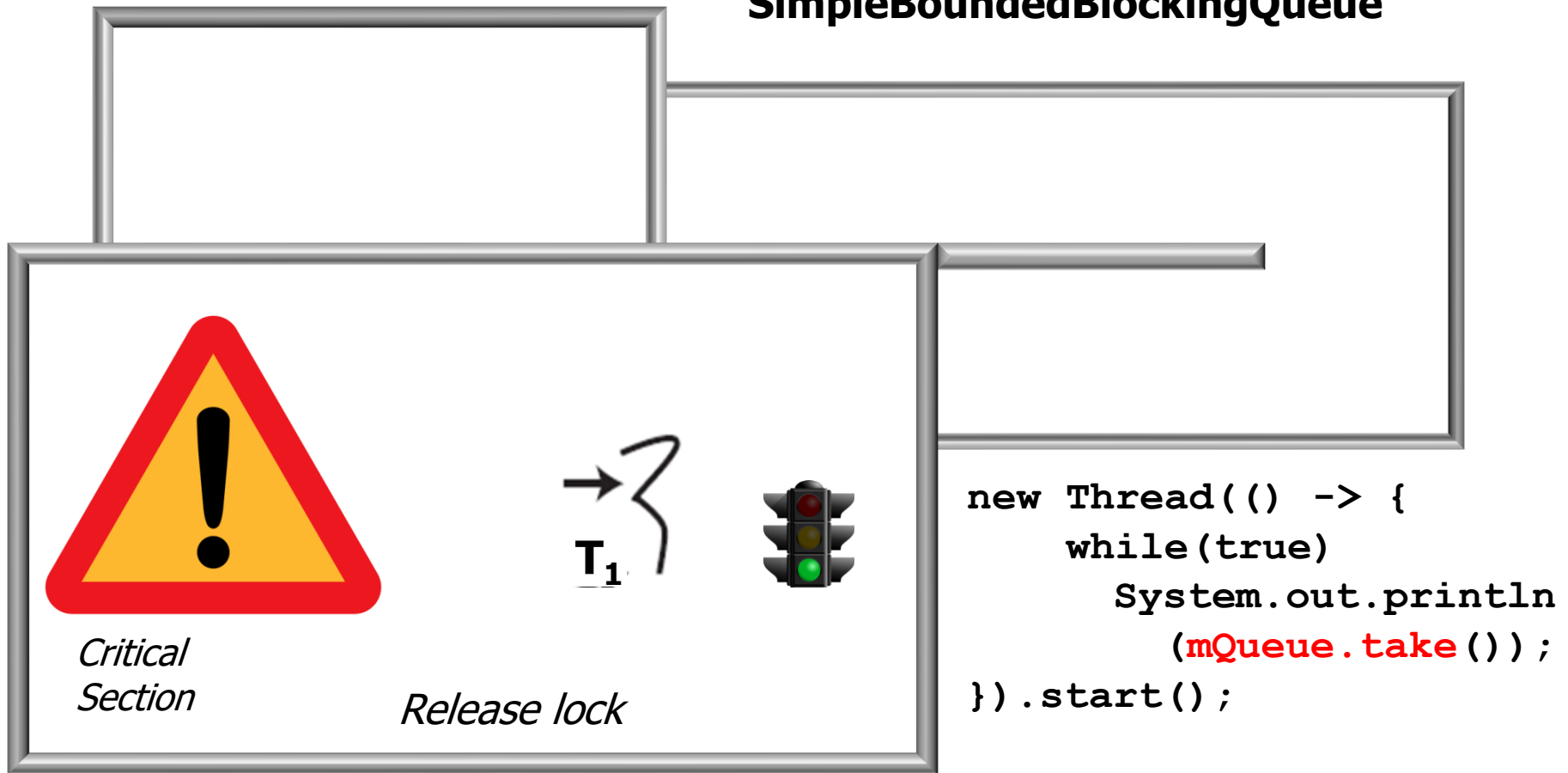
Visual Analysis of SimpleBoundedBlockingQueue

SimpleBoundedBlockingQueue

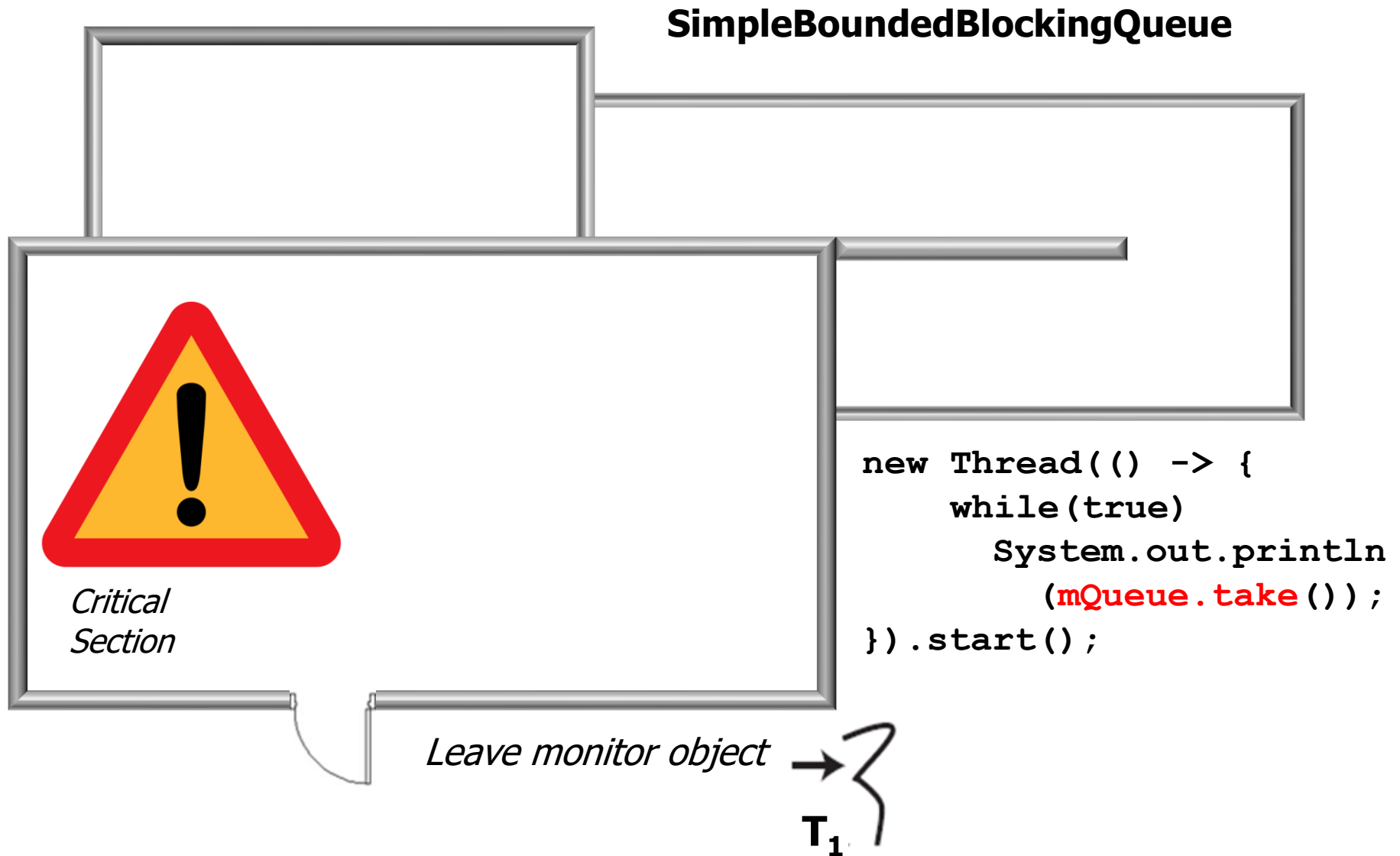


Visual Analysis of SimpleBoundedBlockingQueue

SimpleBoundedBlockingQueue



Visual Analysis of SimpleBoundedBlockingQueue



End of Java Monitor Object: Coordination Example Visualization