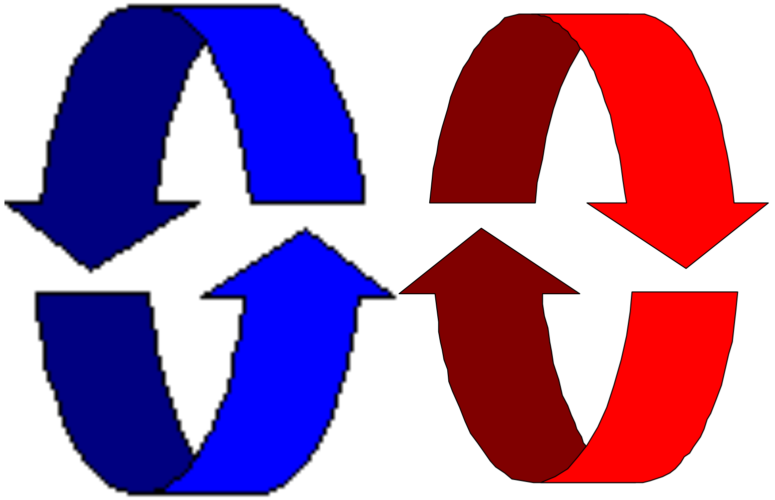
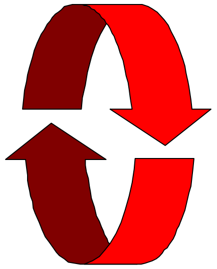
Concurrency

State Models and Java Programs

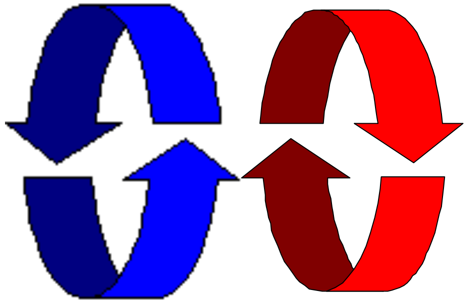


Jeff Magee and Jeff Kramer

What is a Concurrent Program?



A **sequential** program has a single thread of control.

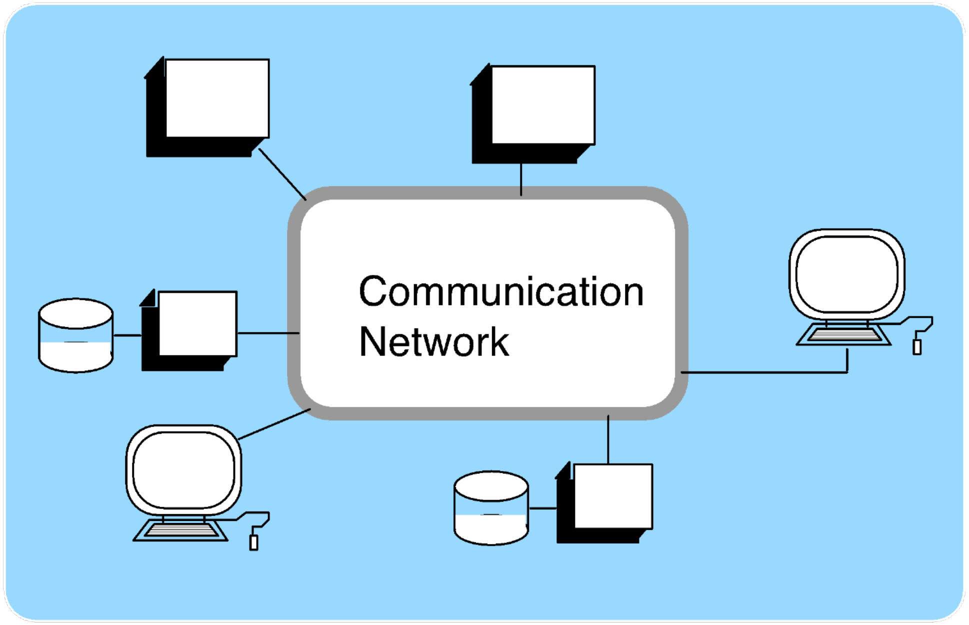


A **concurrent** program has multiple threads of control allowing it

- perform multiple computations in parallel
- control multiple external activities which

occur at the same time.

Concurrent and Distributed Software?



Interacting, concurrent software components of a system:

single machine -> *shared memory interactions*

multiple machines -> *network interactions*

- *GPU (Graphic Process Unit) vs. Multicore CPU*

(ref: https://www.youtube.com/watch?v=WmW6SD-EHVY)

- *Cluster Computing:* from 2-node to supercomputer (same hardware, software)
- *Grid Computing*: Berkeley Open Infrastructure for Network Computing (BOINC)
  - collection of computer resources from multiple locations to reach a common goal.

Why Concurrent Programming?

- Performance gain from multiprocessing hardware
  - parallelism.
- Increased application throughput
  - an I/O call need only block one thread.
- Increased application responsiveness
  - high priority thread for user requests.
- More appropriate structure
  - for programs which interact with the environment, control multiple activities and handle multiple events.

Do I need to know about concurrent programming?

Concurrency is widespread but error prone.

- Therac - 25 computerized radiation therapy machine

  Concurrent programming errors contributed to accidents causing deaths and serious injuries

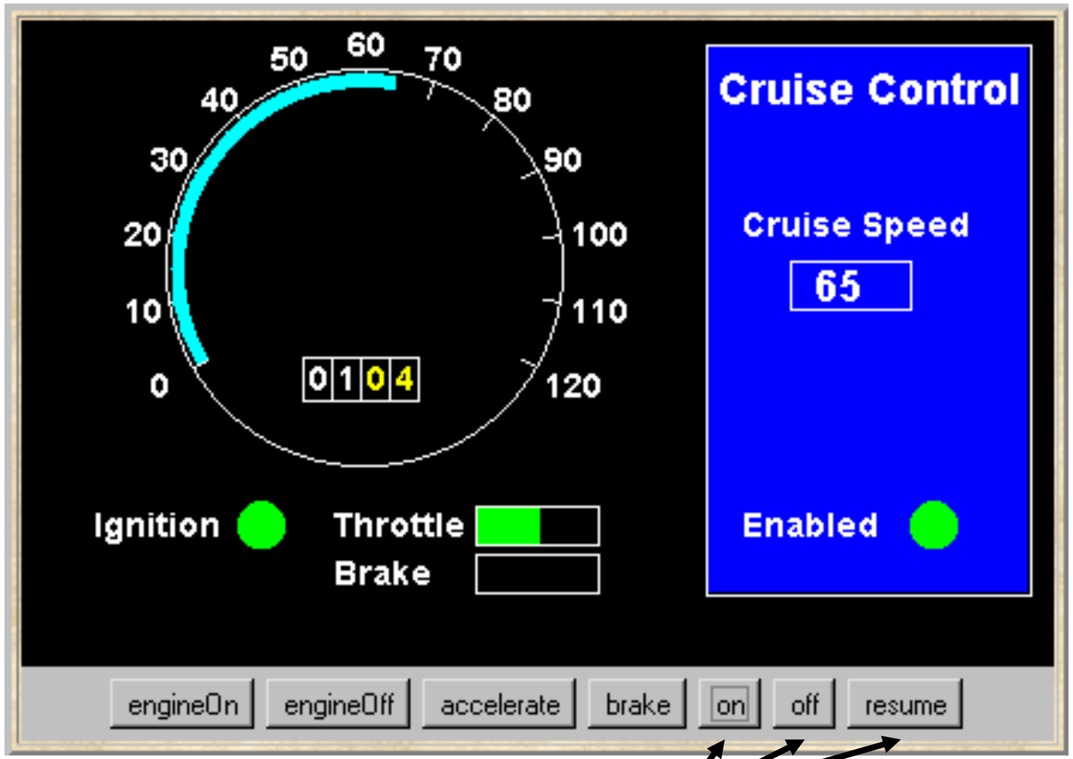
- Mars Rover

  Problems with interaction between concurrent tasks caused periodic software resets reducing availability for exploration.

Models

- Model:
  - is a simplified representation of the real world
  - Engineers use models to gain confidence in the adequacy and validity of a proposed design
  - use Finite State Processes (**FSP**) to **specify/code** processes
  - use Labelled Transition Systems **(LTS) to model the** process **as state machine**
  - is displayed and analysed by the Labelled Transition System Analyser (***LTSA***) analysis tool
- Model Animation:
  - to visualise a behaviour.

a Cruise Control System

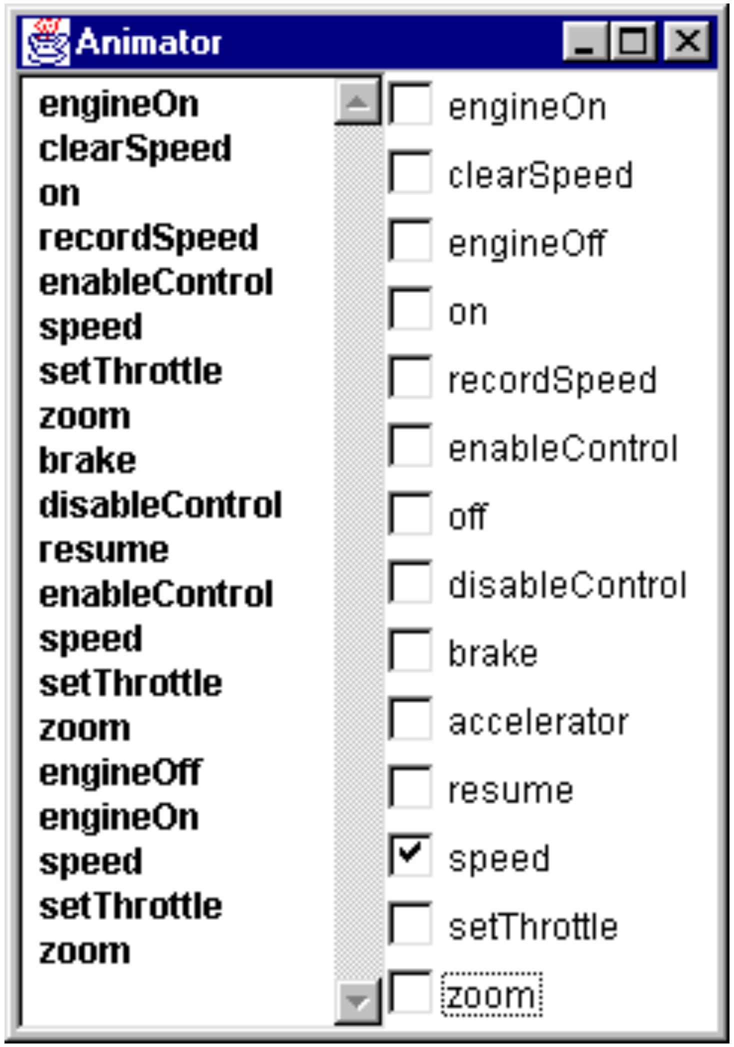


buttons

When the car ignition is switched **on** and the **on** button is pressed, the current speed is recorded and the system is enabled: *it maintains the **speed** of the car at the recorded setting.*

Pressing the **brake**, **accelerator** or **off** button disables the system. Pressing **resume** re-enables the system.

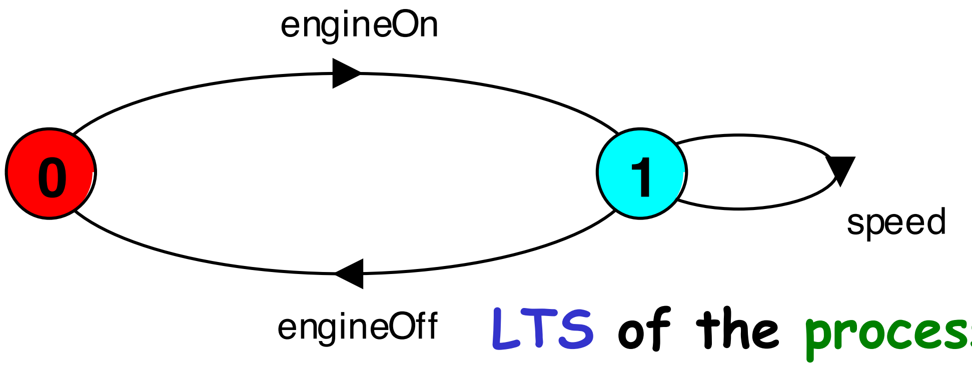
Modelling the Cruise Control System



engineOn
clearSpeed
on
recordSpeed
enableControl
speed
setThrottle
zoom
brake
disableControl
resume
enableControl
speed
setThrottle
zoom
engineOff
engineOn
speed
setThrottle
zoom

☐ engineOn
☐ clearSpeed
☐ engineOff
☐ on
☐ recordSpeed
☐ enableControl
☐ off
☐ disableControl
☐ brake
☐ accelerator
☐ resume
☑ speed
☐ setThrottle
☐ zoom

LTSA **Animator** **to step through system actions and events.**



LTS of the process that monitors speed.

Later chapters will explain how to construct models such as this so as to perform animation and verification.

Programming practice in Java

Java is

- ♦ widely available, generally accepted and **portable**
- ♦ provides sound set of concurrency features

Hence Java is used for all the illustrative examples, the demonstrations and the exercises. Later chapters will explain how to construct Java programs such as the Cruise Control System.

"Toy" problems are also used as they exemplify particular aspects of concurrent programming problems!

Course objective

- intend to provide a sound understanding of
    - ***concepts**, **models** and **practice*** involved in designing concurrent software
    - **emphasis** on **principles** and **concepts** provide a thorough understanding of both the **problems** and the **solution techniques**
- **Modeling**
    - provides insight into concurrent behavior
    - aids reasoning about particular designs
- Concurrent programming in **Java** provides
    - the **programming practice** and **experience**

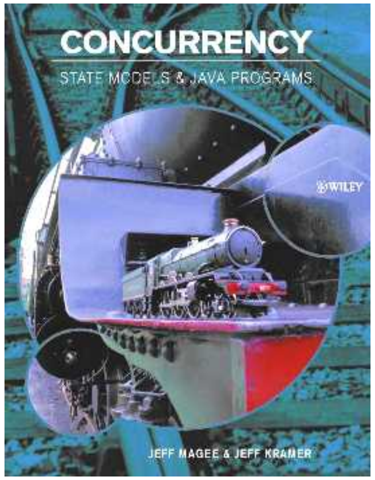
Book

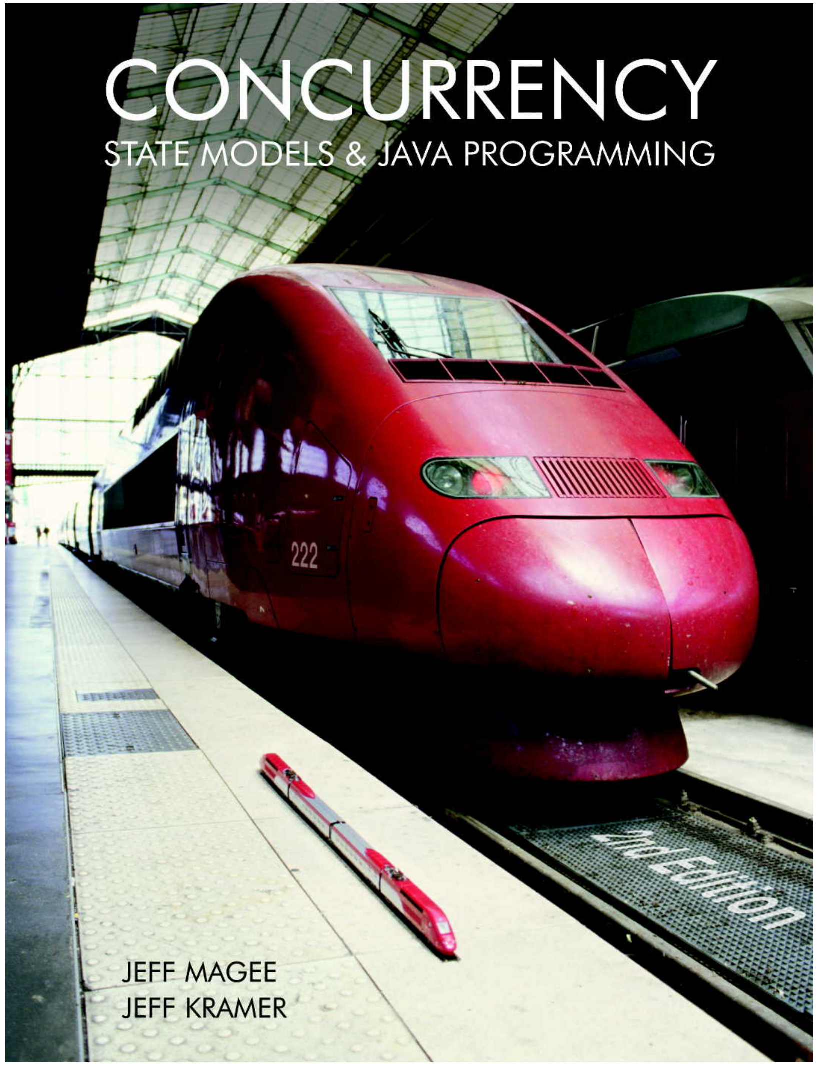
Concurrency: State Models & Java Programs, 2nd Edition

Jeff Magee & Jeff Kramer

WILEY

1st edition





Course Outline

2. Processes and Threads
3. Concurrent Execution
4. Shared Objects & Interference
5. Monitors & Condition Synchronization
6. Deadlock
7. Safety and Liveness Properties
8. Model-based Design

The main basic

Concepts

Models

Practice

Advanced topics …

9. Dynamic systems
10. Message Passing
11. Concurrent Software Architectures
12. Timed Systems
13. Program Verification
14. Logical Properties

Web based course material

http://www.wileyeurope.com/college/magee
http://www.doc.ic.ac.uk/~jnm/book/
http://www.doc.ic.ac.uk/~jnm/book/ltsa/LTSA_applet.html

- Java examples and demonstration programs
  - http://www.doc.ic.ac.uk/~jnm/book/
  - Use Safari to view Applet samples
- Download LTSA:
  - **http://www.doc.ic.ac.uk/~jnm/book/ltsa/download.html**
- To run LTSA: unix> java -jar ltsa.jar

Summary

- Concepts
  - **we adopt a model-based approach for the design and construction of concurrent programs**
- Models
  - use finite state processes (**FSP**) to **code processes**
  - use Labelled Transition Systems **(LTS) to model the process as state machine**
- Practice
  - **we use Java for constructing concurrent programs.**

Examples are used to illustrate the concepts, models and demonstration programs.