

# *Concurrent & Distributed Systems 2007*

*Alistair Rendell and  
Peter Strazdins*

*Department of Computer Science  
The Australian National University*

# What's it all about

*Fundamentals & Overview  
as well as perspectives, paths, methods,  
implementations, and questions*

*of/into/for/about*

*Concurrent & Distributed Systems*

# Relevant to

- anybody who ...
- ... works with real-world scale computer systems
- ... would like to learn how to analyse and design operational and robust systems
- ... would like to understand more about the existing trade-off between theory, the real-world, traditions, and pragmatism in computer science
- ... would like to know what you do not know about concurrent systems

# The People



Alistair Rendell (convener)

<http://cs.anu.edu.au/~Alistair.Rendell>



Peter Strazdins

<http://cs.anu.edu.au/~Peter.Strazdins>

## The High Performance Computing Group

Warren Armstrong

**TUTORS**

Pete Janes



©Department Computer Science,  
Australian National University

# Content

- Lectures:
  - 3 per week ... all the nice stuff and theory  
Monday, 11:00 (Chem-T1); Tuesday 11:00 (Chem-T1); Friday 11:00 (Phys-T)
- Laboratories:
  - 2 hours per week from week 3... on-line tutorial in weeks 1-2  
Mon 14:00, Tue 14:00, Wed 14:00, Thu 14:00, Fri 09:00, all N115
  - Laboratory-enrolment: <https://cs.anu.edu.au/streams/>
- Resources:
  - Introduced in the lectures and collected on the course page:  
<http://cs.anu.edu.au/student/comp2310/>  
... as well as schedules, slides, sources, etc. ... watch this page!
- Assessment (To discuss):
  - exam at the end of the course (70%) plus two assignments (15% each), and mid-term quiz (0%)

# Text Book and Reference Material

- M. Ben-Ari, *Principles of Concurrent and Distributed Programming*, 2006, second edition, Prentice-Hall, ISBN 0-13-711821-X
- See web page for other reference material

# Topics

- 1. Concurrency [3]**
- 2. Mutual exclusion [3]**
- 3. Condition  
synchronization [4]**
- 4. Non-determinism in  
concurrent systems [2]**
- 5. Scheduling [2]**
- 6. Safety and liveness [3]**
- 7. Architectures  
for CDS [3]**
- 8. Distributed systems [8]**

# Topics

## 1. *Concurrency [3]*

- **Forms of concurrency [1]**
  - Coupled dynamical systems
- **Models and terminology [1]**
  - Abstractions
  - Interleaving
  - Atomicity
  - Proofs in concurrent and distributed systems
- **Processes & threads [1]**
  - Basic definitions
  - Process states
  - Implementations

# Topics

## 1. *Concurrency* [3]

## 2. *Mutual exclusion* [3]

- **by shared variables [2]**
  - Failure possibilities
  - Dekker's algorithm
- **by test-and-set hardware support [0.5]**
  - Minimal hardware support
- **by semaphores [0.5]**
  - Dijkstra definition
  - OS semaphores

# Topics

1. **Concurrency [3]**
  2. **Mutual exclusion [3]**
  3. **Condition synchronization [4]**
- **Shared memory synchronization [2]**
    - Semaphores
    - Cond. variables
    - Conditional critical regions
    - Monitors
    - Protected objects
  - **Message passing [2]**
    - Asynchronous / synchronous
    - Remote invocation / rendezvous
    - Message structure
    - Addressing

# Topics

1. **Concurrency [3]**
  2. **Mutual exclusion [3]**
  3. **Condition synchronization [4]**
  4. **Non-determinism in concurrent systems [2]**
- **Correctness under non-determinism [1]**
    - Forms of non-determinism
    - Non-determinism in concurrent/distributed systems
    - Is consistency/correctness plus non-determinism a contradiction?
  - **Select statements [1]**
    - Forms of non-deterministic message reception

# Topics

1. *Concurrency [3]*
  2. *Mutual exclusion [3]*
  3. *Condition synchronization [4]*
  4. *Non-determinism in concurrent systems [2]*
  5. *Scheduling [2]*
- **Problem definition and design space [1]**
    - Which problems are addressed / solved by scheduling?
  - **Basic scheduling methods [1]**
    - Assumptions for basic scheduling
    - Basic methods

# Topics

1. **Concurrency [3]**
  2. **Mutual exclusion [3]**
  3. **Condition synchronization [4]**
  4. **Non-determinism in concurrent systems [2]**
  5. **Scheduling [2]**
  6. **Safety and liveness [3]**
- **Safety properties**
    - Essential time-independent safety properties
  - **Livelocks, fairness**
    - Forms of livelocks
    - Classification of fairness
  - **Deadlocks**
    - Detection
    - Avoidance
    - Prevention (& recovery)
  - **Failure modes**
  - **Idempotent & atomic operations**
    - Definitions

# Topics

1. **Concurrency [3]**
  2. **Mutual exclusion [3]**
  3. **Condition synchronization [4]**
  4. **Non-determinism in concurrent systems [2]**
  5. **Scheduling [2]**
  6. **Safety and liveness [3]**
  7. **Architectures for CDS [3]**
- **Academic**
    - CSP
    - occam
  - **Production**
    - Ada95
    - JAVA
  - **Historical roots: UNIX**
    - UNIX processes
    - UNIX communication schemes
  - **Dedicated hardware**
    - Communication controllers
  - **Embedded systems**

# Topics: Distributed Systems [8]

- **Networks [1]**
  - OSI model
  - Network implementations
- **Global times [1]**
  - synchronized clocks
  - logical clocks
- **Distributed states [1]**
  - Consistency
  - Snapshots
  - Termination
- **Distributed communication [1]**
  - Name spaces
  - Multi-casts
- **Distributed safety and liveness [1]**
  - Elections
  - Network identification
  - Dynamical groups
- **Distributed deadlock detection [1]**
- **Forms of distribution/redundancy [1]**
  - computation
  - memory
  - operations
- **Transactions [2]**

# Laboratories

- ***Concurrency language support basics (in Ada) [3]***
  - **Structured, strongly typed programming**
  - Program structures
  - Data structures
  - **Generic, re-usable programming**
  - Generics
  - Abstract types
  - **Concurrent processes:**
  - Creation
  - Termination
  - Rendezvous
- ***Concurrent programming [3]***
  - **Synchronization**
  - Protected objects
  - **Remote invocation**
  - Extended rendezvous
  - **Client-Server architectures**
  - Entry families
  - Requeue facility
- ***Concurrency in UNIX [3]***
  - **UNIX process creation, termination**
  - **UNIX process communication**
  - Pipes
  - Sockets

# Pre-Labs: On-Line Tutorials

- As noted, this course will use Ada
- In weeks 1 and 2 you are strongly encouraged to complete the on-line tutorial available from the lab web page
- Discuss lab web page and rest of comp2310 web page

# Assessment Options

---

	———— Option ————			
	<b>1</b>	<b>2</b>	<b>3</b>	<b>4</b>
<b>Assign 1</b>	15	15	20	20
<b>Assign 2</b>	15	15	20	20
<b>Mid Semester</b>	None	Open Questions not Marked	10	10 Redeemable
<b>Exam</b>	70	70	50	50

---

# Questions?

©Department Computer Science,  
Australian National University