Dependable and Distributed Systems

Professor Matthew Leeke
School of Computer Science
University of Birmingham

Topic 1 - Introduction

Module Administration

Module Administration

Module Organiser:

Matthew Leeke (CS 116, m.leeke@bham.ac.uk)

Lectures:

Two lectures per week

Topic Videos:

Video lectures released for each topic

Website and Resources:

Canvas page provides everything you need, including links to relevant reading and support

Module Administration

Module Aim:

To provide you with a knowledge of issues and concepts in the design, implementation and evaluation of dependable and distributed systems. We concentrate on the principles and technologies that can be applied in the design, development and measurement of dependable and distributed systems under varied assumptions.

Module Administration

Learning Outcomes:

Understand and design distributed algorithms.

Analyse and reason about the properties of distributed algorithms.

Understand and apply fundamental concepts in the design of dependable systems.

Analyse the dependability of systems use of qualitative and quantitative approaches.

Understand the design and implementation of synchronisation methods in distributed systems.

Demonstrate the capacity to independently study, understand, and critically evaluate advanced materials or research articles in the subject areas covered by this module.

Module Themes and Topics

Module Themes

Dependability Analysis:

How do we quantify dependability? Can we have representative models?

Software:

How do we think about defensive programming techniques, N-version programming, recovery blocks, etc.?

Hardware:

What approaches allow us to design dependable hardware?

Module Themes

Distributed systems:

How can we solve classical problems with broad applicability, e.g., leader election, consensus, clock synchronisation, Byzantine agreement, etc.?

How do synchronous, asynchronous and partially synchronous system models change things?

Are there situations where we apply dependability concepts to achieve effective algorithms?

Synchronisation:

How can we design systems based on generalised synchronisation mechanisms?

Module Topics

01 Introduction

Context and administration

Module Topics

01 Introduction → Context and administration

02 Dependability Concepts

03 Dependability Analysis and Evaluation

04 Software Fault Tolerance

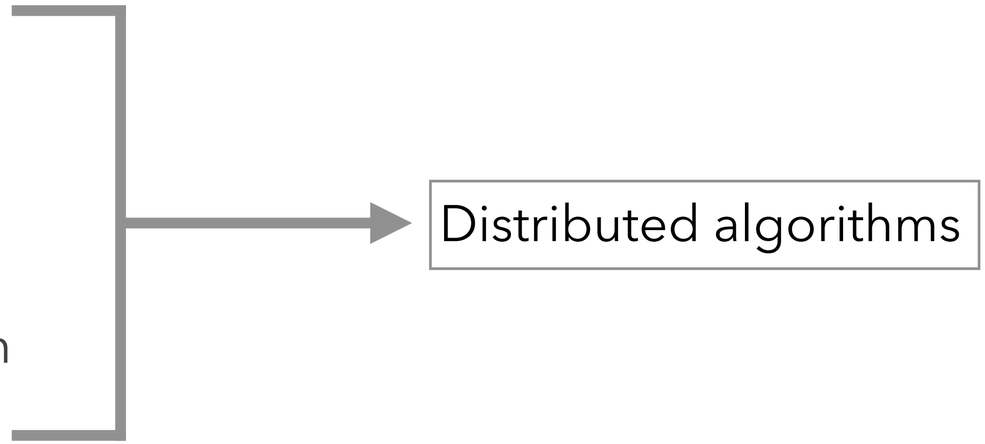
→ Dependable systems, including analysis, modelling and theory

Module Topics

05 Leader Election

06 Consensus

07 Byzantine Generals Problem

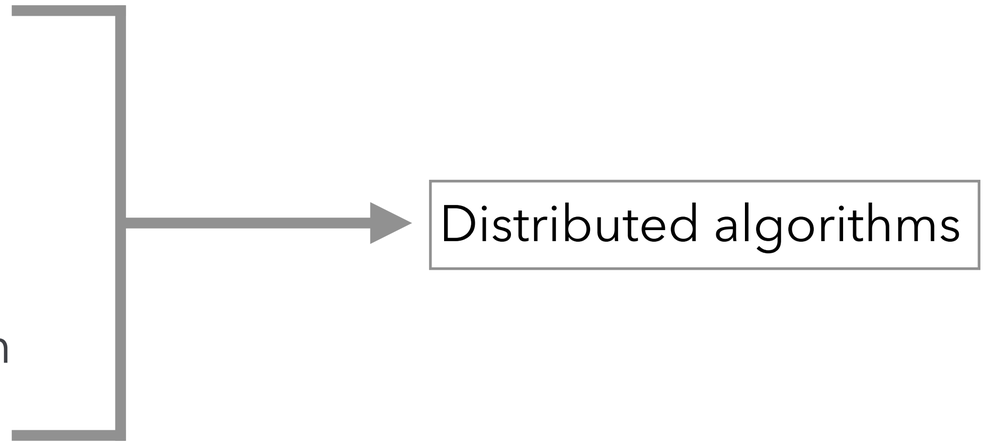


Module Topics

05 Leader Election

06 Consensus

07 Byzantine Generals Problem

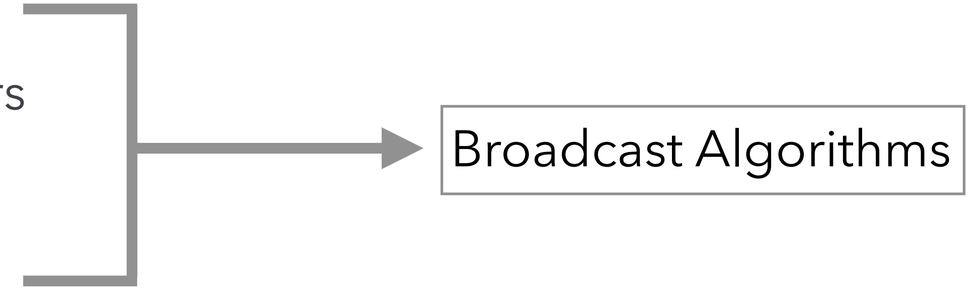


08 Clock Synchronisation, Logical Clocks and Vector Clocks

Module Topics

09 Reliable Links and Failure Detectors

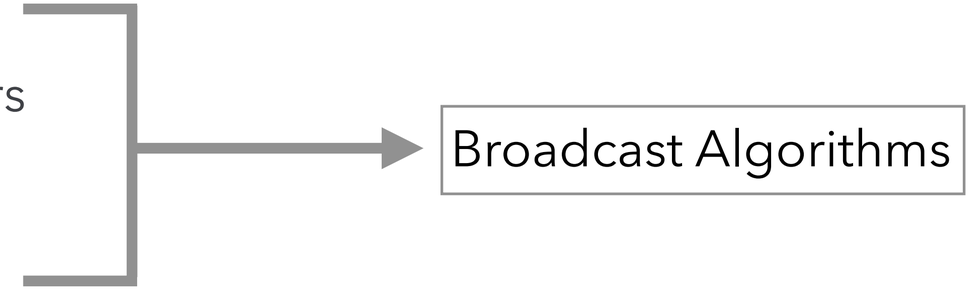
10 Reliable Broadcast



Module Topics

09 Reliable Links and Failure Detectors

10 Reliable Broadcast



What can I be expected to know about the module topics?

You will be expected to know the breadth and depth of the content covered in lectures. On top of that, you will be be expected to apply some problem solving, particularly the more challenging part of the examination that require you to combine knowledge and problem solving.

Module Assessment and Support

Module Assessment

Coursework (20%) and formal examination (80%)

Coursework

Released shortly and due before the end of term

Exam

One exam in the main period, i.e., May / June

Two hours to answer three questions

Module Coursework

Carries 20% module credit

A single assignment that will be released immediately

Should take no more than a few days a dedicated work

Designed to help you with your project by providing time and relevant skills

Deliberately designed to help you with one of the most important parts of your dissertation

Weekly in-lecture support and the opportunity to have me review complete drafts before submission

Module Exam

Carries 80% module credit

Two hours to answer three questions

One exam in the main period, i.e., May / June

You will be supported in preparing through past papers and comprehensive exam guidance

Attending lectures and/or watching topics videos will provide indications of topic importance

Module Support

Lectures

Reading

Email

Baseline for most modules, but these will enhance the additional support

Coursework Support

Exam Revision Lecture

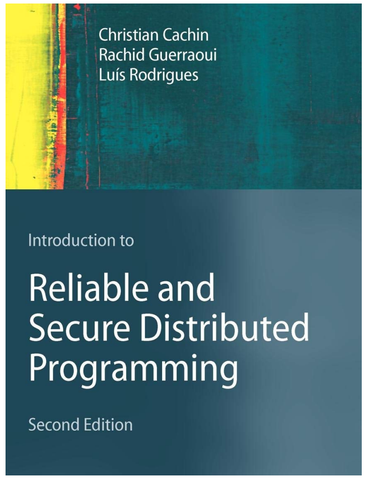
Exam Guidance

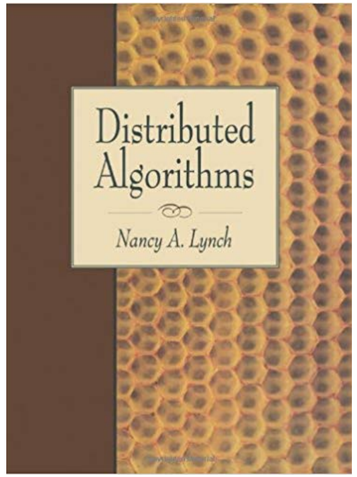
Past Exam Papers

Reasons why you will succeed if you attend and engage

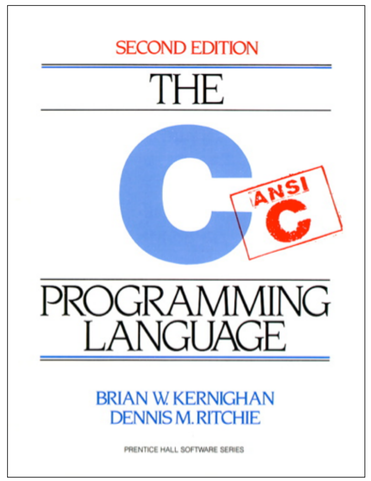
Recommended Books

C. Cachin, R. Guerraoui, L. Rodrigues, Introduction to Reliable and Secure Distributed Programming (2nd Edition), Springer, February 2011



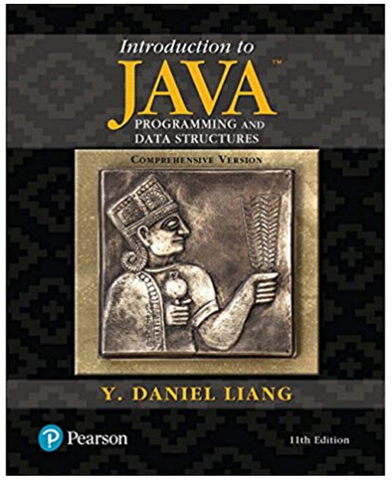


N. Lynch, Distributed Algorithms (1st Edition), Morgan Kaufmann, April 1996



B. W. Kernighan, D. Ritchie. The C Programming Language (2nd Edition), Prentice Hall, March 1988

D. Liang, Introduction to Java Programming and Data Structures (11th Edition), Pearson, April 2018



Motivating Dependability

Why Dependability?

Dependability has been an distinct are of research and development for at least 50 years, with the implicit study of the discipline having been around for much longer

We will look at classical and state-of-the-art approaches to dependability

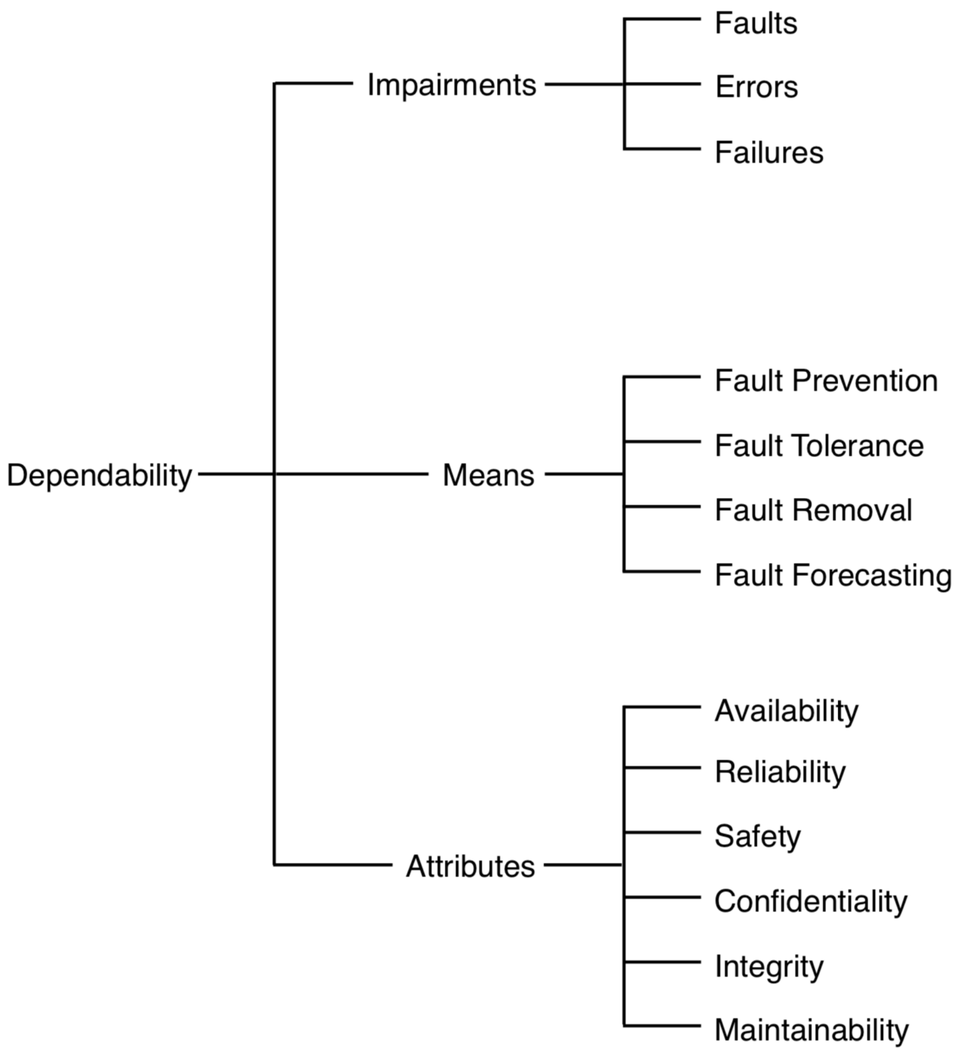
- Hardware
- Software
- Distributed Systems

What Is Dependability?

Preventing catastrophic consequences seems like a challenging task

Impossible to foresee all failure modes and circumstances

We defined and about dependability in a structured way to facilitate progress



Why Dependability?

Dependability is typically associated with safety critical systems

- Avionics
- Nuclear reactors
- X-by-wire systems

Increasingly common Quality-of-Service (QoS) focus on dependability for consumer goods and services

Why Dependability?

Murphy's Law: If something can go wrong, it will go wrong

We need to consider our assumptions carefully

Dependability is generally achieved through some form of redundancy

Redundancy can be spacial or temporal

- Replicating data storage is an example of spatial redundancy
- Repeated execution is an example of temporal redundancy

A Case Study - Apollo 13

NASA's Apollo 13 mission of 1970 is famously documented

Bestselling book - "Failure Is Not An Option"

Award winning film - "Apollo 13"

Classic academic text by Leveson - "Safeware"

They intended to land on the moon, which had been done in 1969

What happened? What can we learn?

Apollo 13 - Background

The rocket itself comprised three modules at launch

- Service Module, Command Module, and Lunar Module

Preparations for the mission were not incident-free

- Liquid oxygen leaked, causing sparks in car ignition systems to set them on fire
- One of the two tanks supplying oxygen for the command module would not drain properly
- Engineers decided to ignore the issues because of schedule pressures, claiming that each problem “did not affect performance”

Apollo 13 - Preparation

It gets worse, quite quickly…

Wire in on of the oxygen tanks were almost devoid of any insulation as a result of accidental overheating during test runs of the emptying process

Crew did not have experience of working together in critical situations, not least because a member has German measles a few days before launch

Apollo 13 - Launch

No hardware problems formally reported at launch, despite:

One of the oxygen tanks readings being off-the-scale

Erratic helium pressure in the lunar module

Low pressure in a hydrogen tank

Second-stage engines cutting off two minutes early, requiring other engines to fire for longer than planned, using more fuel - still plenty left under normal operating conditions

Apollo 13 - 56 Hours After Launch

Amber warning light indicates low pressure again in the service module hydrogen tank, forcing the crew to stir the tanks

Threw four switches to turn on the global fans in the tanks

16 seconds later, as arc of electricity jumped across two wires, causing an oxygen tank fire

Rapid increase in oxygen pressure goes unnoticed because the hydrogen pressure warning light overrode the oxygen pressure warning

Leaking oxygen spreads the fire throughout the service bay

Apollo 13 - 56 Hours After Launch

Fire caused a blowout in the Service Module, resulting in an amber warning light for low power, but the explosion has already closed the oxygen supply lines to fuel cells

The same fuel cells were critical to producing electricity, water and cabin oxygen but the instruments had no way of indicating that the oxygen tank had exploded, just erratic readings

Crew member notices a white cloud of surrounding the service module, realising that venting will be throwing the spacecraft off course

Two fuel cells gone with the power of the third dropping, one oxygen tank reporting empty and the other with little pressure - no more electricity meant just two hours of oxygen

Apollo 13 - A Great Hope

Lunar Module has its own supply of oxygen and water, serving as a life raft

Need Command Module to survive reentry so all power consumption is dropped to zero

No power makes the punishingly cold - would the instruments survive?

Lunar Module design to hold and keep two people alive for 50 hours but had to support three for 84 hours - rotating sleeping in the Command Module

No spoilers for the ending!

Apollo 13 - What Can We Take Away?

The Underlying Problem: **incorrect failure assumptions**

- Assumed low probably of quadruple failures
- Assumed independence of failure

Recommendations:

- Eliminate dangers associate with use of pure oxygen
- All teflon insulation to be replaced with stainless steel
- Fourth oxygen tank located apart from others to provide fault containment

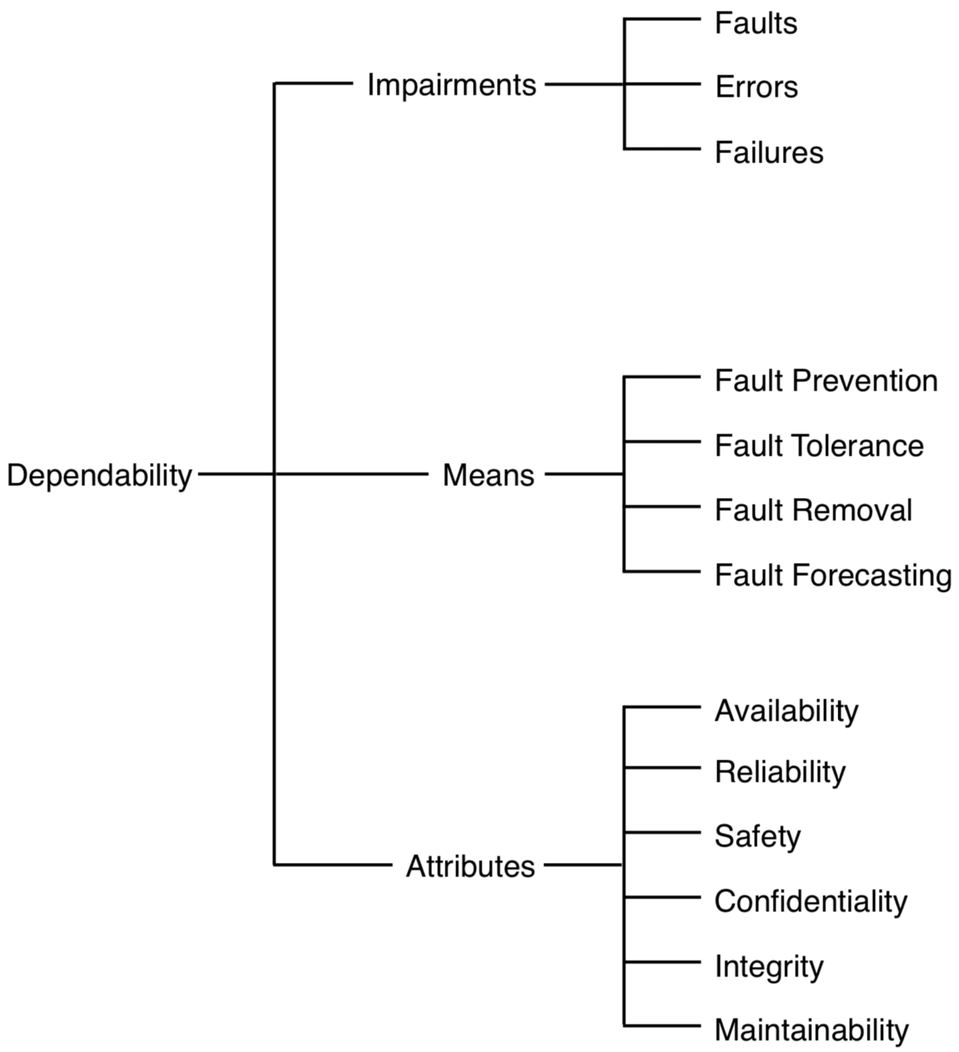
A Peek Ahead

What Is Dependability?

Preventing catastrophic consequences seems like a challenging task

Impossible to foresee all failure modes and circumstances

We defined and about dependability in a structured way to facilitate progress



Trustworthiness

How can we define trustworthiness for dependability?

If your computer crashes once per year, is it trustworthy? Once per day? Once per hour?

Really depends on the system requirements

Specification defined **what the system is supposed to do**

Can we relate trustworthiness and system specification?

If a system fulfils its specification, is it trustworthy?

Dependability

A system is said to be **dependable** if it is trustworthy enough for reliance to be justifiably placed on the services it delivers

How trustworthy should such a system be?

How do we measure trustworthiness?

How can we ensure trustworthiness by design?

What Do We Need To Develop Dependable Systems?

We have a **system model**

How we view our system - black box, white box, interconnected, etc.

Define a **fault model**

What are the potential problems that we must consider

System has a **specification**

Essentially an oracle for system behaviour

System Model

A system is a set of **elements** that work together to provide **service** to a **user**

> Elements can be hardware, software, human, etc., whilst user can mean another system, human, etc.
>
> Interaction through well-defined interfaces, e.g., GUI connects human and software, remote procedure call connects software elements

An element is an entity that provide a predefined service and is able to communicate with other elements

Depending on the analysis requirements, we can view an element at varying levels of abstraction, e.g., memory or individual registers in hardware, method or object in software

What is a Service?

A service is the behaviour that a user perceiver at the system interface

For trustworthy systems, services need to be **correct** and **timely**

What Are The Problems

A system is a collection is elements

Elements may start failing

- Services not provided as expected, e.g., due to bugs in software
- Hardware problems, e.g., due to ageing mechanical components
- Unexpected circumstances, e.g., spikes in workload

The Dependability Tree

A taxonomy generalises and structures our consideration of dependability

Impairments - What endangers the dependability of a system

Means - What we can we do to impart dependability

Attributes - What measures we uses to establish the dependability of a system