COMP2121: PRINCIPLES OF DISTRIBUTED SYSTEMS AND NETWORKS

Semester 2, 2017 | 6 Credit Points | Mode: Normal-Day
Sessions Valid: Semester 2
Coordinator(s): Vincent Gramoli

WARNING: This unit version is currently under review and is subject to change!

1. INTRODUCTION

The unit will provide a broad introduction to the principles of distributed systems and their design; provide students the fundamental knowledge required to analyse and construct various types of distributed systems; explain the common architectural principles and approaches used in the design of networks at different scales (e.g. shared medium access and routing); introduce the programming skills required for developing distributed applications, and will cover the use of Java class libraries and APIs; cover common approaches and techniques in distributed resource management (e.g. task scheduling).

2. LEARNING OUTCOMES

Learning outcomes are the key abilities and knowledge that will be assessed in this unit. See assessment summary table below for details of which outcomes are assessed where. Outcomes are listed according to the course goals that they support.

Design (Level 2)

1. Students will have experience in distributed implementation of algorithms. They will be able to able to apply some common distributed algorithms (e.g. searches, shortest path, trees) towards solving problems.

Engineering/IT Specialisation (Level 2)

2. At the end of the course, students will understand the general properties of distributed systems and networks. They should be familiar with various types of distributed applications and how information is shared between components in distributed systems. They will have knowledge of the basic building blocks in distributed systems. They should also understand the functions of resource management and task scheduling and be familiar with standards commonly used in distributed systems and networks, e.g. network protocols and information representation. The students should also understand programming paradigms for distributed systems (e.g. sockets) and be able to apply them in a Java API.

Information Seeking (Level 2)

3. Students will be aware of fundamental constraints and performance metrics of distributed systems and networks. They will understand the layered network model and the functions at each layer, and be familiar with some common realisations of those functions. They will be able to produce good quality distributed software and be aware of professional expectations for such software. They will also be aware of trade-offs arising in distributed systems between system requirements and available resource and cost constraints, and understand how to resolve such trade-offs by prioritizing the system capabilities.

Communication (Level 2)

4. Students will have the experience to produce professional quality written assignments and reports as well as well-documented software for reuse.

Professional Conduct (Level 3)

5. Students will be made aware of the implications of sharing of information and the importance of privacy and security. They will also appreciate the importance of ethical behaviour among users of distributed systems.

For further details of course goals related to these learning outcomes, see online unit outline at http://cusp.eng.usyd.edu.au/students/view-unit-page/alpha/COMP2121 .

3. ASSESSMENT TASKS

ASSESSMENT SUMMARY

Assessment name	Team-based?	Weight	Due	Outcomes Assessed
Programming Assignment 1	No	10%	Week 5 (Wednesday, 12 pm)	2, 3, 4
Mid-Sem Quiz	No	10%	Week 7	1, 2
Programming Assignment 2	No	10%	Week 8 (Wednesday, 12 pm)	2, 3, 4
Programming assignment 3	No	20%	Week 12 (Wednesday, 12 pm)	1, 2, 3
Final Exam	No	50%	Exam Period	1, 2, 5

ASSESSMENT DESCRIPTION

The unit will use programming assignments, a mid-term quiz and a final exam.

The mid-semester quiz and final exam will test the students' understanding of the theoretical material and concepts and ability to put it in the appropriate context of solving problems. The programming assignments will enable students to develop and test their practical skills and benchmark them against set criteria.

ASSESSMENT FEEDBACK

Marks will be awarded for the assignments and mid-term exam and will reflect the students' understanding of the assessed material and their

ability to apply it in a programming task.

The tutor(s) will encourage interactive learning and provide an opportunity for students to test their understanding in a classroom setting.

The solutions of the assignments will be discussed in the tutorial/lab classes and provide an opportunity for the students to learn by comparing their solutions to the recommended ones.

ASSESSMENT GRADING

Final grades in this unit are awarded at levels of HD for High Distinction, DI (previously D) for Distinction, CR for Credit, PS (previously P) for Pass and FA (previously F) for Fail as defined by University of Sydney Assessment Policy. Details of the Assessment Policy are available on the Policies website at http://sydney.edu.au/policies . Standards for grades in individual assessment tasks and the summative method for obtaining a final mark in the unit will be set out in a marking guide supplied by the unit coordinator.
It is a policy of the School of Information Technologies that in order to pass this unit, a student must achieve at least 40% in the written examination. For subjects without a final exam, the 40% minimum requirement applies to the corresponding major assessment component specified by the lecturer. A student must also achieve an overall final mark of 50 or more. Any student not meeting these requirements may be given a maximum final mark of no more than 45 regardless of their average.

4. ATTRIBUTES DEVELOPED

Attributes listed here represent the course goals designated for this unit. The list below describes how these attributes are developed through practice in the unit. See Learning Outcomes and Assessment sections above for details of how these attributes are assessed.

Attribute	Method
Design (Level 2)	The tasks and assignments in the unit will provide ample opportunity for students to exercise design and problem solving skills, particularly in the programming tasks that will require development of original solutions. The issue of plagiarism will be addressed.
Engineering/IT Specialisation (Level 2)	The task and assignment give students opportunities to identify, integrate and synthesise knowledge on distributed systems and programming to solve problems under constraints.
Information Seeking (Level 2)	The students will be given various problems that they will need to solve, requiring research of the appropriate background information using resources such as the university library and the Internet. Students will also be required to understand different types of information and its representation and use in distributed systems, and will be exposed to standards that ensure the consistency and quality of such information.
Communication (Level 2)	The students will be required to produce written assignments and reports and develop professional quality, well-documented software that can be understood and reused by other programmers.
Professional Conduct (Level 3)	The students will become familiar with the ethical issues pertaining to the use of large-scale distributed systems, and will understand the risks involved with the access to and processing of large quantities of information and the importance of its security and privacy.

For further details of course goals and professional attribute standards, see the online version of this outline at http://cusp.eng.usyd.edu.au/students/view-unit-page/alpha/COMP2121 .

5. STUDY COMMITMENT

The lectures will provide the theoretical and conceptual foundations of the material. The lab classes will present more specific examples of particular systems and network protocols in detail, with a focus on the programming aspects of the material.

The use of these complementary modes of delivery will enhance the students` learning experience, by reinforcing the concepts presented in the lectures with practical examples and realisation of solutions to conceptual problems, and by allowing the students to gain hands-on experience with implementing those solutions.

Activity	Hours per Week	Sessions per Week	Weeks per Semester
Lecture	2.00	1	13
Lab/Tutorial	1.00	1	12

Standard unit of study workload at this university should be from 1.5 to 2 hours per credit point which means 9-12 hours for a normal 6 credit point unit of study. For units that are based on research or practical experience, hours may vary. For lecture and tutorial timetable, see University timetable site at: web.timetable.usyd.edu.au/calendar.jsp

6. TEACHING STAFF AND CONTACT DETAILS

COORDINATOR(S)

Name	Room	Phone	Email	Contact note
Dr Gramoli, Vincent		02 9036 9270	Vincent.gramoli@sydney.edu.au	

LECTURERS

Name	Room	Phone	Email	Contact note
Dr Gramoli, Vincent		02 9036 9270	Vincent.gramoli@sydney.edu.au	

TUTORS

Teaching assistant:

Jiaan Guo, jguo4890@uni.sydney.edu.au

Tutors:

Jiaan Guo, jguo4890@uni.sydney.edu.au,

Omid Tavallaie, otav8458@uni.sydney.edu.au,

Rabia Chaudry, rcha3704@uni.sydney.edu.au,

Julia Wong, jwon5553@uni.sydney.edu.au,

Patrick Nappa, pnap4580@uni.sydney.edu.au,

Tyson Thomas, ttho6664@uni.sydney.edu.au,

Parinya Ekparinya, pekp6601@uni.sydney.edu.au,

Michael Spain, mspa1382@uni.sydney.edu.au,

Joshua Murray, jmur9664@uni.sydney.edu.au

7. RESOURCES

PRESCRIBED TEXTBOOK(S)

Tanenbaum & van Steen, *Distributed Systems Principles and Paradigms, 2nd edition*. Prentice-Hall , 2007.

RECOMMENDED REFERENCES

Hagit Attiya and Jennifer Welch, *Distributed Computing, 2nd edition*. WILEY, 2004. 978-0-471-45324-6.

Silberschatz, Galvin, Gagne, *Operating System Concepts with Java, 8th edition*. Wiley, 2010. 978-0-470-50949-4.

Saltzer & Kaashoek, *Principles of Computer System Design, an Introduction*. Morgan & Kaufmann, 2009. 978-0-12-374957.

8. ENROLMENT REQUIREMENTS

ASSUMED KNOWLEDGE

Introductory Java programming unit, Data Structures, Algorithms

PREREQUISITES

(INFO1103 OR INFO1903) AND (INFO1105 OR INFO1905).

COREQUISITES

COMP2007 OR COMP2907.

ADDITIONAL NOTES

The unit will provide the introductory platform for students interested in more advanced units in the area of distributed systems and networks, such as ELEC3506 (Data Communications and the Internet), COMP5116 (Internet Protocols), COMP5416 (Advanced Network Technologies), and COMP5426 (Parallel and Distributed Computing).

9. POLICIES

ACADEMIC HONESTY

While the University is aware that the vast majority of students and staff act ethically and honestly, it is opposed to and will not tolerate academic dishonesty or plagiarism and will treat all allegations of dishonesty seriously.

All students are expected to be familiar and act in compliance with the relevant University policies, procedures and codes, which include:

- *Academic Honesty in Coursework Policy 2015*
- *Academic Honesty Procedures 2016*

- *Code of Conduct for Students*
- *Research Code of Conduct 2013* (for honours and postgraduate dissertation units)

They can be accessed via the University"s Policy Register: http://sydney.edu.au/policies (enter "Academic Honesty" in the search field).

Students should never use document-sharing sites and should be extremely wary of using online "tutor" services. Further information on academic honesty and the resources available to all students can be found on the Academic Integrity page of the University website: http://sydney.edu.au/elearning/student/EI/index.shtml

Academic Dishonesty and Plagiarism

Academic dishonesty involves seeking unfair academic advantage or helping another student to do so.

You may be found to have engaged in academic dishonesty if you:

- Resubmit (or "recycle") work that you have already submitted for assessment in the same unit or in a different unit or previous attempt;
- Use assignment answers hosted on the internet, including those uploaded to document sharing websites by other students.
- Have someone else complete part or all of an assignment for you, or do this for another student.
- Except for legitimate group work purposes, providing assignment questions and answers to other students directly or through social media platforms or document ("notes") sharing websites, including essays and written reports.
- Engage in examination misconduct, including using cheat notes or unapproved electronic devices (e.g., smartphones), copying from other students, discussing an exam with another person while it is in progress, or removing confidential examination papers from the examination venue.
- Engage in dishonest plagiarism.

Plagiarism means presenting another person's work as if it is your own without properly or adequately referencing the original source of the work.

Plagiarism is using someone else's ideas, words, formulas, methods, evidence, programming code, images, artworks, or musical creations without proper acknowledgement. If you use someone's actual words you must use quotation marks as well as an appropriate reference. If you use someone's ideas, formulas, methods, evidence, tables or images you must use a reference. You must not present someone's artistic work, musical creation, programming code or any other form of intellectual property as your own. If referring to any of these, you must always present them as the work of their creator and reference in an appropriate way.

Plagiarism is always unacceptable, regardless of whether it is done intentionally or not. It is considered dishonest if done knowingly, with intent to deceive or if a reasonable person can see that the assignment contains more work copied from other sources than the student's original work. The University understands that not all plagiarism is dishonest and provides students with opportunities to improve their academic writing, including their understanding of scholarly citation and referencing practices.

USE OF SIMILARITY DETECTION SOFTWARE

All written assignments submitted in this unit of study will be submitted to the similarity detecting software program known as **Turnitin**. Turnitin searches for matches between text in your written assessment task and text sourced from the Internet, published works and assignments that have previously been submitted to Turnitin for analysis.

There will always be some degree of text-matching when using Turnitin. Text-matching may occur in use of direct quotations, technical terms and phrases, or the listing of bibliographic material. This does not mean you will automatically be accused of academic dishonesty or plagiarism, although Turnitin reports may be used as evidence in academic dishonesty and plagiarism decision-making processes.

Computer programming assignments may also be checked by specialist code similarity detection software. The Faculty of Engineering & IT currently uses the MOSS similarity detection engine (see http://theory.stanford.edu/~aiken/moss/) . These programs work in a similar way to TII in that they check for similarity against a database of previously submitted assignments and code available on the internet, but they have added functionality to detect cases of similarity of holistic code structure in cases such as global search and replace of variable names, reordering of lines, changing of comment lines, and the use of white space.

IMPORTANT: School policy relating to Academic Dishonesty and Plagiarism.

In assessing a piece of submitted work, the School of IT may reproduce it entirely, may provide a copy to another member of faculty, and/or to an external plagiarism checking service or in-house computer program and may also maintain a copy of the assignment for future checking purposes and/or allow an external service to do so.

Other policies

See the policies page of the faculty website at http://sydney.edu.au/engineering/student-policies/ for information regarding university policies and local provisions and procedures within the Faculty of Engineering and Information Technologies.

10. WEEKLY SCHEDULE

Note that the "Weeks" referred to in this Schedule are those of the official university semester calendar https://web.timetable.usyd.edu.au/calendar.jsp

Week	Topics/Activities
Week 1	Lecture: Introduction to distributed systems. Goals of the UoS, definition and challenges.
Week 2	Lecture: Concurrency in operating systems. UNIX processes, context switches, Java threads.
	Lab: Java multithreading.
Week 3	Lecture: Communication 1/2. Network layers, routing protocols, sockets.
	Tutorial: Routing protocols.
Week 4	Lecture: Communication 2/2. TCP/IP, RPC-like mechanisms, Java RMI.
	Lab: Java sockets and server implementation.
Week 5	Lecture: Synchronization 1/2. Notions of physical time and logical time, network time protocol, logical and vector clocks.
	Tutorial: Network time protocol.
	Assessment Due: Programming Assignment 1
Week 6	Lecture: Naming. Name spaces, decription of DNS, comparison of distributed file systems.
	Lab: Remote Method Invocation
Week 7	Lecture: Synchronization 2/2. Multiprocessor, Mutual exclusion, Transactional Memory.
	Assessment Due: Mid-Sem Quiz
Week 8	Lecture: Consistency. The notion of consistency among entities of a distributed system.
	Lab: Transactional Memory.
	Assessment Due: Programming Assignment 2
Week 9	Lecture: Failures. Crash and Byzantine failures. Consensus and two-phase commit problems.
	Lab: Consensus.
Week 10	Lecture: Security. Hashing function, symmetric and asymmetric cryptosystems.
	Tutorial: Security.
Week 11	Lecture: Blockchain and proof-of-work.
	Lab: Gossip-based protocols.
Week 12	Lecture: Security and Efficiency in Blockchain Systems.
	Lab: Demo of final assignment.
	Assessment Due: Programming assignment 3
Week 13	Lecture: Summary and review
Exam Period	Assessment Due: Final Exam