

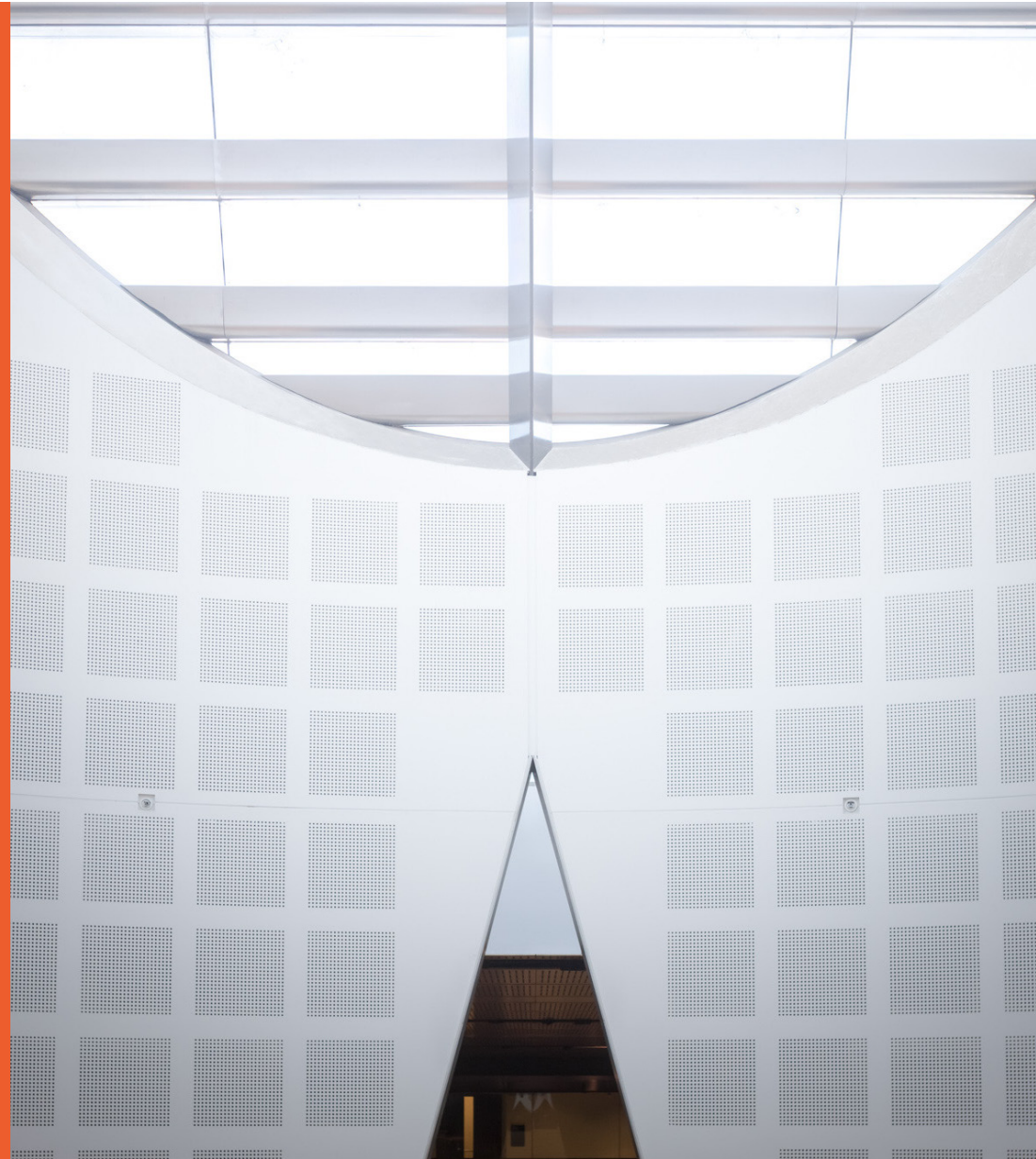
Software Design and Construction 1

SOFT2201 / COMP9201

Introduction to Software Construction & Design

Dr. Xi Wu

School of Computer Science



Copyright Warning

COMMONWEALTH OF AUSTRALIA

Copyright Regulations 1969

WARNING

This material has been reproduced and communicated to you by or on behalf of the University of Sydney pursuant to Part VB of the Copyright Act 1968 (**the Act**).

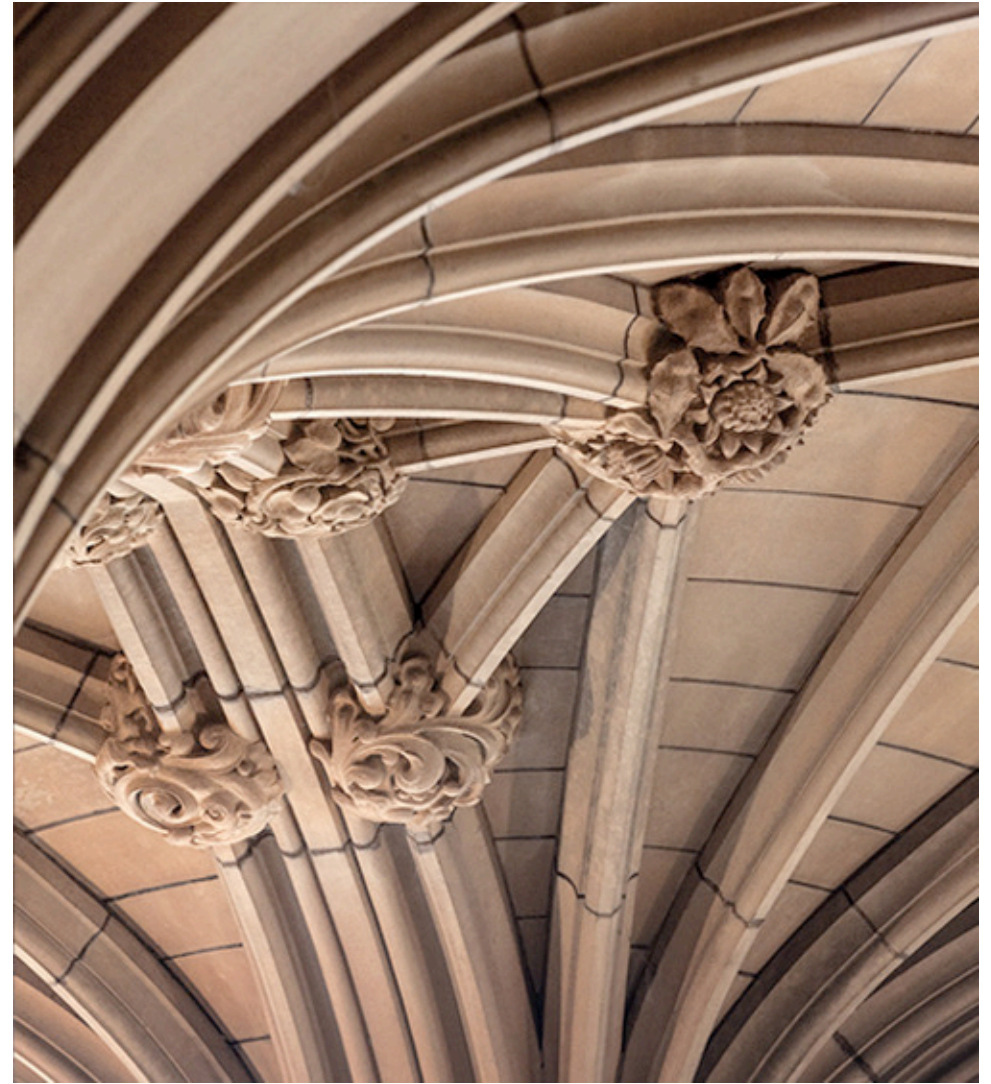
The material in this communication may be subject to copyright under the Act. Any further copying or communication of this material by you may be the subject of copyright protection under the Act.

Do not remove this notice.

Agenda

- Workplace Health and Safety
- Unit of Study Overview
- Software Engineering
- Software Modeling
- The Unified Process

WHS Induction



Keeping our campus COVID safe

- The University is following NSW Government and NSW Health guidance as a minimum standard in our response to the COVID-19 pandemic.
- NSW Government restrictions can change at short notice.
- Check your student email for updates about University operations and COVID safety precautions.
- Visit our website: sydney.edu.au/covid-19

Follow COVID safety precautions



Stay home if you are sick



Wash hands regularly



Avoid physical greetings



Cough or sneeze into your elbow or tissue



Keep 1.5m away from others where possible



Avoid crowding entrances and exits

sydney.edu.au/covid-19



Feeling unwell?

– Stay at home

- if you are feeling unwell with any COVID-19 symptoms
- If you have been directed to self-isolate

– Get tested

- If you are feeling unwell with COVID-19 symptoms, please get tested as soon as possible

– Did you test positive?

Yes? If you have visited campus within the infectious period, i.e. 72 hours before taking the test, you must advise the University via:

- email covid19.taskforce@sydney.edu.au, or
- call +61 2 9351 2000 (select option 1)

– Stay informed

- Monitor [the list of confirmed COVID case locations on campus page](#) to check for potential exposure and [follow NSW Health isolation and testing requirements](#).

COVID-19 support and care

- Most large lectures will be delivered online and accommodations will be made for international students who have not yet returned to Australia.
- If you become infected with COVID-19 during the semester, or need to isolate, please notify your unit of study coordinator, as with any unexpected absence.
- If COVID-19 isolation or illness impacts assessment, use the usual mechanisms, for example, special consideration to arrange reasonable adjustments.
Visit <https://www.sydney.edu.au/covid-19/students/study-information/test-exams-assessment.html#consideration>.
- Further information on student support can be found on the University website at <https://www.sydney.edu.au/students/support.html>
- Other helpful study information can be found on the website at <https://www.sydney.edu.au/covid-19/students/study-information.html>.

Emergency procedures (on campus)

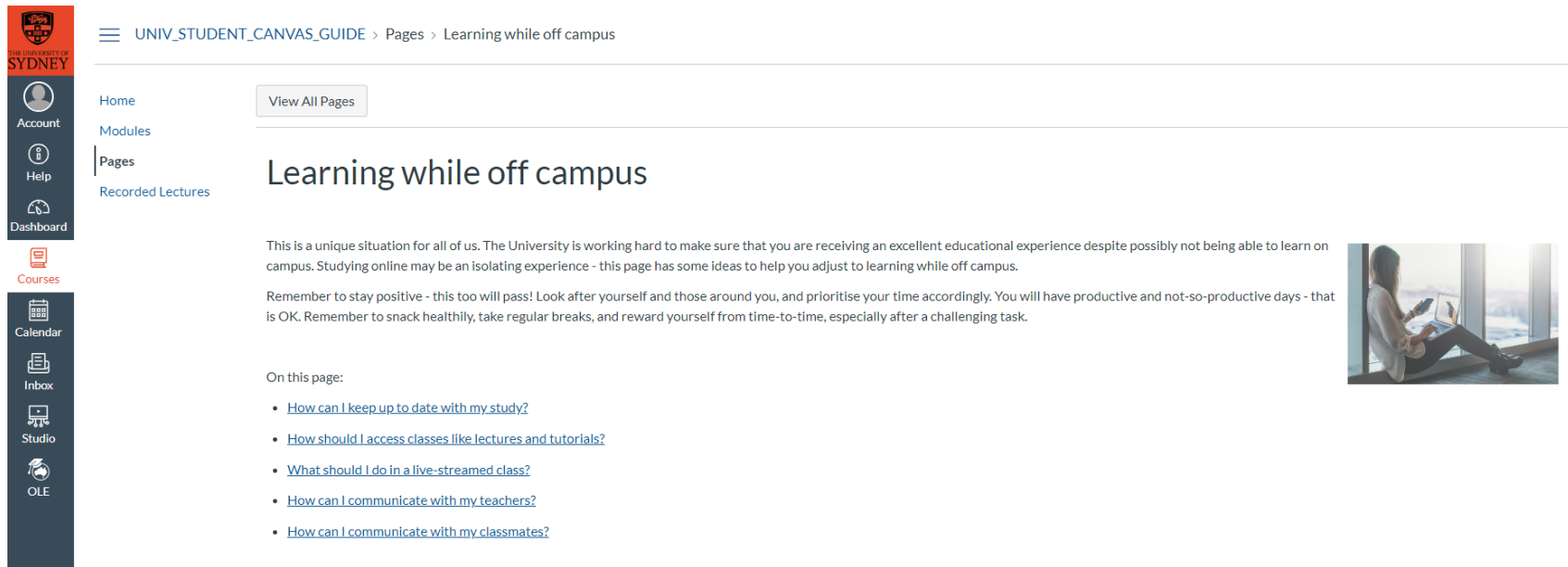
- In the unlikely event of an emergency, we may need to evacuate the building.
- If we need to evacuate, we will ask you to take your belongings and follow the green exit signs.
- We will move a safe distance from the building and maintain physical distancing whilst waiting until the emergency is over.
- In some circumstances, we might be asked to remain inside the building for our own safety. We call this a lockdown or shelter-in-place.
- More information is available at www.sydney.edu.au/emergency.

Tips for students learning online

- Remember that you are still in a space with other students.
- Mute your microphone when not speaking.
- Use earphones or headphones - the mic is better and you'll disturb others less.
- If you have a webcam, please switch it on so we can see you, if you are comfortable doing so.
- Try not to talk over someone else.
- Some classes may use breakout rooms – engaging fully in these is a great way to meet classmates and your teachers.
- Help your teachers know you're there by participating in chat, polls and other activities during class - we're all in this together.

Tips for learning online

- For tips and guides on learning online and the tools you will use, refer to [Learning while off campus resources](#) in Canvas. This is especially useful if it's your first time learning online at university.



The screenshot displays the Canvas LMS interface for The University of Sydney. On the left is a dark blue sidebar with icons for Account, Help, Dashboard, Courses, Calendar, Inbox, Studio, and OLE. The top navigation bar shows the breadcrumb 'UNIV_STUDENT_CANVAS_GUIDE > Pages > Learning while off campus'. Below this, a 'View All Pages' button is visible. The main content area is titled 'Learning while off campus' and contains the following text:

This is a unique situation for all of us. The University is working hard to make sure that you are receiving an excellent educational experience despite possibly not being able to learn on campus. Studying online may be an isolating experience - this page has some ideas to help you adjust to learning while off campus.

Remember to stay positive - this too will pass! Look after yourself and those around you, and prioritise your time accordingly. You will have productive and not-so-productive days - that is OK. Remember to snack healthily, take regular breaks, and reward yourself from time-to-time, especially after a challenging task.

On this page:

- [How can I keep up to date with my study?](#)
- [How should I access classes like lectures and tutorials?](#)
- [What should I do in a live-streamed class?](#)
- [How can I communicate with my teachers?](#)
- [How can I communicate with my classmates?](#)

An image of a student sitting on a windowsill, using a laptop, is positioned on the right side of the page.

Assistance

- There are a wide range of support services available for students
 - e.g., <http://sydney.edu.au/study/academic-support/disability-support.html>
 - it is advisable to do this as early as possible
- Please make contact, and get help
- You are not required to tell anyone else about this
- If you are willing to inform the unit coordinator, they may be able to work with other support to reduce the impact on this unit
 - eg provide advice on which tasks are most significant

Special Consideration (University policy)

- If your performance on assessments is affected by illness or misadventure
- Follow proper bureaucratic procedures
 - Have professional practitioner sign special USyd form
 - Submit application for special consideration online, upload scans
 - Note you have only a quite short deadline for applying
 - http://sydney.edu.au/current_students/special_consideration/
- Also, notify coordinator by email *as soon as anything begins to go wrong*
- There is a similar process if you need special arrangements, e.g., for religious observance, military service, representative sports

Academic Integrity (University policy)

- “The University of Sydney is unequivocally opposed to, and intolerant of, plagiarism and academic dishonesty.”
 - Academic dishonesty means seeking to obtain or obtaining academic advantage for oneself or for others (including in the assessment or publication of work) by dishonest or unfair means.
 - Plagiarism means presenting another person’s work as one’s own work by presenting, copying or reproducing it without appropriate acknowledgement of the source.” [from site below]
 - <http://sydney.edu.au/elearning/student/EI/index.shtml>
- Submitted work is compared against other work (from students, internet, etc)
 - Turnitin for textual tasks (through eLearning), other systems for code
- Penalties for academic dishonesty or plagiarism can be severe
- Complete self-education AHEM1001 (required, canvas → assessment info)

What is academic dishonesty?

The following are some behaviours that are academically dishonest:

- **Plagiarism** (this is the most common form)
- **Collusion** or illegitimate co-operation
- **Recycling** (using your own work from previous assessments)
- **Cheating**, including **contract cheating**
 - sharing questions or accessing solutions on online “help sites”
 - receiving coaching from a private tutoring company on how to complete an assignment
 - asking someone else to write your assignment (for payment or not)
- **Exam cheating** (using prohibited materials, working with others)
- **Fabrication** or falsification of sources, data or results

What are the consequences?

- The University has strong mechanisms for detection of potential academic dishonesty.
- Suspected breaches are reported to the faculty educational integrity team for investigation.
- The University is deeply committed to ensuring the integrity of its educational programs and treats integrity breaches seriously. As a result, the **academic consequences** for cheating are numerous.
- You may:
 - need to resubmit a task with a mark penalty or
 - receive a 0 for the assessment or even the unit of study
 - be suspended or even excluded from your studies for serious misconduct

Unit of Study Overview



Prerequisites

- This course has the following prerequisites:
 - INFO1113 OR INFO1103 OR INFO1105 OR INFO1905
- This means that we expect students who enroll in this course to be:
 - Confident Java programmers
 - Familiar with data-structures
- Prohibitions
 - INFO3220 OR COMP9201 (for SOFT2201 students)
 - INFO3220 OR SOFT2201 (for COMP9201 students)

SOFT2201/COMP9201 Lectures

- Online Lecture:
 - Monday, 2pm to 4pm (AEST), Zoom (w1 to w13)
- When you have questions during lectures:
 - if questions are public, feel free to type them on zoom chat and I will read and answer them during the lecture
 - if questions are private, please take a quick note and post it on Ed forum **in private mode after lecture**. I will then answer your private questions there
- How you are suggested to use the Zoom chat:
 - Highly appreciate it that you use the Zoom chat only for the lecture related stuff

SOFT2201/COMP9201 Tutorials

- Tutorial:
 - 2h tutorial, hybrid mode, from Tuesdays to Fridays (w2 to w13)
 - Check your timetable
 - Attend ONE (go to the tutorial you're scheduled for)
 - Great opportunity for interactive and hands-on practical exercises
 - Tutors to supervise your learning and provide guidance
 - Not to debug your code, or solve the problems for you
 - Respect your tutors and value their feedback
 - Respect your classmates

SOFT2201/COMP9201 Staff

Course Coordinator and Lecturer:

- Dr. Xi Wu (xi.wu@sydney.edu.au)
 - Consultation: Monday, 4pm to 5pm (AEST), the same Zoom link for lectures

Tutors:

Abbey Lin	Alex Maher
Daniel Friedrich	Dylan Williams
Hetush Gupta	Liza Fishman
Rayman Tang	Tim Yarkov

Tutor allocation can be found via [Staff and Contact Details](#) on canvas

Language and Tools

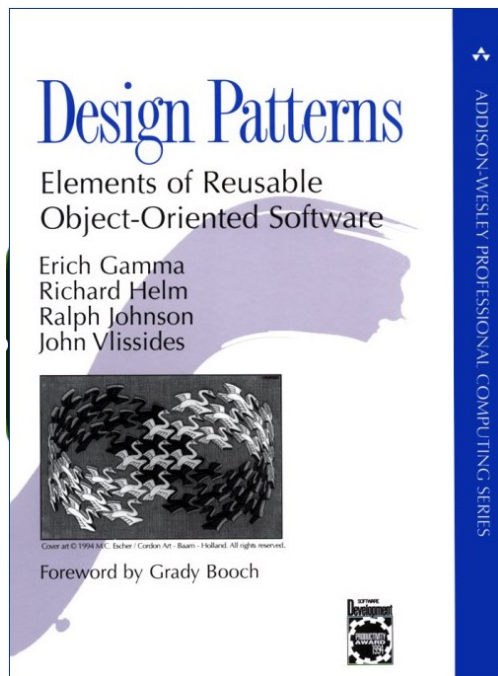
- Your coding will be in Java 17
- You will use IntelliJ (2022.1) for writing your code
- You will use JavaFX for designing your GUIs
- You will use supporting tools such as Gradle, Git, etc.

SOFT2201/COMP9201 Resources

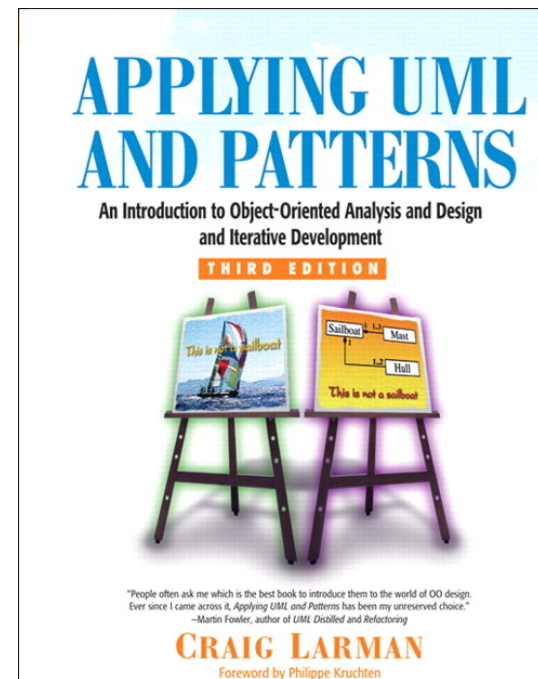
- Canvas (eLearning)
 - Announcements (**important**)
 - Modules: Copies of lecture slides, Tutorial instructions
 - Assignments: Assignment instructions
 - Recorded Lecturers
 - Lecture recordings (usually upload after 1~2 hours of the live lectures)
 - Zoom: Live lectures and tutorials
 - Ed
 - The best place to ask **technical questions** about the course.
 - You will get faster answers here from staff and peers than through email.
 - You can also ask private questions in the private mode

Main Resources

- We recommend the following textbooks



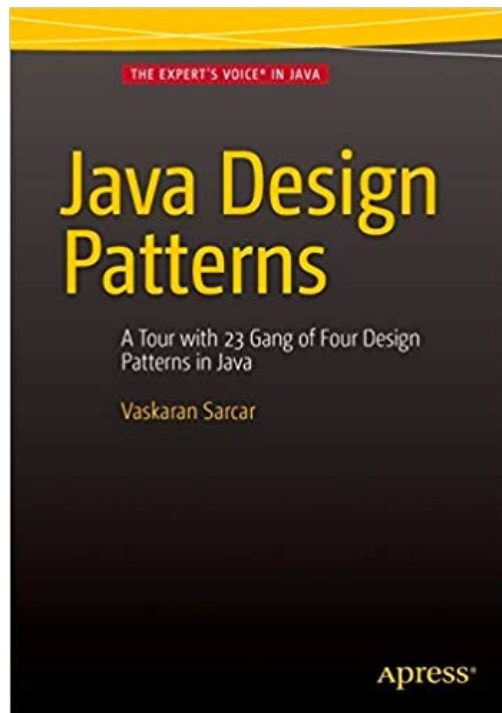
[Link to USYD Library](#)



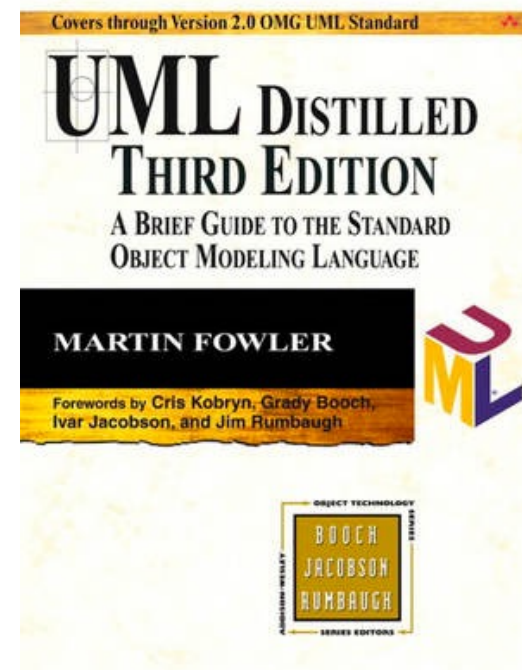
[Link to USYD Library](#)

Additional Resources

- We recommend the following textbooks



[Link to USYD Library](#)



[Link to USYD Library](#)

Topics Overview

Week	Contents		Week
1	Introduction; Software Engineering, Software Modeling, The Unified Process		
2	OO Theory I: Java Knowledge Revisited		
3	UML & Software Modelling Case Studies		
4	OO Theory II: Software Design Principles and Design Smells		
5	Design Pattern: Factory Method & Builder	Design Pattern: Strategy & State	6
7	Testing	Code Review	8
10	Design Pattern: Adapter & Observer	Design Pattern: Prototype & Memento	11
12	Design Pattern: Singleton, Decorator and Facade		
13	Unit Review		

Assessment

What (Assessment)*	When (due)	How	Value
Weekly Exercises	Weekly	Submission on Canvas	10%
Software Construction Stage 1	Week 4	Individual Submission on Canvas	5%
Software Construction Stage 2	Week 8	Individual Submission on Canvas	15%
Software Construction Stage 3	Week 12	Individual Submission on Canvas	20%
Exam	Exam period	Individual exam	50%

Weekly Exercises

- Weekly exercises can be found under each week module on Canvas.
- Exercises cover contents from recent tutorials and lectures
- Timeframe (from week 2 to week 12):
 - Exercises and submission portals are released at 8am (AEST) Tuesday
 - Exercises due and submission portals are closed at 23.59pm (AEST) Saturday
 - You can do the exercise any time after it is open and before its due
 - **However, once it is closed, without CON, you cannot re-take it if you miss it**
 - **However, only 10 with highest marks out of 11 will be count for final mark**
- Total marks: 10%

Assignment

- Three individual assignments (40% marks):
 1. Design UML diagrams for a given scenario
 - You are going to submit UML diagrams + presentation recording (5 minutes)
 2. Implement a program for a given scenario
 - You are going to submit implementation code with test cases + report about the design principles/design patterns in your implementation
 3. Review and extend somebody else's code
 - You are going to submit implementation code + report about the design principles/design patterns of the receiver code and your extensions

Late Assignments

- Suppose you hand in your assignments after the deadline:
- If you have not been granted special consideration or arrangements
 - A penalty of 5% of the maximal mark will be taken, per day late
 - E.g., your work would have scored 60% and is 1 day late you get 55%
 - **Assignments after 10 calendar days late get 0**
 - Late penalty doesn't apply to weekly exercises; you cannot re-take the exercise if you miss it
- Warning: submission sites get very slow near deadlines
 - Submit early; you can resubmit if there is time before the deadline

Online Final Exam

- 2hrs, will be organized by Exam Office during Exam period
- More details will be announced at the end of the semester (w13)
- Total marks: 50%

Passing this unit

- To pass this unit you must do all of these:
 - Get a total mark of at least 50%
 - Get at least 40% for your final exam mark

SOFT2201/COMP9201 Expectations

- Students attend scheduled classes, and devote an extra 10 hours per week
 - Prepare and review lecture and tutorial materials
 - Revise and integrate with ideas
 - Carry on the required assessments
 - Practice and self-assess
- Students are responsible learners
 - Participate in classes, constructively
 - Respect for one another (criticize ideas, but not people)
 - Humility: none of us knows it all; each of us knows valuable things
 - Check eLearning site very frequently
- Know the Uni/school policies

Q&A and Feedback

- A discussion forum is setup:
 - Please **use ED for technical questions** so that everybody can benefit from the questions and answers
- Talk to us:
 - After lectures/tutorials
 - During teaching staff consultation
- Feedback to you will take many forms:
 - Verbally by your tutor
 - As comments accompanying hand marking of your assignment work and exercises work

Advice for doing well in this unit

- To do *well* in this unit you should
 - Organize your time well
 - Devote extra 10 hours a week in total to this unit
 - Read.
 - Think.
 - Practice.

“Tell me and I forget, teach me and I may remember, involve me and I learn.” – Benjamin Franklin

Why Software Engineering?



Software is Everywhere!

- Societies, businesses and governments dependent on SW systems
 - Power, Telecommunication, Education, Government, Transport, Finance, Health
 - Work automation, communication, control of complex systems
- Large software economies in developed countries
 - IT application development expenditure in the US more than \$250bn/year¹
 - Total value added GDP in the US²: \$1.07 trillion
- Emerging challenges
 - Security, robustness, human user-interface, and new computational platforms

¹ Chaos Report, Standish group Report, 2014

² softwareimpact.bsa.org

Why Software Engineering?

Need to build high quality software systems under resource constraints

- **Social**
 - Satisfy user needs (e.g., functional, reliable, trustworthy)
 - Impact on people's lives (e.g., software failure, data protection)
- **Economical**
 - Reduce cost; open up new opportunities
 - Average cost of IT development ~\$2.3m, ~\$1.3m and ~\$434k for large, medium and small companies respectively³
- **Time to market**
 - Deliver software on-time

³Chaos Report, Standish group Report, 2014

Software Failure - Ariane 5 Disaster

What happened?

- European large rocket - 10 years development, ~\$7 billion
- Unmanaged software exception resulted from a data conversion from 64-bit floating point to a 16-bit signed integer
- Backup processor failed straight after using the same software
- Exploded 37 seconds after lift-off



Why did it happen?

- Design error, incorrect analysis of changing requirements, inadequate validation and verification, testing and reviews, ineffective development processes and management

<http://iansommerville.com/software-engineering-book/files/2014/07/Bashar-Ariane5.pdf>

Nissan Recall – Airbag Defect

What happened?

- ~3.53 million vehicles recall of various models 2013-2017
- Front passenger airbag may not deploy in an accident

Why did it happen?

- Software that activates airbags deployment improperly classify occupied passenger seat as empty in case of accident
- Software defect that could lead to improper airbag function (failure)
- No warning that the airbag may not function properly
- Software sensitivity calibration due to combination of factors (high engine vibration and changing seat status)

<http://www.reuters.com/article/us-autos-nissan-recall/nissan-to-recall-3-53-million-vehicles-air-bags-may-not-deploy-idUSKCN0XQ2A8>

<https://www.nytimes.com/2014/03/27/automobiles/nissan-recalls-990000-cars-and-trucks-for-air-bag-malfunction.html>

Software Project Failures

Project	Duration	Cost	Failure/Status
e-borders (UK Advanced passenger Information System Programme)	2007 - 2014	Over £ 412m (expected), £742m (actual)	Permanent failure - cancelled after a series of delays
Pust Siebel - Swedish Police case management (Swedish Police)	2011 - 2014	\$53m (actual)	Permanent failure – scraped due to poor functioning, inefficient in work environments
US Federal Government Health Care Exchange Web application	2013 – ongoing	\$93.7m (expected), \$1.5bn (actual)	Ongoing problems - too slow, poor performance, people get stuck in the application process (frustrated users)

https://en.wikipedia.org/wiki/List_of_failed_and_overbudget_custom_software_projects

Software Engineering - No Silver Bullet⁷

No Silver Bullet - Essence and Accidents in Software Engineering

“There is no single development, in either technology or management technique, which by itself promises even one order-of-magnitude improvement within a decade in productivity, in reliability, in simplicity.” ! - Frederick P. Brooks

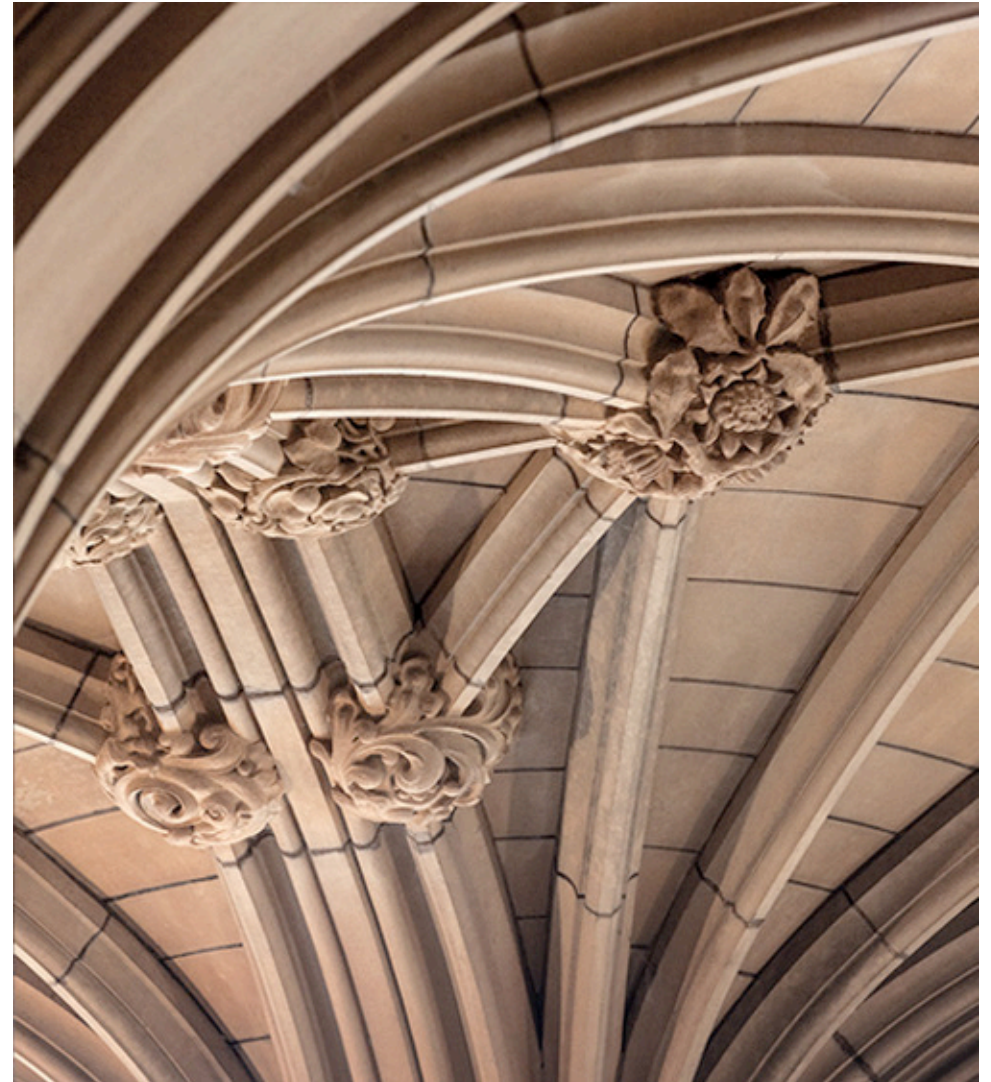
- **Essence:** difficulties inherent (or intrinsic) in the nature of SW
- **Accidents:** difficulties related to the production of software
- *Most techniques attack the accidents of software engineering*

⁷ No Silver Bullet - Essence and Accident in Software Engineering - <http://www.itu.dk/people/hesj/BSUP/artikler/no-silver-bullit.pdf>

Software Engineering - Essence⁷

- **Complexity**
 - Many diverse software entities - interactions increase exponentially
 - Intrinsic complexity cannot be abstracted - aircraft software, air traffic control
- **Conformity**
 - Arbitrary changes from environment (people, systems) - no unifying principle
- **Changeability**
 - Changing a building model vs. a software
 - Stakeholders understanding of software changes
- **Invisibility**
 - Software is intangible (invisible)
 - Building model vs software models (UML - 13 diagram types)

What is Software Engineering?



What is Software Engineering?



<http://www.purplesoft.com.au/wp-content/uploads/2017/03/software.jpg>

Software Engineering

“An engineering discipline that is concerned with all aspects of software production from the early stages of system specification through to maintaining it after it has gone into use.”

- **NOT** programming/coding! a lot more is involved
- Theories, methods and tools for cost-effective software production
- Technical process, project management and development of tools, methods to support software production
- System Engineering (Hardware & Software) - software often dominates costs

Software Engineering

- Theories, methods, tools, techniques and approaches
 - Solve concrete SWEN problems
 - Increase productivity of the engineering process
 - Produce effective software
 - Produce efficient software
 - Control social and technical aspects in the development process
 - Manage complexity, changeability, invisibility and conformity

Software Engineering

“The Roman bridges of antiquity were very inefficient structures. By modern standards, they used too much stone, and as a result, far too much labour to build. Over the years we have learned to build bridges more efficiently, using fewer materials and less labour to perform the same task.” !

- Tom Clancy (The Sum of All Fears)

- The art of managing social, economical and technical factors
 - Efficient and effective development processes and management
 - Delivering software on-time and on-budget with all the necessary requirements
- The art of analysing and managing complexity
 - Ability to understand complex systems
 - Apply various abstraction and analytical thinking

Software Engineering Fundamentals

- Software processes for managing and developing of SW systems
 - Waterfall vs. Incremental and agile software development
- Attributes of good software system
 - Maintainability, dependability and security, efficiency and acceptability
- Importance of Dependability and Performance
- Need for specifications and requirements
- Software reuse to save costs
 - Careful consideration – Ariane 5 reused software from Arian 4!

Software Engineering Body of Knowledge

- Software Requirements
- **Software Design / Modelling**
- **Software Construction**
- Software Testing
- Software Maintenance
- Software Configuration Management
- Software Engineering Process
- Software Engineering Tools and Methods
- Software Quality



Software Engineering Body of Knowledge (SWEBOK) <https://www.computer.org/web/swebok/>

Software Design/Modelling & Construction



Software Modelling

- The process of developing conceptual models of a system at different levels of abstraction
 - Fulfil the defined system requirements
 - Focus on important system details to deal with complexity
- Object-oriented design approach
 - Concepts in the domain problem are modelled as interacting objects
- Using graphical notations (e.g., UML) to capture different views or perspectives

Software Modelling – The Art of Abstraction

- Conceptual process to derive general rules and concepts from concrete information in a certain context
- Analysis and understanding of domain-specific problems
 - Decompose large problems into smaller understandable piece
 - Language required to break down complexity, i.e., find abstractions
- SW models with different levels of abstraction
 - Focus on certain details in each phase of the SW development process
- SW models with different views/perspectives
 - Time, structural, requirements, etc.

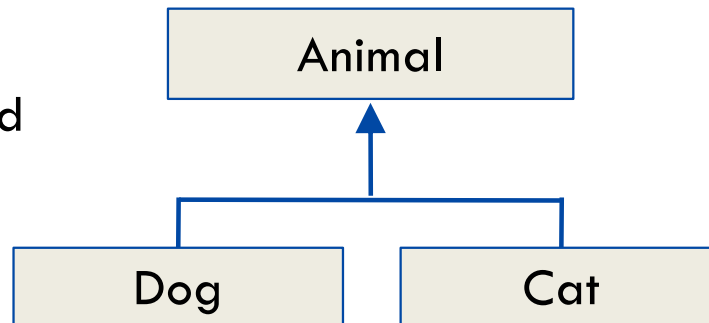
Software Abstraction – Example

Real world



Abstraction

Software world



```
1 public class Animal {  
2     public void sleep () {}  
3 }  
4  
5 public class Dog extends Animal {  
6     public void woof {}  
7 }  
8  
9 public class Cat extends Animal {  
10     public void meow {}  
11 }
```

Cat <http://s1.thingpic.com/images/ZP/L4FtpQYKNZzCXV8r34PWhqCF.jpeg>

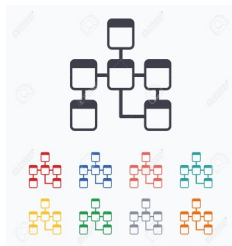
Dog [https://s7d2.scene7.com/is/image/PetSmart/SV0401_CATEGORY_HERO-Dog-Dog-20160818?\\$SV0401\\$](https://s7d2.scene7.com/is/image/PetSmart/SV0401_CATEGORY_HERO-Dog-Dog-20160818?$SV0401$)

Data Abstraction – Example

View Table Data - MRCWORKLIB.DMCMP100

215 Items Page 1 of 22 Page size: 10

Customer Number	Company Number	Customer Name	Customer Address Line 1	Customer address line 2
EQ	EQ	EQ	EQ	EQ
100001	25	BIKES R US	495 GRANVILLE STREET	SUITE 2
100002	25	AAM-EQUIPTO BIKE SALES	2412 FAR HILLS AVE.	SUITE 214
100003	25	ABCO CORPORATION	1870 S. HIGH ST.	SUITE 5
100004	25	AB CYCLE SHOPPE	535 S. FRONT STR.	
100005	25	ACE HARDWARE	899 BROADWAY	
100006	25	STATE OF NEW YORK	163 WEST 22ND ST	
100007	25	A-1 CYCLE CITY	500 W NORTH AVE	
100008	25	BERNARDS SCHWINN CYCLERY	510 FREDERICK ST	
100009	25	WESTFORTH SPORTS INC	4704 ROOSEVELT PL	
100010	25	GRAND CYCLE	1417 CLARK STREET	



View level



Logical level
(conceptual data
model)



Physical level
(data model)



UML Principles



- Graphical notations to visually specify and document design artifacts software systems using OO concepts
 - Industry standard managed by Object Management Group (OMG)
 - Is not OO Analysis and Design (OOA&D) or method!
 - Is a language for OOA&D communicating visual models of the software being designed
 - Is not a skill, but how to design software with different level of abstractions
 - Many software diagramming tools, hand sketches are good too
- Combines techniques from data modeling (ER diagrams), business modeling (workflows), object and component modeling
- Capture system activities (jobs), system components and its interaction, software entities and its interaction, run-time behavior, external interfaces

UML Principles (Cont.)



- UML is not “Silver Bullet”
 - No tool/technique in software will make dramatic order-of-magnitude difference in productivity, reliability or simplicity
 - Design knowledge and thinking in objects is critical
- Three ways to apply UML
 - Sketching to explore part of the problem or solution space
 - Blueprint: detailed design diagrams for:
 - Reverse engineering to visualize and understand existing code
 - Forward engineering (code generation)
 - Programming language (Model Driven Engineering): executable specification of a system
- “Agile Modeling” emphasizes UML as a sketch
 - Not waterfall mindset

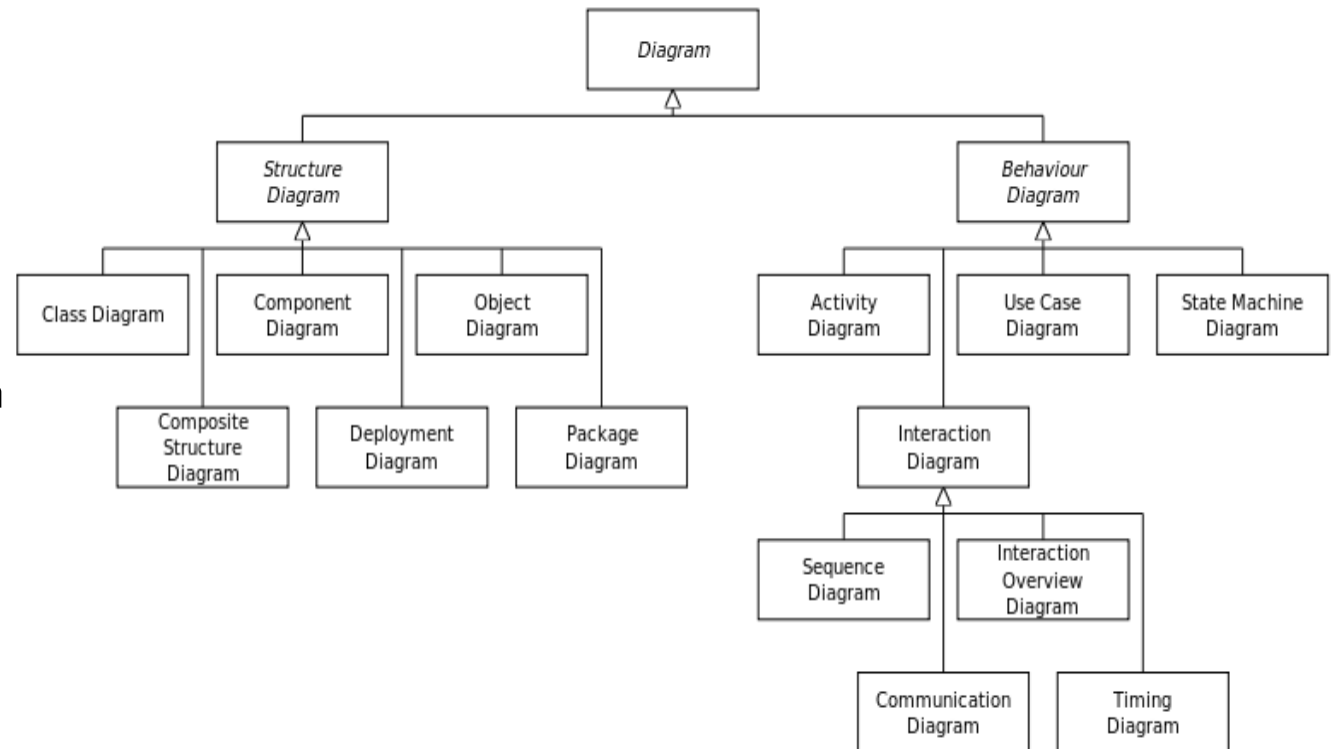
UML Diagrams

Structural (static) View

- Static structure of the system (objects, attributes, operations and relationships)

Behavioural (dynamic) View

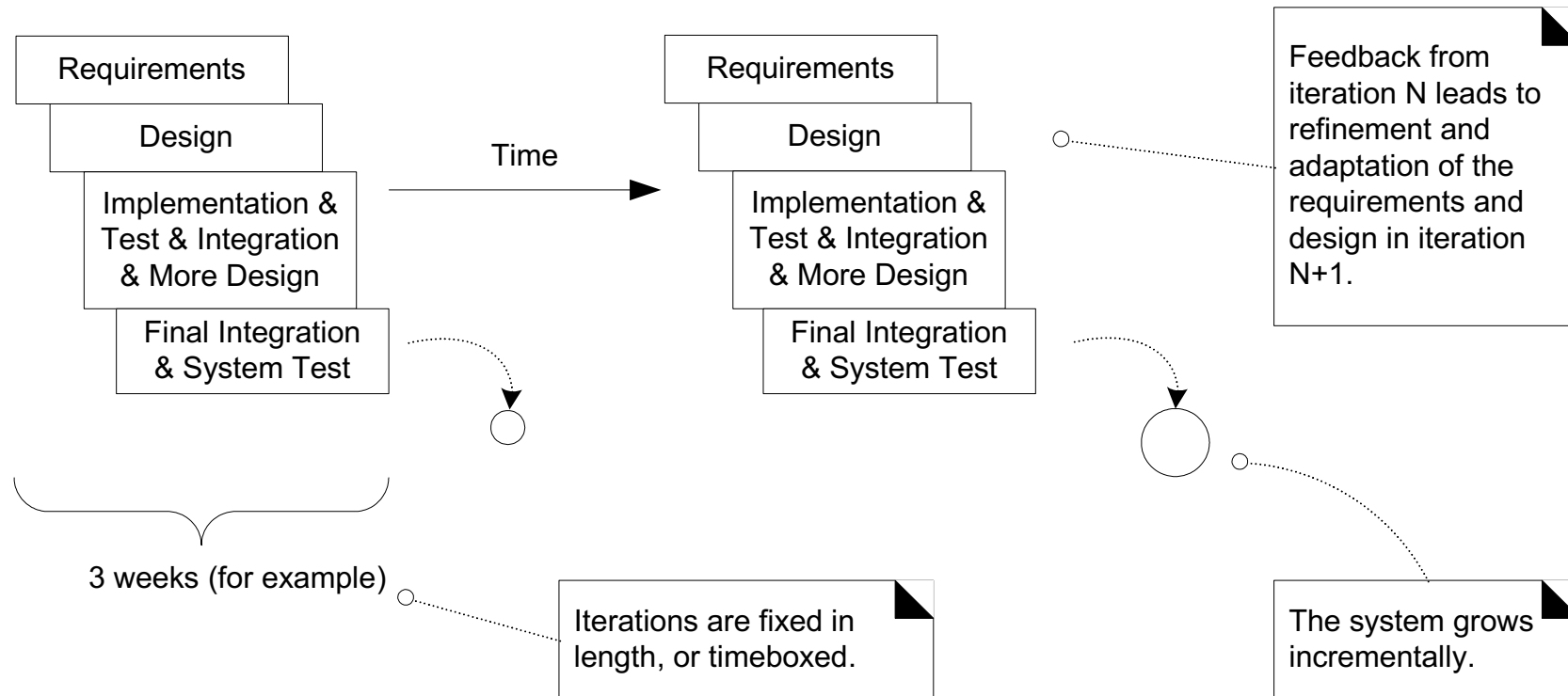
- Dynamic behavior of the system (collaboration among objects, and changes to the internal states of objects)
- Interaction (subset of dynamic view) - emphasizes flow of control and data



The Rational Unified Process

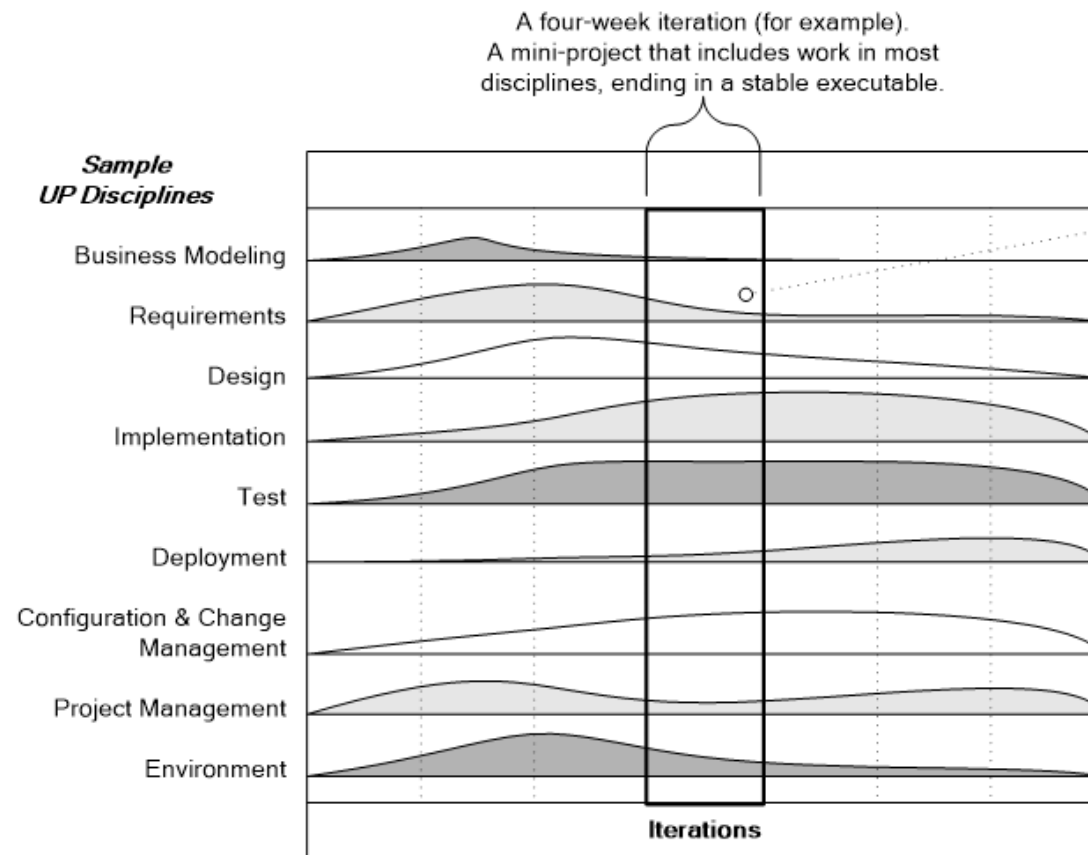


Iterative and Evolutionary Development



Rational Unified Process (UP)

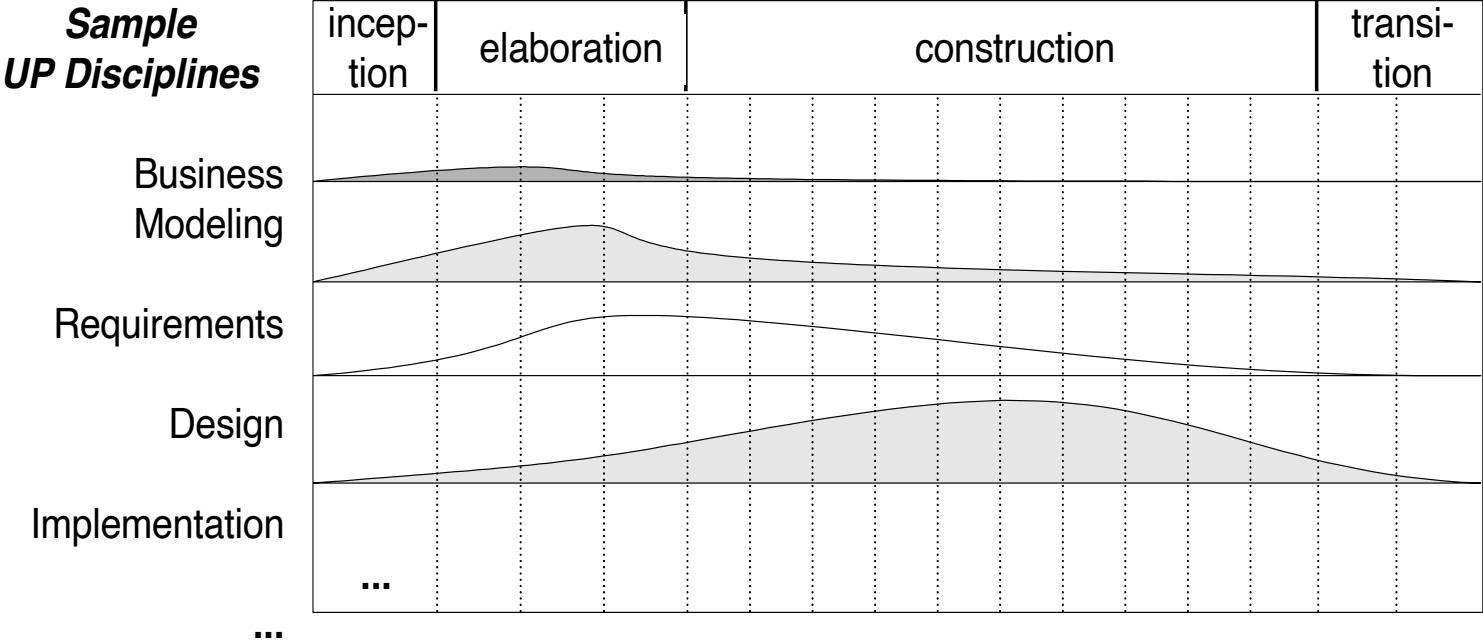
- Software development process utilizing iterative and risk-driven approach to develop OO software systems
- Iterative incremental development
- Iterative evolutionary development



Note that although an iteration includes work in most disciplines, the relative effort and emphasis change over time.

This example is suggestive, not literal.

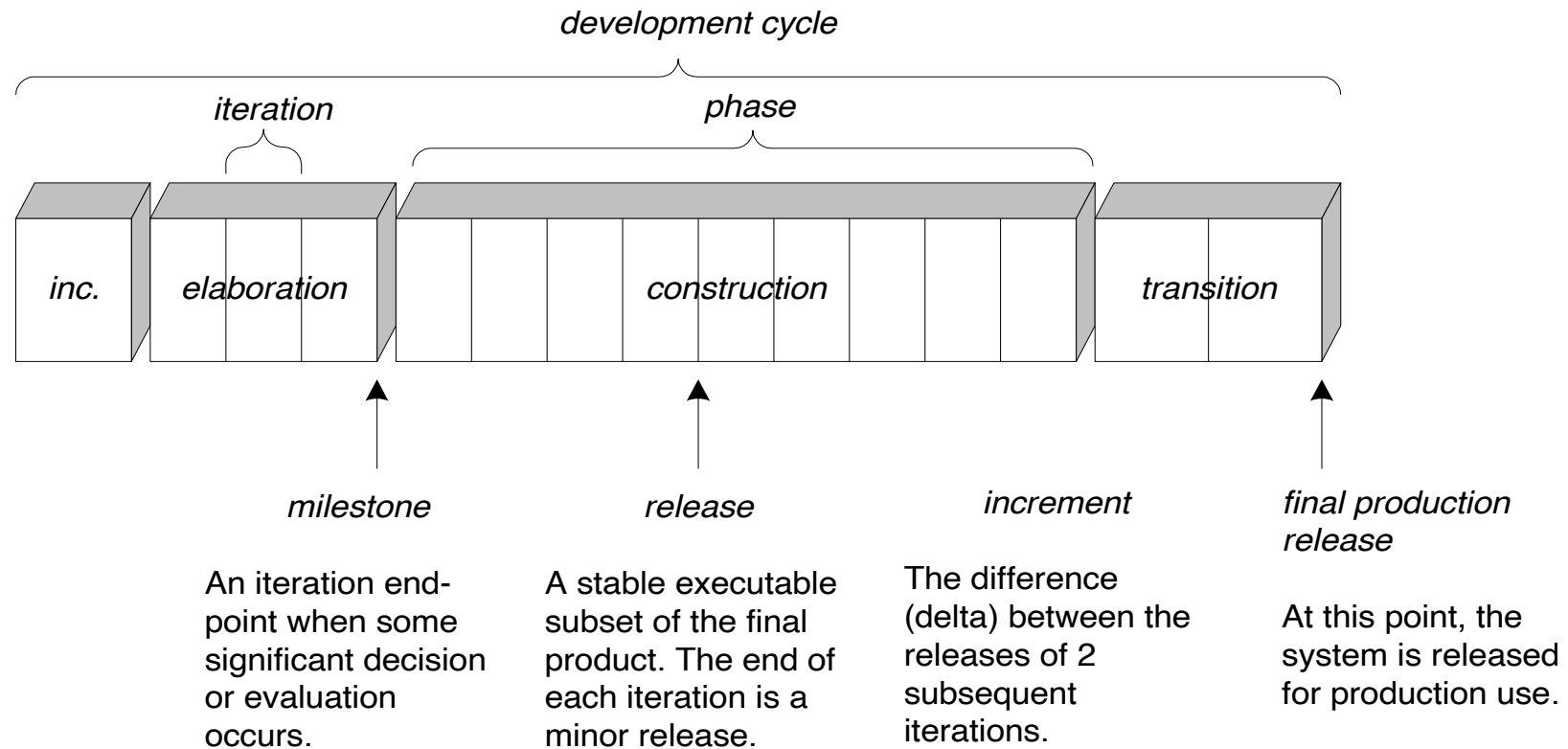
UP Phases and Disciplines



The relative effort in disciplines shifts across the phases.

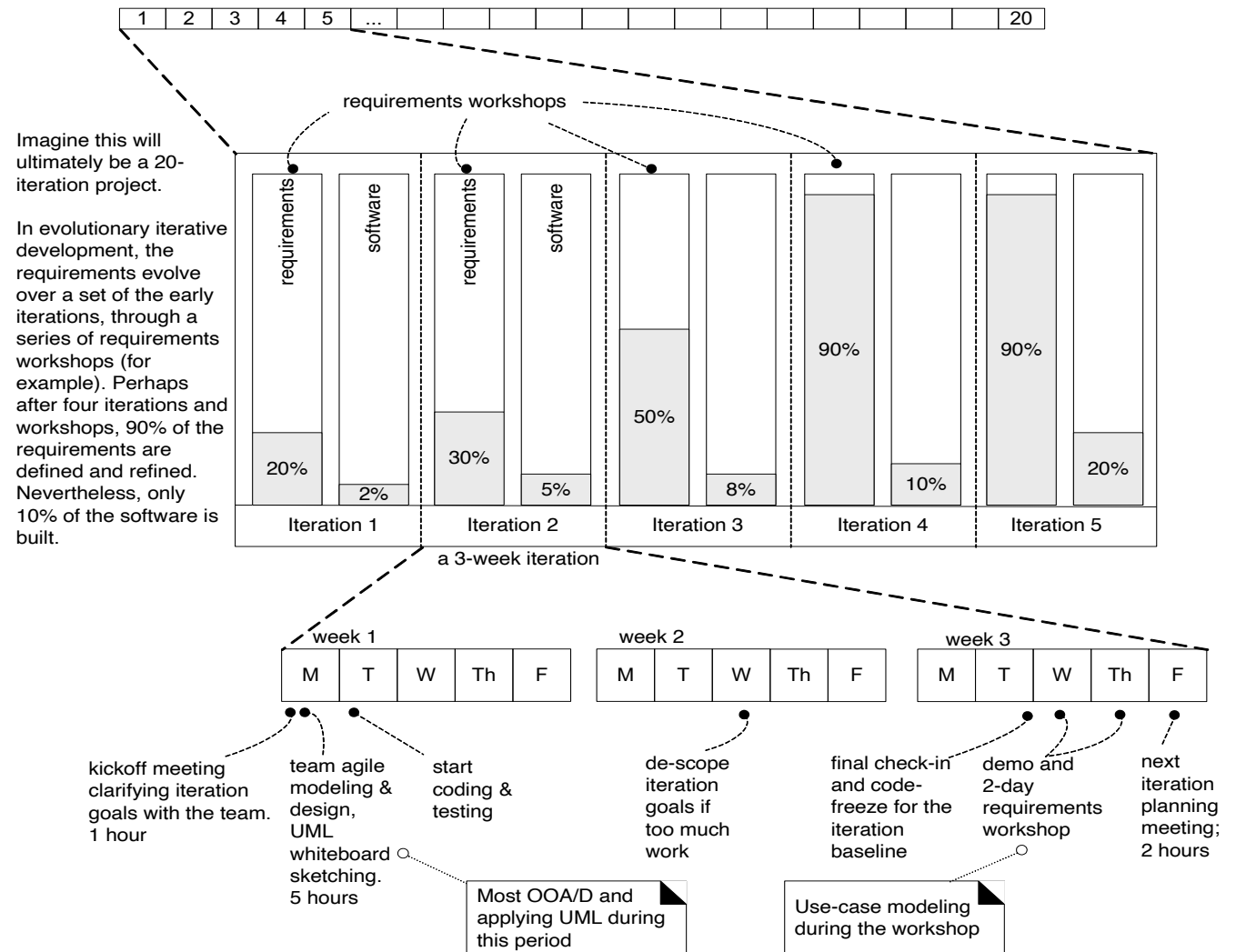
This example is suggestive, not literal.

UP Phases and Disciplines



UP - Example

- Iterative and evolutionary analysis and design
 - First five iterations of 20
 - 3-week iteration



Software Construction / Implementation

- Realization of design to produce a working software system
 - Meet customer requirements
- Design and implementation activities often interleaved
 - Agile development to accommodate for changes
- Object-Oriented design and Implementation model
 - Encapsulation
 - Abstraction
 - Reuse
 - Maintenance

Tasks for Week 1

- Install gradle and IntelliJ
 - Instruction can be found in Ed announcement (Tuesday Morning)
 - Get help from staffs on Ed forum if you have questions on installation

What are we going to learn next week?

- OO Theory I: Java Knowledge Revisited
 - Encapsulation
 - Inheritance
 - Variable Binding & Polymorphism
 - Virtual Dispatch
 - Abstract Classes
 - Interfaces