Algorithms and Distributed Systems 2019/2020 (Lecture Zero)

MIEI - Integrated Master in Computer Science and Informatics

Specialization block

João Leitão (jc.leitao@fct.unl.pt)

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

Lecture Zero

- Context of the Course
- Course Structure
- Project (some news here)
- Evaluation Rules

Context of the Course

- We live in very interesting times...

Context of the Course

- We live in very interesting times...

Context of the Course

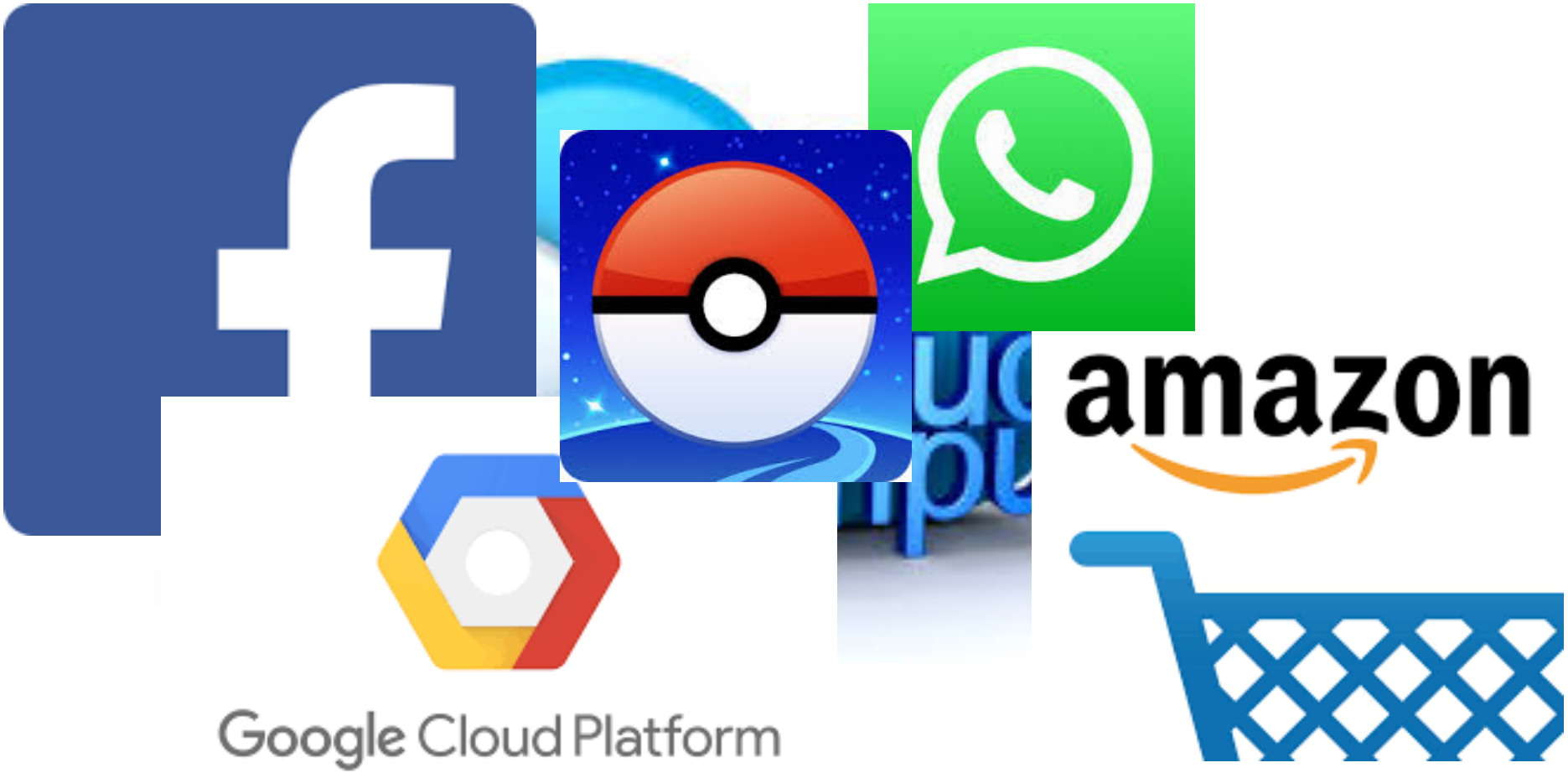
- We live in very interesting times...

Context of the Course

- We live in very interesting times...

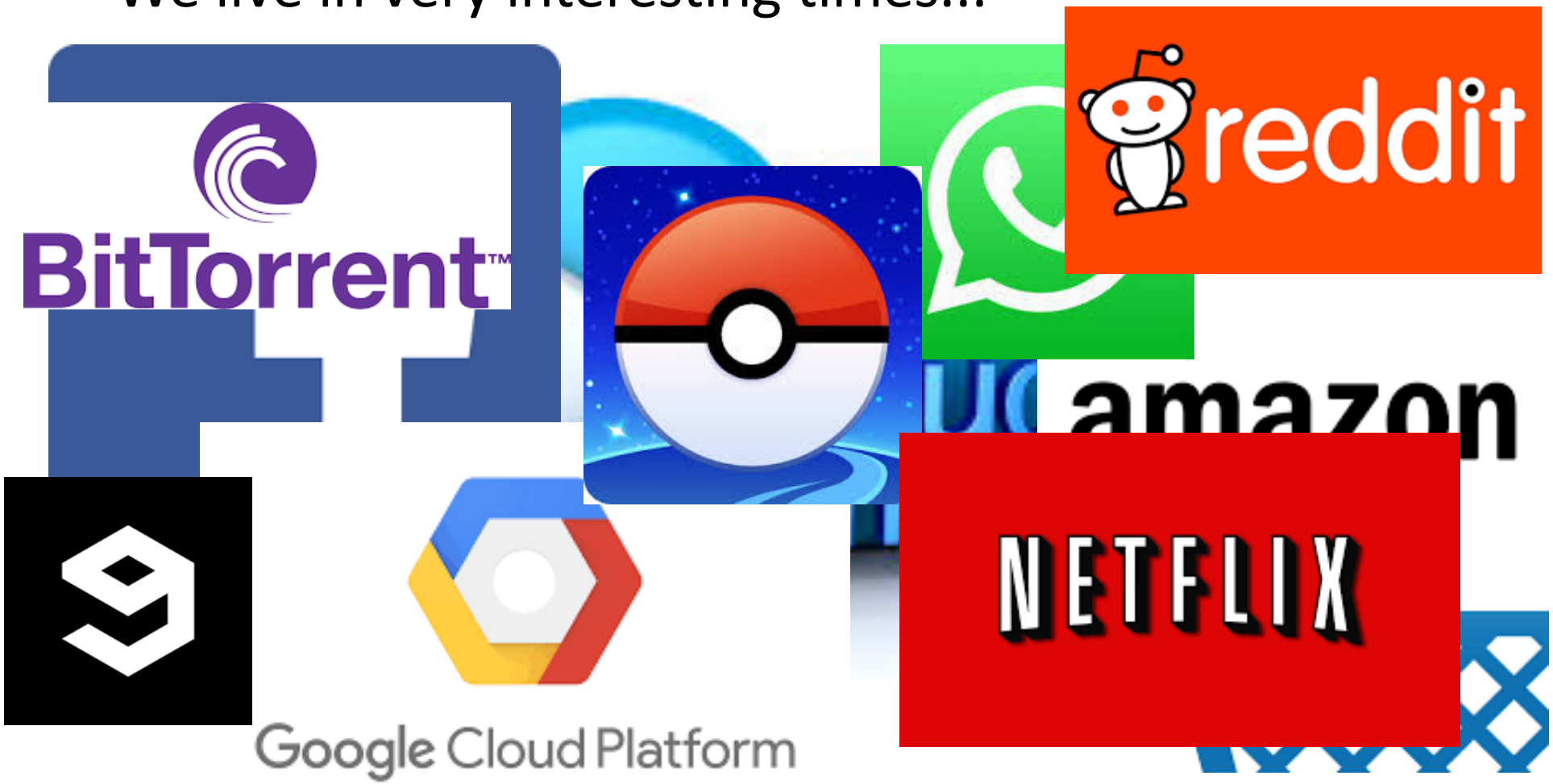
Context of the Course

- We live in very interesting times...



Context of the Course

- We live in very interesting times...



Context of the Course

- We live in very interesting times...

Google Cloud

Context of the Course

- Virtually all of these applications and systems have aspects in common:
  - They have huge (global-scale) user base (and users are so annoying with all their demands and expectations).
  - They are composed of a myriad of services (storage systems, web services, membership services, ...)
  - They are materialized by a huge number of machines (most of the times scattered through-out the world).
  - They are very very profitable (with maybe one or two exceptions).

Context of the Course

- A key commonality, is developers of such applications/systems have to deal with similar problems:
  - How do you know what are the processes (i.e, application components) that are part of your system at a given point in time?
  - How do you propagate information to a large number of processes?
  - How do you ensure that user data is not lost?
  - How do you ensure that different processes do not make different decisions and mess-up?
  - How do you know if a component (e.g., server) is still active?

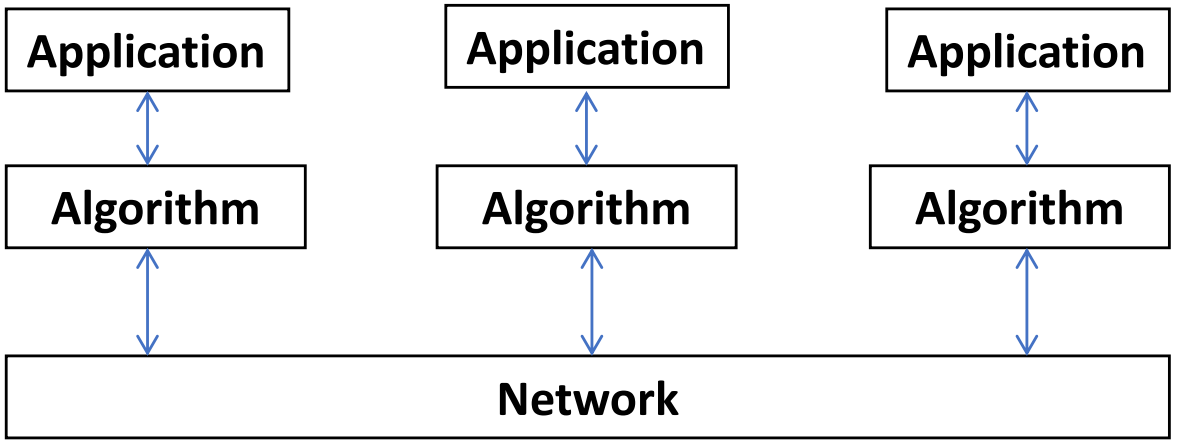
Context of the Course

- Dealing with these problems require:
  - Option 1: Throwing into a bag a lot of *very enligtned (shoud be read "pseudo-arbitrary")* lines of codes that try to deal with every possible bad scenario.
  - Option 2: Understanding these problems, understanding under which conditions they can be solved, employing verified and correct (modular) solutions.

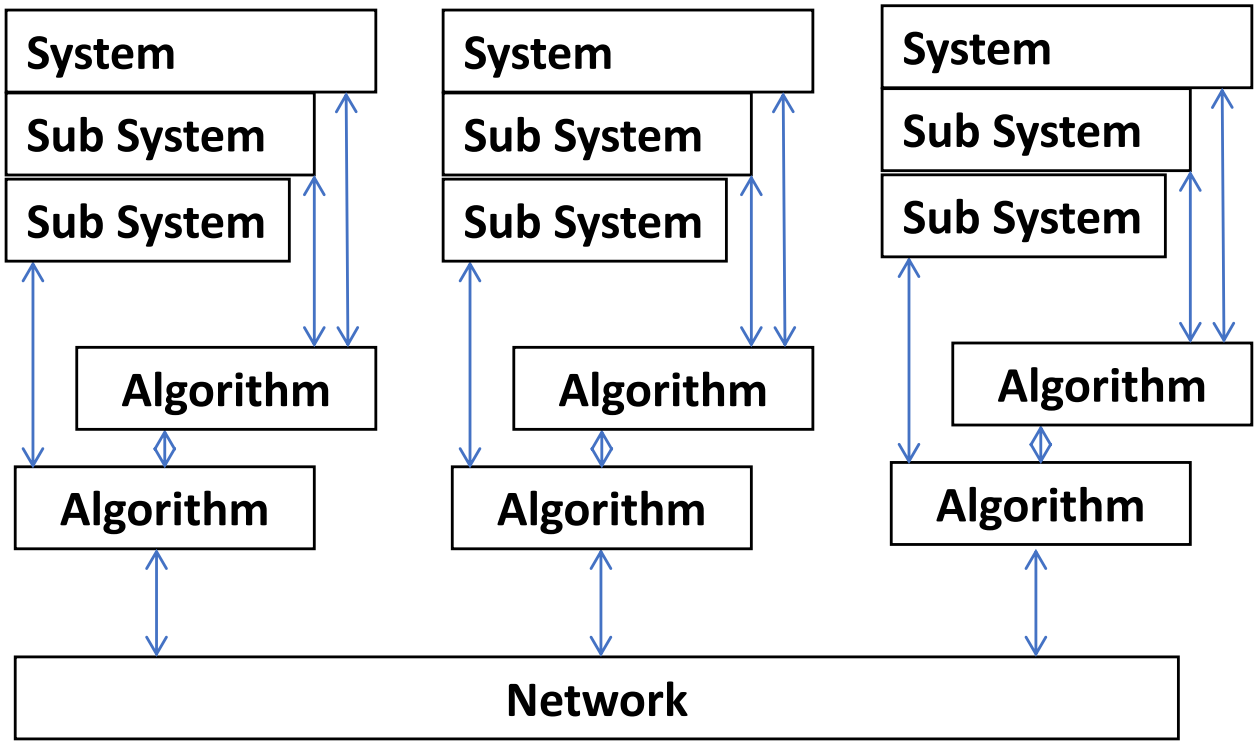
Context of the Course

- Dealing with these problems require:
  - ~~Option 1: Throwing into a bag a lot of *very enligtned (shoud be read "pseudo-arbitrary")* lines of codes that try to deal with every possible bad scenario.~~
  - Option 2: Understanding these problems, understanding under which conditions they can be solved, employing verified and correct (modular) solutions.

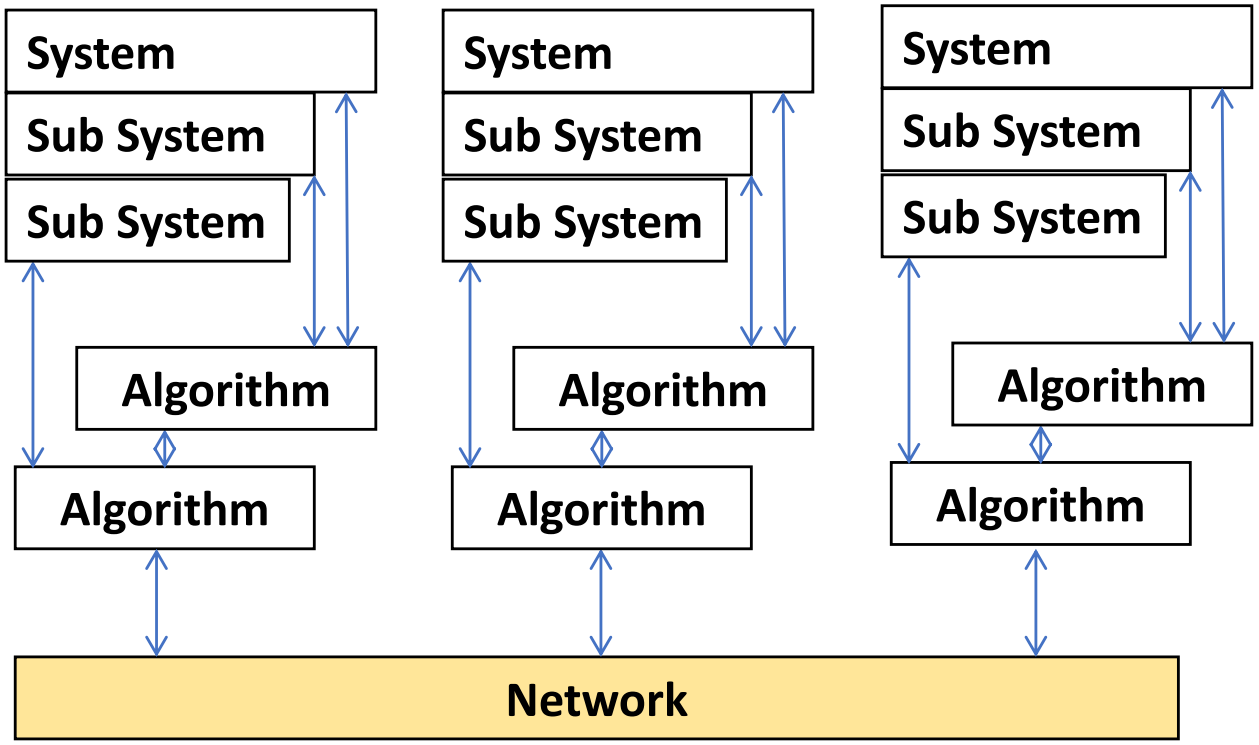
Distributed Algorithms (and Protocols)



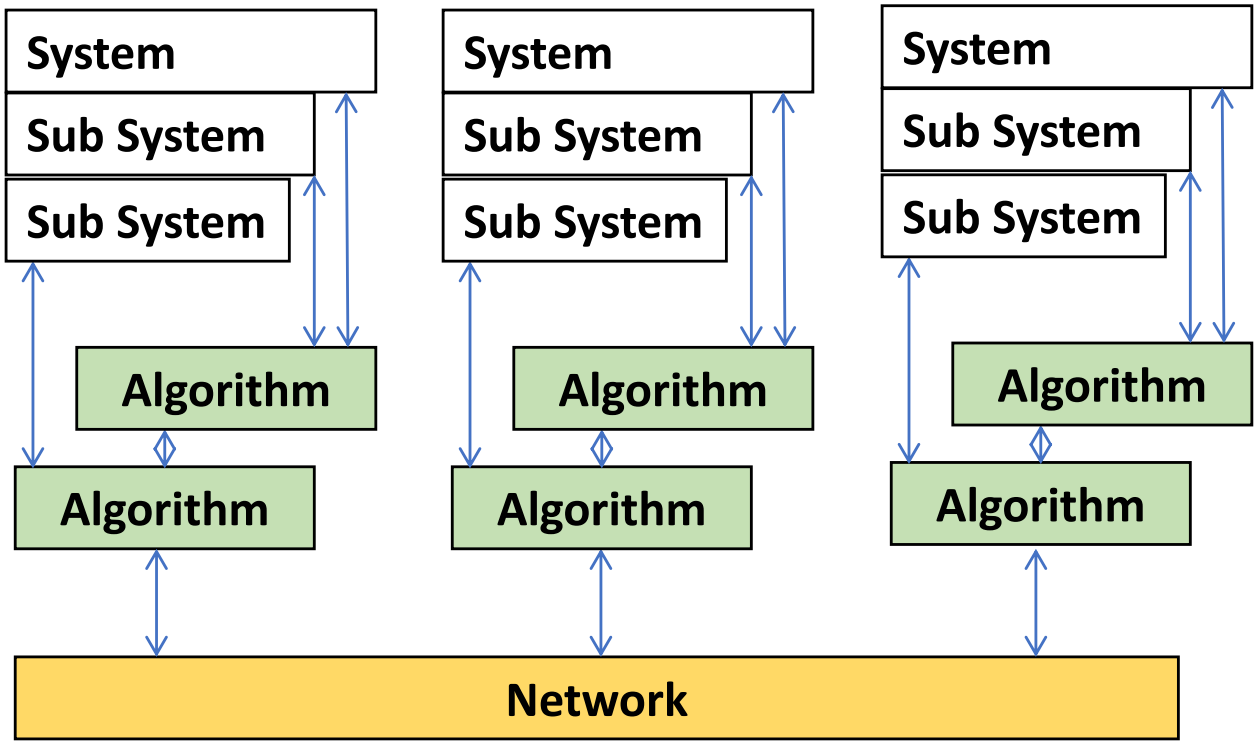
Distributed Algorithms (and Protocols) – More realistically…



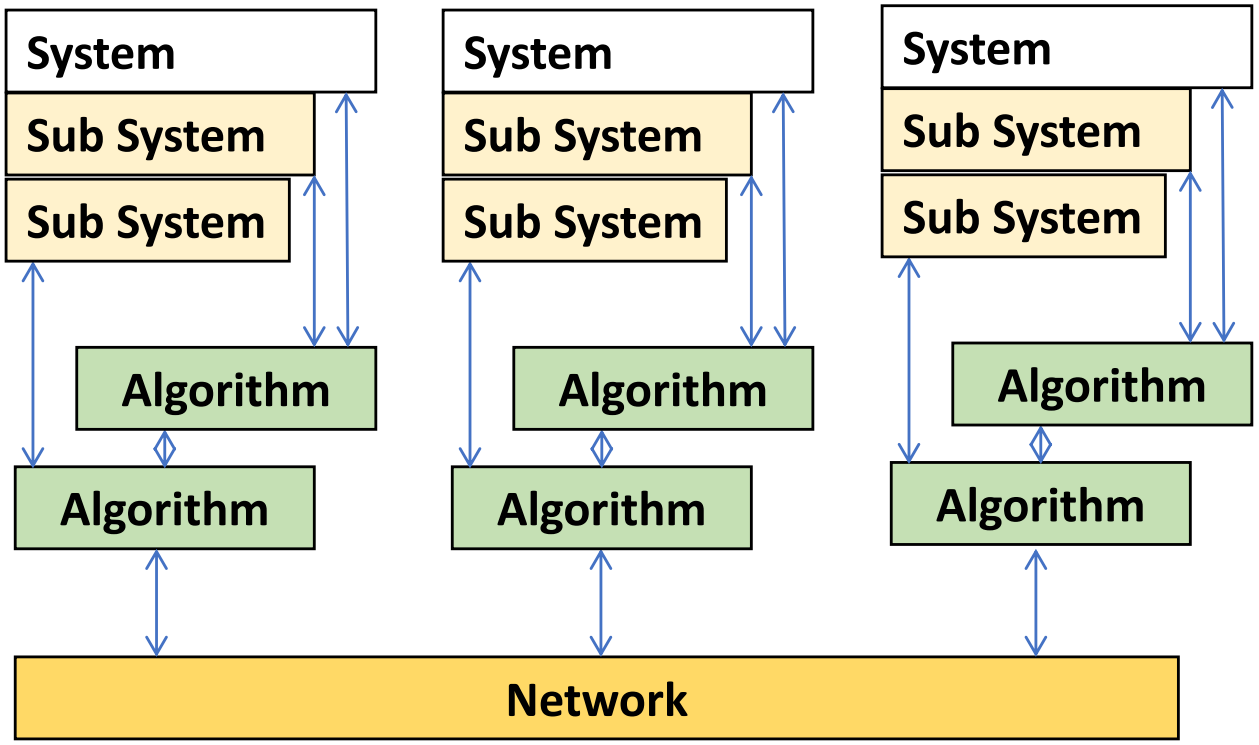
Distributed Algorithms (and Protocols) – More realistically…



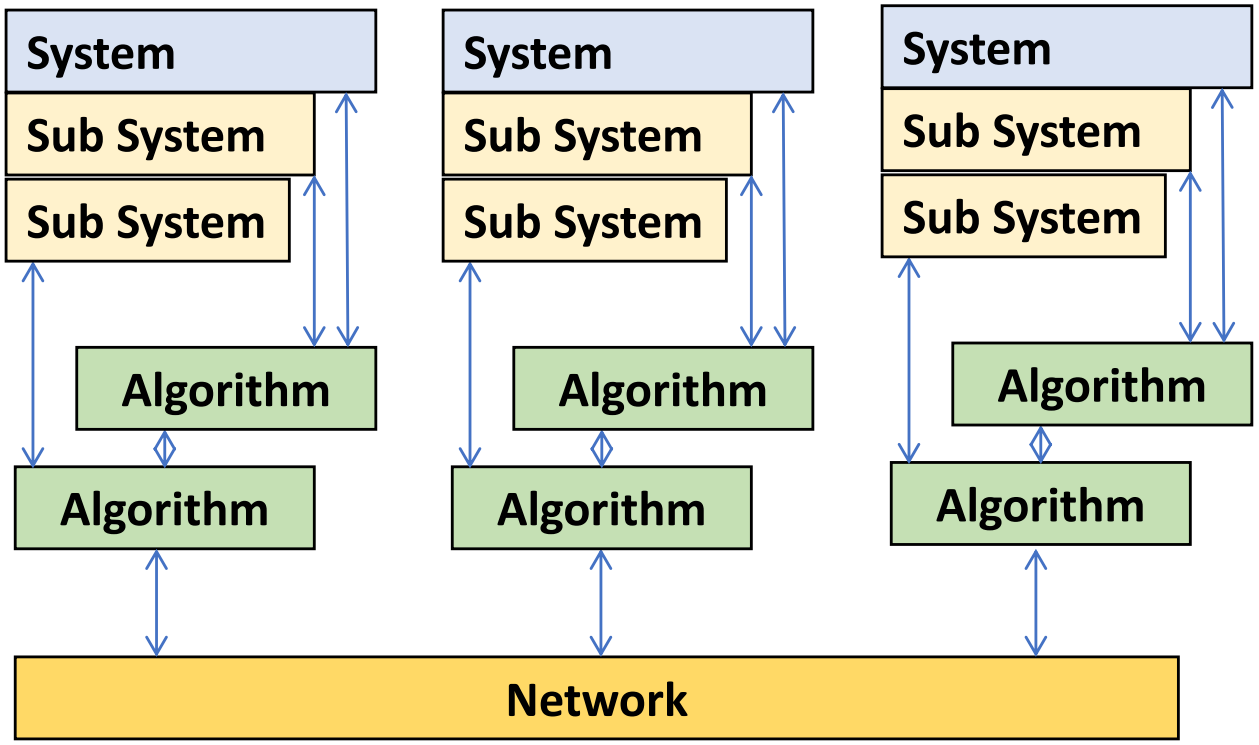
Distributed Algorithms (and Protocols) – More realistically…



Distributed Algorithms (and Protocols) – More realistically…



Distributed Algorithms (and Protocols) – More realistically…



Course Structure

- Lectures
  - 2 hours / week (Friday, 09h-11h)
  - Somewhat interactive
- Labs
  - 2 hours / week (Friday, 11h-13h & 16h-18h)
  - Focused on:
    - Analysis of use cases.
    - Implementation and support for the course project.
    - Project feedback & evaluation.

Course Bibliography:

- *Main course reading:*
  - C. Cachin, R. Guerraoui, L. Rodrigues "Introduction to Reliable and Secure Distributed Programming", 2nd ed, Springer, 2011.
  - N. Lynch. Distributed Algorithms Morgan Kauffman, 1996.

  - Selected research papers (more in a bit).

Course Bibliography:

- *Additional course reading:*
  - H. Attiya and J. Welch. Distributed Computing: Fundamentals, Simulations, and Advanced Topics (2nd Ed.), Wiley 2004.
  - S. Mullender (editor) Distributed Systems, Second Edition, ACM Press, Addison–Wesley, MA, 1994.
  - A.S. Tanenbaum and M. van Steen. Distributed Systems. Principles and Paradigms. (2nd Ed.) Prentice Hall, 2007.
  - Rodrigo Rodrigues, Peter Druschel. Peer–to–Peer Systems. Communications of the ACM. Vol. 53 N10, pp 72 – 82.

Course Bibliography:

- ***Some of the selected research papers:***

- Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. 2001. Chord: A scalable peer-to-peer lookup service for internet applications. In Proceedings of the 2001 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM '01). ACM, New York, NY, USA, 2001.
- Leslie Lamport. Paxos Made Simple. ACM SIGACT News (Distributed Computing Column) 32, 4 (Whole Number 121, December 2001).
- Spyros Voulgaris, Daniela Gavidiam and Maarten van Steen. CYCLON: Inexpensive Membership Management for Unstructured P2P Overlays. Journal of Network and Systems Management June 2005, Volume 13, Issue 2, pp 197–217.
- Giuseppe DeCandia, Deniz Hastorun, Madan Jampani, Gunavardhan Kakulapati, Avinash Lakshman, Alex Pilchin, Swaminathan Sivasubramanian, Peter Vosshall, and Werner Vogels. 2007. Dynamo: amazon's highly available key-value store. SIGOPS Oper. Syst. Rev. 41, 6 (October 2007), 205–220.
- J. Leitão, J. Pereira and L. Rodrigues. HyParView: a membership protocol for reliable gossip-based broadcast. Proceedings of the 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, Ed- inburgh, UK, June, 2007.
- B. Maniymaran, M. Bertier and A. M. Kermarrec, Build One, Get One Free: Leveraging the Coexistence of Multiple P2P Overlay Networks. 27th International Conference on Distributed Computing Systems (ICDCS '07), Toronto, ON, 2007, pp. 33– 33.
- J. Leitão, J. Pereira and L. Rodrigues. Epidemic Broadcast Trees. Proceedings of the 26th IEEE International Symposium on Reliable Distributed Systems, Beijing, China, October, 2007.
- Robbert Van Renesse and Deniz Altinbuken. 2015. Paxos Made Moderately Complex. ACM Computer Surveys 47, 3, Article 42 (February 2015), 36 pages.

Course Project

- Some novelty here in relation to previous editions of this course.
- Motivated by student sugestions provided in the course pedagogical inquiries.

Course Project

- Single Project with three phases.
- Each phase enriches what was done previously with additional functionality.
- There is a delivery for each phase (which is evaluated and you do get feedback) and you can make adjustments/corrections/improvements in the following phases.

- Each phase is evaluated with a grade from 0 to 20.
- Project Grade =

25% Phase 1 + 25% Phase 2 + 50 % Phase 3

Course Project

- Operational Aspects:
- Group Composition: 3 Students (not required to be on same lab class, but recomended for improved feedback).
- Programming Model:
  - Java
  - Network Layer: Netty
  - Threads and “Concurrent Programming” will be employed*

https://netty.io

* Concurrent Programming will be used so that you can avoid to deal with concurrency ☺

Course Project

- Project Topic: Build a Publish-Subscribe system with flavours of Message Queues

  (think Kafka or RabbitMQ but larger and better ☺)

Course Project

- Phase 1 (due on 11 Oct, 23:59:59)
  - Membership Issues
  - Broadcast
  - Will be developing Peer-to-Peer and Gossip protocols
- Phase 2 (due on 10 Nov, 23:59:59)
  - Scalability and Performance
  - Decentralized topology control
  - Peer-to-Peer and Fault-Tolerance protocols
- Phase 3 (due on 06 Dez, 23:59:59)
  - Replication and Fault Tolerance
  - State machine replication and Consensus protocols

Evaluation Rules

- "Attendance" evaluation with a weight of 45% in the final grade.
- Four individual homework assignements with a total weight of 10% in the final grade.
- Two theoretical quizzes or one repeat exam with a total weight of 45% in the final grade (both quizzes have the same weight for this component).
- Final grades are rounded to the nearest integer. Intermediate grades are rounded to the decimal point.

Attendance Evaluation

- Course Project (groups of three students).
- In person discussion in the end of the semester, with a final score between zero and one (Discussion grades are individual).
- The attendance final grade is computed through the following equation:

  Attendance grade = Project grade × Discussion score.

Homeworks

- 4 homework assignment (individual) each will have a final weight in of 0.5 in the final course score.
- Homework is provided at the end of (some) of the lectures.
- Delivery in hand to the professor before the start of the next lecture.

Home Works

- 4 homework assignment (individual) each will have a final weight in of 0.5 in the final course score.
- Homework is provided at the end of (some) of the lectures.
- Delivery in hand to the professor before the start of the next lecture.
- Homeworks will fundamentally require you to (**hand**) write pseudo-code for solving concrete problems, usually less than a A4 page (each homework assignment can be done within 30 min).
- First lab will cover how to write **comprehensive** pseudo-code)

Home Works

- 4 homework assignment (individual) each will have a final weight in of 0.5 in the final course score.
- Homework is provided at the end of (some) of the lectures.
- Delivery in hand to the professor before the start of the next lecture.
- Homeworks will fundamentally require you to (**hand**) write pseudo-code for solving concrete problems, usually less than a A4 page (each homework assignment can be done within 30 min).
- First lab will cover how to write **comprehensive** pseudo-code)
- You want to do this, specially because you also have to write **comprehensive** pseudo-code with project reports and in quizzes.

Quizzes

- Quizzes will follow a structure similar to the ones in the previous two editions of the course (previous quizzes will be made available in clip at least two weeks before the quizz)
- Expect quizzes to:
  - Ask you about fundamental knowledge covered in the lectures;
  - Ask you to analyse algorithms;
  - Ask you to write pseudo-code or manipulate pseudo-code;
  - Ask you to explain how would you design a protocol/sub-system/system.
- You will be allowed to use 2 pages of notes during each quizz:
  - Must be handwritten (no prints, no fotocopies, no exceptions)

Requirements for Approval in ASD

- Attendance grade greater or equal to 8.5;
- Either the average of the two theoretical quizzes or the repeat exam greater or equal to 8.5; and
- Final course grade greater or equal to 9.5 (including all components).

Evaluation (Important) Dates

- *Project Phase 1 available @ 20 September*
- Project Phase 1: 11 October, 23:59:59
- *Project Phase 2 available @ 11 October*
- First Mid-Term: 29 October, 18:30
- *Project Phase 2 available @ 08 November*
- Project Phase 2: 10 November, 23:59:59
- Project Phase 3: 06 December, 23:59:59
- Second Mid-Term: 10 December, 18:00
- Exam: To be defined

Lecture Planning

Lecture	Date	Lecture Contents
0	13-Sep	Course Overview & Introduction
1	20-Sep	Distributed Computing Model and Analysis Methods. Communicaiton Primitives and Broadcast.
2	27-Sep	Peer-to-Peer introduction: Membership, Overlays, and Gossip Protocols
3	04-Oct	No Classes (out of Country)
4	11-Oct	Resource Location Problem, Structured and Partially Structured Overlays
5	18-Oct	Replication: Active Replication, Passive Replication. Register Replication, Quorums.
6	25-Oct	State Machine Replication. Consensus in Synchronous Systems.
x	01-Nov	No Classes (Holiday)
7	08-Nov	Consensus in Asynchronous Systems. FLP. Paxos.
8	15-Nov	Paxos, State Machine Replication and Paxos, Multi-Paxos
9	22-Nov	Strong and Eventual Consistency.
10	29-Nov	CAP theorem and causal consistency.
11	06-Dec	Distributed Transactions, 2 Phace Commit/3 Phase Commit
12	13-Dec	No class planned (Potentially Project discussions)

Lecture Planning

Lecture	Date	Lecture Contents
0	13-Sep	Course Overview & Introduction
1	20-Sep	Distributed Computing Model and Analysis Methods. Communicaiton Primitives and Broadcast.
2	27-Sep	Peer-to-Peer introduction: Membership, Overlays, and Gossip Protocols
3	**04-Oct**	**No Classes (out of Country)**
4	11-Oct	Resource Location Problem, Structured and Partially Structured Overlays
5	18-Oct	Replication: Active Replication, Passive Replication. Register Replication, Quorums.
6	25-Oct	State Machine Replication. Consensus in Synchronous Systems.
x	**01-Nov**	**No Classes (Holiday)**
7	08-Nov	Consensus in Asynchronous Systems. FLP. Paxos.
8	15-Nov	Paxos, State Machine Replication and Paxos, Multi-Paxos
9	22-Nov	Strong and Eventual Consistency.
10	29-Nov	CAP theorem and causal consistency.
11	06-Dec	Distributed Transactions, 2 Phace Commit/3 Phase Commit
12	13-Dec	No class planned (Potentially Project discussions)

Next Lecture Preview

- Fundamental concepts and definitions.
- How to model a distributed system.
- Timing assumptions.
- Fault models.