TiSF - Modelling for Comprehension

Spring 2023 Offering

1 Class Topics

Table 1 lists the topics covered in the class.

Table 1: Class Topics

Class #	**Topic**	**Details**
1	Course Introduction	Recap of transition systems, Perspective from industry and expectations from campus hires, Motivating the need to model software products as systems, an example of a large product and how to approach modelling
2	Modelling software systems	What is modelling? How is it different from architecture and design? Why model? Examples of system descriptions from well-known product companies of their cloud products
3	Modelling System of Systems (SoS)	A typical web application is an SoS; discuss what makes an SoS complex and its key challenges: a) communication, b) UI-backend separation, c) Modeling database access, d) microservices (API-API), an example of a commercial product to illustrate the ideas
4	Modelling Communications	How do we connect systems and what are the semantics of communication between them, the notion of communicating systems, abstracting underlying network into send and receive primitives for modelling
5	Modelling UI systems	Why focus on UI, Modeling UI as a separate system, Model-View-Controller (MVC) pattern and other related patterns and their system model, problems that need to be addressed by a UI system
6	Class presentations on UML	Teams of two students pick two UML diagram types to present to the entire class to learn more about UML
7	Challenges in modelling data systems	Transactional and analytical data, typical data access flow, challenges that need to be addressed by a data system model
8	Modelling data systems	System modelling of the typical data access flow, how to address the challenges discussed in previous class
9	Closure - Architecture and Modeling	Recap of this unit, discuss system models of common architecture patterns and identify commonalities and differences, summarize transition systems vocabulary used for modelling large systems and architecture patterns in this unit, closure

2 List of case studies used

Following is the list of the case studies and their discussion theme:

1. Netflix Viewing data[1]. Show how architecture and design are represented informally and what kind of information is ambiguous in this representation.
2. Netflix billing migration to Cloud [2]. What components of the system can be identified? Why are these components chosen?
3. Netflix - System Architectures for Personalization and Recommendation [3]. The diagram shows control and data flow. Can this be called an architecture diagram? Why/Why not?
4. Turbine: Facebook's Service Management Platform for Stream Processing - Meta Research [4]. What are the structures and behaviour of the system? What type of users are identified? How do you model them?
5. Kangaroo: A new flash cache optimized for tiny objects (Facebook) [5]. What infrastructure components produce specific behaviours in the system?
6. Scribe: Transporting petabytes per hour via a distributed, buffered queueing system (Facebook) [6]. What component systems exist in this architecture? How do you model asynchronous communication?
7. Unified Payments Data Read at Airbnb [7]. What decisions created the need to refactor the architecture to a better one? What could have been done to anticipate and factor in the problems?
8. How pacing works at Twitter ads [8]. This primarily describes the behaviour; what structure and interconnections produce this behaviour? What are the relevant system diagrams?
9. A Brief History of Scaling LinkedIn [9]. How have the design choices in initial versions impacted the need to scale the later versions?
10. Processing billions of events in real-time at Twitter [10]. This is a deployment architecture diagram that includes functional components. How do you produce multiple models at different levels of abstraction so that functional and infrastructural components are depicted more clearly?

[1] https://bit.ly/netflixviewingdata
[2] https://bit.ly/netflixbillingmigration
[3] https://bit.ly/netflixpersonalization
[4] https://bit.ly/metaturbine
[5] https://bit.ly/metakangaroo
[6] https://bit.ly/metascribe
[7] https://bit.ly/airbnbpayments
[8] https://bit.ly/xadpacing
[9] https://bit.ly/linkedinscaling
[10] https://bit.ly/xeventprocessing

3 LO-Teaching-Learning Map

Table 2 lists the mapping of the learning outcomes with teaching approach and evaluation mechanism.

Table 2: LO-Teaching-Learning map

Learning Outcomes	Teaching approach	Learning evaluation
LO1: Create a system model of a complex software system	Demonstrated non-software examples of systems (light bulb, random walk) and then simple software systems (LinkedIn reactions) during lectures and revisited them through class activities. Modelled architecture patterns and their components in class (MVC, Observer, Blackboard, etc.) incrementally and used the discussion sessions to collaborate on open-ended questions around the patterns	Assessment of the individual system models as well as the SoS in the final project
LO2: Map informal engineering descriptions to the system model descriptions	Class activities done in groups where a case study (engineering blog from Meta, for example) was shared; the students described them as models and discussed them within the groups and shared with the instructors.	Submissions and discussions during class activities that focused on engineering blogs (ungraded)
LO3: Articulate their understanding of the system	Class activities to write models of the examples discussed in the class, discussed the ways product companies developed products	Evaluation of the final project submission