CS-A1113 Basics in Programming Y1

Final Lecture 15.11.2022

What Have we Leaned?

- Procedural programming
  - Data and code
  - Functions, loops and branches
  - Most programming languages are procedural
- Object oriented model
  - Many languages support object oriented model

What do You Get out of this Course?

- A start on programming
  - The principles will work on any language
  - Assignments, loops, branches, functions
- An understanding on how computers and software work
  - This will be useful for all of us
  - You can program your future house to use the market price of electricity
  - You can use spreadsheets and analysis software more efficiently
- You can make utility programs
  - E.g., read a file in one format and output it in another format

Programming Paradigms

- Procedural or imperative programming
  - Data and code are separate
  - Program manipulates data
- Object oriented model
  - Data and code are packaged in objects
- Functional programming
  - Theoretical approach, data is passed as parameters between functions
- Declaritive languages
  - E.g., database query language SQL
  - Describe the result you want

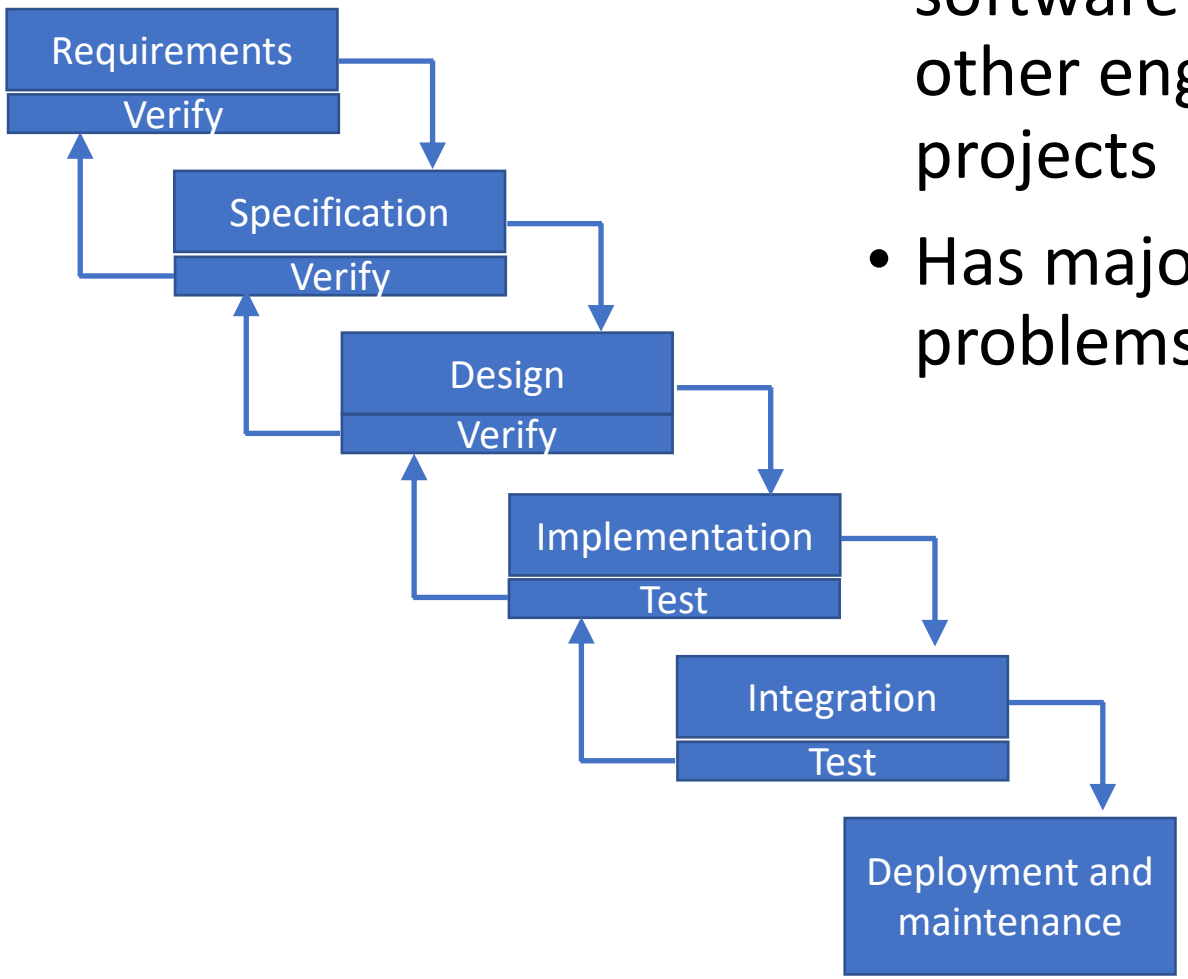
Software Engineering: Design and Development

Software Engineering

- Software is created to fulfill some requirements
  - A reason and need
  - Like other engineering tasks
- Software can be changed during and after creating
  - Differs from most types of engineering
  - Even major changes are feasible
- There are two main approaches to programming
  - Waterfall model: analyze, design, build, test, and deploy
  - Agile: "We don't really know what we are needing, so let's get started and have good work practices"

Waterfall Model

- Collect Requirements
- Create specification
- Design
- Implement
- Test and accept
- Maintenance
  - Fixing bugs
  - Adding functionality



- Approaches software like other engineering projects
- Has major problems

Why The Waterfall Model Often Fails?

- The requirements for the software were not understood well
- Requirements were not interpreted correctly to specification
- Specification was not correctly implemented in code
- Too much coding is done before testing, making it very hard to find causes of failure

- Pure waterfall is not recommended
- Waterfall can be used iteratively when the requirements are clearly understood
  - Work in sprints, add functionality, integrate and test

Iterative Software Development Methodologies

- Software is developed in small iterations (sprints)
  - Weeks or months
- Each iteration ends with working, tested software that can be presented to the customer
  - Initially minimal, functionality is added during the process
- Requirements are re-evaluated after each sprint
  - Even dramatic changes can be made (pivot)
- Many methodologies: Agile, DevOps, Test Driven, Scrum, Kanban...
  - Overlapping and subject to much debate

Common Practices

- Requirements management
  - Requirements and their completion is tracked (ticketed)
  - There is a measurable awareness of progress
  - Completion of a task or requirement is unambiguous, "Definition of Done"
- Work discipline
  - Tasks are assigned to people in a clear manner
  - Documentation to maintain the resulting software
- Continuous integration and automated testing
- Acceptance procedures

From Idea to a System

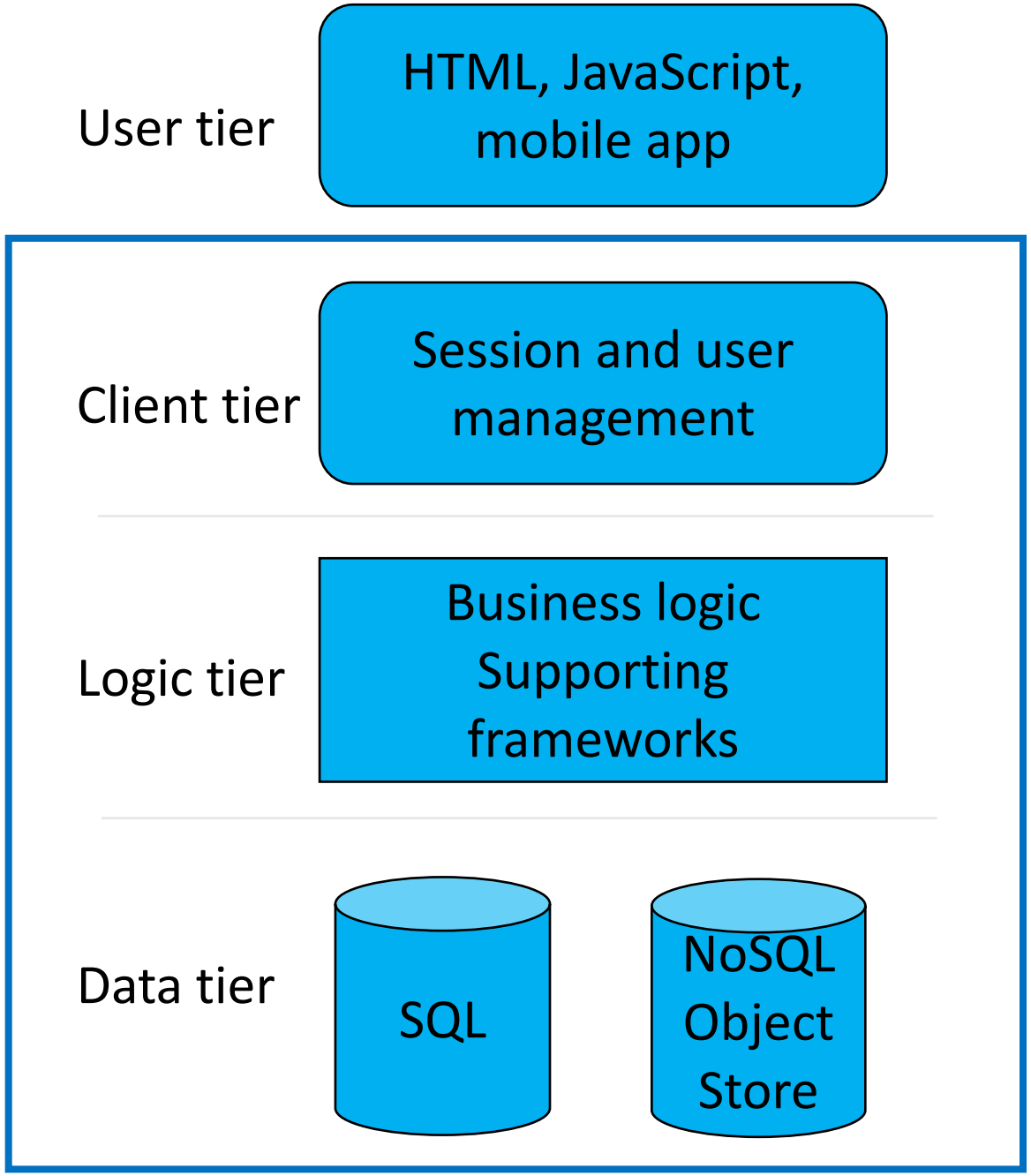
- Proof of concept (PoC)
  - Showing that the technical challenges (performance etc.) can be solved
- Prototype
  - A working model for experimenting
- Mock-up
  - A visual representation of what the product could look like
- None of the above should be put into production…

- Minimum Viable Product (MVP)
  - The smallest working product and set of features that can be deployed

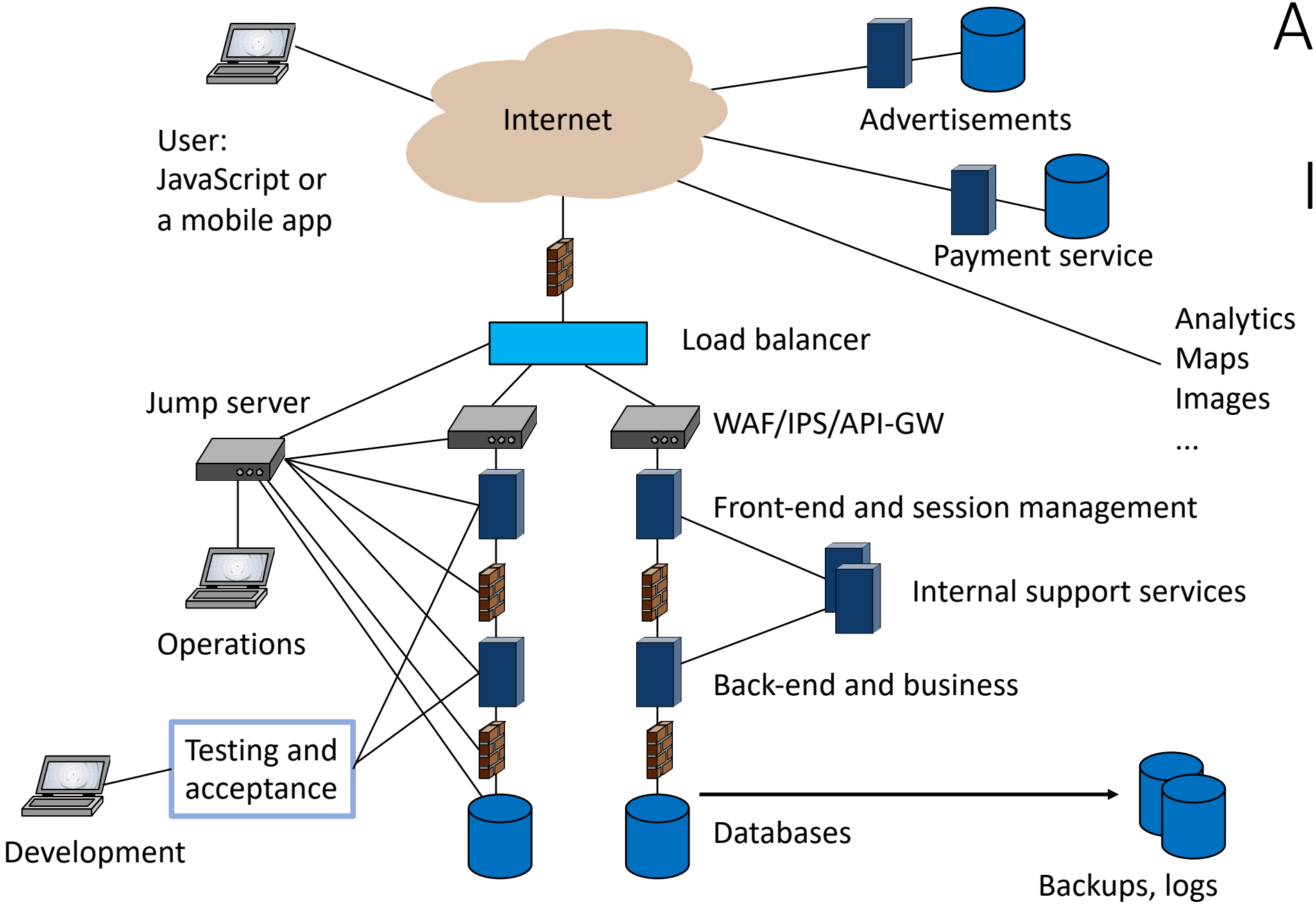
Architecture

3-Layer Architecture

- One common architecture
- Presentation layer
  - Front end
  - Talks to web browser / mobile app
  - Manages user sessions
- Application / logic layer
  - Server software
  - The actual business logic
- Database / data layer
  - Data storage
  - SQL, NoSQL ...



A System on the Internet



IT and Business

Is IT a Critical Component for Business?

- Computers entered business in the support role
  - Billing, accounting, numerical analysis
  - "Faster people"
- Now many businesses can't exist without computers/digitalization
  - Uber, AirBnB, Amazon, delivery services, logistics
  - Process industries, manufacturing
- Computing is still developing
  - Understanding its possibilities is important for business
  - Also understanding what computers can't do

The Future of IT in Work and Business

- Computers are good with well defined and organized tasks
  - Much more efficient than humans
- Humans are good at open ended reasoning
  - Computers fail spectacularly with undefined tasks
- Computing is evolving
  - AI and machine learning are current hot topics (early 2020's)
- A likely scenario for near future is better human-machine co-working
  - "Human in loop" and semi-autonomous systems

Careers in and near IT

Roles and Skill Sets

Who creates the system?

- Programmer
  - Writes functions, modules and small programs
- Lead Programmer
  - Directs the other programmers
- Architects
  - System architect designs program modules and their locations (usually several servers or instances in the cloud)
  - Data architect designs data models for the software and databases

Who decides what the system is?

- Product owner, business owner
  - Controls what the system does, decides product's features
- Technical product owner
  - Controls the technical features
- Project manager
  - Makes sure the project is on schedule and budget and has the required features
  - Often not reaching all goals
- Sales and marketing

Roles and Skill Sets

Support roles

- Tester
- UI designer
- UX designer
- Security specialist
- Operator/Administrator
- Database manager
- …

Specific skill areas

- Front end developer
- Back end developer
- Full stack developer

Other roles

- Sales engineer
- Consultant

Roles in IT are often flexible

Digitalization and Importance of Software

- Everything will be computerized and networked
- The field is evolving and changing
- There will be new roles and tasks
- We still don't know how to write correct programs or work in an efficient way
- Lots of practical work and research to do

Study Paths

Interesting Areas in Computing

- Programming, architectures, data (big and small)
- Distributed and embedded systems
  - Cyber-physical systems
  - Machine to machine interaction
- Artificial intelligence and machine learning
- Human-computer interaction
  - User interfaces and experience
- Graphics and media
  - Games, VR, AR
- Quantum computing

Where to Go from Here?

- CS-A1123 - Basics in Programming Y2
  - Continues from this course with object oriented Python
- ELEC-A7100 - Basic Course in C programming
  - C is a more low level language, useful especially in embedded systems
- CS-A1143 Data Structures and Algorithms Y
  - Fundaments of computing theory, also required for CS minor
- CS-A1153 – Databases
  - Modeling data and relations
  - CS-A1150 Tietokannat in Finnish, can be done in English with self study
- Minor in Computer Science (20 – 25 cr)
  - https://into.aalto.fi/pages/viewpage.action?pageId=53681468
- Ask around your own study program for recommendations


"That's all Folks!"
Isberg