
LUT
University
LAND
OF THE
CURIOUS

LECTURE 13

SOFTWARE BUSINESS

Software Engineering, December 8th, 2021

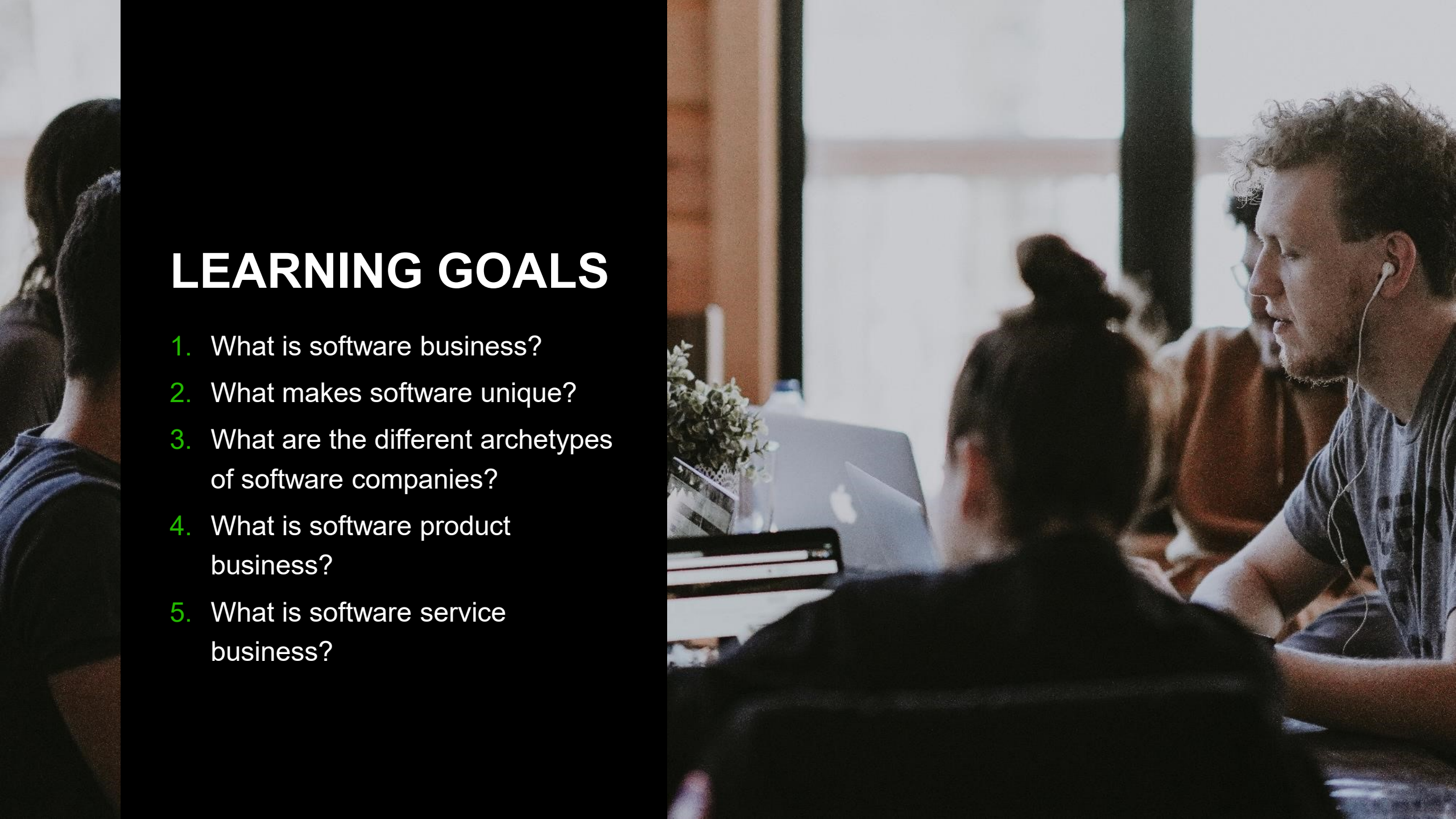
Sami Hyrynsalmi

SAMI HYRYNSALMI

- **Associate Professor** (tenure track) of Software Engineering at Lahti Campus
  - Previously worked for Tampere University, Tampere University of Technology, University of Turku, and Turku School of Economics.
- Academic Head of **Software Product Management and Business** Master’s of Science programme at Lahti Campus

LEARNING GOALS

1. What is software business?
2. What makes software unique?
3. What are the different archetypes of software companies?
4. What is software product business?
5. What is software service business?



LUT University

SOFTWARE BUSINESS

LECTURE 13

WHAT IS SOFTWARE BUSINESS?

RÖNKKÖ & PELTONEN (2011):

"SOFTWARE BUSINESS ***is business of selling software*** *(including systems software, application software, and games) either as* ***licenses*** *or as* ***services*** *and* ***services related to development and deployment*** *activities of this software."*

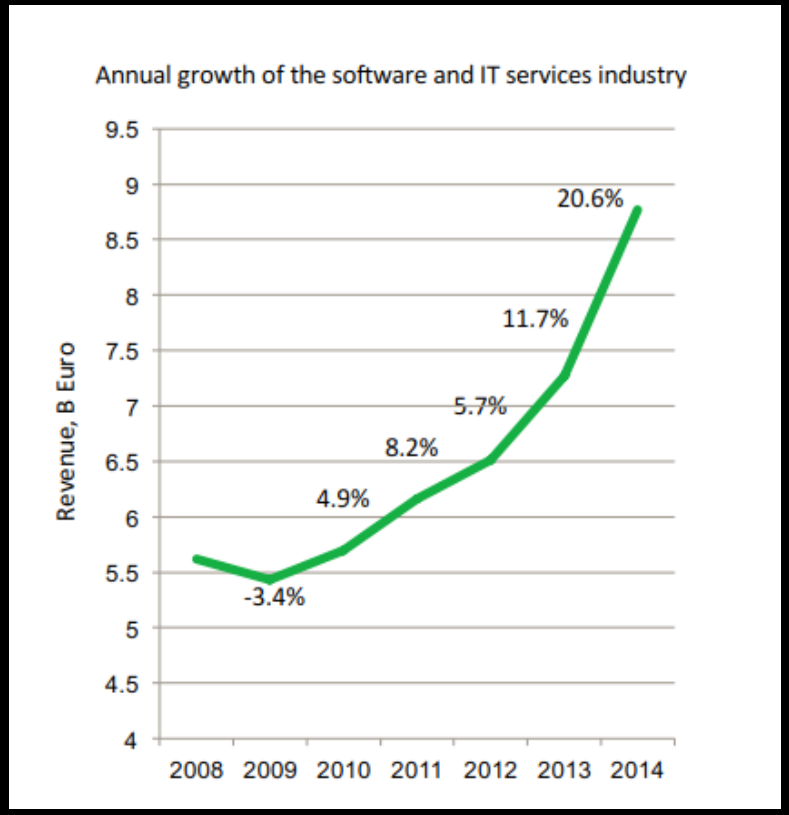
M. Rönkkö, J. Peltonen (2011) *Software Industry Survey*. Finland: Aalto University

- It **does not** include **operation of software produced by third parties** (e. g. operating a server farm), **business and operations consulting** related to software systems, and **deployment projects of third-party software**.
- It is worth to note that **not all revenue** of firms operating in the software industry is necessarily from software business.

LECTURE 13

SOFTWARE INDUSTRY

- The term ‘software industry’ refers to the industrial field of the companies doing software business.
- The software industry **was born on June 23rd, 1969**, when IBM announced that it will separate software and hardware sales from each other.
- The entire software industry’s revenue was estimated to be **USD 407 billion in 2013** (with yearly growth of +4.8%) and **USD 968 billion** in 2021 (+4%)
- For a comparison, the budget of Finland was 65 B$ in 2013 and 73 B$ in 2021.



Eetu Luoma (2015) Finnish Software Industry Survey 2015. Univ. Jyväskylä.

LECTURE 13

SOFTWARE COMPANY TYPES

- A classic approach, by Hoch et al (2000), divides software companies into three archetypes.

- **Professional software services**
  - Usually producing tailor-made software solutions
  - A low level of productization
  - Focuses on a small niche market

  Examples: CGI, Fujitsu

- **Enterprise product solutions**
  - Products and services for business-to-business markets
  - Larger markets

  Examples: SAP, IBM, Oracle

- **Packaged software**
  - Highly productized consumer products
  - Largest market

  Examples: Microsoft, Adobe

D.J. Hoch et al (2000) Secrets of software success. Harvard Business Press.

LECTURE 13

SOME MODERN EXAMPLES

- **Supercell** is a mobile game company and producer of *Hay Day*, *Clash of Clans* etc.
- 1.3 B€ revenue (2020) with 340 employees (2021).

- **Vincit** is a software service and consultancy company, which sells tailor-made projects and software.
- 52.4 M€ revenue (2020) with 460 employees.

- **Vaadin** is a producer of open-source software tools. The core product is free, but they sell consultancy, support, and education services.
- 9.6 M€ revenue (2020) with 110 employees.

BUSINESS FOR ENGINEERS?

LECTURE 13

WHY SHOULD I, AS A SOFTWARE ENGINEER, CARE?

Software business and its engineering are highly intertwined – they affect to each others.

- **Some technical decision affect on the business and revenue model of the software product and *vice versa*. For example, pay-per-use pricing requires that the selected software architecture supports logging.**
- Commercial software is always done in a business environment. In software service companies, the business is close to everyday work. In software product companies, it is good be able to support and serve the business management.
- For those interested to found startup or work in senior management positions in a software companies, the business of software aspect will be important.

Software is eating the world.

Over the next 10 years, I expect many more industries to be disrupted by software, with new world-beating Silicon Valley companies doing the disruption in more cases than not.

-- Marc Andreessen, 2011

Marc Andreessen (August 20, 2011) Why Software is Eating the World? Wall Street Journal.

LUT University

SOFTWARE IS UNIQUE

But software is not like other businesses.

These instructions form products that companies can standardize for many users, customize for individual users, or do something in between. Companies that rely on this highly malleable technology for their livelihoods must be unique in many ways, particularly in how they deal with business models, product strategy, people (especially software engineers), and management of a core activity: software development.

-- SMR Distinguished Professor **Michael A. Cusumano**, 2004

M.A. Cusumano (2004) The business of software. Free Press.

WHAT MAKES SOFTWARE UNIQUE?

A GROUP DISCUSSION EXERCISE

Task:

- Form a group of 2-4 people sitting around you or work alone.
- Please, discuss and try to identify what **characteristics of software** make it so unique that it is studied separately from the traditional industries?
- Write your answers down in **menti.com** with a code **8033 5800** – or read the QR code.
- You have approximately **10-15 minutes**.

Examples of a characteristics

- *Intangibility* i.e., one can't touch or feel the software, it is hard to assess.
- *Expensive to develop, cheap to copy* i.e., the traditional pricing logic doesn't work.
- *Piracy* i.e., unlicensed use of software products is/was easy.

A GROUP DISCUSSION EXERCISE

WHAT MAKES SOFTWARE UNIQUE?

Far from being a complete list, but a few noteworthy characteristics:

- **Small upfront investment needs**
  In theory, a few people, laptops and a garage is enough to built a new product or service for large markets.
  (E.g. WhatsApp had 55 employees in 2014)
- **Built-in internationalism**
  Moving software from a country to country is practically free, yet there might be need for localization (e.g., different laws).
- **Ubiquitous**
  There is hardly an industrial domain which haven't been impacted by software.
- **Vendor lock-in / high cost of change**
  Changing a software from one another can be expense due to the indirect costs.
- **Updatability**
  Functionality can be added or altered after the launch.
- **Low entry barrier, high competition**

PRODUCT VS. SERVICE

LECTURE 13

A COMMON DICHOTOMY IN ECONOMICS

SERVICES

A transaction between a seller and a buyer where no physical goods are transferred.

- Services are **intangible** (can't be manufactured, stored or transported).
- Services are **perishable** (i.e., they vanish after a specific period in time).
- Services are **inconsistent** (i.e., each service is, in some degree, unique).

PRODUCTS / GOODS

Products are often – but not always – tangible goods that are transferred from a seller to a buyer.

- Often **identical**, thus allowing **mass-production.**
- (To some degree,) **imperishable** and can be resold, stored and stocked.

LECTURE 13

PRODUCTS AND SERVICES IN SOFTWARE INDUSTRY

SOFTWARE **PRODUCT**

- Restricted **licenses** (and updates) to use a software product are sold.
- Note, companies rarely sell **the product,** but licenses i.e., right to use it for a certain number of people, processors etc.
- That is, one doesn't own the software and can't modify it.

SOFTWARE **SERVICE**

- Services and consultancy related to software development and deployment.
- E.g., a custom product ordered from a company; customization or deployment project of enterprise software; or hiring an expert or buying training…

PRODUCT-SERVICE **HYBRID**

- Products and services are combined into innovative offerings – licenses and services are both offered.
- E.g., Oracle on Demand offers consultancy and management services to get the full benefits out of the packaged solution.

LECTURE 13

MASS-PRODUCTION AND MARGINAL COSTS

- In the classical dichotomy of goods and services, software is (kind of) **product.**
  - Surprisingly, this is not that simple.
- In traditional pricing models of products, the price is based on production costs plus desired profit margins.
- **Development cost** = a cost of producing the first copy.
  Marginal cost = a cost of producing an extra copy (at the current output).
- The question of pricing is not that simple in software business as development costs are relatively high and marginal costs are trivial.
- **How to set the price?**
  To cover the development costs, a relatively good estimate of customers should be used (if the price is a share for development cost plus margin). Otherwise, the price might be too high or revenue too low to cover the costs.

LECTURE 13

REVENUE MODEL

- *Business model*, *pricing model* and *revenue model* are three different, yet interlinked concepts.
  - **Business model** is a blueprint of a company – how it creates, delivers and capture value.
  - **Revenue model** tells how a company is getting compensated from the offered product or service. (The compensation can be other than money.)
  - **Pricing model** tells how the price of a single product or service is formed. For example, what is the pricing base and are there any bundling.
- These three are all inherently different for software product and software service business.

LECTURE 13

REVENUE MODELS IN SOFTWARE INDUSTRY

SOFTWARE **SERVICE** BUSINESS

- Revenue model is **usually** based on, e.g., billed hours and individually made contracts.
- For example, a customer can either
  - buy a **fixed-price project** (i.e., the seller estimates the workload, adds margin and offer a price for the project); or
  - "**rent a team**" (e.g., pays a fixed sum for the services of a development team per Scrum sprint);
  - an agreed percentage of made profit; or
  - some combinations of above; or…

SOFTWARE **PRODUCT** BUSINESS

- Revenue model is **usually** based on selling **licenses** and updates to the customer.
- For example, a Microsoft Windows 10 64-bit license for a single computer (can't be transferred) is **154,90 €**. Minor updates are either free or with a time-limited license (e.g., F-Secure Safe **29,90€ / year**) and major updates (e.g., Windows 11) is sold with a new license.

LECTURE 13

SOFTWARE-AS-A-SERVICE

- In a traditional software product model, a customer buys a license (via one time fee) to use the software. The customer is responsible for deployment (i.e., installing) and operation of the software. The customer might receive minor updates as well as buy new versions of the software.
- **Software-as-a-Service (SaaS)** refers to a model where a customer is **renting** the software (via monthly or yearly fee) and uses it online. The customer always uses the newest version and does not operate the software.
  - SaaS can also be offered in **a pay-per-use model** where a customer is charged based on the metered usage. For example, the number of hours that the customer used the service, processor time or memory used, the number of (certain) function calls made while using etc.

LECTURE 13

SAAS EXAMPLES

LECTURE 13

TREND OF SERVITIZATION

- There is an overall trend of *servitization,* where products are turned into services, visible in the global business. E.g., no one wants to anymore own music or movies, but rent them as a service (Spotify, Netflix). Thus, outcome is becoming more important than the method.
- Similar transition is happening also in manufacturing industry where, e.g., Rolls Royce sells jet engines as a service. RR is not anymore asking upfront payment for an engine, instead it rents them as a function of flying time plus engine's efficiency and is responsible for maintenance.
- This is also to some extend visible in software business – more and more software are moved out from on-premises towards SaaS.

HOMEWORK

LECTURE 13

HOMEWORK

- **Essay:** *Compare advantages and disadvantages of pay-per-use vs software rental revenue models from a software provider's viewpoint.*
- **Reading materials:**
  - Ojala, A. (2013). Software-as-a-Service Revenue Models. *IT Professional* 15(3), 54-59. doi:10.1109/MITP.2012.73. https://jyx.jyu.fi/bitstream/handle/123456789/38554/Software-as-a-Service%20Revenue%20Models.pdf?sequence=4&isAllowed=y
  - Buxmann P., Diefenbach H., Hess T. (2012) The Rules in the Software Industry. In: *The Software Industry*. Springer, Berlin, Heidelberg. doi:10.1007/978-3-642-31510-7_1. https://link-springer-com.ezproxy.cc.lut.fi/chapter/10.1007/978-3-642-31510-7_1


LUT
University