software architectures

course overview

objectives

provide an introduction to the concept
of software architecture and its
various implementations

provide the opportunity to acquire theoretical
and practical understanding of the different
dimensions of a software architecture

practical issues

lectures: Internef 237
exercises: Internef 143

doplab.unil.ch/software-architecture

evaluation

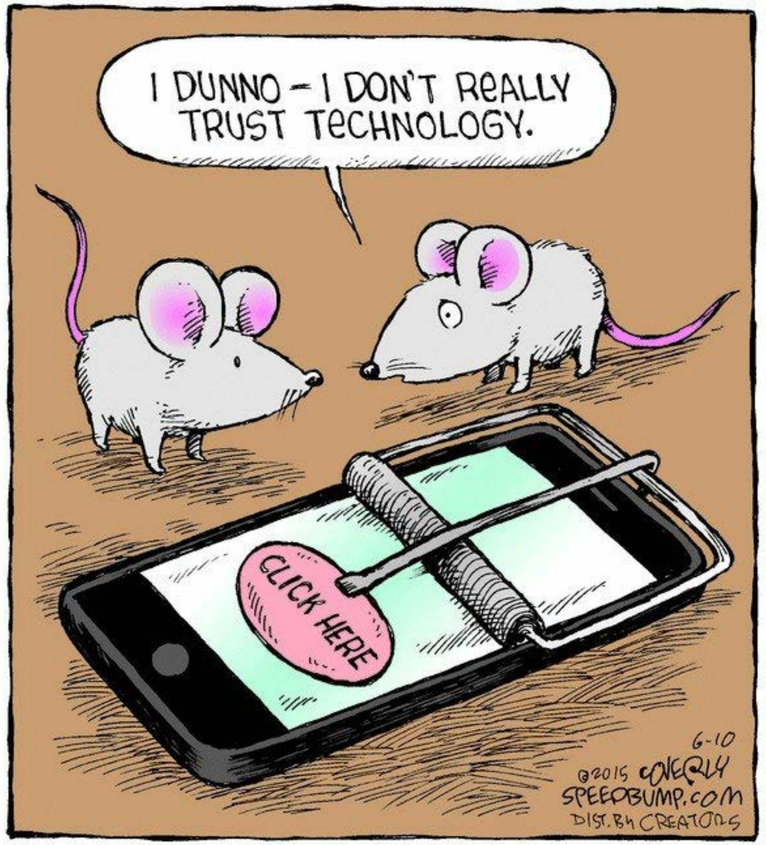
a written exam at the regular exam session

written or oral exam at the retake session, depending on the number of candidates

some development tools

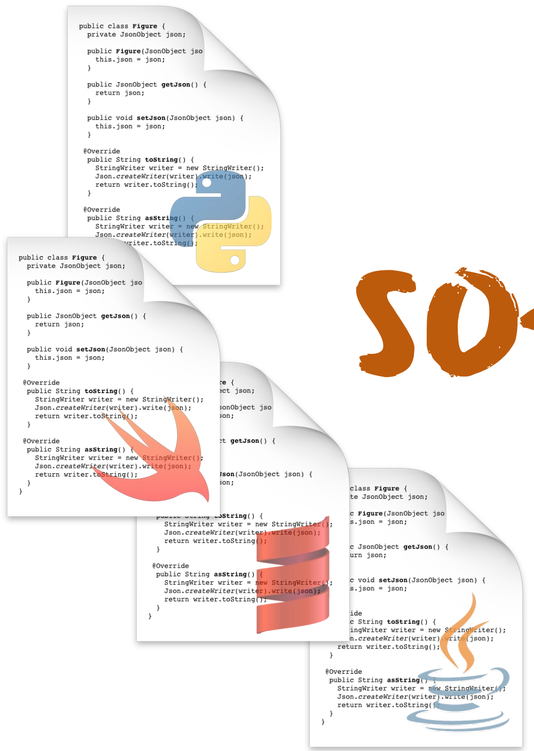
caveat

students are expected to possess basic knowledge of programming and algorithms, typically acquired through a course like Algorithms and Computational Thinking



this is a brand-new course on an important innovation-driven domain, so expect some changes along the journey

okay, but what is a



software

architecture?

software...

... can be continuously corrected, extended, refactored, optimized

the nature of the software is unique and brings it closer to a living organism than to an inert and passive artifact

because of its unique nature, software is at the heart of the digital transformation and makes this transformation profound

the nature of software

software is closely related to the notion of algorithm

an algorithm is a finite and unambiguous sequence of operations used to solve a problem

the word "algorithm" comes from Muhammad ibn Musa al-Khwarizmi, Latinized to Algoritmi, a Persian mathematician who lived in Baghdad in the 9th century

he was the first to describe a systematic procedure (actually an algorithm) for constructing solutions to linear and quadratic equations

the software revolution

software is not longer being simply used as support for existing human activities but rather becoming the driver of profound changes in the way we do things and even the source of totally new activities

this is the essence of digital transformation

a new mindset

the "old" mindset

- align software usage to existing business practices
- business is the driver, software is merely a support

the "new" mindset

- extend business models to support new digital channels
- embrace the intrinsic fungibility offered by software

> property of goods or services whose individual instances are capable of mutual substitution

software is now ruling the world

digital disruption is what happens when you let others drive the digital transformation of your business

did you know...

...the largest taxi company owns no vehicles?

...the largest accommodation provider owns no building?

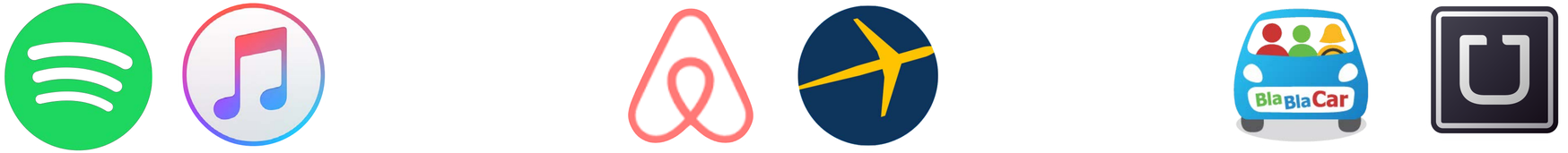
...the most valuable retailer has no inventory?

...the largest travel agency has no public offices?

digital disruption

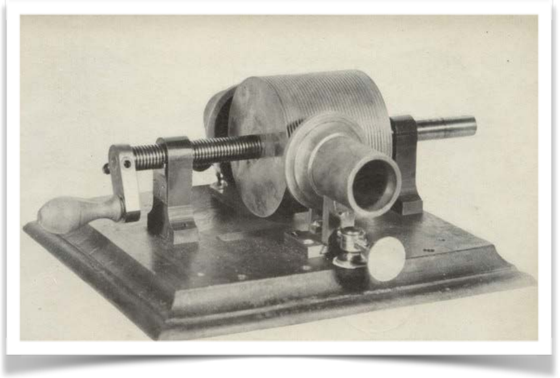
typical sequence of events

1. digitization dematerializes the product and makes the existing business more efficient
2. some digital technology innovation makes the product fungible but existing actors do not see it
3. some outsider who understands the potential of that digital innovation disrupt the market by proposing a cheaper and/or better alternative



digital disruption

the example of the record industry



1877 – Phonograph (Edison)

1890 – Gramophone (Berliner)

1925 – Standard 78rpm

1948 – Long Play (33rpm)

1963 – Cassette (Philips)

1979 – Walkman (Sony)

1980 – CD (Philips/Sony)

1992 – MiniDisc (Sony)

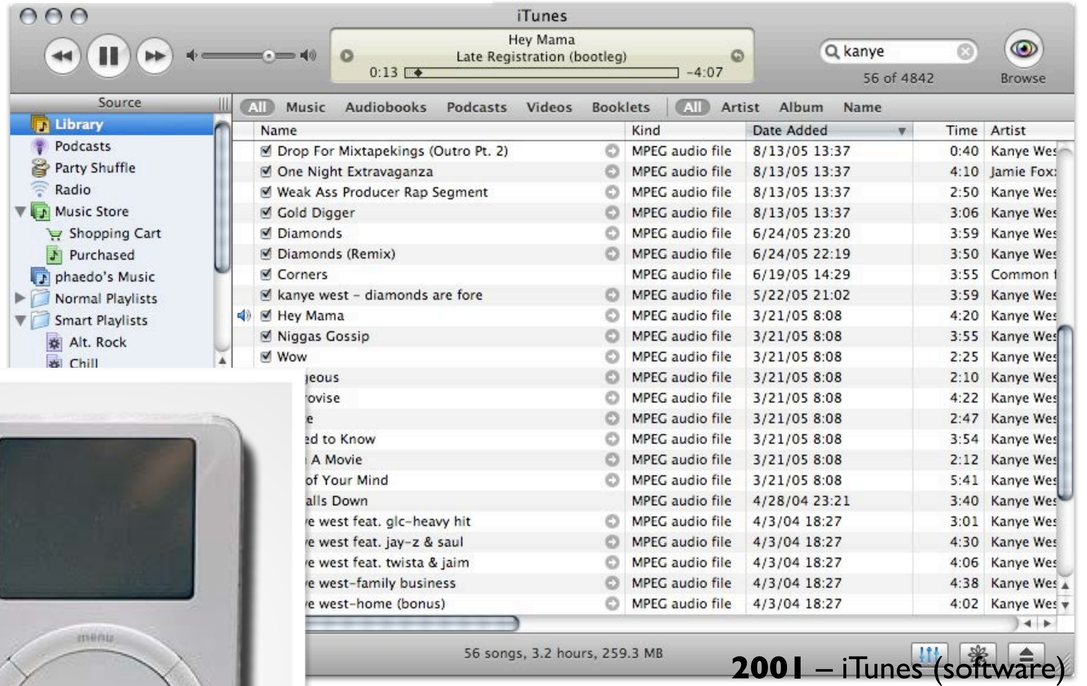
digital disruption

the example of the record industry

1993 – MP3 Format

1998 – MPMan (10 to 30 songs)

1999 – Napster Service



2001 – iTunes (software)

2001 – iPod (1000 songs)

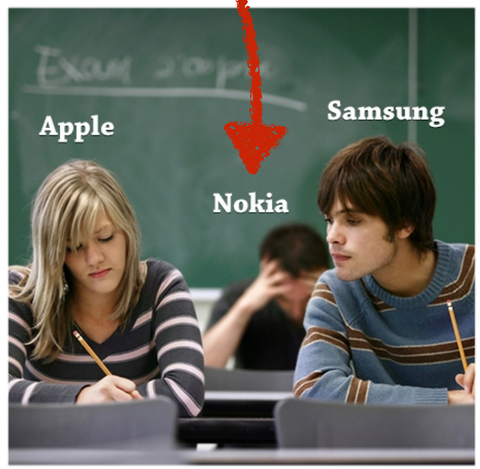
digital disruption

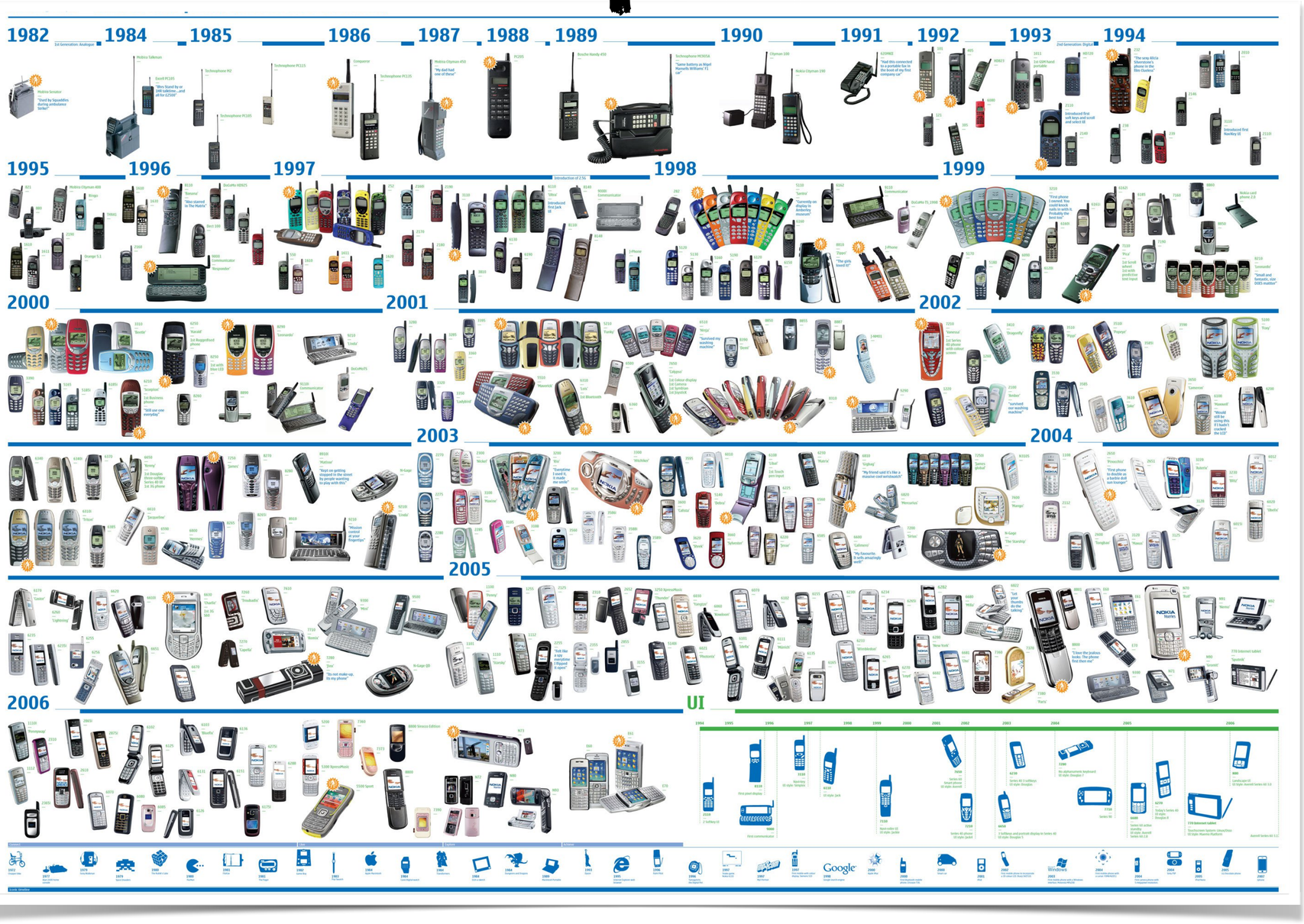
the example of the record industry

	sony	apple
was building audio devices	✔	✘
co-invented the compact disk	✔	✘
invented the walkman	✔	✘
owned a record company	✔	✘
understood software	✘	✔

digital disruption

November 2007

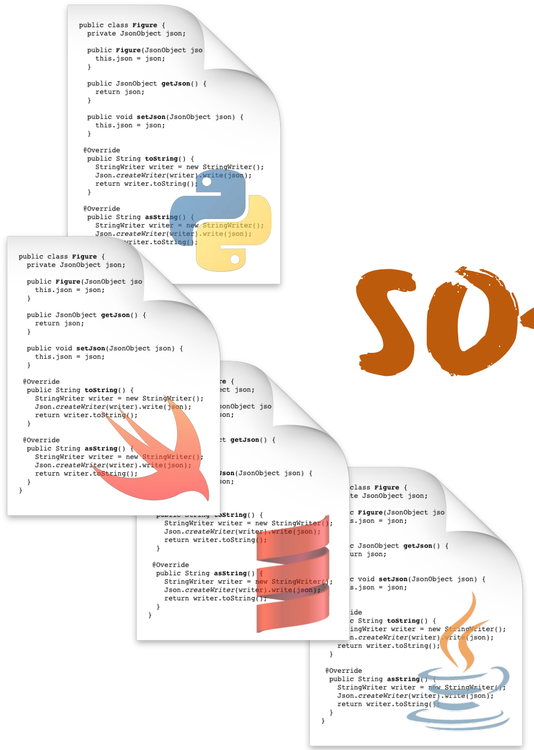




okay, so what is a

software

architecture?



software architecture

multiple definitions

Overall, macroscopic system structure.

Garlan & Shaw (1994). An Introduction to Software Architecture.

The important stuff—whatever that is.

Fowler, M. (2003). Design – Who needs an architect? IEEE Software. 20 (5)

Architecture is the fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding its design and evolution.

[IEEE 1471 Standard]

That which is fundamental to understanding a system in its environment

[ISO/IEC/IEEE 42010 Standard]

no strict consensus but an overall agreement

software architecture

LIFECYCLE PHASE	TYPICAL QUESTIONS
Development Associated Terms • designing • programming • coding • static	• how effective and efficient are the algorithms I am using? • how are the different functional and technical concerns dealt with? • how is the growing complexity of my code base managed? • how easy is it to correct bugs and add new functionalities? • are side effects possible? if so, how are they managed?
Deployment Associated Terms • distribution • provisionning • curating / updating	• on how many processors / machines will my code be deployed? • how many distinct tiers are being used for deployment? • how do remote pieces of software communicate? • how is distributed trust and security ensured?
Operation Associated Terms • execution-time • run-time • monitoring/profiling • dynamic	• who is managing the machines running my software? • where are those machine located? • how do I monitor my running software? • how do I update my running software? • how is scalability achieved?

the software lifecycle as guide

beware

decision made in each phase influences the other phases

software architecture

a tentative plan

	FRIDAY	8:30 – 10:00	10:15 – 11:00	11:15 – 12:00	
1	Sep 20	course overview	discover development tools	assess your programming skills	Development
2	Sep 27	programming paradigms – a refresher course	basic object-oriented and functional programming exercises		
3	Oct 04	programming methodology and tools	programming methodology and tools execises		
4	Oct 11	modularity, unit testing and separation of concerns	separation of concerns | remote invocations and transactions		Deployment
5	Oct 18	modularity and unit testing exercises	remote invocations and transactions exercises		
6	Oct 25	separation of concerns | persistence and object-relational mapping	object-relational mapping exercises		
7	Nov 01	basics of web applications	web application exercises		
8	Nov 08	THEMATIC WEEK			
9	Nov 15	asynchronous interactions	asynchronous interactions exercises		
10	Nov 22	basics of web services services and micro-services	web services and micro-services exercises		
11	Nov 29	basics of distributed trust with blockchains	blockchain exercises		
12	Dec 06	basics of mobile applications	mobile application exercises		
13	Dec 13	cloud computing– virtualization and devops	virtualized services exercises		Operation
14	Dec 20	cloud computing – containerized services and clustering	containerized services and clustering exercises		