The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Advanced Topics in Software Architecture (E23)

Tools and Technologies 2

SDU Sune Chung Jepsen and Torben Worm

1

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Agenda

→ Last weeks exercises
→ Programming paradigms
→ Database systems
→ Peer review
→ Pitches

SDU Sune Chung Jepsen and Torben Worm

2

The Maersk Mc-Kinney Moller Institute

sdu.dk

Learning Objective

#sdupdk

→ Explain tools and technologies for implementing software architecture
→ Basic understanding of object-oriented programming languages and concepts
→ Select and combine tools and technologies to implement software architecture
→ Ability to apply software architectures for different quality attributes using tools and technologies

3

The Maersk Mc-Kinney Moller Institute

sdu.dk

Last weeks exercise

#sdupdk

→ How did you approach the exercise?
→ What were the main challenges?
→ What did you decide to produce (if anything)?
→ What were your essential use cases?
→ What systems and sub-systems did you identify?

4

The Maersk Mc-Kinney Moller Institute

sdu.dk

Motivation

#sdudk

→ Modern software systems are constructed from many different components programmed in many different languages
→ Software architecture exits both ”in the small” and ”in the large”
→ The architecture you can program is to a large extend independent of the programmeing language
  → In the sense that in the end it all compiles to instructions to be executed on a CPU
→ What you can express with at programming language affects the ease with which you can implement certain architecutal ”styles” or principles

5

The Maersk Mc-Kinney Moller Institute

sdu.dk

Motivation

#sdudk

→ Modern systems tends to use multiple languages, e.g. layers

→ Problems?

→ Programming paradigms

Jul 2023	Jul 2022	Change	Programming Language	Ratings	Change
1	1		Python	13.42%	-0.01%
2	2		C	11.56%	-1.57%
3	4	↑	C++	10.80%	+0.79%
4	3	↓	Java	10.50%	-1.09%
5	5		C#	6.87%	+1.21%
6	7	↑	JavaScript	3.11%	+1.34%
7	6	↓	Visual Basic	2.90%	-2.07%
8	9	↑	SQL	1.48%	-0.16%
9	11	↑	PHP	1.41%	+0.21%
10	20	↑↑	MATLAB	1.26%	+0.53%
11	18	↑↑	Fortran	1.25%	+0.49%
12	21	↑↑	Scratch	1.07%	+0.35%
13	12	↓	Go	1.07%	-0.07%

6

The Maersk Mc-Kinney Moller Institute

sdu.dk

Paradigm

#sdudk

→ a model of something, or a very clear and typical example of something (Cambridge dictionary)

→ a philosophical and theoretical framework of a scientific school or discipline within which theories, laws, and generalizations and the experiments performed in support of them are formulated. broadly : a philosophical or theoretical framework of any kind (Merriam-Webster)

7

The Maersk Mc-Kinney Moller Institute

sdu.dk

Programming Paradigm

#sdudk

→ A way to classify programming languages according to some properties, e.g.

→ Execution model, i.e. sematics

```
s1; s2;
a+b*c;
```

→ Side effects

```
int a=0;
int f(int x) {a=10; x++; return x }
```

→ Structuring of the code

```
DEFINITION MODULE M1;
    EXPORT QUALIFIED a, b, c, P;
    ...
MODULE M2;
    IMPORT M1;
    ...
...
    M1.a := 0;
    M1.c := M1.P(M1.a + M1.b);
    ...
```


8

The Maersk Mc-Kinney Moller Institute

sdu.dk

Programming Paradigms

#sdudk

- → Imperative
  - → procedural
  - → object-oriented
- → Declarative
  - → functional
  - → logic

9

The Maersk Mc-Kinney Moller Institute

sdu.dk

Imperative

#sdudk

- → Imperative, noun (Merriam-Webster)
- → : something that is imperative (see IMPERATIVE entry 1): such as
- → a: COMMAND, ORDER
- → b: RULE, GUIDE
- → c: an obligatory act or duty
- → d: an obligatory judgment or proposition
- → 2: the grammatical mood that expresses the will to influence the behavior of another or a verb form or verbal phrase expressing it

10

The Maersk Mc-Kinney Moller Institute

sdu.dk

Imperative programming language

#sdudk

→ Imperative programming uses statements that change a program's state
→ Describes how a program operates step by step

```
label x;
int a=10;
int b=20;
int c = 0;
c=a+b;
b=c;
goto x
…
```

Implications on architecture?

11

The Maersk Mc-Kinney Moller Institute

sdu.dk

Procedural (structured) programming language

#sdudk

→ Goto considered harmful[Dijkstra1968]

→ —>

→ Structured programming

→ Functions
→ If/then/else
→ Do/while/until

→ Structured programming imposes discipline on the direct transfer of control

→ Implications on architecture?

12

The Maersk Mc-Kinney Moller Institute

sdu.dk

Object-oriented programming

#sdudk

→Abstract datatypes
→Encapsulation
→Information hiding and interfaces

→Inheritance
→Polymorphy

→Object-orientated programming imposes discipline on the indirect transfer of control

→Implications for software architecture?

Let S be a stack.

Initialization:
S = ∅

Push Operation: (element 'x') onto the stack S:
S' = S ∪ {x}

Pop Operation:
S' = S - {y}

Top Operation:
Top(S) = y

IsEmpty Operation:
IsEmpty(S) = (S = ∅)

Size(S) = |S|

13

The Maersk Mc-Kinney Moller Institute

sdu.dk

Functional Programming

#sdudk

→ Pure functions are the foundation of functional programming.
  → They always produce the same output for the same input.
  → They don't have side effects, meaning they don't modify external state or data.
→ Immutability
  → In functional programming, data is immutable, meaning once it's created, it cannot be changed
→ First-Class and Higher-Order Functions
  → In functional programming, functions are first-class citizens, meaning they can be treated as values
→ Recursion
  → Functional programming often uses recursion as a looping mechanism instead of traditional loops
→ Avoiding State
  → Functional programming discourages the use of mutable variables and global state.

```
def factorial(n):
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)
```

Functional programming imposes discipline upon assignment

Implications on software architecture?

14

The Maersk Mc-Kinney Moller Institute

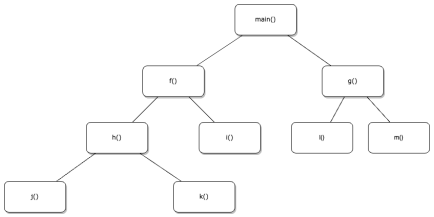
sdu.dk

#sdudk

Source code dependencies and flow of control

→ Divide and conquer, i.e. functional decomposition implies that the flow of control and the source code dependencies follow each other

- main()
  - f()
    - h()
      - j()
      - k()
    - i()
  - g()
    - l()
    - m()



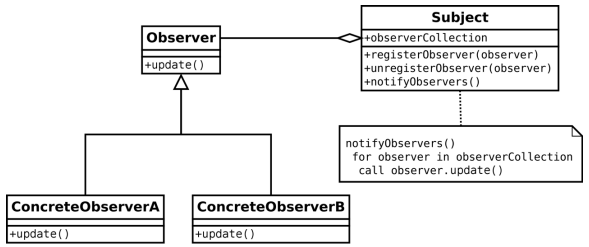
15

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Example: Observer Pattern



https://en.wikipedia.org/wiki/Observer_pattern#/media/File:Observer_w_update.svg

16

The Maersk Mc-Kinney Moller Institute

sdu.dk

SOLID

#sdudk

→ **S**RP: The Single Responsibility Principle
- → What is the responsibility of your class/component/microservice?
- → There should be no "and" in the aswer

→ **O**CP: The Open-Closed Principle
- → Software entities (classes, modules, functions, etc.) should be open for extension, but closed for modification
- → Polymorphy vs. interfaces

→ **L**SP: The Liskov Substitution Principle
- → Let Φ(x) be a property provable about objects x of type T. Then Φ(y) should be true for objects y of type S where S is a subtype of T.
- → Don't implement any stricter validation rules on input parameters than implemented by the parent class.
- → Apply at the least the same rules to all output parameters as applied by the parent class.

→ **I**SP: The Interface Segregation Principle
- → Clients should not be forced to depend upon interfaces that they do not use

→ **D**IP: The Dependency Inversion Principle
- → High-level modules should not depend on low-level modules. Both should depend on abstractions.
- → Abstractions should not depend on details. Details should depend on abstractions.

[Martin 2018]

17

The Maersk Mc-Kinney Moller Institute

sdu.dk

Motivation

#sdudk

→ Cyber physical systems

→ Emerging technologies

→ Data intensive applications

→ Scalability and availability

→ High performance database when reading and writing

420 systems in ranking, August 2023

Rank Aug 2023	Rank Jul 2023	Rank Aug 2022	DBMS	Database Model	Score Aug 2023	Score Jul 2023	Score Aug 2022
1.	1.	1.	Oracle	Relational, Multi-model	1242.10	-13.91	-18.70
2.	2.	2.	MySQL	Relational, Multi-model	1130.45	-19.89	-72.40
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	920.81	-0.78	-24.14
4.	4.	4.	PostgreSQL	Relational, Multi-model	620.38	+2.55	+2.38
5.	5.	5.	MongoDB	Document, Multi-model	434.49	-1.00	-43.17
6.	6.	6.	Redis	Key-value, Multi-model	162.97	-0.80	-13.43
7.	↑ 8.	↑ 8.	Elasticsearch	Search engine, Multi-model	139.92	+0.33	-15.16
8.	↓ 7.	↓ 7.	IBM Db2	Relational, Multi-model	139.24	-0.58	-17.99
9.	9.	9.	Microsoft Access	Relational	130.34	-0.38	-16.16
10.	10.	10.	SQLite	Relational	129.92	-0.27	-8.95
11.	11.	↑ 13.	Snowflake	Relational	120.62	+2.94	+17.50
12.	12.	↓ 11.	Cassandra	Wide column, Multi-model	107.38	+0.86	-10.76
13.	13.	↓ 12.	MariaDB	Relational, Multi-model	98.65	+2.55	-15.24
14.	14.	14.	Splunk	Search engine	88.98	+1.87	-8.46
15.	↑ 16.	15.	Amazon DynamoDB	Multi-model	83.55	+4.75	-3.71
16.	↓ 15.	16.	Microsoft Azure SQL Database	Relational, Multi-model	79.51	+0.55	-6.67
17.	17.	17.	Hive	Relational	73.35	+0.48	-5.31
18.	18.	↑ 22.	Databricks	Multi-model	71.34	+2.87	+16.72
19.	19.	↓ 18.	Teradata	Relational, Multi-model	61.31	+1.06	-7.76
20.	20.	↑ 24.	Google BigQuery	Relational	53.90	-1.52	+3.87
21.	21.	↑ 23.	FileMaker	Relational	53.85	+0.53	+0.73
22.	22.	↓ 19.	Neo4j	Graph	51.42	-0.64	-7.93

19

The Maersk Mc-Kinney Moller Institute

sdu.dk

SQL and noSQL databases

#sdudk

→ SQL Databases
- → Structured Data
- → ACID Transactions
- → Relational Model
- → Consistency
- → Vertical Scalability

→ noSQL Databases
- → Flexible Schema
- → BASE Transactions
- → Variety of Data Models
- → Horizontal Scalability
- → Eventual Consistency

→ Impacts on software architecture?

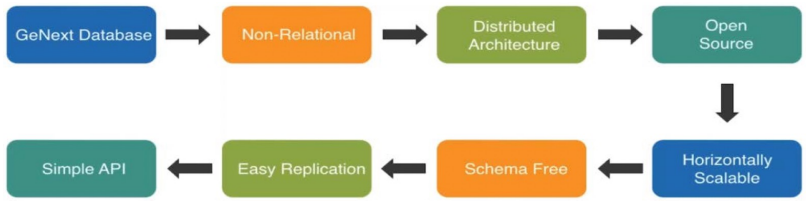
20

The Maersk Mc-Kinney Moller Institute

sdu.dk

NoSQL – Essential Features

#sdudk

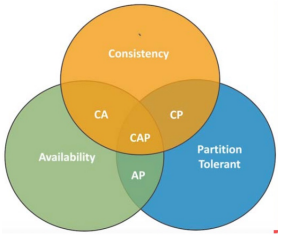


Khan 2023

21

The Maersk Mc-Kinney Moller Institute

CAP



Consistency	Availability	Partition Tolerance
• Consistency means that the data in the database remain consistent after the execution of an operation. • For example, after an update operation, all clients see the same data.	• Availability means that the system will not have downtime (100% service uptime guaranteed). • Every node (if not failed) always executes the query.	• Partition tolerance means that the system continues to function even when the communication among servers is unreliable. • The servers may be partitioned into multiple groups that cannot communicate with one another.

Khan 2023

The Maersk Mc-Kinney Moller Institute

RDBMS vs. NoSQL

DBMSs	Features								
	Data	Schema	Scalability	Compliance	Architecture	Consistency	Query Language	Performance	Best Suited For
RDBMS	S	Fixed	Vertical	ACID	Centralized	Strict	SQL	Slow	BFT
NoSQL	SUSm	Dynamic	Horizontal	BASE	Distributed	Eventual	OO API, SQL Like	Fast	LSWA, SD

S: Structured; SUSm: Structured, Unstructured, and Semi-Structured; LSWA: Large-Scale Warehouse Applications, Sensor Data; BFT: Banking, Financial Transaction.

Khan 2023

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Peer Review

→ A formal assessment that
- → Finds errors
- → Shortcomings
- → Excellences

→ A review checks whether an exercise meets the requirements and is professionally adequate on the basis of
- → the exercise description
- → the criteria in the journal writing rubric

→ A review should primarily be factual and sober and not just praise the content.

24

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Review contents

→ A review must include the following sections:

→ Review Formalia (Title, Date, Group reviewed, Reviewer)

→ General comments about the journal, followed by specific comments on
- → Exercise formalia and template (if applicable)
- → Rubric

Criteria	Poor	Sufficient	Good	Excellent
Description of the domain	Incomplete or inaccurate	Somewhat accurate but lacks depth	Accurate and reasonably thorough	Comprehensive and highly representative
Quality of use cases	Vague and uninformative	Somewhat informative but lacks detail	Clear, but could be more detailed	Clear, thorough, and very well-defined
System structure supports use cases	Does not support use cases	Partially supports use cases	Supports most use cases	Supports all use cases excellently
Quality of arguments	Unconvincing or lacking	Somewhat convincing, but not strong	Convincing for the most part	Highly convincing and well-supported

25

The Maersk Mc-Kinney Moller Institute

sdu.dk

Review Process

#sdupdk

→ There are two exercises created on itslearning:
- → Exercises from Lecture 2
  - → Hand in your answers to the exercises from Lecture 2 (as far as you are).
  - → No formal requirements but remember to write your group number and group member names in the ecercise
  - → Save you answer to a file named Group<no>_ecercise_lecture_2 in pdf format
  - → Deadline 15-09-2023 18:00
  - → After the deadline the hand-ins are distributed on itslearning in the folder "Review Material" under "Resources" and reviewers are assigned to the exercises
- → Peer review
  - → Hand in your peer review
  - → Use the rubric in the exercise to perform the review
  - → Deadline 20-09-2023 23:59
  - → After the deadline the reviews are distributed on itslearning in the folder "Review Material" under "Resources"

26

The Maersk Mc-Kinney Moller Institute

sdu.dk

Pitch

#sdupdk

→ You can find (business) inspiration here
→ https://ie.sdu.dk/index.php/videos-explaining-ie/pitch/

→ A pitch may contain these four elements:
- → Need
- → Approach
- → Benefits
- → Competition -> Alternatives

→ In the next lecture you should create a 3 minutes pitch that can convince the audience that you have created the right software architecture to fulfill you quality attributes
→ All groups will present their pitch in the second half of the lecture

27

The Maersk Mc-Kinney Moller Institute

sdu.dk

Exercises

#sdupdk

→ Programming languages for your system
  → Research, match, justify
→ Databases for your system
  → Research, match, justify

→ Peer review
  → See exercise and slides on peer review

28

The Maersk Mc-Kinney Moller Institute

sdu.dk

References

#sdupdk

1. Robert C.Martin. Clean Architecture, Pearson Education, 2018, ISBN 978-0-13-449416-6
2. Wisal Khan et al. “SQL and NoSQL Database Software Architecture Performance Analysis and Assessments—A Systematic Literature Review”. English. In: Big Data and Cognitive Computing 7.2 (2023). Copyright - © 2023 by the authors

29