The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Advanced Topics in Software Architecture (E23)

Tools and Technologies 3

1

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Agenda

→ Last weeks exercises
→ Message busses
→ Exercise(s)
→ Pitches

2

The Maersk Mc-Kinney Moller Institute

sdu.dk

Learning Objectives

#sdupdk

→ Explain tools and technologies for implementing software architecture
→ Select and combine tools and technologies to implement software architecture
→ Ability to apply software architectures for different quality attributes using tools and technologies
→ Explain and argue for software architecture and associated qualities attributes and architectural problem
→ Document, describe, and communicate modern software architectures

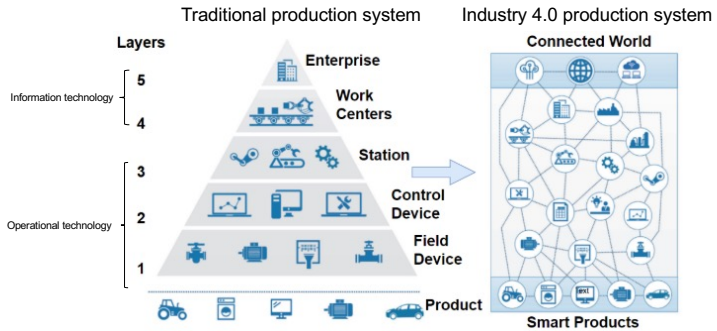
3

The Maersk Mc-Kinney Moller Institute

sdu.dk

Flexible Production System

#sdupdk



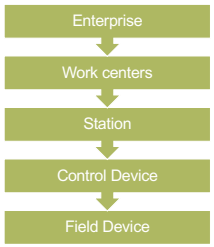
5

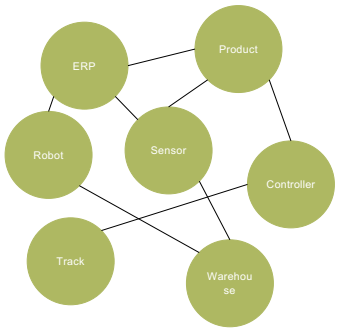
The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Layered and distributed systems





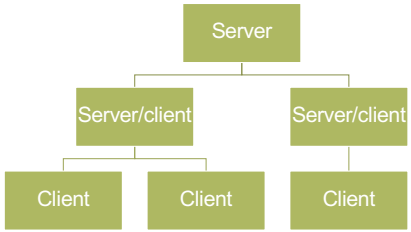
6

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Client server architecture



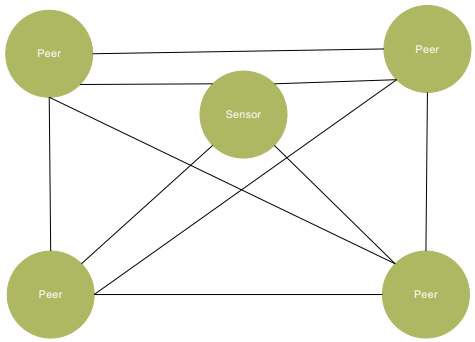
8

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Peer-to-peer



SDU Torben Worm and Sune Chung Jepsen February 2023 9

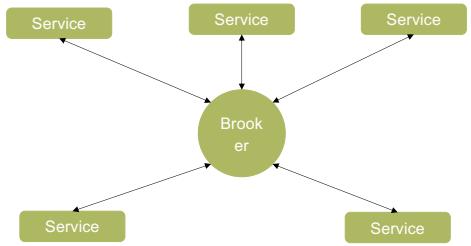
9

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Brooker



SDU Torben Worm and Sune Chung Jepsen February 2023 10

10

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Message Oriented Middleware

→ Middleware can realize the peer-to-peer model

→ A middleware is defined as a layer between software applications and system resources, which abstracts functionality of, for example, interfaces and operating system to afford interoperability.

→ Middleware implements and encapsulates additional functionality by providing a reusable interface for the user

→ A message oriented middelware in particular creates an interface for sending messages between distributed services

→ A programmer does not need to implement a communication technology, but can focus on his implementation task and can use a message oriented middelware to communicate with other processes

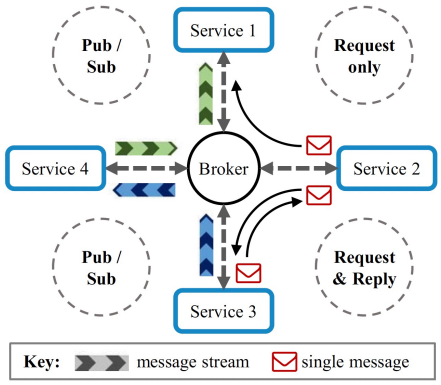
11

The Maersk Mc-Kinney Moller Institute

sdu.dk

#sdudk

Message Types



12

The Maersk Mc-Kinney Moller Institute

sdu.dk

Quality of service

#sdupdk

→ At least once
→ At most once
→ Exactly once

→ Load balancing
→ Clustering

13

The Maersk Mc-Kinney Moller Institute

sdu.dk

Security

#sdupdk

→ Encryption
→ Authentication
→ Authorization

14

The Maersk Mc-Kinney Moller Institute

sdu.dk

MQTT

#sdudk

→ MQTT is a Client Server publish/subscribe messaging transport protocol. It is lightweight, open, simple, and designed so as to be easy to implement. These characteristics make it ideal for use in many situations, including constrained environments such as for communication in Machine to Machine (M2M) and Internet of Things (IoT) contexts where a small code footprint is required and/or network bandwidth is at a premium.

→ MQTT uses a binary message format for communication between clients and brokers. This is in contrast to other protocols that use text-based formats, such as HTTP or SMTP.

→ The binary format used by MQTT is designed to reduce the size of messages and increase the efficiency of communication.

Bit	7	6	5	4	3	2	1	0
byte 1	MQTT Control Packet type				Flags specific to each MQTT Control Packet type			
byte 2…	Remaining Length							

https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/

15

The Maersk Mc-Kinney Moller Institute

sdu.dk

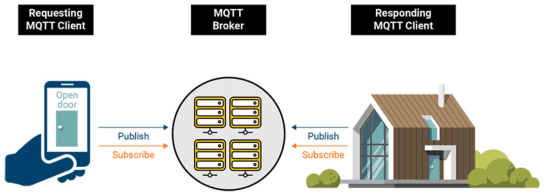
MQTT

#sdudk

→ MQTT's messaging model is based on topics and subscriptions.

→ Topics are strings that messages are published to and subscribed to. Topics are hierarchical and can contain multiple levels separated by slashes, like a file path as shown below.

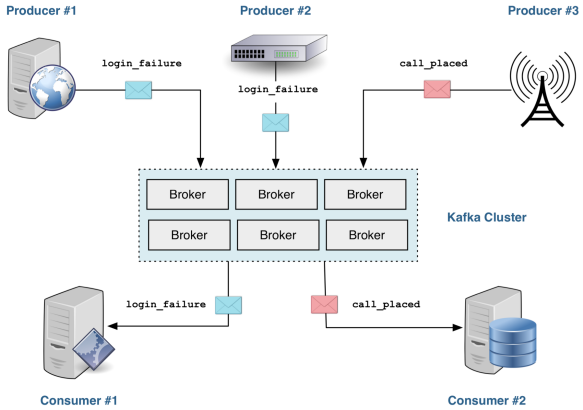
→ myhome/house/door



https://www.hivemq.com/blog/mqtt-essentials-part-1-introducing-mqtt/

16

Kafka



https://files.ondemand.cloudera.com/Kafka201601/Cloudera_Kafka_Training_Slides.pdf

18

Messages

→ Messages in Kafka are variable-size byte arrays
- → Allows for serialization of data in any format your application requires Common formats include strings, JSON, and Avro
- → There is no explicit limit on message size
- → Optimal performance usually occurs with messages of a few KB in size We recommend that you do not exceed 1MB per message

→ Kafka retains all messages for a defined time period
- → This period can be set on global or per-topic basis
- → Messages will be retained regardless of whether they were read
- → They are discarded automatically after the retention period is exceeded

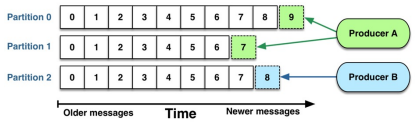
Z

19

The Maersk Mc-Kinney Moller Institute

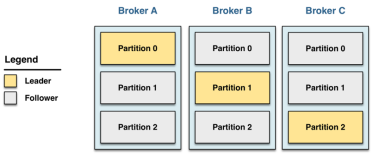
sdu.dk

Topics



#sdudk

- → There is no explicit limit on the number of topics
  - → Kafka works better with a few large topics than many small ones
- → A topic can be created explicitly or simply by publishing to the topic
  - → Controlled by the auto.create.topics.enable property
- → Each topic is divided into some number of partitions
  - → Partitioning improves scalability and throughput
- → A topic partition is an ordered and immutable sequence of messages
  - → New messages are appended to the partition as they are received
  - → Each message is assigned a unique sequential ID known as an offset
- → Each partition can be replicated across a configurable number of brokers
- → Each broker acts as a leader for some partitions and a follower for others
- → Followers passively replicate the leader
- → If the leader fails, a follower will automatically become the new leader



20

The Maersk Mc-Kinney Moller Institute

sdu.dk

Message Oriented Middleware Characteristics

#sdudk

MOM / Features	MQTT	AMQP 0.9	KAFKA	ZeroMQ
Implementation used in benchmark	Eclipse Mosquitto	RabbitMQ	Apache Kafka	JeroMQ
CP: Messaging types	Pub/Sub	Pub/Sub Request/Reply Point-to-point	Pub/Sub	Pub/Sub Request/Reply
CP: Technical communication realization	Broker	Broker	Broker	Brokerless / Point-to-point
CP: Transport Protokoll	TCP	TCP	TCP	Arbitrary: TCP, IPC, etc.
Payload type	Binary defined by user	Binary defined by user	Binary defined by user	Binary defined by user
QoS: Message delivery	At least once At most once Exactly once	Exactly once, with ack-nowledgement settings (To be extended in 1.0)	At least once At most once Exactly once	-
QoS: Load balancing, clustering	Yes*	Yes*	Yes	No
Security: Authentication	SASL PLAIN *	SASL with challenge-response	SASL PLAIN and Kerberos	SASL full
Security: Encryption	TSL	TSL *	SSL	CurveZMQ over TCP
Security: Authorization	ACL*	ACL *	ACL	-
I+S: Standardization	Yes (ISO/IEC 20922)	Yes (ISO/IEC 19464)	No	No
I+S: Open Source	Yes	Yes	Yes	Yes
I+S: License	Royalty-free / OASIS Standard	Royalty-free / OASIS Standard	Apache License 2.0	LGPLv3
I+S: Programming language of clients	●	●	●	●
I+S: Commercial support	●	●	●	◔
I+S: Active community**	●	●	●	●
I+S: Tutorial and documentation**	◑	◕	◕	●
I+S: Ease of implementation **	◕	●	◑	◔

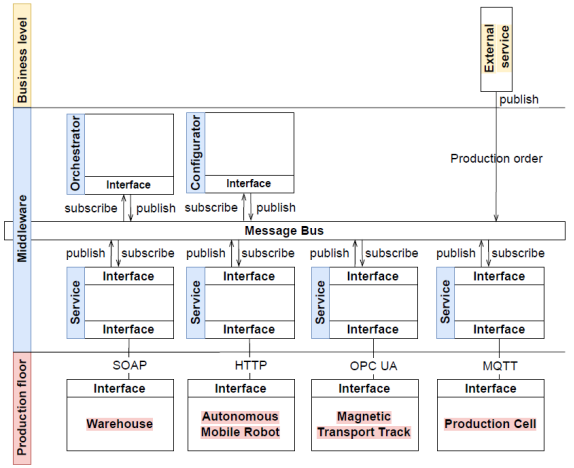
24

The Maersk Mc-Kinney Moller Institute

Example

→ SDU Industry 4.0 Middleware Software Architecture

→ Experimental system for working with I4.0 challenges



sdu.dk

#sdupdk

25

The Maersk Mc-Kinney Moller Institute

Exercises

→ If you haven't worked with the peer-reviews, now is the time
  → Use the feed-back to reflect on your solustions and your arguments

→ After the exercise you will present your pitches for inspiration, knowledge sharing, and discussion

sdu.dk

#sdupdk

26