Aalto University
School of Science

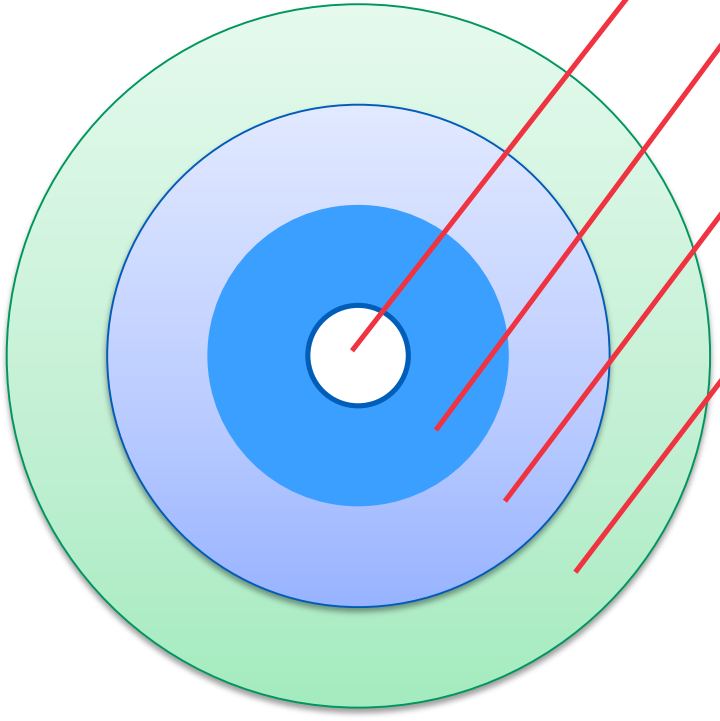
Architecting Big Data Platforms

Hong-Linh Truong
Department of Computer Science
linh.truong@aalto.fi, https://rdsea.github.io

What is this lecture about

- **Your big data platform story**
- **Different big data architectures**
- **Our picture of big data platform technologies in this course**
- **Key architecture design issues**
  - Interaction, Partition, Elasticity, API

Recall: Big data platforms



Data-centric Virtualized Infrastructures

Middleware Platforms: building, deploying, operating and scaling big data services and applications

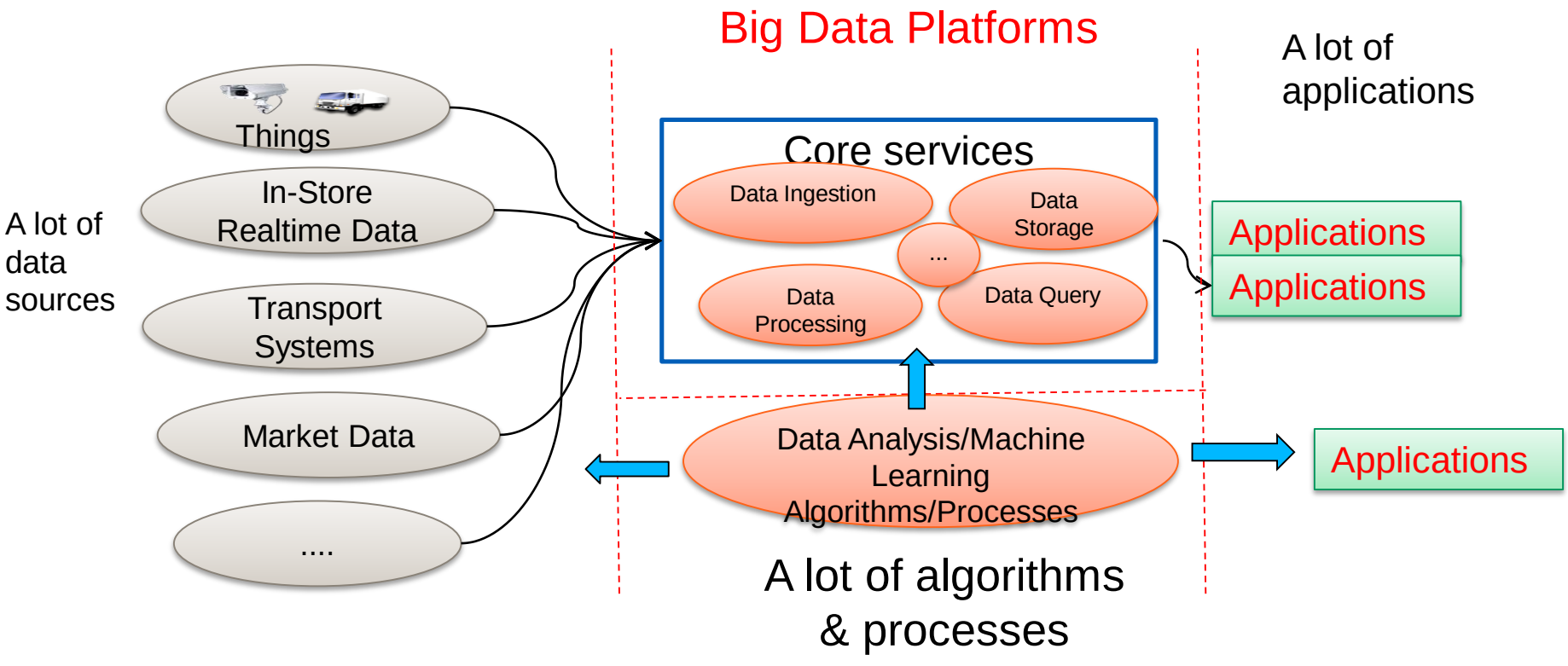
Big data services & applications

Big data platforms consumers and producers: Sensors, Things, Equipment, Industrial Processes, People

Movement of the data in the platform

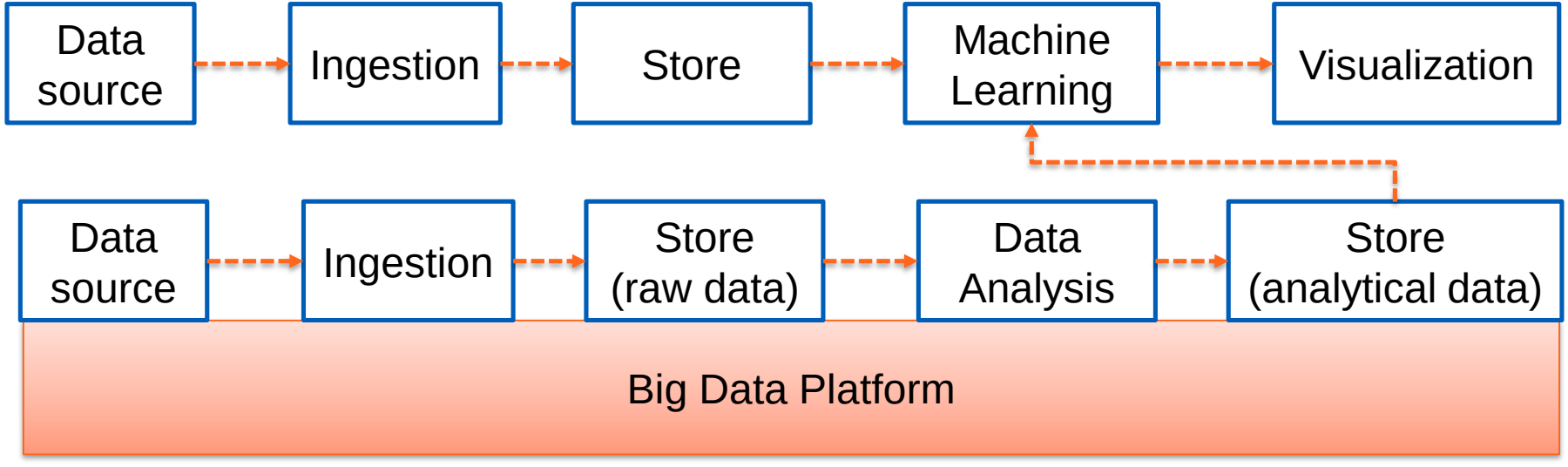
- **Ingestion**
  - From various data sources we move data into the platform
- **Storing**
  - Ingested data will be stored and managed using different types of storages and databases
- **Analyzing and (Machine) Learning**
  - Data within platforms will be processed, analyzed and learned to improve data, find insights and to create models
- **Reporting and Visualization**
  - Patterns in data will be discovered and interpreted for decision-making, reporting and creating stories

Recall: A bird view of big data platforms



Big Data Pipelines

Multiple big data pipelines can be constructed atop a big data platform



Your big data platform story - an evolving scenario

"Your team has to build a big data platform for X types of data. Data will be generated/collected from N sources. We expect to have 10+ GBs/day of data to be ingested into our platform. We will have to serve K thousands of requests for different types of analytics – to be determined. Our response time should be in t milliseconds. Our services should not be …"

Remember big data V* and platform definition?

What would be your approach? Tell us your first first action?

@All: Things described will not be the same after a while!

You may have similar questions?

- How to design elastic big data infrastructures?
- Do we have to support multiple types of data?
- How do data pipelines look like?
- How to enable different data processing models?
- How do support various SLA ?
- Which part of the platform we must do self-manage and which part will be fully managed by others?
- To where we should distribute our components?
- Etc.

Your Big Data Platform story starts with Big Data Platform architectures!

To architect the platform centered around data!

Handling multiple types of data?

- **First important aspect: you don't have to support multiple types of data**
  - But are you sure that you will not have it in the future?
- **Multiple types of data**
  - Any linked models among them?
- **Any good solution that enable changes with minimum changes**
  - *E.g., multi-model databases, microservices of multiple of databases (Lectures:#storage#integration)*

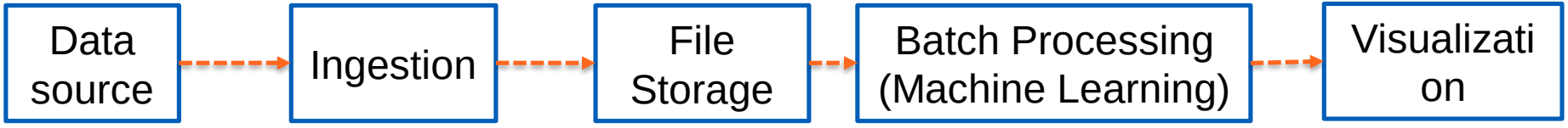
Ingesting, Storing and SLA

- **Ingesting data**
  - Mapping and transforming data
  - Ingestion under V*
  - Data validation during ingestion
- **Storing data**
  - Data Sharding and Consistency, data backup, retention, etc.
- **SLA Multitenancy versus single tenancy**
  - Security, privacy, performance and maintenance?

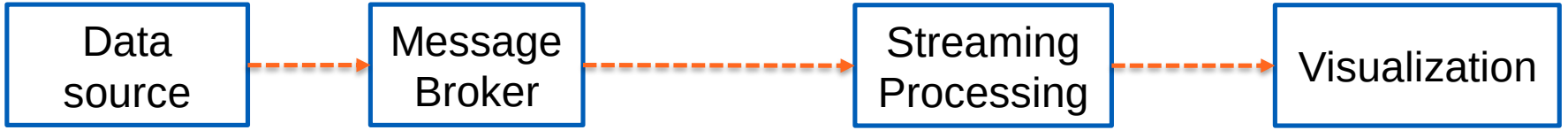
Lectures: #ingestion, #storage

Basic Big Data Pipelines

big data but not real-time, e.g., take customer transaction files from companies and move to data centers for analytics



fast, small IoT data in real-time flows, e.g. position of cars



But

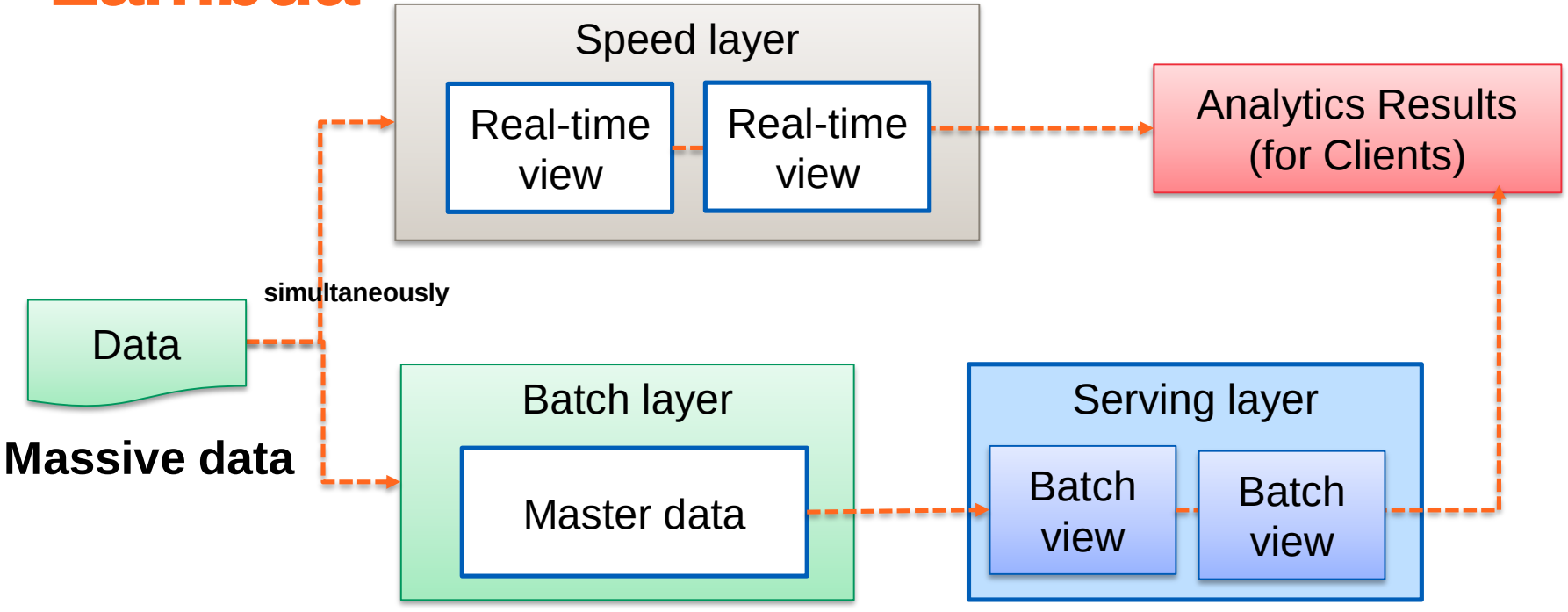
if you have mixed types of data

Or

if you have big data you want to do analytics with different quality of analytics (cost, performance, quality of data)?

Then ?

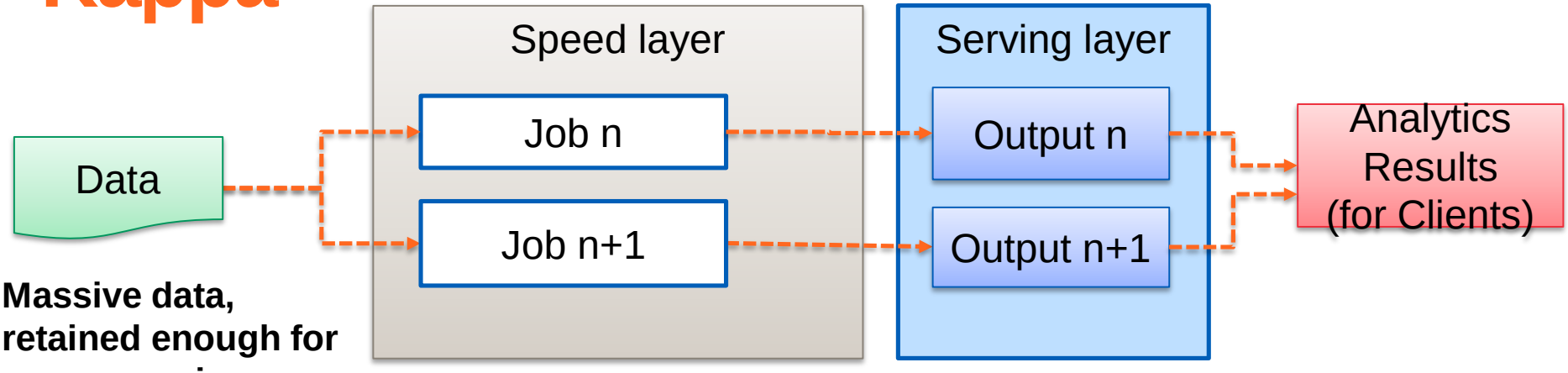
Lambda



Check: http://lambda-architecture.net/

Lectures: #ingestion, #streaming

Kappa



Check: https://milinda.pathirage.org/kappa-architecture.com/

Lectures: #ingestion, #streaming

The set of big data tools/frameworks (and configurations) used is dependent on the big data architecture

be aware of your #techradar!

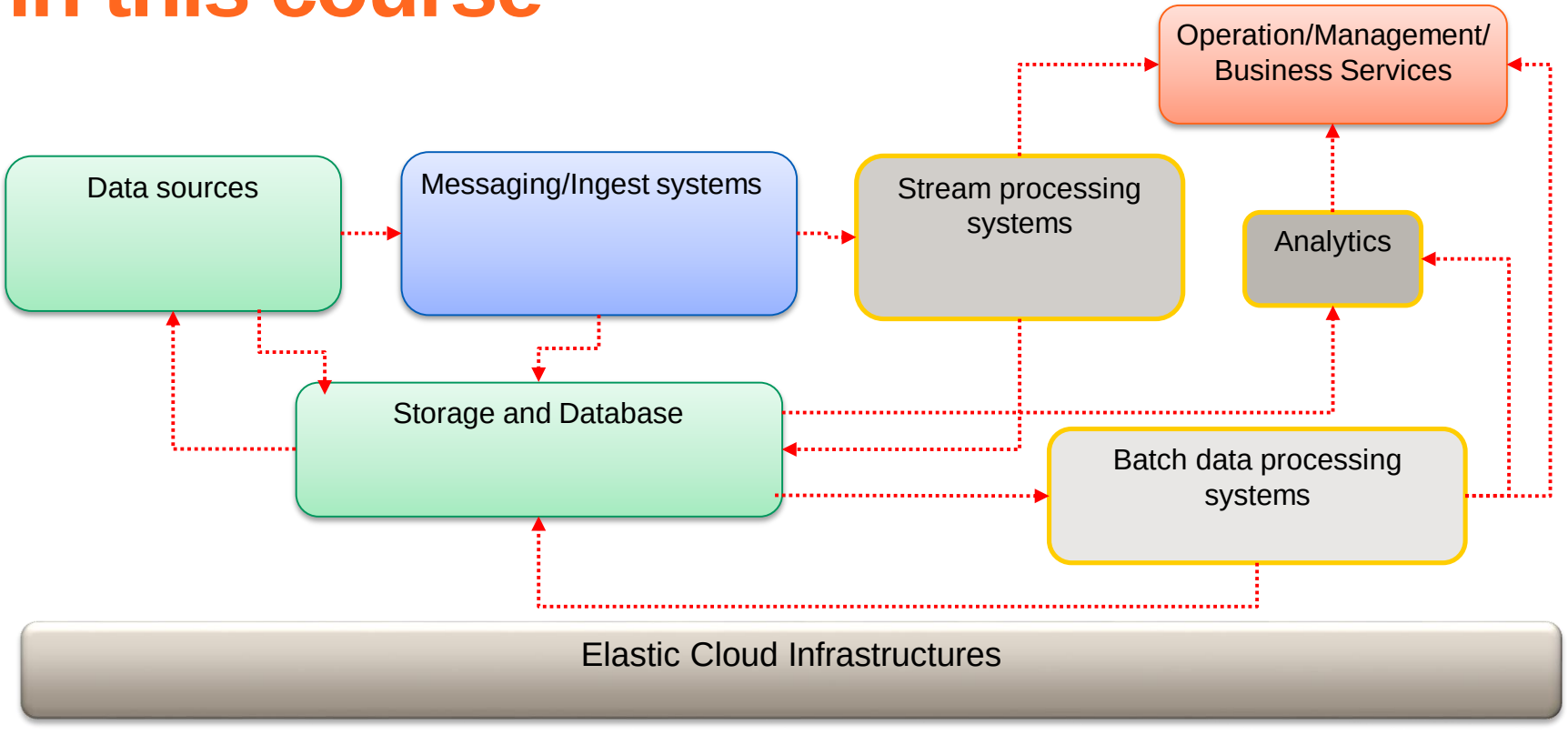
Quick check

“A big data platform monitors network usage of devices from million+ customers. We have different levels: Sensor/Customer, Node (concentrator of multiple customers), Agent (concentrator of multiple Nodes) and the whole network. In a region, the real operator can generate 1.4 billion records per day ~ 72GB per day”

Quickcheck

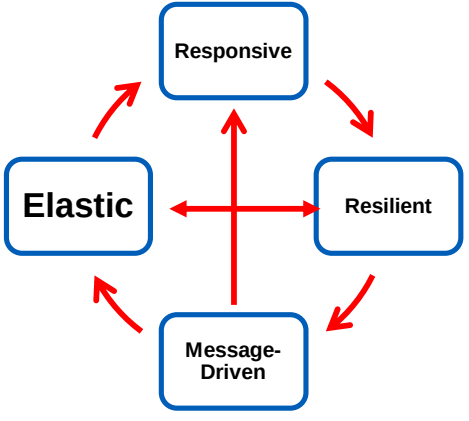
https://bit.ly/2kPBdD8

Big data at large-scale: the big picture in this course



How to architect big data platforms and pipelines as reactive systems?

Reactive systems



Source: https://www.reactivemanifesto.org/

Why? For dealing with V*

- Responsive: quality of services
- Resilient: deal within failures
- Elastic: deal with different workload and quality of analytics
- Message-driven: allow loosely coupling, isolation, asynchronous

Designs must address various aspects

- **Responsive:**
  - distributed computing, multi layer optimization
- **Resilient:**
  - replication, containment, isolation
- **Elastic:**
  - sharding, replication, load balancing, scale up/out
- **Message-driven:**
  - loosely coupling of services with messages, non-blocking protocols, location-independent

Open sources and build stuffs yourself?

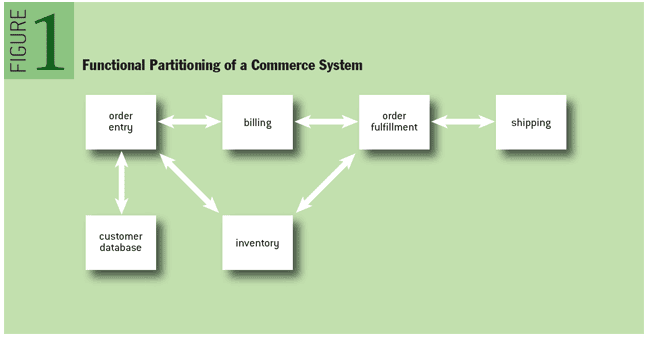
- **From open sources or existing enterprise version?**
  - Many hard problems for design decisions: cost, skills/support, regulation, …
- **Be aware of your technical debt**
  - I am familiar with XYZ so I just select it!
  - Our company uses Apache Storm in the past so …

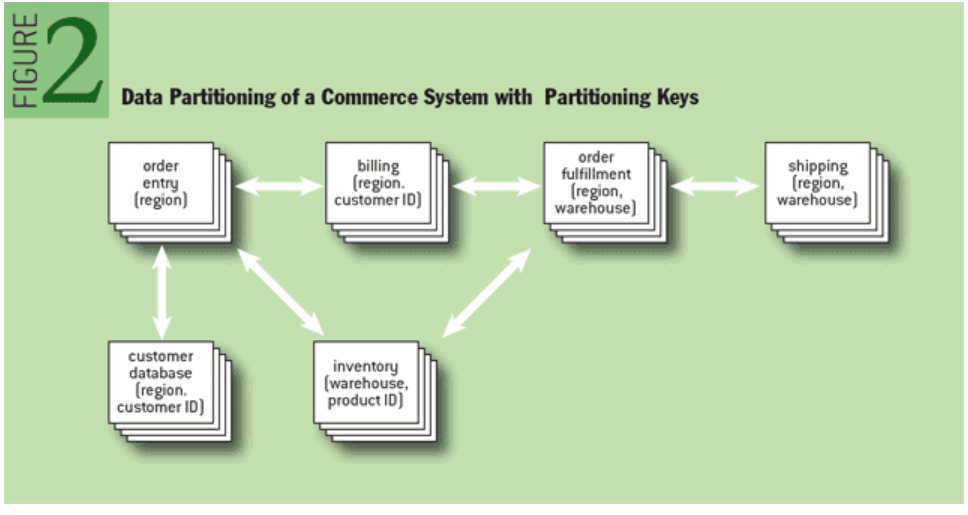
Tutorial 1 #techradar (Thu, 19.09)

Partitioning: Splitting functionality & data

- Breakdown the complexity
- Easy to implement, replace, and compose
- Deal with performance, scalability, security, etc.
- Support teams in DevOps
- Cope with technology changes

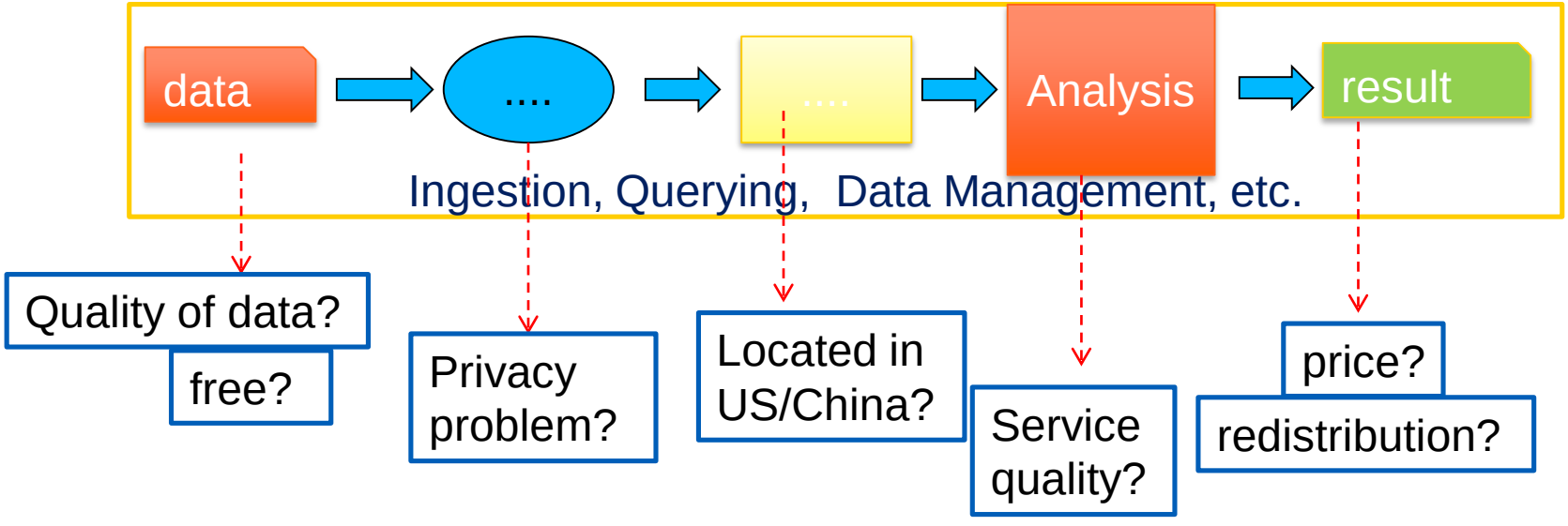
Example of Functional and Data Partitioning





Figures source: http://queue.acm.org/detail.cfm?id=1971597

Data concerns: data validation and quality of analytics



- Ethical consequence?
- Regulation-compliant platforms: e.g., GDPR

Lecture: #governance,#quality

Aalto University
School of Science

Distributed systems of components are used to manage, ingest data and process data

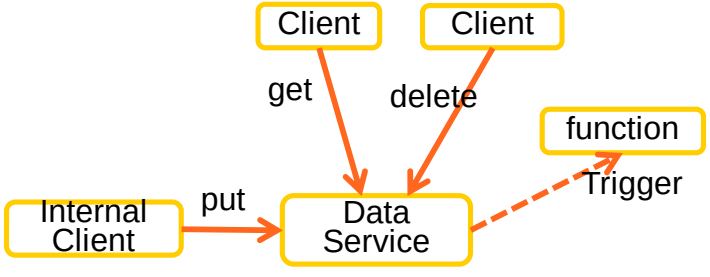
Interaction: protocols & interfaces

- **Large number of communication protocols and interfaces**
- **Interaction styles, protocols and interfaces**
  - REST, gRPC, Message Passing, Stream-oriented Communication
  - Your own protocols
- **Other criteria**
  - Architectural styles: microservices/serverless
  - Scalability, Elasticity, Performance, Monitoring, Logging, etc.

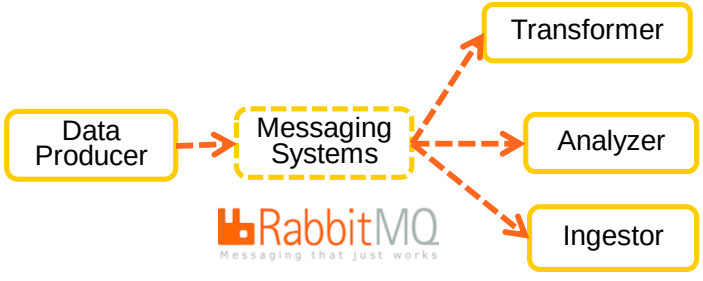
Lectures: #integration, #quality

Interaction: Complex interactions

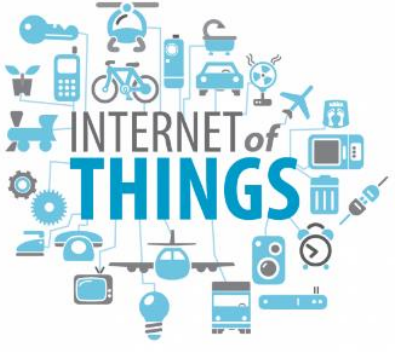
- One-to-many, Many-to-one, Many-to-Many
- Synchronous/Asynchronous calls
- Public/Subscribe, Message-oriented Middleware
- Internal data exchange versus open/external exchange

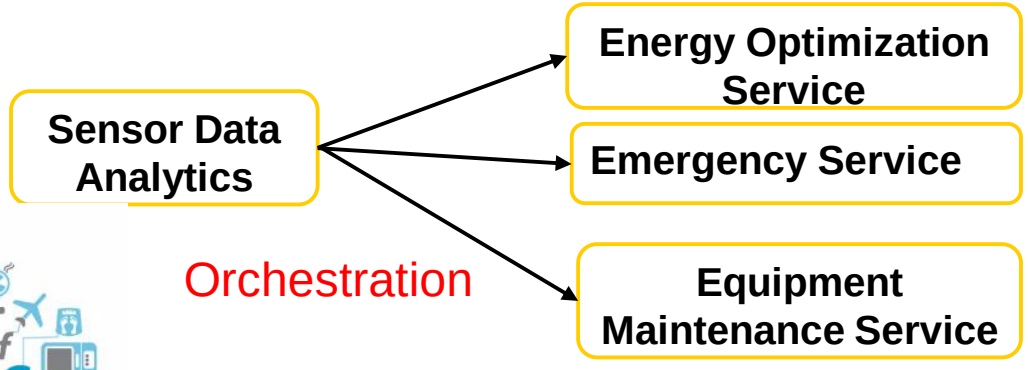


Amazon S3/MongoDb

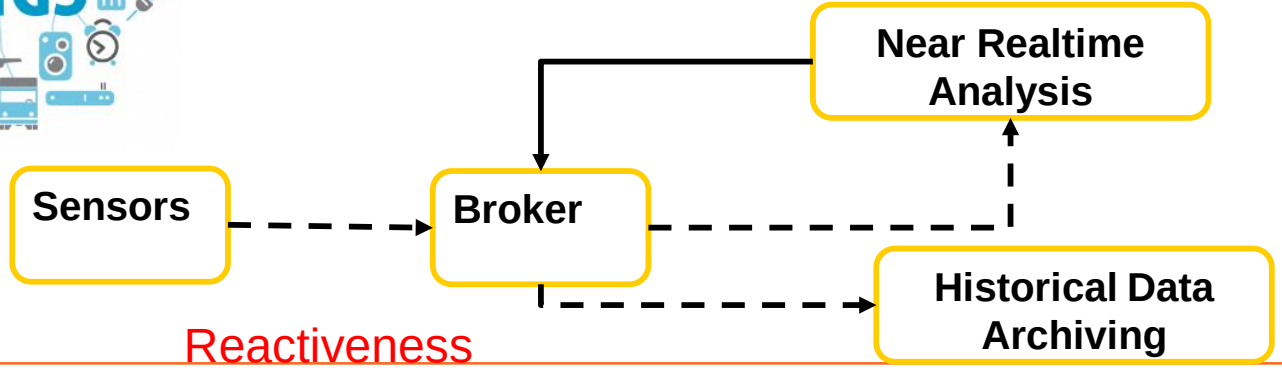


Coordination: Orchestration and Reactiveness





Lectures: #integration, #workflow



Lectures: #integration, #streaming

Distribution: Edge or Data Centers?

Big data & components components can be distributed in different places!

Global deployment or not?

Move analytics/work or move data?

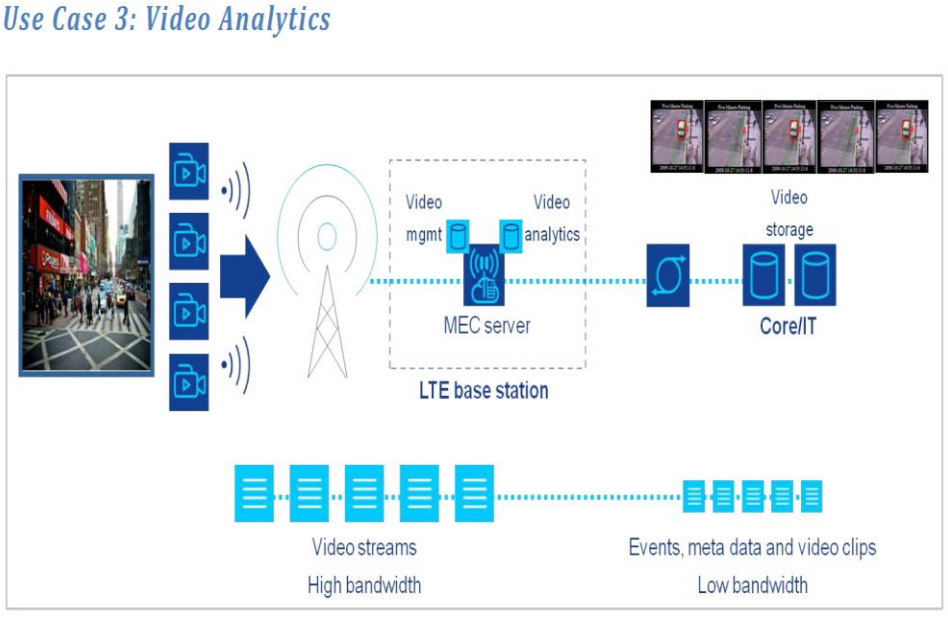


Figure 4: Example of video analytics

Figure source: https://portal.etsi.org/portals/0/tbpages/mec/docs/mobile-edge_computing_-_introductory_technical_white_paper_v1%2018-09-14.pdf

Quick check

“A big data platform monitors network usage of devices from million+ customers. We have different levels: Sensor/Customer, Node (concentrator of multiple customers), Agent (concentrator of multiple Nodes) and the whole network. In a region, the real operator can generate 1.4 billion records per day ~ 72GB per day”

Quick check

https://tinyurl.com/y3vyd777

Scalability and Elasticity: Scale out

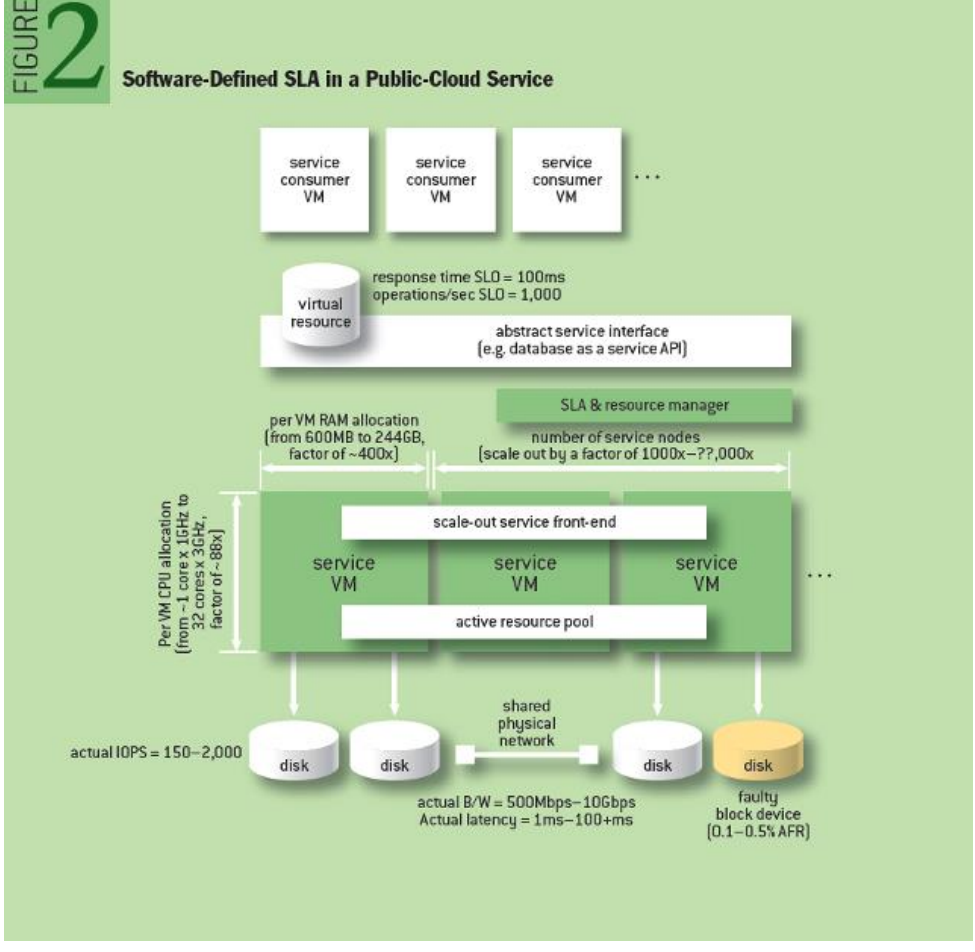


Figure source: http://queue.acm.org/detail.cfm?id=2560948

Scalability and Elasticity: Load balancing

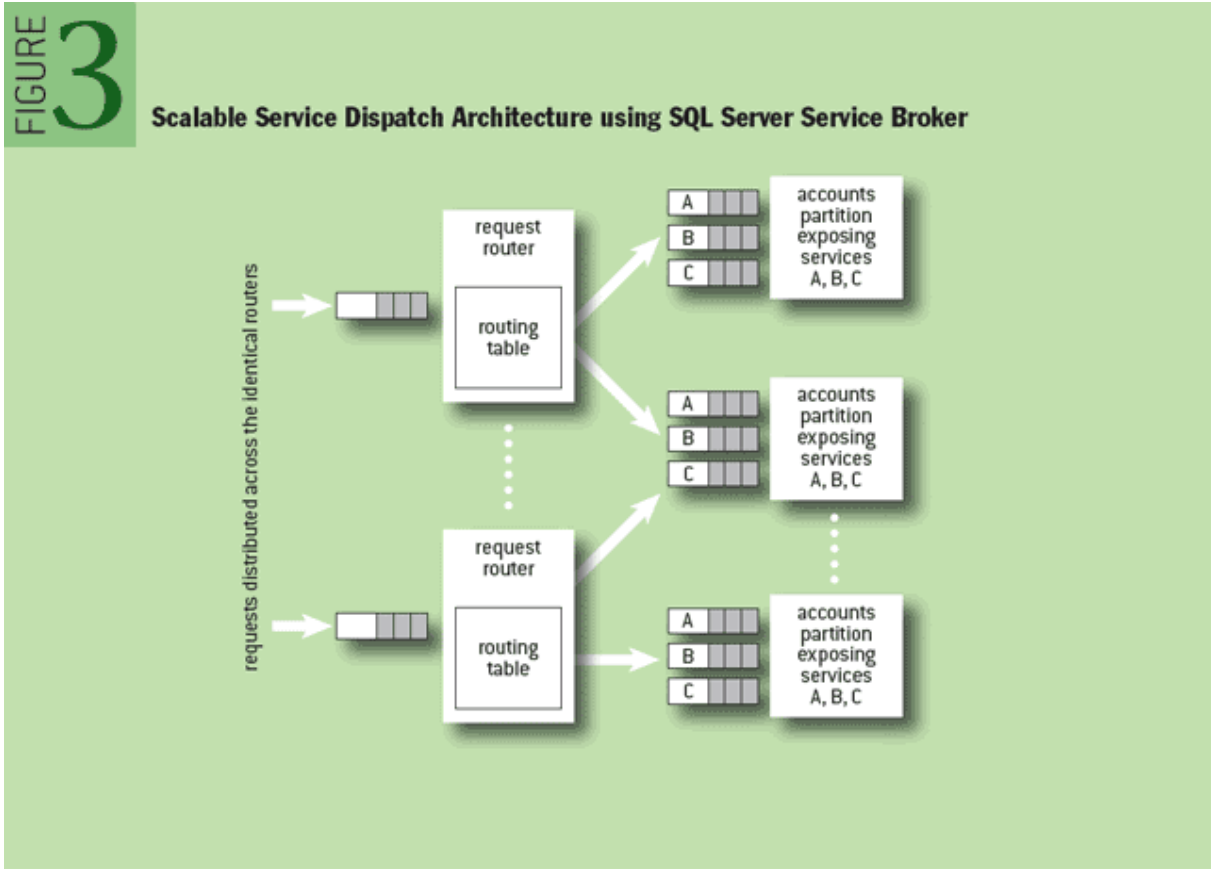


Figure source: http://queue.acm.org/detail.cfm?id=1971597

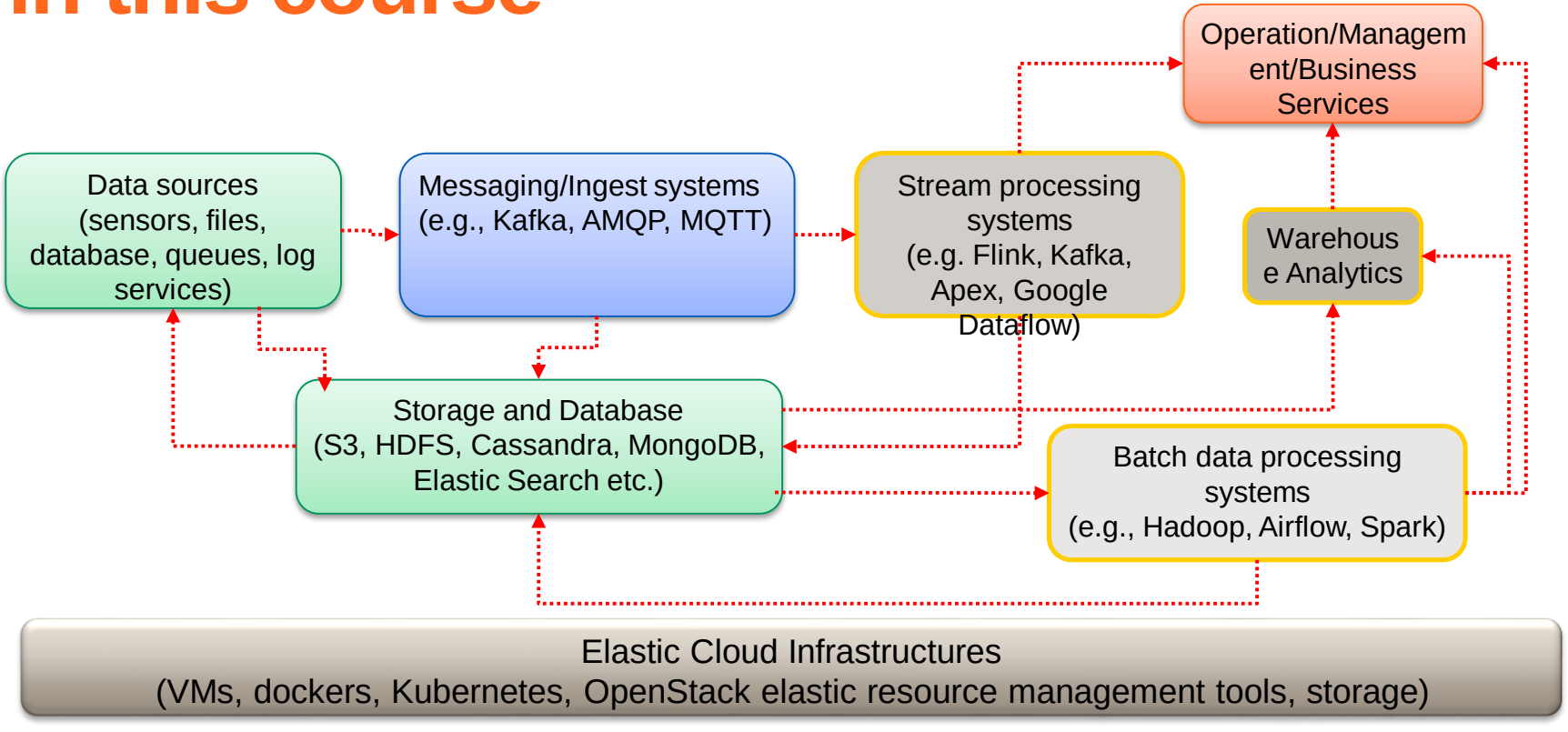
API for Platform as a Service

- **APIs are key! Why?**
  - Enable access to data and function from entities in your ecosystem without worrying about changes within your organization
  - Virtualization (hide internal, control access, throttling)

Which API would you publish? And how other concepts are related, e.g. API Gateways for Load balancing and Fault-Tolerance?

Common, high-level architecture view

Big data at large-scale: the big picture in this course



Note the next activities

- **Thu 19.09: Tutorial Walkthrough (9.15-10)**
  - Location: TU1
- **Wed 26.09: Lecture on Service & Integration Models**
  - Done with Module 1: on big picture of big data platforms, design/architecture
  - Start the first assignment

Thanks!

Hong-Linh Truong
Department of Computer Science

rdsea.github.io