

Trends / observations / tech
discussion over coffee

Agenda

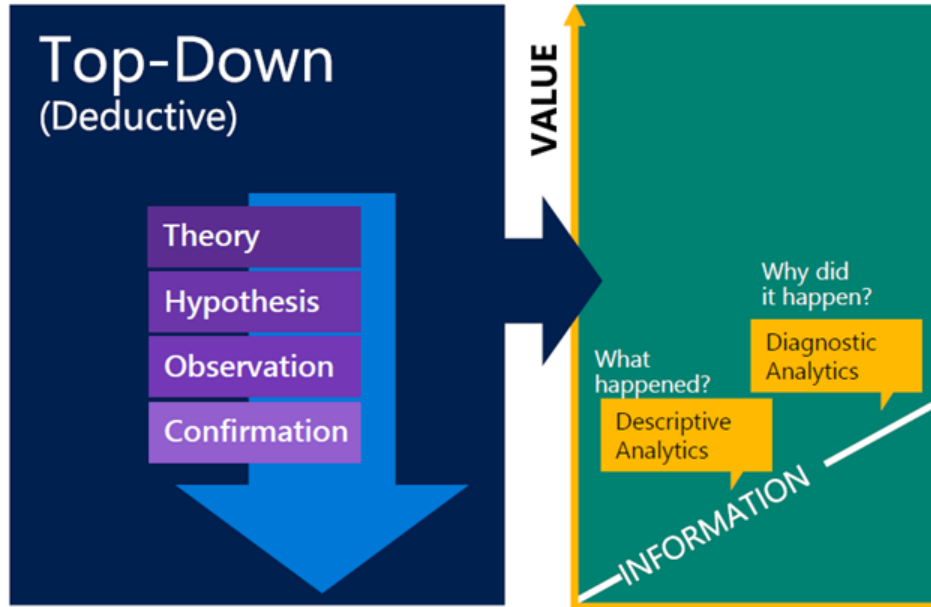
Data platform components and architecture

1. Data warehouse vs data lake
2. The rise of Kubernetes
3. What is data mesh?

Patterns and principles

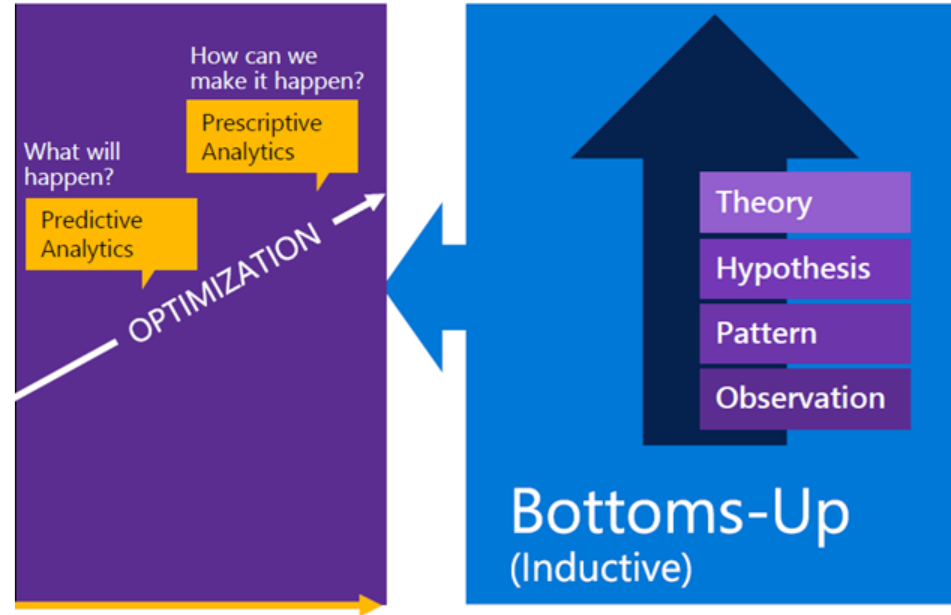
1. Data integrations options
2. Data acquisition options
3. Fitness functions

Data warehouse vs data lake



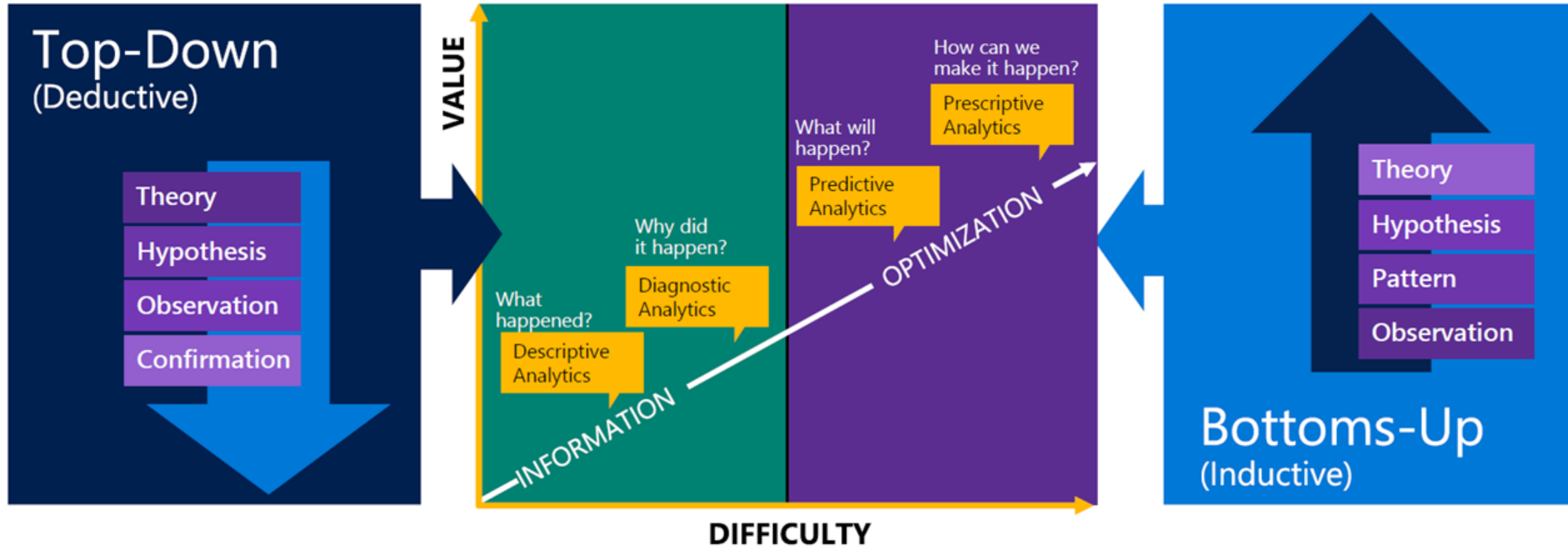
- Know the questions to ask
- Lot's of upfront work to get the data to where you can use it
- Model first

Data warehouse vs data lake



- Don't know the questions to ask
- Little upfront work needs to be done to start using data
- Model later

Data warehouse vs data lake



- Know the questions to ask
- Lot's of upfront work to get the data to where you can use it
- Model first

- Don't know the questions to ask
- Little upfront work needs to be done to start using data
- Model later

Is Hadoop Dead? How Kubernetes and Cloud-Native Could Displace Hadoop

Is Hadoop Officially Dead?

Alex Woodie



(Christos-Georgiou/Shutterstock)

The merger of
applauded by
and even Wa
confetti from
questions, li
of Hadoop
forward. Th

The Octob
join force

single company with about \$730 million in annual re
market valuation took a lot of people by surprise.

"My first reaction was I think it's good news for p
Zweben, who had a front-row seat to the first d
ity that we didn't see two



The New Age of Big Data: Is It the Death of Hadoop?

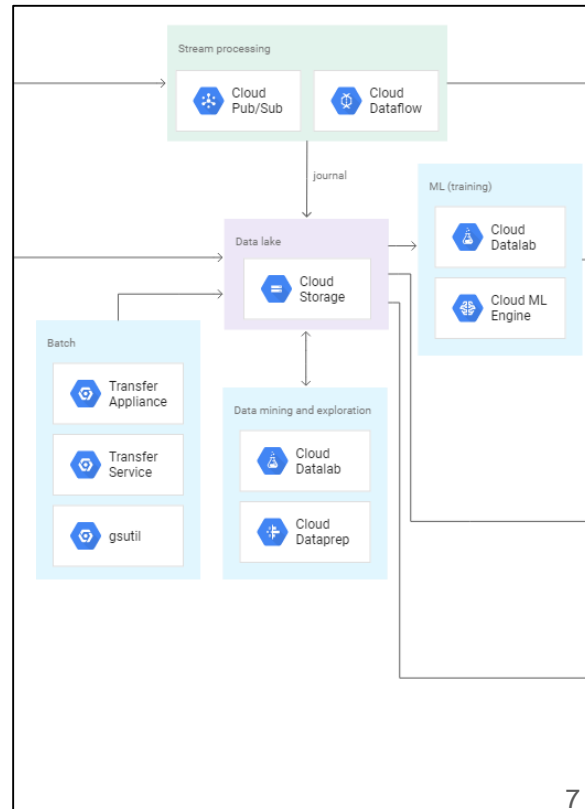
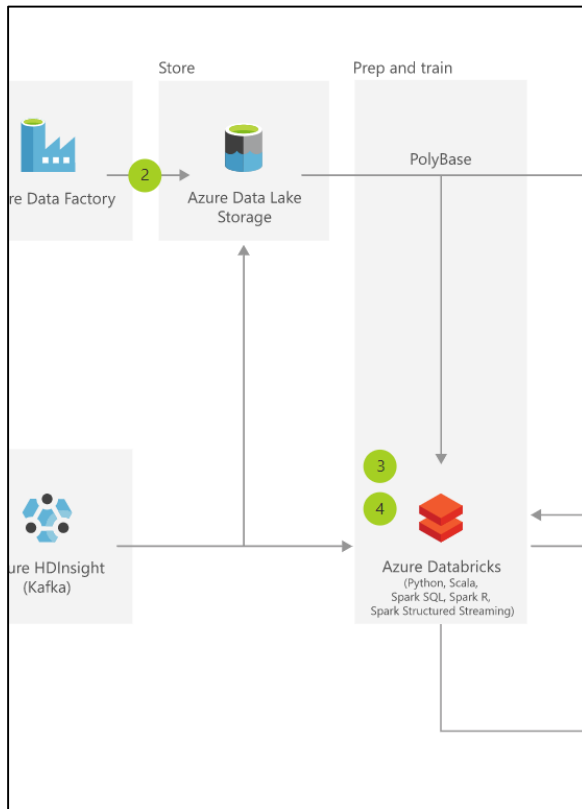
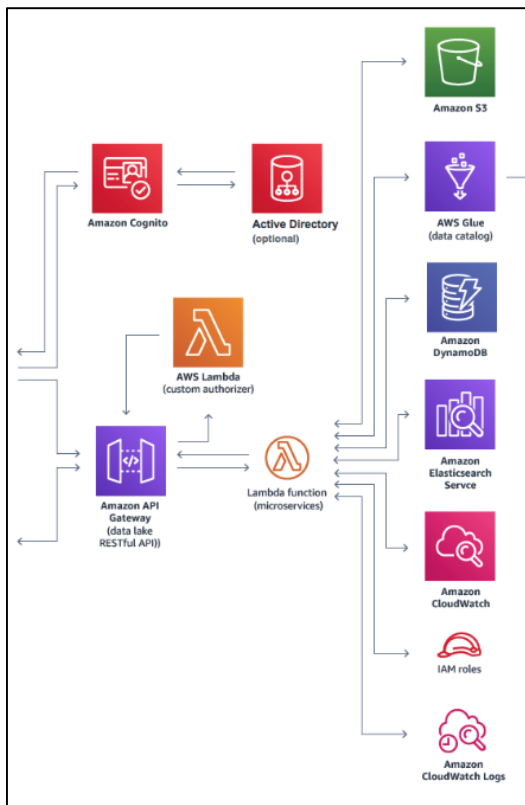
July 22, 2019 / in Big Data, Main Category / by Edward Huskin

Big Data had gone through several transformations
first identified use on the back of Hadoop
technologies such as K

It's Time to Stop Using Hadoop for Analytics

years, growing into the phrase we identify it as today. From
w age of Big Data has been ushered in with the spread of n
ually, but they fill the same niche and

Big data cloud vendors race



Kubernetes to help?

- No vendor lock-in, run everywhere
- Declarative configurations and deployment
- Microservice architectures
- Save money by optimizing infrastructural resources thanks to the more efficient use of hardware

The origins of Data Mesh


[martinFowler.com](#)

[Refactoring](#) [Agile](#) [Architecture](#) [About](#) [ThoughtWorks](#) [RSS](#) [Twitter](#)

How to Move Beyond a Monolithic Data Lake to a Distributed Data Mesh

Many enterprises are investing in their next generation data lake, with the hope of democratizing data at scale to provide business insights and ultimately make automated intelligent decisions. Data platforms based on the data lake architecture have common failure modes that lead to unfulfilled promises at scale. To address these failure modes we need to shift from the centralized paradigm of a lake, or its predecessor data warehouse. We need to shift to a paradigm that draws from modern distributed architecture: considering domains as the first class concern, applying platform thinking to create self-serve data infrastructure, and treating data as a product.

20 May 2019

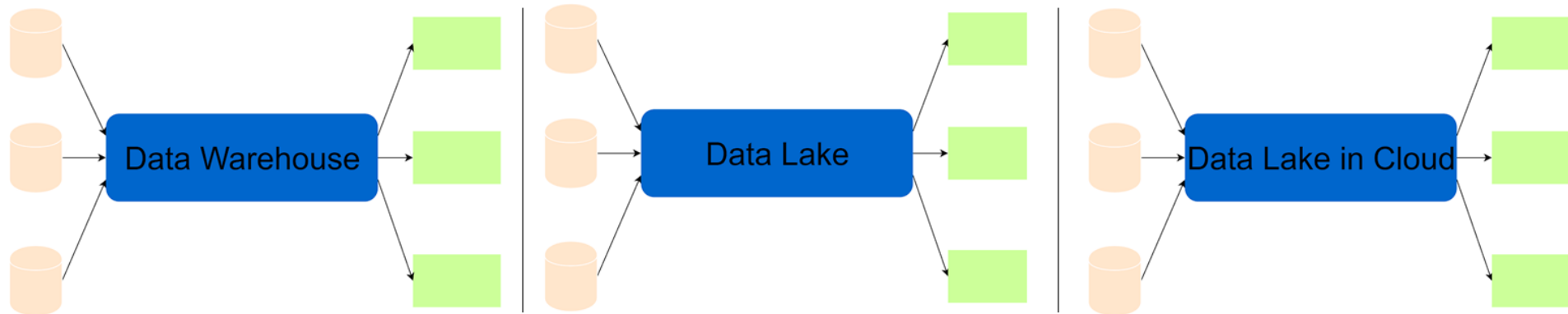


Zhamak Dehghani
Zhamak is a principal technology

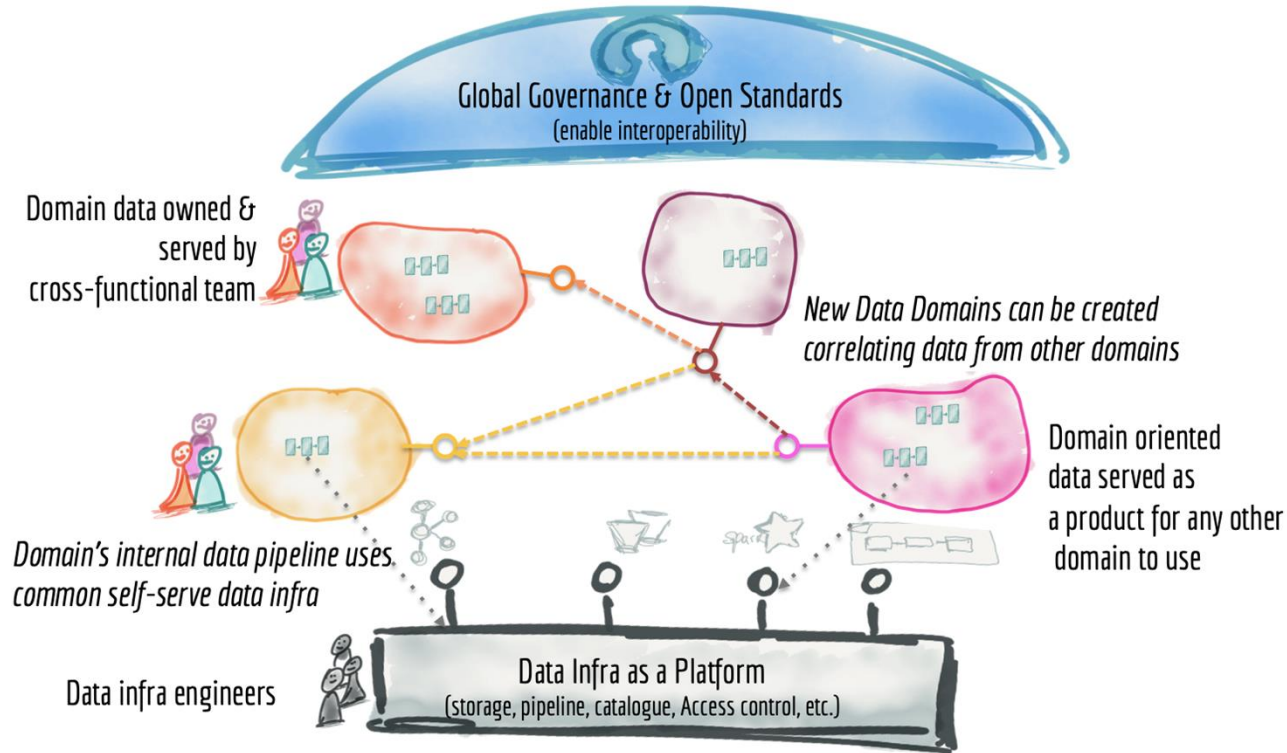
CONTENTS

- [The current enterprise data platform architecture](#)
- [Architectural failure modes](#)
 - [Centralized and monolithic](#)
 - [Coupled pipeline decomposition](#)
 - [Siloed and hyper-specialized ownership](#)

The flawed evolution of data platforms



Data Mesh



Data Mesh drivers

Delivery:

- I want to get the data
- I want to build my solutions
- I want to be responsible for end-to-end
- I want to decide on tools and way of working

Organization:

- Data is a common asset, teams have to share it with others
- Ensure we comply legal and privacy
- Use skills we have in a company

Business:

- I want to take business decisions based on data
- I can't wait months to validate my ideas

Architecture:

- Ensure we can expand, change our architecture in the future
- I want to focus on domains and products rather on systems
- I want to ensure architecture is not monolithic

Data Mesh – the three pillars

Domain Driven Design

Autonomous teams with clear bounded context building and running products independently

Product Thinking

Produce datasets, both raw and transformed. Ensure data lifecycle, versioning, SLA, documentation, quality

Platform Thinking

Create self-service platform for storage, computation, access rights, pipelines, etc.

“Data Mesh” by Microsoft

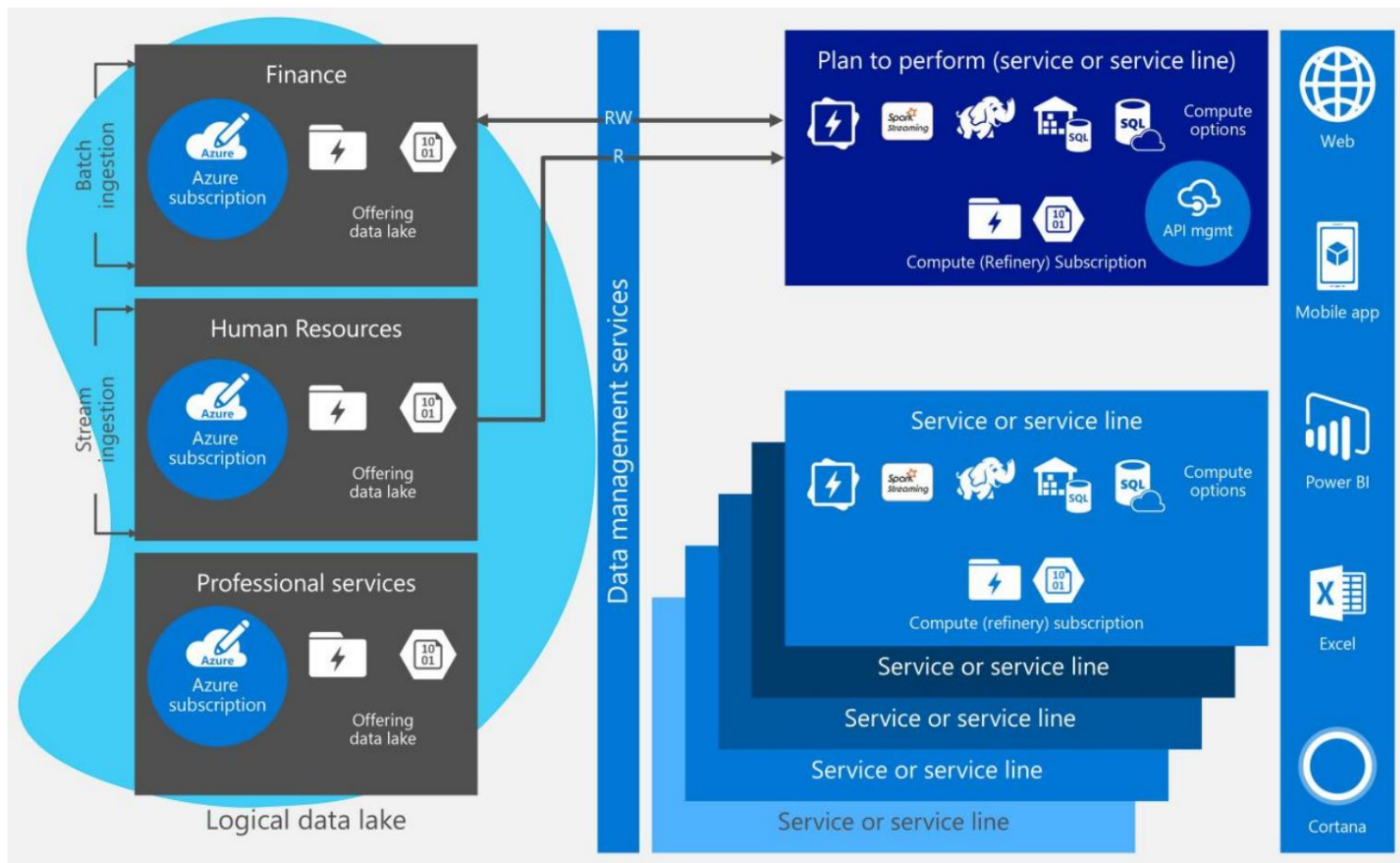
Microsoft tames the ‘wild west’
of big data with modern data
management

December 20, 2018

DOWNLOAD PDF >



"Data Mesh" by Microsoft



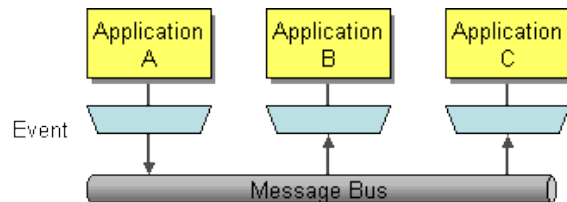
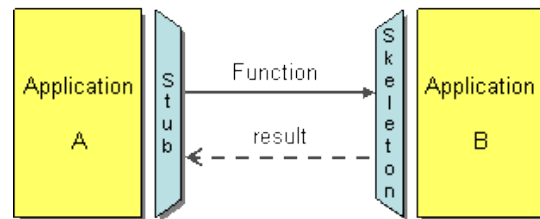
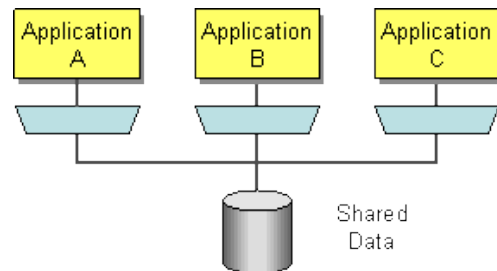
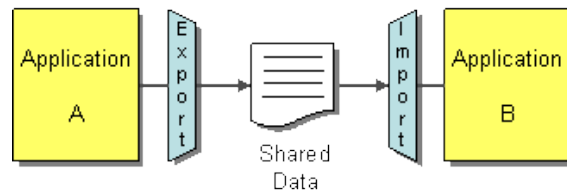
Data integration options

File Transfer — Have each application produce files of shared data for others to consume, and consume files that others have produced.

Shared Database — Have the applications store the data they wish to share in a common database.

Remote Procedure Invocation — Have each application expose some of its procedures so that they can be invoked remotely, and have applications invoke those to run behavior and exchange data.

Messaging — Have each application connect to a common messaging system, and exchange data and invoke behavior using messages.



Data acquisition options

Direct

- For example, access data by using connection string
- Creating dependencies

APIs

- Abstraction layer
- Generic approach

SDKs

- Abstraction layer
- Language specific
- For example, within Databricks notebooks as Python/Scala helper library

3rd-party

- Data virtualization, e.g. Denodo
- Schema-on-read

Fitness functions to measure a system's alignment with architectural goals (measure how good your solution is)

For example:

- How long does it take to deliver a feature, from conception to release?
- How long does it take to provide data (table)?
- How often are deployments being made and how many of them fail?
- How long does it take to on-board a new team?
- How many new support incidents are being recieved?
- How much unscheduled down-time is going on?

Create fitness functions for all important dimensions, e.g. data design, security, scalability, ...

