

Julian Kunkel

Data Models & Data Processing Strategies



Learning Objectives

- Define important terminology for data handling and data processing
- Sketch the ETL process used in data warehouses
- Sketch a typical HPDA data analysis workflow
- Sketch the lambda architecture
- Construct suitable data models for a given use-case and discuss their pro/cons
- Define relevant semantics for data

Outline

- 1 Data: Terminology
- 2 Processing
- 3 Data Models
- 4 Summary

Terminology

Data [1, 10]

- **Raw data:** collected information that is not derived from other data
- **Derived data/data product:** data produced with some computation/functions
- **View:** presents derived data to answer specific questions
 - ▶ Convenient for users (only see what you need) + faster than re-computation
 - ▶ Convenient for administration (e.g., manage permissions)
 - ▶ Data access can be optimized

Processing makes data valuable!

Dealing with unstructured data

- We need to extract information from raw unstructured data
 - ▶ e.g., perform text-processing using techniques from computer linguistics
- **Semantic normalization** is the process of reshaping free-form information into a structured form of data [11]
- Store raw data when your processing algorithm improves over time

Data Management in Projects Involves Various Considerations

1. Data Collection and Documentation



What kind of data are generated

How will data be generated

What metadata are needed

2. Ethics, legal and security Issues



How will ethical issues be handled

How are the data accessed

Are there copyright issues

Are there sensitive data

What about intellectual property rights

3. Data Storage and Preservation



How are the data stored?

Are there back up systems

How are data safely preserved

4. Data Sharing and reuse



How and where will the data be shared?

How are sensitive data protected

How can data be accessed

Figure: Data Management Plan. Source:

<https://www.uzh.ch/blog/hbz/2018/11/15/data-management-plan-in-a-nutshell/>

- **Data management plan (DMP):** describes the strategies and measures for handling research data during and after a project
- **Data life cycle:** creation, distribution, use, maintenance & disposition
 - ▶ For every data object, should define its life cycle
- **Information lifecycle management (ILM):** business term; practices, tools and policies to manage the data life cycle in a cost-effective way

Terminology for Managing Data [1, 10]

- **Data governance:** *“control that ensures that the data entry ... meets precise standards such as business rule, a data definition and data integrity constraints in the data model”* [10]
 - ▶ Think about reasons that invalidate data that lead to catastrophic results...
 - ▶ Example: missinterpretation of data value "NaN" as "0" in a survey
- **Data provenance:** the documentation of input, transformations of data and involved systems to support analysis, tracing and reproducibility
 - ▶ e.g., Input (file.csv) ⇒ Calculate means via x.py (result: means.csv) ⇒ Create diagrams via d.py (result fig1.pdf)
- **Data-lineage** (Datenherkunft): forensics; allows to identify the source data used to generate data products (part of data provenance)
 - ▶ e.g., fig1.pdf has been produced from ... using Z...
 - ▶ I'm able to reproduce results and track errors from the product
- **Service level agreements** (SLAs): contract defining quality, e.g., performance/reliability & responsibilities between service user/provider

Outline

- 1 Data: Terminology
- 2 Processing**
 - Process Model
 - Data-Cleaning and Ingestion
 - Overview
 - Domain-specific Language
- 3 Data Models
- 4 Summary

Process Model [13]

- Process Model: *A model describing processes*
- **Process** [15]:
 - ▶ “A series of events to produce a result, especially as contrasted to product.”
- Qualities of descriptions
 - ▶ Descriptive: Describe the events that occur during the process
 - ▶ Prescriptive
 - Define the intended process and how it is executed
 - Rules and guidelines steering the process
 - ▶ Explanatory
 - Provide rationales for the process
 - Describe requirements
 - Establish links between processes
- **Workflow**
 - ▶ *A defined process, i.e., all events/processing steps are clearly defined*

Data-Cleaning and Ingestion

- Importing of raw data into a big data system is an important process
 - ▶ Wrong data results in wrong conclusions: Garbage in – Garbage out
- **Data wrangling**: process & procedures to clean/convert data from one format to another [1]
 - ▶ **Data extraction**: identify relevant data sets and extract raw data
 - ▶ **Data munging**: cleaning raw data, converting it to a format for consumption
- **ETL process** (Extract, Transform, Load): data warehouse term for importing data (from databases) into a data warehouse

Necessary steps

- Define and document data governance policies to ensure data quality
 - ▶ Identifying and dealing with duplicates, time(stamp) synchronization
 - ▶ Handling of missing values (NULL or replace them with default values)
- Document the conducted transformations (for data provenance)
 - ▶ Data sources
 - ▶ Conversions of data types, complex transformations
 - ▶ Extraction of information from unstructured data (semantic normalization)
- Implementation of the procedures for bulk loading and cleaning of data

Datawarehousing ETL Process

- Extract: read data from source (transactional) databases
- Transform: alter data on the fly
 - ▶ Perform quality control
 - ▶ Treat errors and uncertainty to improve quality
 - ▶ Change the layout to fit the schema of the data warehouse
- Load: integrate the data into the data warehouse
 - ▶ Restructure data to fit needs of business users
 - ▶ Rely on batch integration of large quantities of data

Data Analysis Workflow

The traditional approach proceeds in phases:

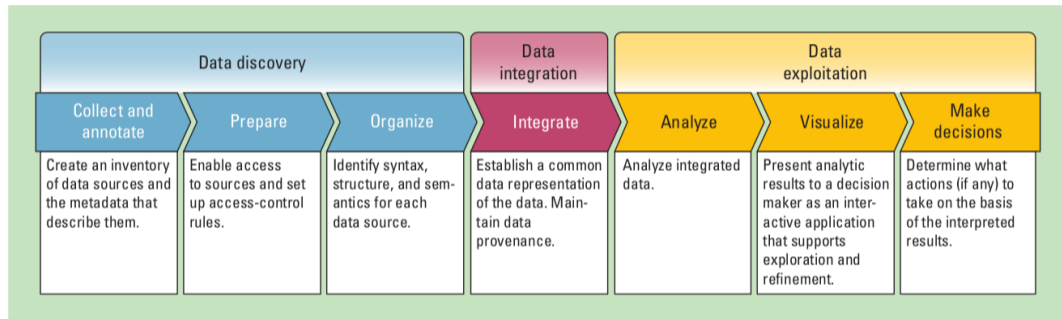


Figure: Source: Gilbert Miller, Peter Mork From Data to Decisions: A Value Chain for Big Data.

- Analysis tools: machine learning, statistics, interactive visualization
- Limitation: Interactivity by browsing through prepared results
- Indirect feedback between visualization and analysis

Programming (High-Performance) BigData Analytics

High-level concepts

- SQL and derivatives
- Domain-specific languages (Cypher, PigLatin)

Programming languages

- Java interfaces are widely available but low-level
- Scala language increases productivity over Java
- Python and R have connectors to popular BigData solutions

In the exercises, we are using primarily Python

Programming Paradigms[14]

Programming paradigms are process models for computation

- Fundamental style and abstraction level for computer programming
 - ▶ **Imperative** (e.g., Procedural)
 - ▶ **Declarative** (e.g., Functional, **Dataflow**, Logic)
 - ▶ **Data-driven** programming (describe patterns and transformations)
 - ▶ **Multi-paradigm** support several at the same time (e.g., **SQL**)
- Goals: productivity of the users and performance upon execution
 - ▶ Tool support for development, deployment and testing
 - ▶ Performance depends on single core efficiency but importantly parallelism
- **Parallelism** is an important aspect for processing of large data
 - ▶ In HPC, there are language extensions, libraries to specify parallelism
 - PGAS, Message Passing, OpenMP, data flow e.g., OmpSs, ...
 - ▶ In BigData Analytics, libraries and domain-specific languages
 - MapReduce, SQL, data-flow, streaming and data-driven

Domain-specific Language (DSL)

- DSLs are a contrast to general-purpose languages (GPL)
- Specialized programming language to an application domain
 - ▶ Mathematics, e.g., statistics, modeling
 - ▶ Description of graphs, e.g., graphviz (dot)
 - ▶ Processing of big data (Apache Pig)
- Standalone vs. embedded DSLs
 - ▶ Embedding into a GPL (e.g., regex, SQL) with library support
 - ▶ Standalone requires to provide its own toolchain (e.g., compiler)
 - ▶ Source-to-source compilation (DSL to GPL) an alternative
- Abstraction level
 - ▶ High-level: only covers (scientific) domain
 - ▶ Low-level: includes technical details (e.g., about hardware)

Tools for Data Exploration

Mandatory features

- Interactive
- Rich set: visualization, data manipulations, algorithms
- Real-time processing of big data

Tools (excerpt)

- Closed source: SAS, Spotfire, Domo, Tableau
- Open source: **R, Python/Jupyter**/Bokeh, GoogleVis
- Other open source tools, see [19]

Requirements

- Usability
- Flexible
- Performance

Productivity

Productivity is a very important metric for Big Data tools

Development environments

- 1 Text editor; workflow: edit, save, (compile), run on a server
 - ▶ Notepad, gedit
- 2 Interactive shell; type code and execute it
 - ▶ Python, SQL frontent
- 3 IDE; optimized workflow of the text editor, may run code on a server
 - ▶ NetBeans, Eclipse, VisualStudio
- 4 Interactive lab notebook; type code and store it together with results
 - ▶ Examples: Jupyter, Apache Zeppelin
 - ▶ Embedded in GitHub:
https://github.com/jakevdp/PythonDataScienceHandbook/blob/master/code_listings/03.11-Working-with-Time-Series.ipynb
- 5 Lab notebook + IDE;
 - ▶ Examples: Spyder

Alternative Processing Technology

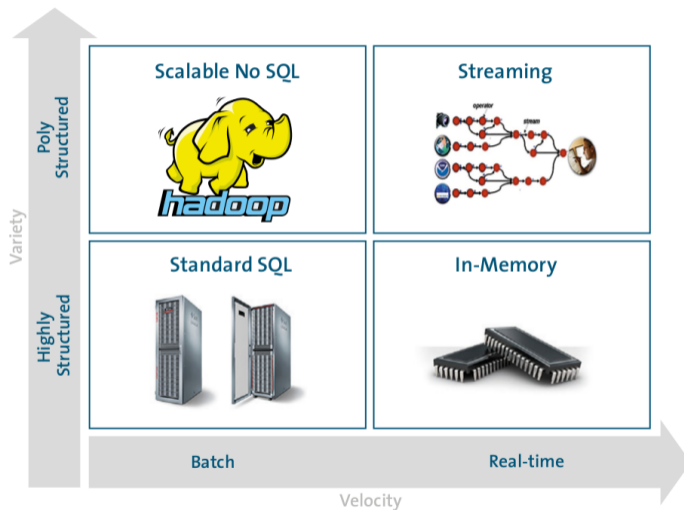
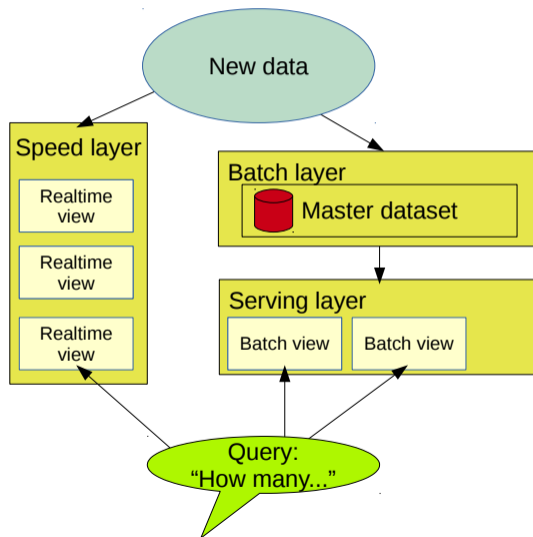


Figure: Source: Forrester Webinar. Big Data: Gold Rush Or Illusion? [4]

The Lambda Architecture [11]

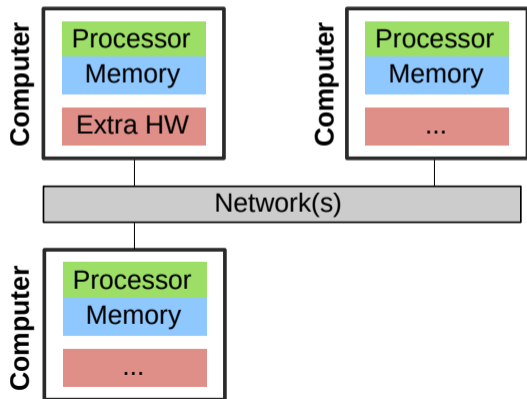


- **Goal:** Interactive Processing
- Batch layer pre-processes data
 - ▶ Master dataset is immutable/never changed
 - ▶ Operations are periodically performed
- Serving layer offers performance optimized views
- Speed layer serves deltas of batch and recent activities, may approximate results
- Robust: Errors/inaccuracies of realtime views are corrected in batch view

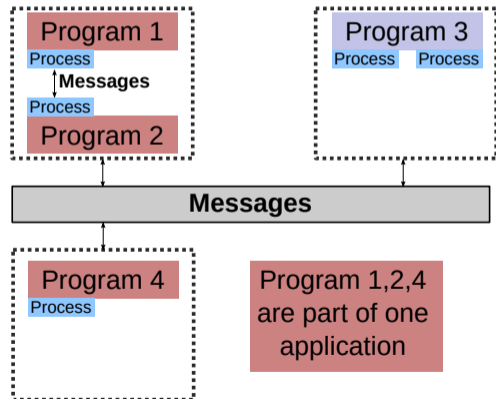
Reminder: Distributed System and Distributed Program

- **Must map a workflow to processes and hardware of a distributed system**
- A **distributed program** (DP) runs on a distributed system

Hardware perspective



Software perspective (mapped to hw)



Group Work

- Discuss the processing stages of a "use case" of your exercise
 - ▶ How is data processed?
 - ▶ How does the use case map to the ETL and data analysis workflow?
 - ▶ Is this an interactive process?
 - ▶ What programming paradigm is used?
 - ▶ What tools are used?
 - ▶ How were the tools mapped to hardware?
 - ▶ How did they manage data? Is there a data management plan, a defined life cycle?
- If you haven't done the exercise participate in the discussion
 - ▶ Google or come up with an example of your choice...
- Time: 10 min
- Organization: breakout groups - please use your mic or chat

Outline

- 1 Data: Terminology
- 2 Processing
- 3 Data Models**
 - Data Model
 - Overview of Data Models
 - Semantics
 - Columnar Model
 - Key-Value Store
 - Document Model
 - Data Lake
- 4 Summary

Data Models¹ and their Instances [12]

- A data model describes how information is organized in a system
 - ▶ It is a tool to specify, access and process information
 - ▶ A model provide operations for accessing and manipulating data that follow certain semantics
 - ▶ Typical information is some kind of entity (virtual object) (e.g., car)

■ **Logical model:** abstraction expressing objects and operations

■ **Physical model:** maps logical structures onto hardware resources (e.g., files, bytes)

■ DM theory: Formal methods for describing data models with tool support

■ Applying theory creates a **data model instance** for a specific application

¹: The term is often used ambivalently for a data (meta) model concept/theory or an instance

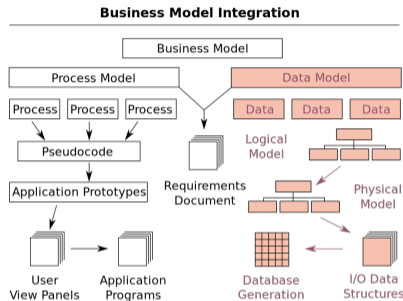


Figure: Source: [12]

Operations

- Operations define how you can interact with the data
 - ▶ Minimal: Need to somehow store and retrieve data
 - ▶ Users may want to search for data, update existing data
- May want to offload some operations to the server side: active storage
 - ▶ Reduce data, e.g., compute mean/sum
 - ▶ Conditional updates

Typical Operations

- POSIX: create, open, write (anywhere), read (anywhere)
 - ▶ Does not distinguish between write and update
- CRUD: Create, Read, Update, Delete
- Amazon S3: Put (Overwrite), Get (Partially), Delete

Selection of Theory (concepts) for Data Models

- I/O Middleware: NetCDF, HDF5, ADIOS
 - ▶ Self-describing formats containing N-dim variables with metadata
- Relational model (tuples and tables)
 - ▶ Can be physically stored in, e.g., a CSV file or database
- Relational model + raster data
 - ▶ Operations for N-dimensional data (e.g., pictures, scientific data)
- NoSQL data models: Not only SQL², lacks features of databases
 - ▶ Example DB models: columnar, document, key-value, named graph
- Fact-based: built on top of atomic facts, well-suited for BI [11]

Data modeling [10]

The process in software-engineering of creating a data model instance for an information system

Semantics

Semantics describe I/O operations and their behavior

- Application programming interface (API)
- **Concurrency:** Behavior of simultaneously executed operations
 - ▶ Atomicity: Are partial modifications visible to other clients
 - ▶ Visibility: When are changes visible to other clients
 - ▶ Isolation: Are operations influencing other ongoing operations
- **Availability:** Readiness to serve operations
 - ▶ Robustness of the system for typical (hardware and software) errors
 - ▶ Scalability: availability and performance behavior depending on the number of clients, concurrent requests, request size, ...
 - ▶ Partition tolerance: Continue to operate even if network breaks partially
- **Durability:** Modifications should be stored on persistent storage
 - ▶ Consistency: Any operation leaves a consistent (correct) system state

Consensus [17]

- **Consensus:** several processes agree (decide) for a single data value
 - ▶ Processes may propose a value (any time)
- Consensus and consistency of distributed processes are related
- Consensus protocols such as Paxos ensure cluster-wide consistency
 - ▶ They tolerate typical errors in distributed systems
 - ▶ Hardware faults and concurrency/race conditions
 - ▶ Byzantine protocols additionally deal with forged (lying) information
- Properties of consensus
 - ▶ Agreement: Every correct process must agree on the same value
 - ▶ Integrity: All correct process decide upon at most one value v . If one decides v , then v has been proposed by some process
 - ▶ Validity: If all process propose the same value v , then all correct processes decide v
 - ▶ Termination: Every correct process decides upon a value

Example Semantics

POSIX I/O

- Atomicity and isolation for individual operations, locking possible

ACID

- Strict semantics for database systems to prevent data loss
- Atomicity, consistency, isolation and durability for **transactions**

BASE

- BASE is a typical semantics for Big Data due to the CAP theorem
- Basically Available replicated Soft state with Eventual consistency [26]
 - ▶ Availability: Always serve but may return a failure, retry may be needed
 - ▶ Soft state: State of the system may change over time without requests due to eventual consistency
 - ▶ Consistency: If no updates are made any more, the last state becomes visible to all clients after some time (eventually)
- Big data solutions usually exploit the immutability of data

Relational Model [10]

- Database model based on first-order predicate logic
 - ▶ Theoretic foundations: relational algebra and relational calculus
- Data is represented as tuples
 - ▶ In its original style, it does not support collections
- Relation/Table: groups tuples with similar semantics
 - ▶ Table consists of rows and named columns (attributes)
 - ▶ No (identical) duplicate of a row allowed
- Schema: specify structure of tables
 - ▶ Datatypes (domain of attributes)
 - ▶ Consistency via constraints
 - ▶ Organization and optimizations

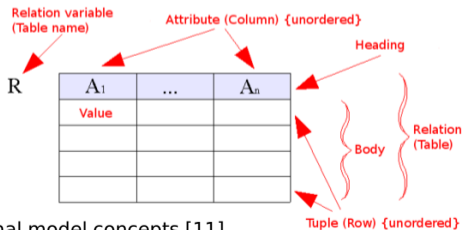


Figure: Source: Relational model concepts [11]

Example Relational Model for Students Data

Matrikel	Name	Birthday
242	Hans	22.04.1955
245	Fritz	24.05.1995

Table: Student table

ID	Name
1	Big Data Analytics
2	Hochleistungsrechnen

Table: Lecture table

Matrikel	LectureID
242	1
242	2
245	2

Table: Attends table representing a relation

Columnar Model

- Data is stored in rows and columns (similar to tables)
- A column is a tuple (name, value and timestamp)
- Each row can contain different columns
 - ▶ Columns can store complex objects, e.g., collections
- Wide columnar model: very sparse table of 100k+ columns
- Example technology: HBase, Cassandra, Accumulo

Row/Columnn:	student name	matrikel	lectures	lecture name
1	"Max Mustermann"	4711	[3]	-
2	"Nina Musterfrau"	4712	[3,4]	-
3	-	-	-	"Big Data Analytics"
4	-	-	-	"Hochleistungsrechnen"

Table: Example columnar model for the students, each value has its own timestamp (not shown). Note that lectures and students should be modeled with two tables

Key-Value Store

- Data is stored as value and addressed by a key
- The value can be complex objects, e.g., JSON or collections
- Keys can be forged to simplify lookup (evtl. tables with names)
- Example technology: CouchDB, BerkeleyDB, Memcached, BigTable

Key	Value
stud/4711	<name>Max Mustermann</name><attended><id>1</id></attended>
stud/4712	<name>Nina Musterfrau</name><attended><id>1</id><id>2</id></attended>
lec/1	<name>Big Data Analytics</name>
lec/2	<name>Hochleistungsrechnen</name>

Table: Example key-value model for the students with embedded XML

Document Model

- Collection of documents
- Documents contain semi-structured data (JSON, XML)
- Addressing to lookup documents are implementation specific
 - ▶ e.g., bucket/document key, (sub) collections, hierarchical namespace
- References between documents are possible
- Example technology: MongoDB, Couchbase, DocumentDB

```
1 <students>
2   <student><name>Max Mustermann</name><matrikel>4711</matrikel>
3     <lecturesAttended><id>1</id></lecturesAttended>
4   </student>
5   <student><name>Nina Musterfrau</name><matrikel>4712</matrikel>
6     <lecturesAttended><id>1</id><id>2</id></lecturesAttended>
7   </student>
8 </students>
```

Table: Example XML document storing students. Using a bucket/key namespace, the document could be addressed with key: “uni/stud” in the bucket “app1”

Graph

- Entities are stored as nodes and relations as edges in the graph
- Properties/Attributes provide additional information as key/value
- Example technology: Neo4J, InfiniteGraph

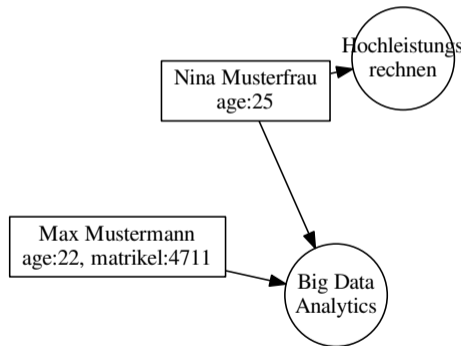


Figure: Graph representing the students (attributes are not shown)

Fact-Based Model [11]⁴

- Store raw data as timestamped atomic facts aka log files of change/current status
- Never delete true facts: Immutable data
- Make individual facts unique to prevent duplicates

Example: social web page

- Record all changes to user profiles as facts
- Benefits
 - ▶ Allows reconstruction of the profile state at any time
 - ▶ Can be queried at any time³

Example: purchases

- Record each item purchase as facts together with location, time, ...

From Big Data to the Data Lake

- With cheap storage costs, people promote the concept of the data lake
- Combines data from many sources (data silos) and of any type and model
- Allows for conducting future analysis and not miss any opportunity

Attributes of the data lake

- Collect everything: all time all data: raw sources and processed data
 - ▶ Decide during analysis which data is important, e.g., no “schema” until read
- Dive in anywhere: enable users across multiple business units to
 - ▶ Refine, explore and enrich data on their terms
- Flexible access: shared infrastructure supports various patterns
 - ▶ Batch, interactive, online, search

Group Work

- Discuss the use case to store information about students with each data model
- Which data model would you prefer, why?
- Time: 10 min
- Organization: breakout groups - please use your mic or chat

Summary

- Data-cleaning and ingestion is a key to successful modeling
- Big data can be considered to be immutable (remember: data lake)
- Data models describe how information is organized
 - ▶ Various I/O middleware, relational model
 - ▶ NoSQL: Column, document, key-value, graphs
- Semantics describe operations and behavior, e.g., POSIX, ACID, BASE
- Process models and programming paradigms describe how to
 - ▶ transform and analyze data
- Apache ecosystem offers means for batch and real-time processing
- Lambda architecture is a concept for enabling real-time processing

Bibliography

- 4 Forrester Big Data Webinar. Holger Kisker, Martha Bennet. Big Data: Gold Rush Or Illusion?
- 10 Wikipedia
- 11 Book: N. Marz, J. Warren. Big Data – Principles and best practices of scalable real-time data systems.
- 12 https://en.wikipedia.org/wiki/Data_model
- 13 https://en.wikipedia.org/wiki/Process_modeling
- 14 https://en.wikipedia.org/wiki/Programming_paradigm
- 15 <https://en.wiktionary.org/wiki/process>
- 16 [https://en.wikipedia.org/wiki/Paxos_\(computer_science\)](https://en.wikipedia.org/wiki/Paxos_(computer_science))
- 17 [https://en.wikipedia.org/wiki/Consensus_\(computer_science\)](https://en.wikipedia.org/wiki/Consensus_(computer_science))
- 20 <http://hortonworks.com/blog/enterprise-hadoop-journey-data-lake/>
- 26 Overcoming CAP with Consistent Soft-State Replication <https://www.cs.cornell.edu/Projects/mrc/IEEE-CAP.16.pdf>