Introduction

School of Information Studies
Syracuse University

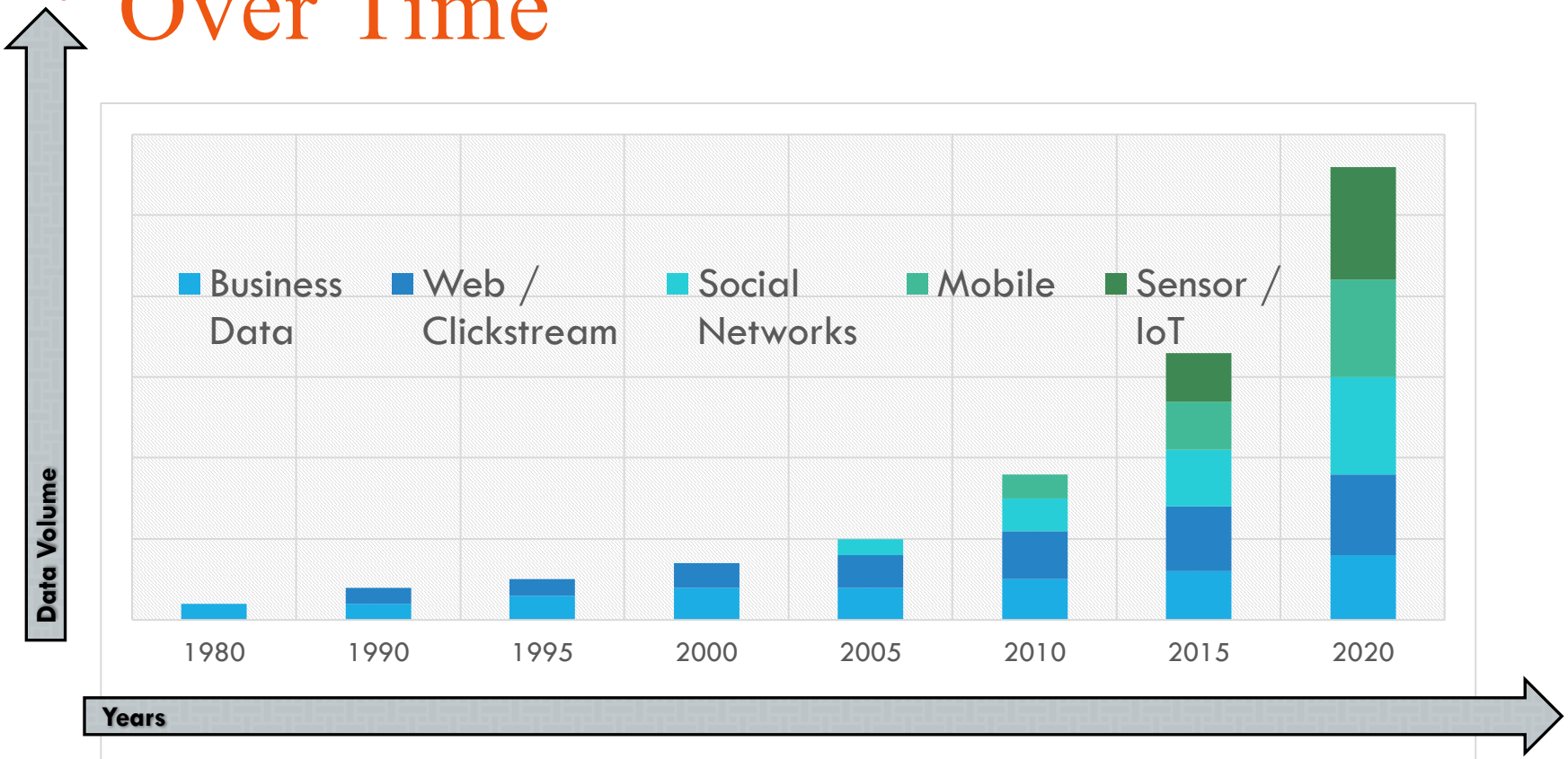
Agenda

- Understanding growth of data in the enterprise
- Defining scalability
- Define Big Data
- Explain how Hadoop, MapReduce, and HDFS work
- Discuss YARN applications used in data warehousing

Growth of Organizational Data

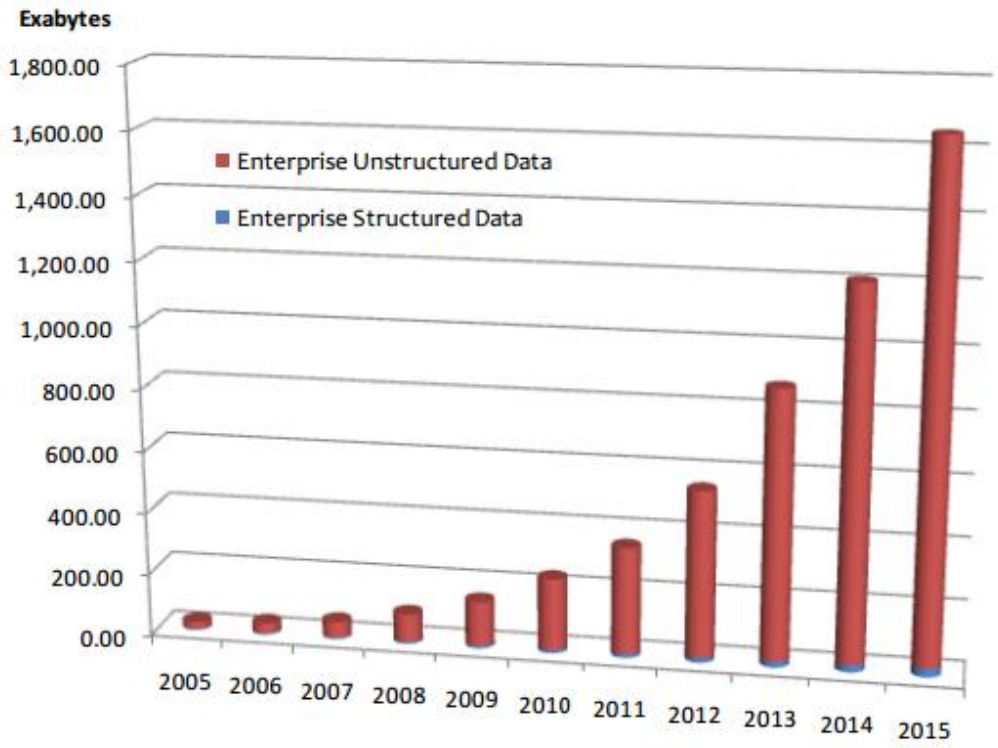
School of Information Studies
Syracuse University

Data Explosion: A Sample Growth of Organizational Data Over Time



The Rise of Enterprise Unstructured Data

Most of the data required for informed decision-making are unstructured data.

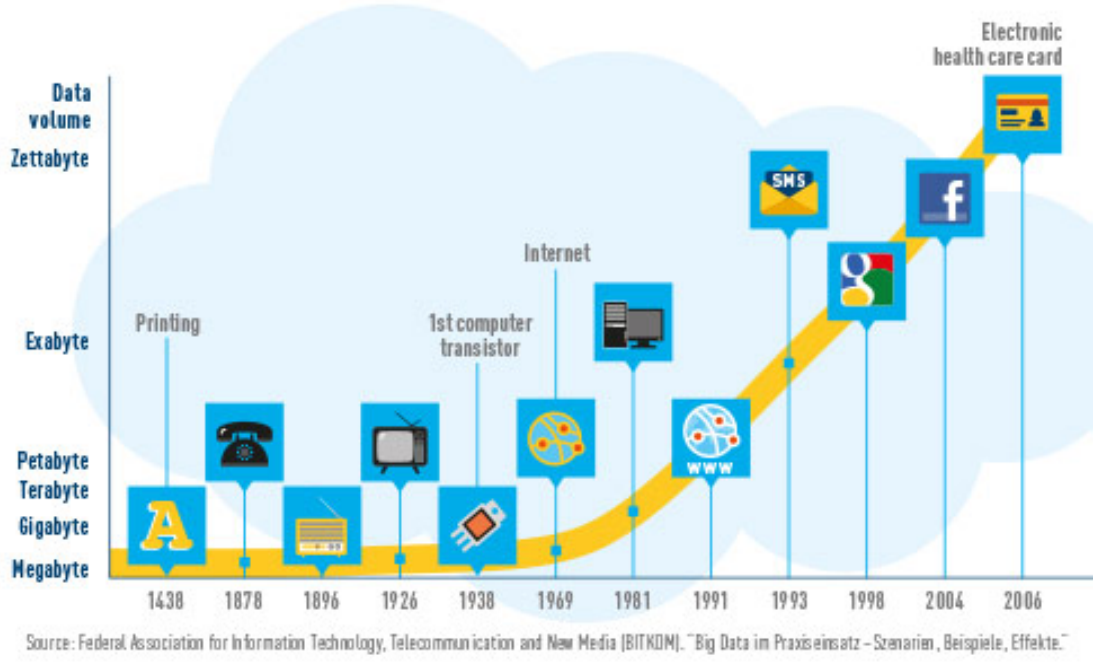


* IDG

Exponential Growth

Exponential growth of data volumes

Technologies such as RFID and smartphones as well as the increasing use of social media applications are resulting in a rapid rise in data volumes.



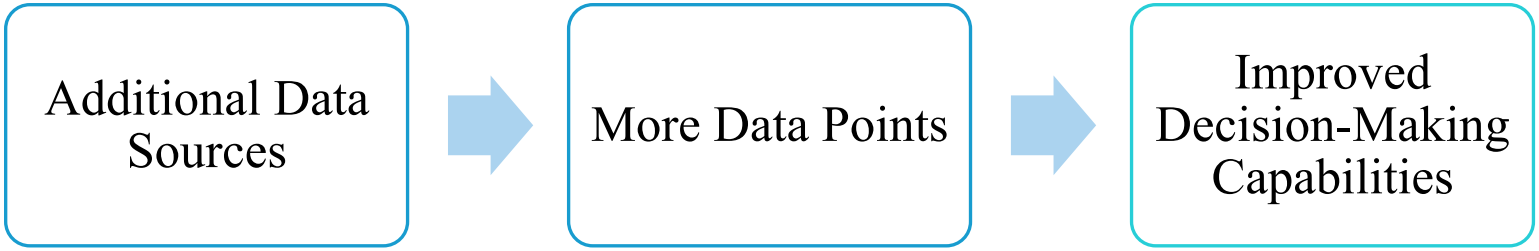
Source: Federal Association for Information Technology, Telecommunication and New Media (BITKOM). "Big Data im Praxiseinsatz – Szenarien, Beispiele, Effekte."


So what?

Data Drives Our Decision-Making

School of Information Studies
Syracuse University

Data Drives Our Decision-Making



More Data ➔ Better Insights

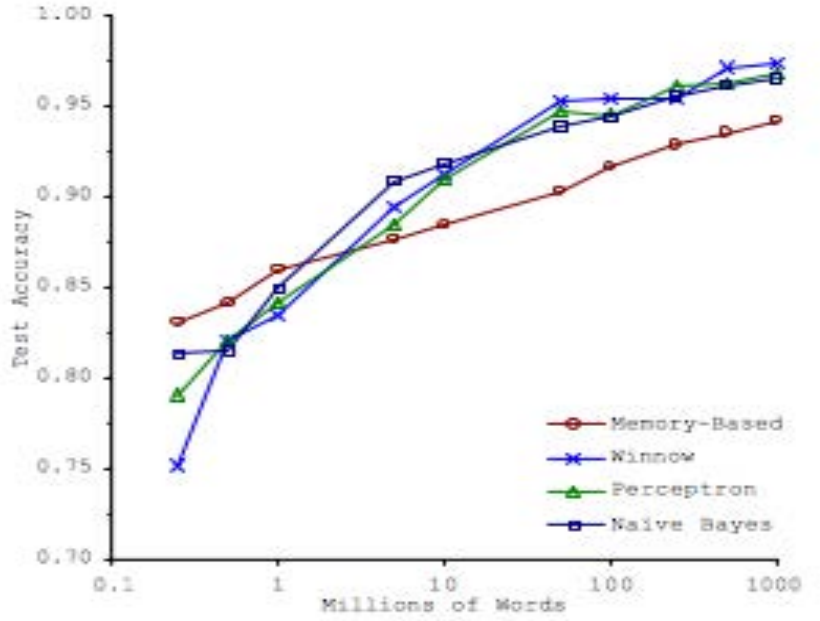
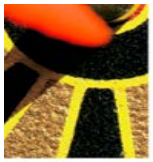


Figure 1. Learning Curves for Confusion Set Disambiguation

Banko & Brill, 2001:
"Algorithms Predict Better With Larger Data Sets"



The Unreasonable Effectiveness of Data

Alon Halevy, Peter Norvig, and Fernando Pereira, *Google*

Halevy, Norvig, & Pereira, 2009:
"Data Set Size Matters More Than Algorithms"

But Added Complexity Makes Decision-Making Difficult

Types of Scalability

School of Information Studies
Syracuse University

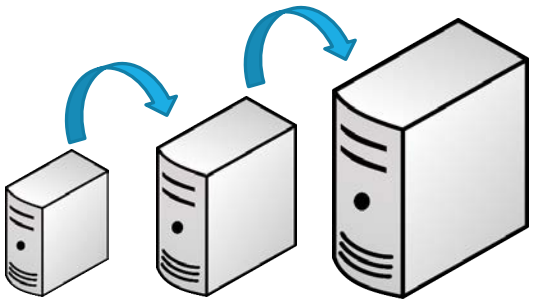
Scaling Services: How Do You Address Growth?

Vertical "Scale-Up"

Add more resources to an existing system running the service.

Easier, but limited scale.

Single point of failure.

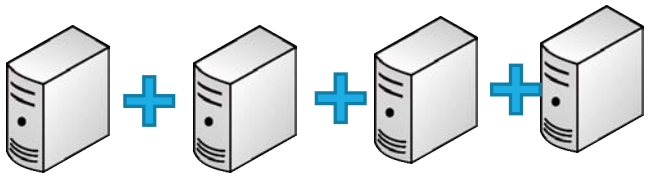


Horizontal "Scale-Out"

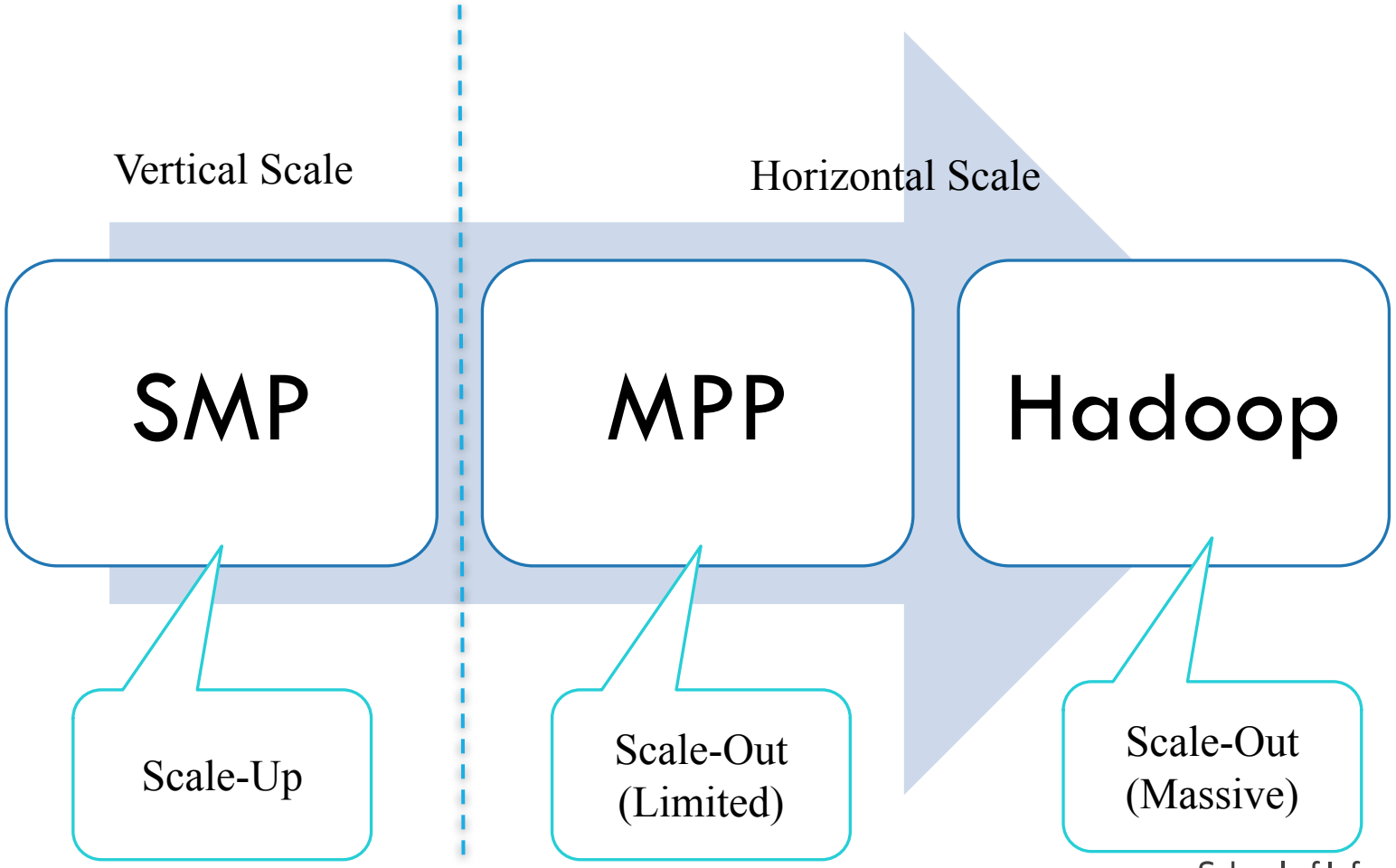
Run the service over multiple systems, and orchestrate communication between them.

Harder, but massive scale.

Overhead to manage nodes.



RECALL: DW System Architectures



CAP Theorem of Distributed Systems

School of Information Studies
Syracuse University

Distributed Systems

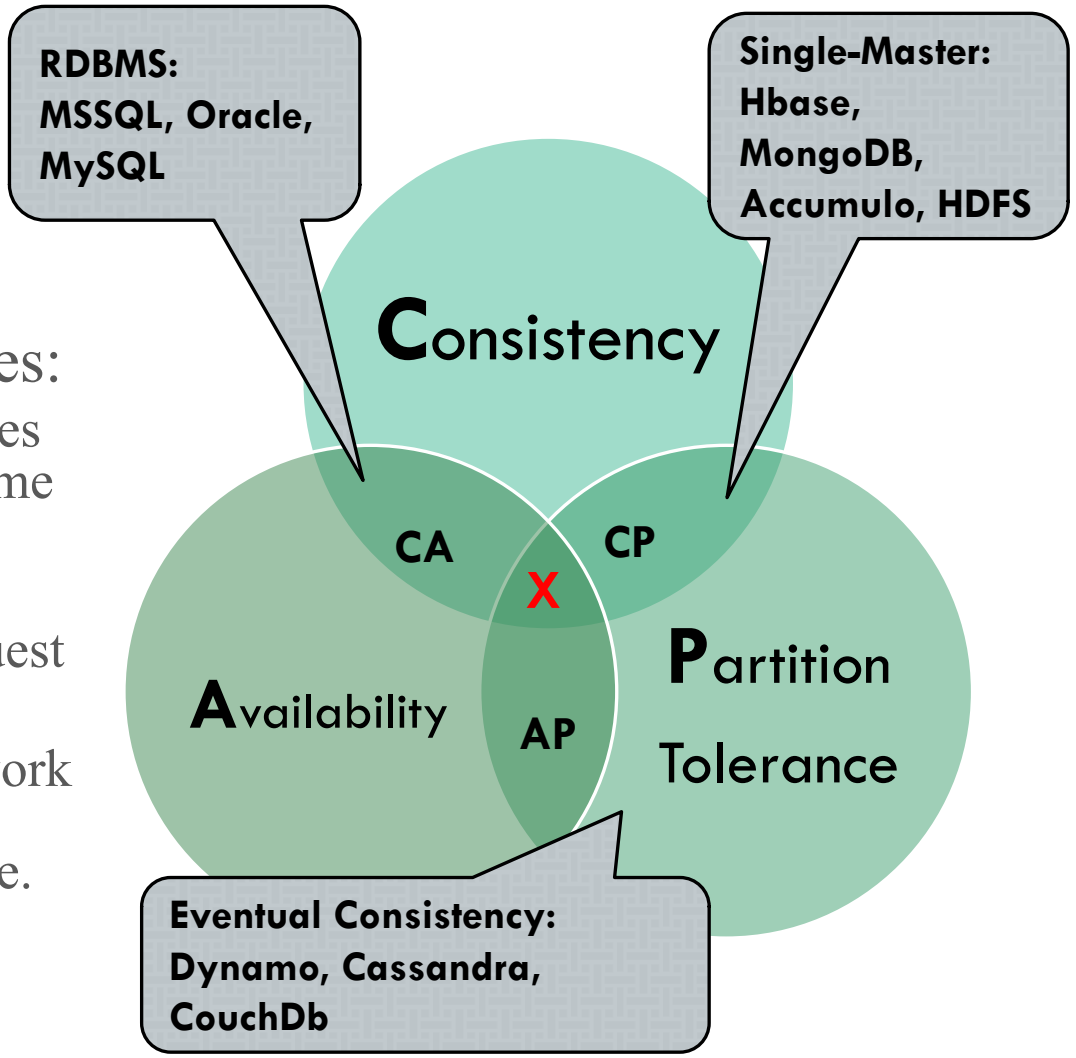
When the data volume is too large for a single system, and you can no longer scale up…

… you scale out.

CAP Theorem of Distributed Systems

At the same time, you can have only two of the following three guarantees:

- **Data consistency**: All nodes see the same data at the same time.
- **Data availability**: Assurances that every request can be processed.
- **Partition tolerance**: Network failures are tolerated; the system continues to operate.



Why Can't You Have All Three? *

A counterexample:

Suppose we lose communication between nodes:

- We must ignore any updates the nodes receive, or sacrifice **consistency**, or we must deny service until it becomes ***available*** again.

If we guarantee ***availability*** of requests, despite the failure:

- We gain ***partition tolerance*** (the system still works) but lose ***consistency*** (nodes will get out of sync).

If we guarantee **consistency** of data, despite the failure:

- We gain ***partition tolerance*** (again, system works) but lose ***availability*** (data on nodes cannot be changed failure is resolved).

* You can have all three, just not at the same time.

CAP: Database Systems for Every Need

RDBMSs like Oracle, MySQL and SQL Server:

- Focus on consistency and availability (ACID principles), sacrificing partition tolerance (and thus they don't scale well horizontally).
- Use cases: business data, when you don't need to scale out.

Single-master systems like MongoDb, Hbase, Redis, and HDFS:

- Provide consistency at scale but data availability runs through a single node.
- Use cases: read-heavy DW, caching, document storage, product catalogs.

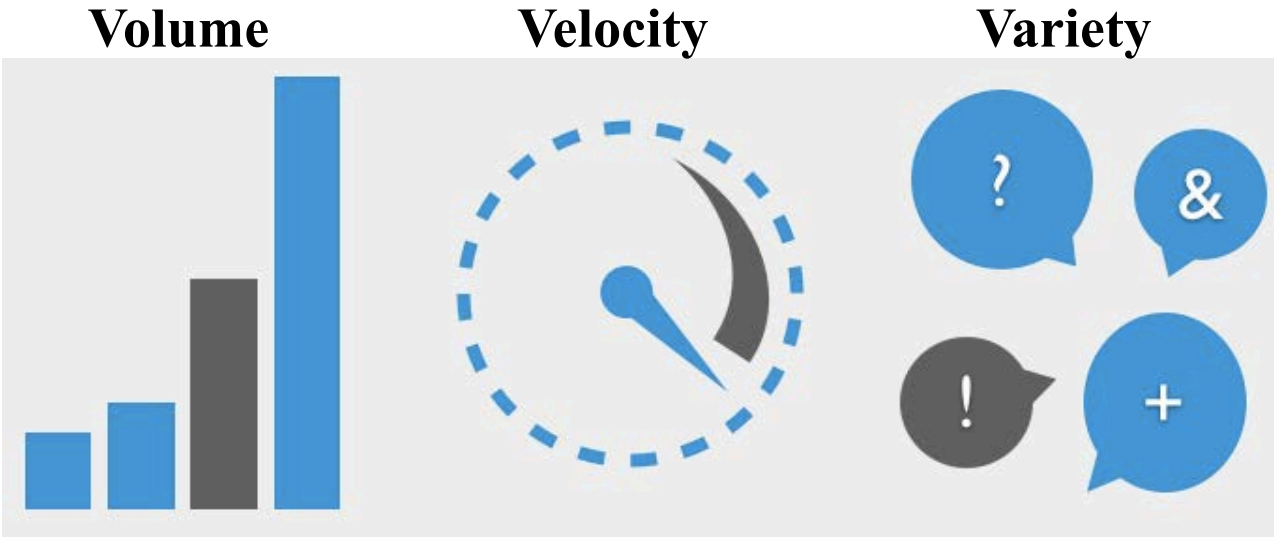
Eventual consistency systems like CouchDb, Cassandra, and Dynamo

- Provide availability at scale but do not guarantee consistency.
- Use cases: write-heavy, isolated activities, e.g., shopping carts, orders, tweets.

The Vs of Big Data

School of Information Studies
Syracuse University

The Three Vs of Big Data



"Too Much" "Too Fast" "Too Complicated"

Three Vs

Volume: Sheer size of data is too large to be processed by a single system.

Velocity: The rate at which new data arrives is too frequent to be processed by a single system.

Variety: Data are unstructured or semistructured, so compute resources must add structure before it can be used.

- E.g., a company's website generates more logging data in 24 hours than its ETL system can process in a 24-hour period.

Implications of the Three Vs

1. Require a scale-out and distribute data over several systems to meet scalability demands.
2. Must have all three Vs. Having one or two of the three can still be met through other methods.
3. Do not scale out because you can; do it because you must.

Other Vs of Big Data

Veracity: Uncertainty of your data. How can we be confident in the trustworthiness of our data sources?

- E.g., matching a tweet to a customer, without knowing his or her Twitter handle.

Viability: Can we predict results from the data? Can we determine which features serve as predictors?

- E.g., discovering patterns among customer purchase habits and unfavorable weather conditions.

Value: What meaning can we derive from our data? Can we use them to make good business decisions?

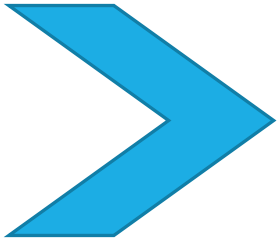
- E.g., increase inventory levels of potato chips two weeks before the Super Bowl.

Birthplace of Big Data

Three companies: Google, Facebook, Yahoo!

These companies had so much data that **enterprise DBMSs** could not meet their reporting requirements.

Time to process one day of data

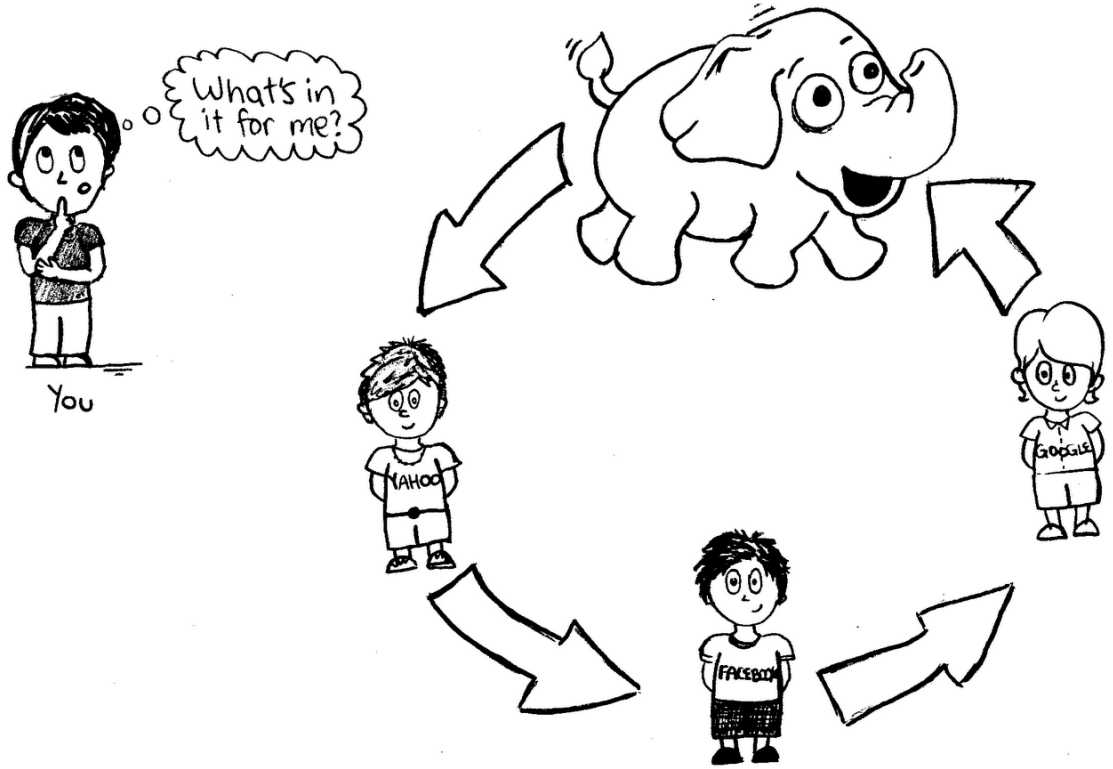


Number of hours in a day

But, I'm Not Google. Do I Need This?

Struggling with data volume, velocity, or variety

Insufficient resources to store process, and analyze your data



Big Data Is Data Warehouse Data

School of Information Studies
Syracuse University

Big Data Is Data Warehouse Data

1. Subject-oriented → web logs, messages, transactions
2. Time-variant → data reflect the point in time
3. Integrated → data come from a variety of sources
4. Nonvolatile → data written does not change

The difference is Big Data are too large, fast, and unstructured to be processed and stored in traditional means.

Examples of Big Data Applications

Clickstream: Analyze website traffic to determine how to invest in site improvements.

Sensor data: Collect data from environmental sensors to identify foot traffic patterns in a retail store.

Geographic data: Analyze online orders to establish consistency between where products are shipped versus ordered.

Server logs: Identify potential intrusions and misconfigured firewalls.

Sentiment: Get a sense of brand through social media.

Unstructured: Detect potential inside trading though e-mail and phone conversations.

What Is Hadoop?

School of Information Studies
Syracuse University

What Is Hadoop?

A system for:

- Storing,
- Processing, and
- Managing

data.

Sound familiar?

Yes. Change "data" to "Big Data."

"It does look similar—but this one is powered by Hadoop"

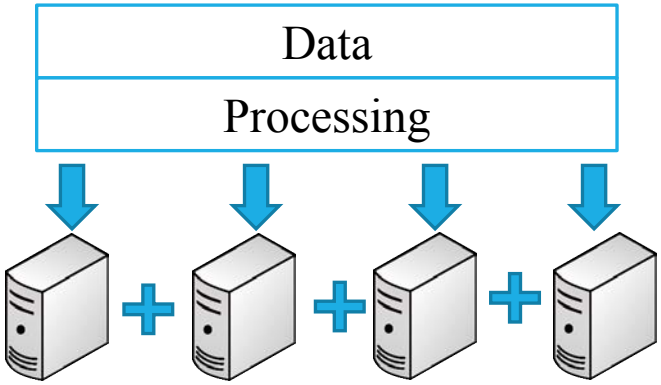
Hadoop Handles Big Data by *Scaling Out*

Problem: File too large to fit on a single storage platform?

Solution: Distribute the file over several computers.

Problem: Server not "fast" enough to perform ETL over your data?

Solution: Distribute data processing over several computers.



Hadoop Is Designed to Use *Commodity Hardware*

Hadoop hardware:

- Modular.
- Easy to add and remove nodes.
- Failure is not only acceptable but *expected*.

This is contrary to enterprise hardware.

- High-redundancy/fault-tolerant
- Vertical scaling
- Storage arrays

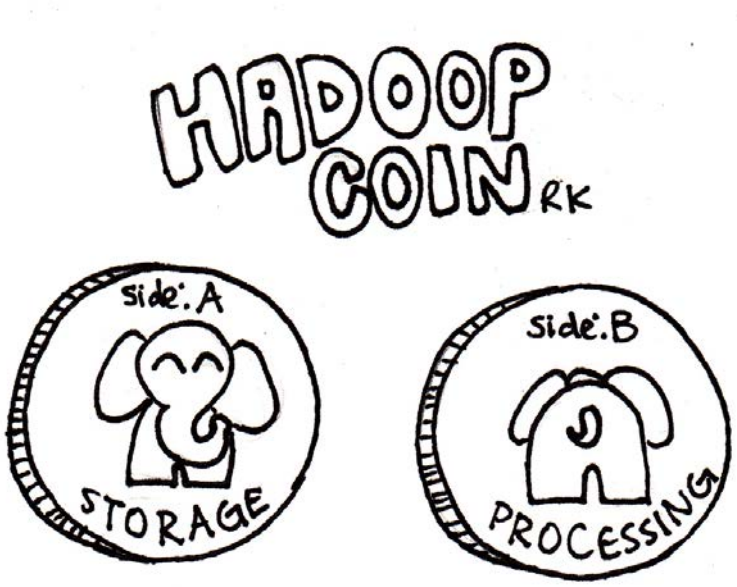
* Google Hardware spec for server source: C|NET

How Hadoop Works

School of Information Studies
Syracuse University

Two Goals of Hadoop

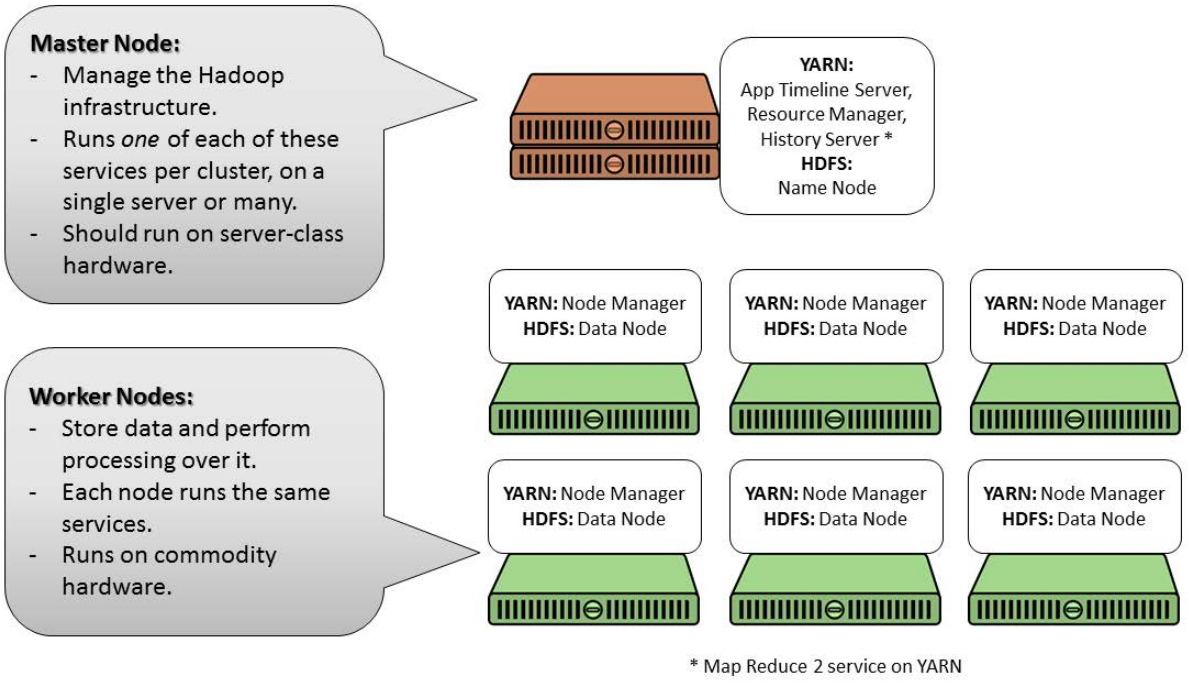
1. Distribute the data.
   HDFS Does this
2. Move processing to the data.
   MapReduce /
   YARN Does this



Hadoop Clusters

3 Node Types:

1. Master Nodes
2. Worker Nodes
3. Client Nodes



How Does Hadoop Differ from Relational?

Relational		Hadoop
Schema On Write	⟷	Schema On Read
Fast Reads	⟷	Fast Writes
Highly Structured Data	⟷	Loosely Structured Data
Declarative Data Processing (SQL)	⟷	Declarative & Procedural Data Processing
Good For: ACID Transactions, Business Data	⟷	Good For: Logs, Data Streams, Unstructured, Data Discovery

What ***Exactly*** is "Schema on Read?" Again?

Traditional RDBMS

You cannot write data without a table.

Cannot insert data unless data fits into table's single design.

Large up front design costs.

- Conceptual Models
- Table Design

"Schema on Write"

Hadoop's HDFS

You write the data "as-is", to HDFS.

Schema applied when data is read – multiple designs.

Very little up-front design costs

- Just Write to disk
- Apply schema when you need it

"Schema on Read"

Example: How Hadoop Works

School of Information Studies
Syracuse University

Collect and Store Data As Is, Schemaless.

Request

```
GET https://www.googleapis.com/youtube/v3/commentThreads?part=snippet&maxResults=25&videoId=O4PXqpv8TAw&key={YOUR_API_KEY}
```

Response

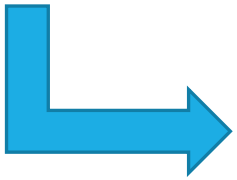
```
200 OK

- Show headers -

-{
  "kind": "youtube#commentThreadListResponse",
  "etag": "\"q5k97EMVGxODeKcDgp8gnMu79wM/lqCaY8zGX8gKVG6NRyNEbR3oSE8\"",
 -"pageInfo": {
   "totalResults": 16,
   "resultsPerPage": 25
  },
 -"items": [
  -{
    "kind": "youtube#commentThread",
    "etag": "\"q5k97EMVGxODeKcDgp8gnMu79wM/3uP8cTARSOm44LzKPV_LIHj5v8Q\"",
    "id": "z13si3gb1r3wdl0wi04ccjsoww2wetoovoc0k",
   -"snippet": {
     "videoId": "O4PXqpv8TAw",
    -"topLevelComment": {
      "kind": "youtube#comment",
      "etag": "\"q5k97EMVGxODeKcDgp8gnMu79wM/MZKledJ37NuMaz2UxwOIvaHqZgk\"",
      "id": "z13si3gb1r3wdl0wi04ccjsoww2wetoovoc0k",
     -"snippet": {
```

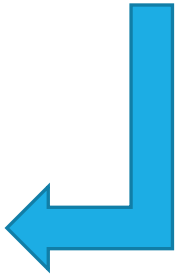
Apply a Schema on Read


```
},
"videoId": "O4PXqpv8TAw",
"textDisplay": "excellent video!\ufeff",
"authorGoogleplusProfileUrl": "https://plus.google.com/100257256306278377832",
"canRate": true,
"viewerRating": "none",
"likeCount": 0,
"publishedAt": "2015-02-02T02:05:16.042Z",
"updatedAt": "2015-02-02T02:05:16.042Z"
```

```
create external table youtube.comments (
    videoId STRING,
    likes INT,
    text STRING,
    pub_date STRING
) location '/user/mafudge/youtubecomments/';
```

videoId	likes	text	pub_date
O4PXqpv8TAw	0	excellent video!\efeff	2015-02-02T02:05:16.042Z
…	…	…	…



Analyze Your Data

videoId	likes	text	pub_date
O4PXqpv8TAw	0	excellent video!\ufeff	2015-02-02T02:05:16.042Z
…	…	…	…

```
select videoId, sum(likes)
    from youtube.comments
    group by videoId
    order by sum(likes) desc
    limit 10;
```

videoId	sum(likes)
O4PXqpv8TAw	56
V56Xqyy-2wq	14
…	…

The Hadoop Ecosystem

School of Information Studies
Syracuse University

The Hadoop Ecosystem: Open Source Tools

Apache Ambari
http://incubator.apache.org/ambari

APACHE FALCON

OOZIE

Spark

HIVE

Hadoop

sqoop

Apache Solr

APACHE HBASE

HDFS

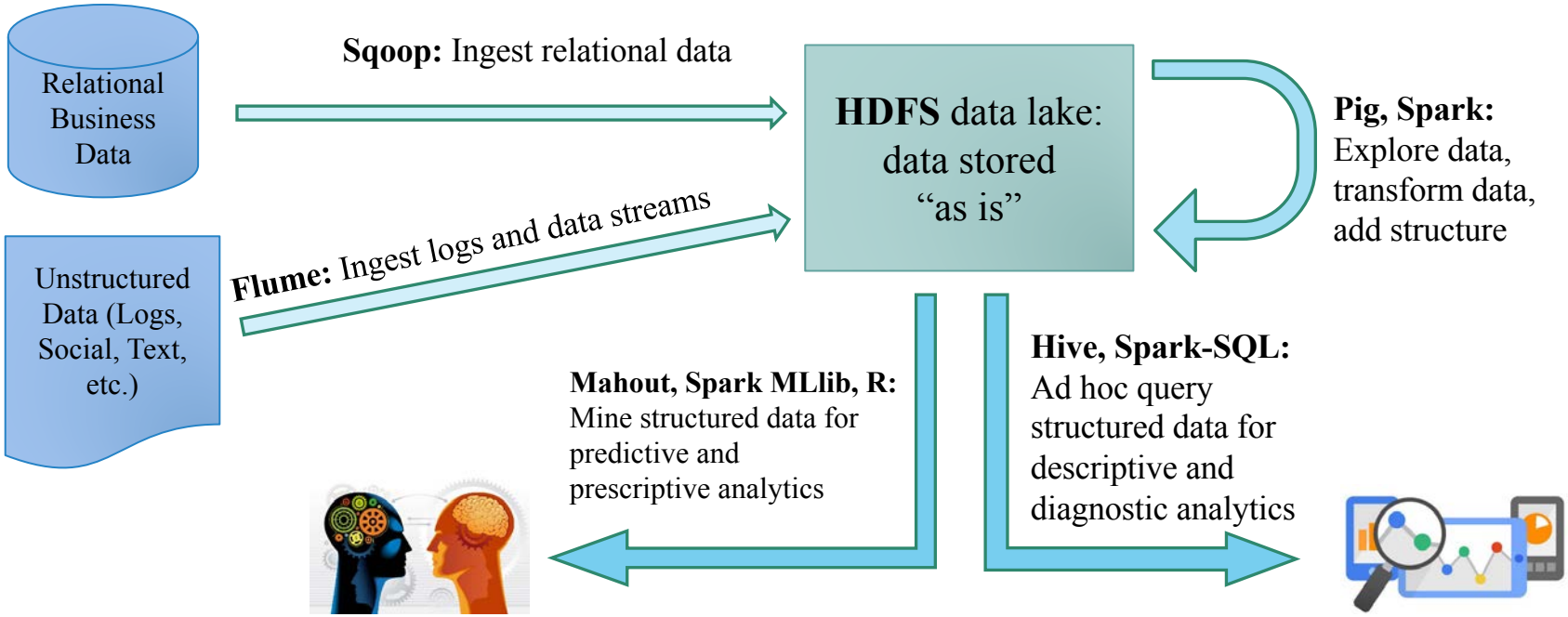
FLUME

STORM

accumulo

mahout

An Oversimplified Version of the Hadoop Ecosystem in Action



How HDFS Works

School of Information Studies
Syracuse University

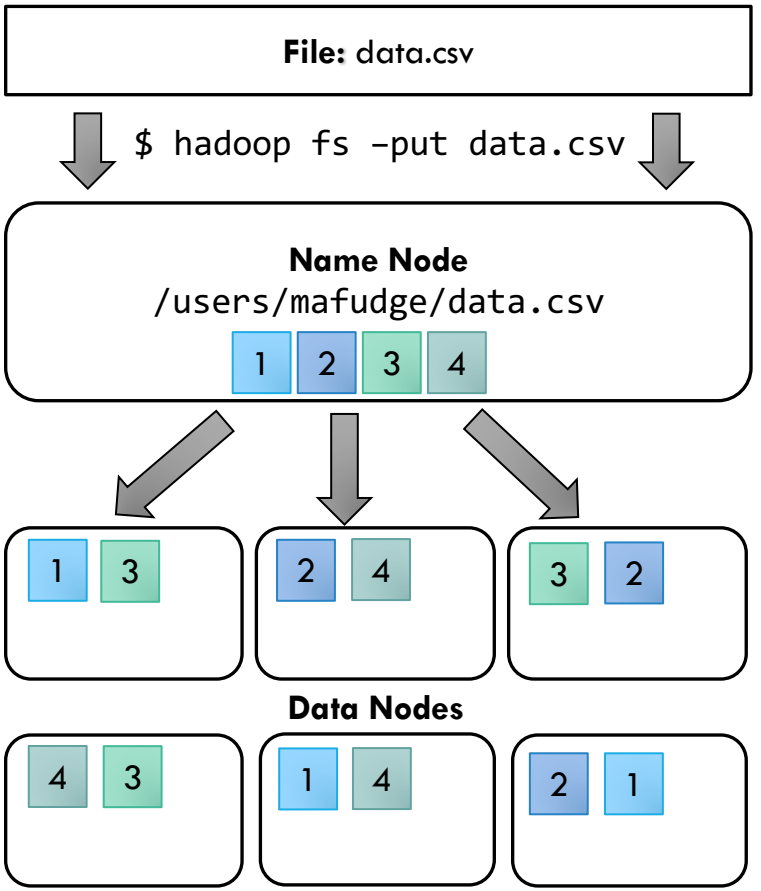
HDFS: Hadoop Distributed File System

- Based on Google's GFS
- Data distributed over physical nodes
- Designed for failover
- Data stored "as is"
- Data split into blocks
- Default replication factor is three

HDFS at Work

Client:
1) Issues command to write data.csv file to HDFS.

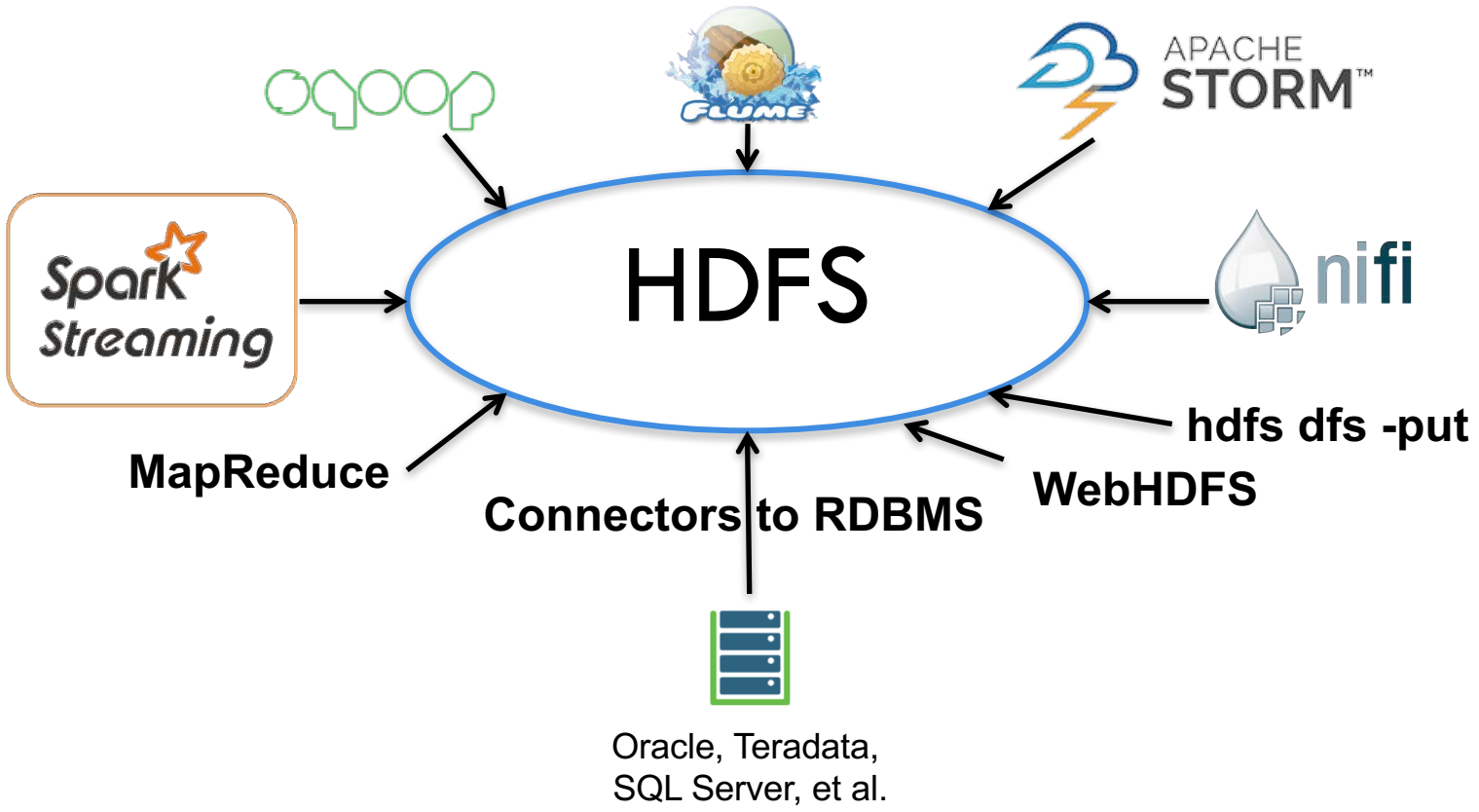
Namenode:
2) Splits the file into 64 MB blocks (size can be changed).
3) Writes each block to a separate data node.
4) Replicates each block a number of times (default is three).
5) Keeps track of which nodes contain each block in the file.



Ingesting Data into HDFS

School of Information Studies
Syracuse University

Options for Data Ingestion



When Integrating…

Remember the "Hadoop way"

- Store data "as is" (the way it was extracted from its source).
- Store common datasets/entities in a folder, not a file.
- Apply a schema to the data when they are read from HDFS.

How MapReduce Works

School of Information Studies
Syracuse University

MapReduce

A programming model for large-scale distributed data processing.

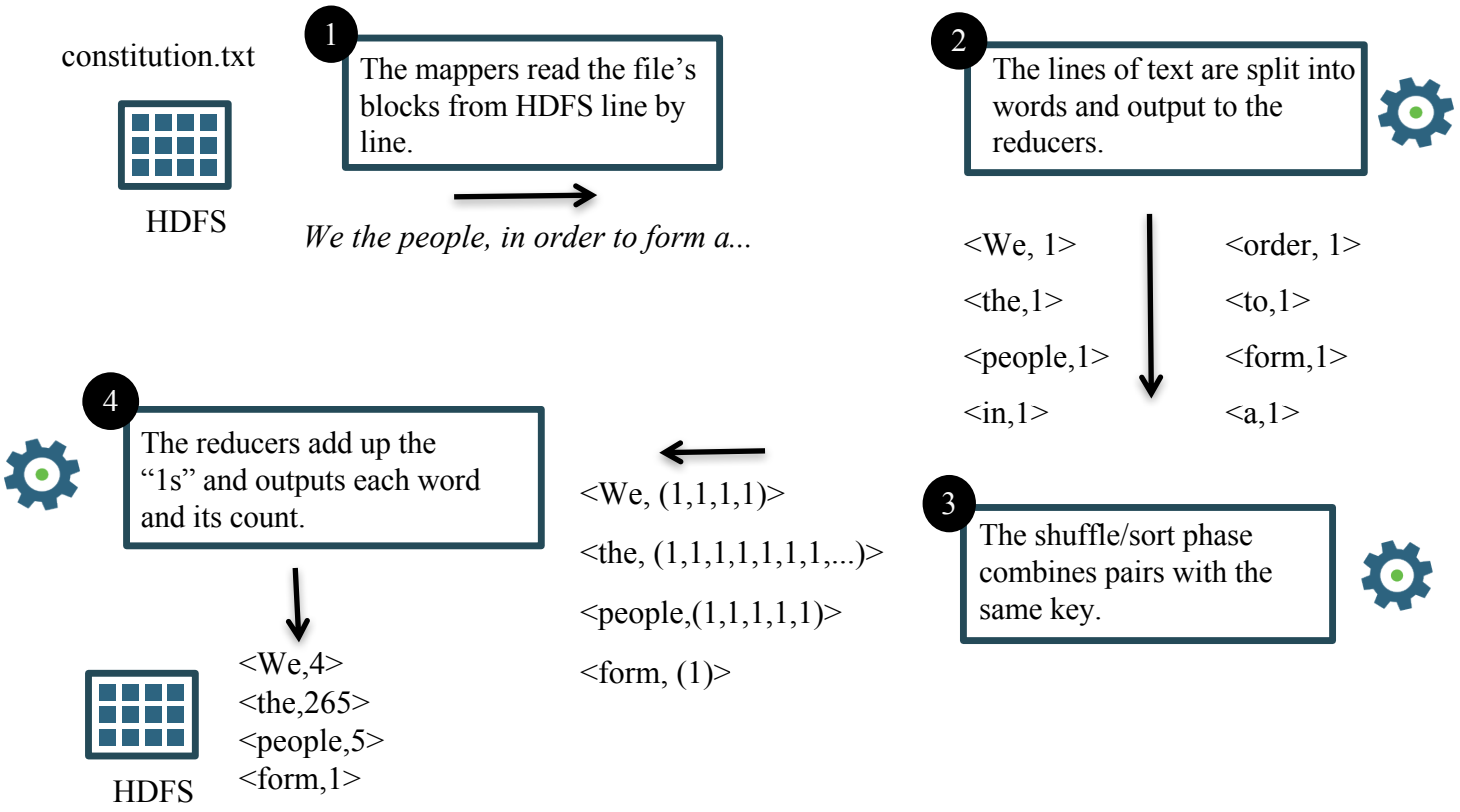
Foundations in functional programming, LISP.

- Map → Apply a transformation to a dataset.
- Shuffle → Transfer output from mapper to reducer nodes.
- Reduce → Aggregate items into a single result.
- Combine → Output of reducer nodes into single output.

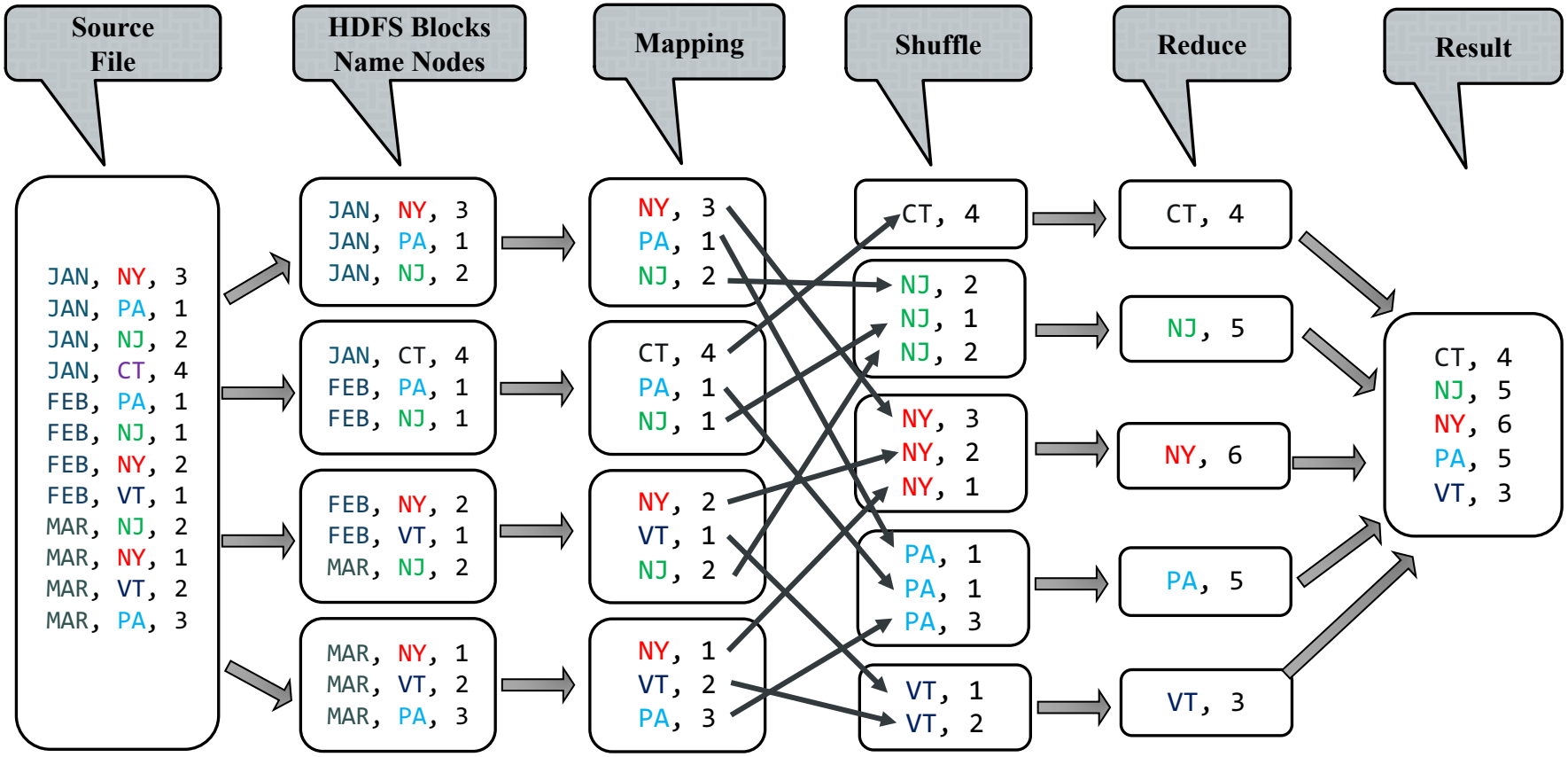
Hadoop 1.0 had only MapReduce.

Hadoop 2.0, MapReduce programs use YARN.

WordCount in MapReduce



MapReduce: Total Orders by State



YARN

School of Information Studies
Syracuse University

Hadoop 1 (MapReduce) vs. Hadoop 2 (YARN)

Hadoop 1 (MapReduce)

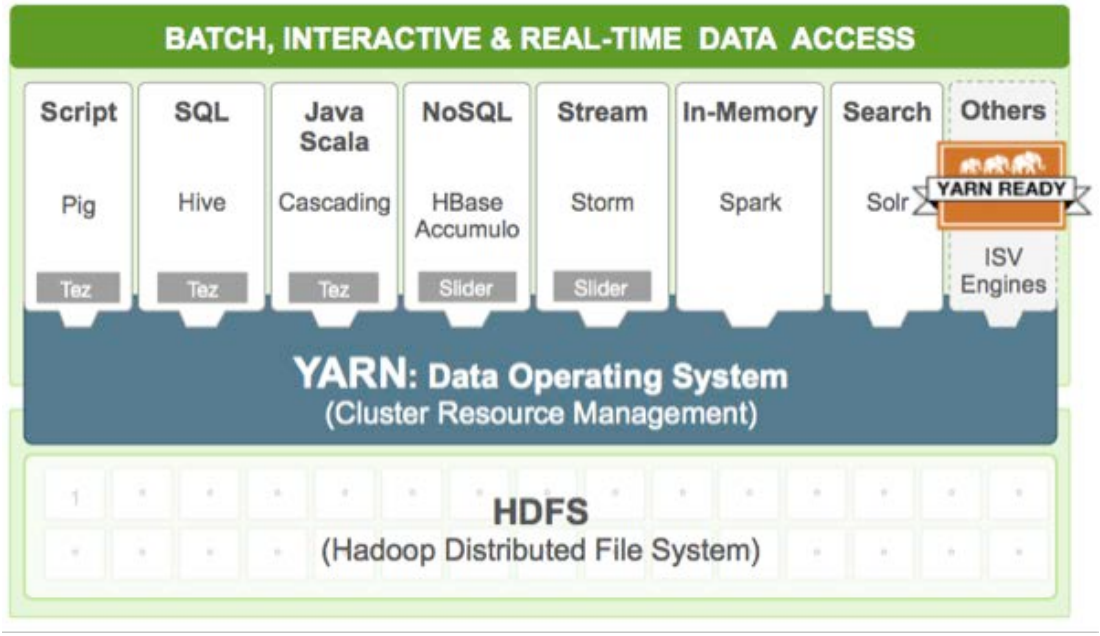
- Batch processing only.
- No high availability or failover capability.
- Only MapReduce applications.
- Data nodes and name nodes are dependent.
- Applications execute and quit.

Hadoop 2 (YARN)

- Suited to a variety of distributed tasks.
- High availability and failover.
- Runs any distributed application, including MapReduce.
- Data node and name nodes (node managers) are independent.
- Applications can remain resident.

YARN: The Data Operating System

- Hadoop 2.0 introduces YARN (yet another resource negotiator).
- Orchestrates processing over the nodes.
- Uses HDFS for storage.
- Runs a variety of applications.
- TEZ: optimization of MapReduce jobs over YARN.
- Slider: a shim to enable applications over YARN.



YARN Applications

School of Information Studies
Syracuse University

YARN Applications

- MapReduce is great, but there's a need for high-level scripting.
- There are also other needs beyond batch capabilities of MapReduce.

Pig

- Platform for analyzing large data sets, performing ETL, data cleanup, etc.
- Write code simpler for MapReduce in “piglatin” instead of Java.

Steps:

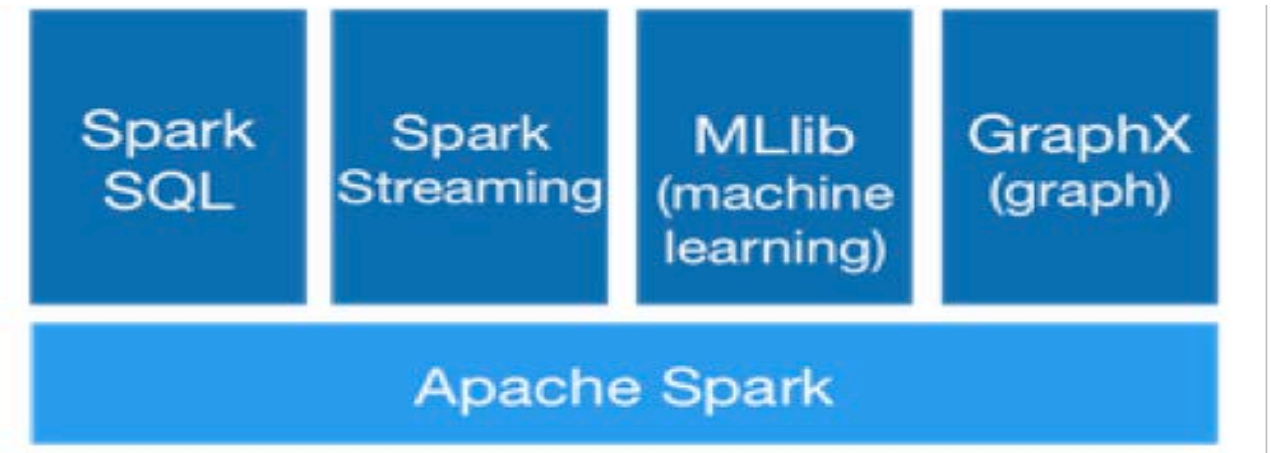
- LOAD
- TRANSFORM
- STORE /DUMP

Hive

- SQL-like syntax over HDFS
- Declarative, not procedural like Pig
- Useful for ad hoc query of HDFS data
- Dimensional models

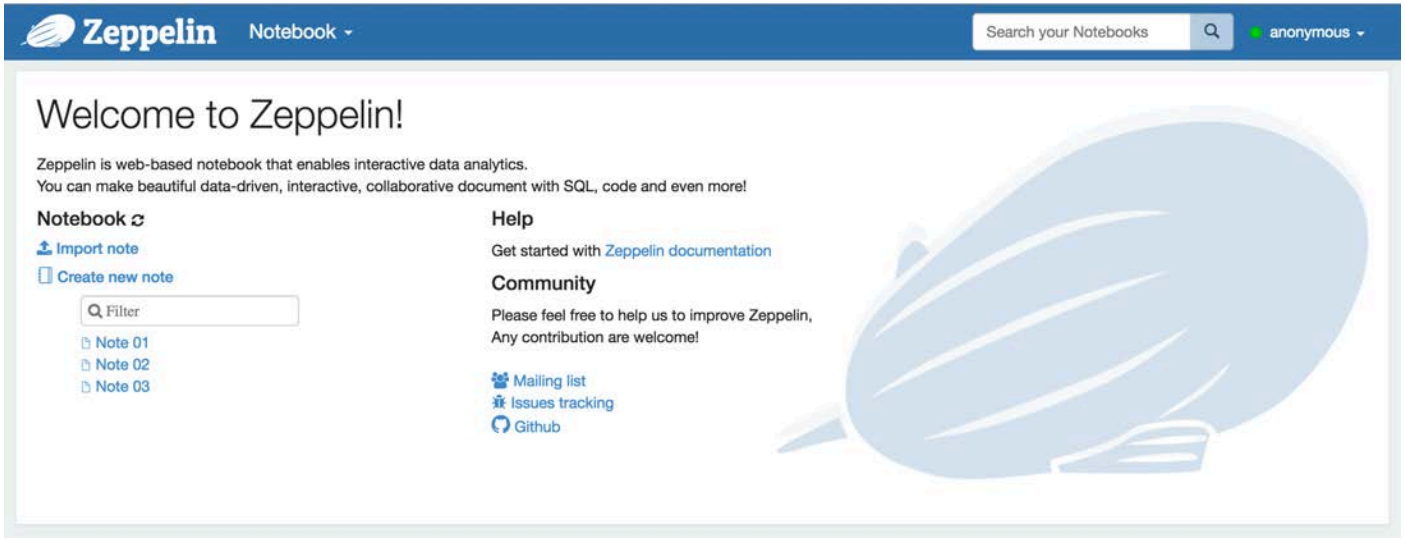
Spark

- Spark is a cluster computing framework
- Extends the M-R metaphor in memory for large-scale data processing.
- Data Mining and ML in Hadoop



Zeppelin

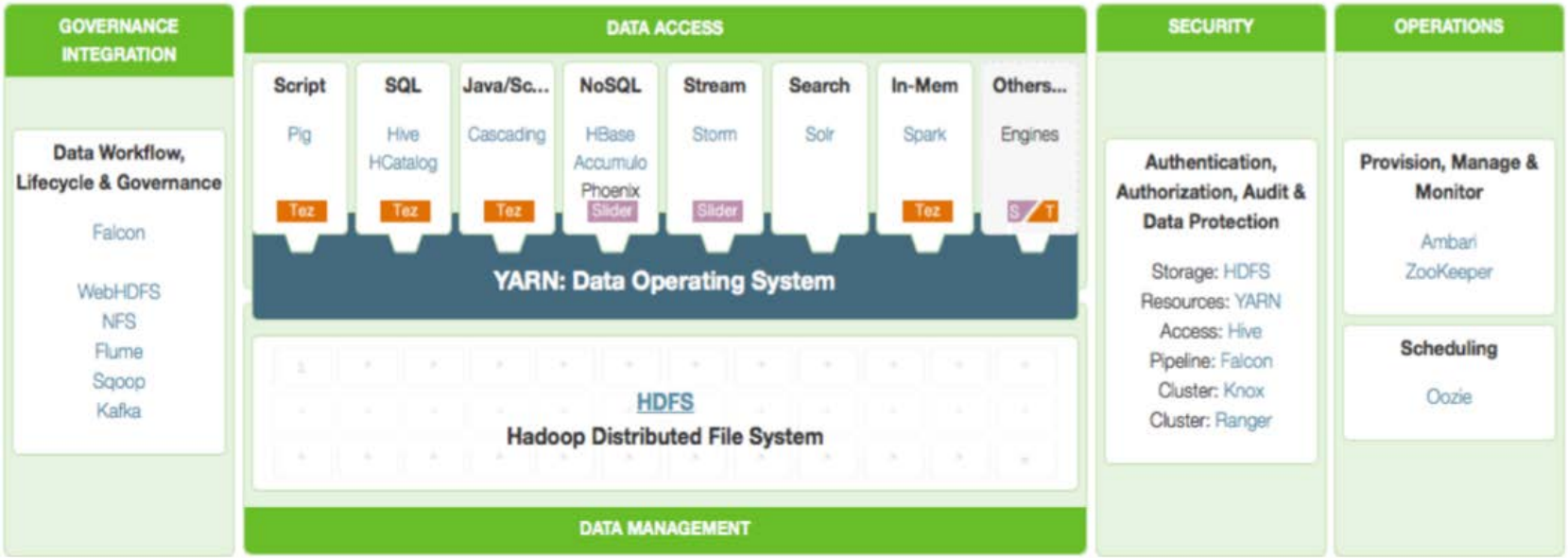
- Web-based notebook for interactive data analytics
- Interactive and collaborative
- Interpreters for Hive, Pig, Spark, Python, R, and more



Example: Zeppelin

School of Information Studies
Syracuse University

Today's Hadoop



Hortonworks

… A suite of open source, distributed technologies, addressing all needs!

School of Information Studies
Syracuse University

Big Data and the Data Warehouse

Michael Fudge