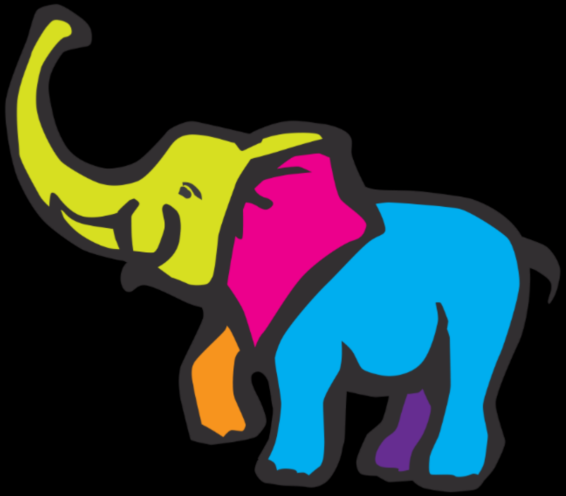
EECS4415: Big Data Systems



Data-Driven Organizations (DDOs)

Overview

- Data Driven Organizations
- Reference Model for DDO Solutions

Data Driven Organizations (DDOs)

How non-DDOs make decisions?

- Intuition
- Ad-hoc or based on few customers feedback
- Look at competition
- Try to be different
- Based on assumptions, that may be wrong
- Without knowing how to validate if it was the right decision

What do DDO's do?

- Make decisions based on data not intuition
- More precise on what they want to achieve
- Measure and validate with data

Is DDO new?

- There are organizations that have been DDO’s for a long-time
  - Walmart
  - GE
  - Airlines
- More data and better tools are enabling more companies to become DDO’s
- You have to become a DDO to compete

How do DDO's do it?

- Collect data
- Develop intuition of the data they got
- Pose questions that they try to answer; Or, search the data for new insights
- Run experiments
- Make decisions and draw insights

Example 1: Email Marketing

Pre DDO

- Did not measure effectiveness of campaigns
- Did not cluster customers
- Did not have tailored campaigns based on data

Result

- Cannibalized their own market
- Offered discounts to customer that would have bought at full price
- Significant loss revenue

Post DDO

- Behavioral clustering
- Predictive analytics
- LTV analysis (Life-time Value)
- Targeted campaigns
- Measure effectiveness

Result

- Increased revenue

Example 2: Application Feature

Pre DDO

- Introduced features on intuition
- No measurable goals

Result

- Sometimes features decreased engagement
- Offered discounts to customer that would have bought at full price
- Occasional lost revenue
- Many features, unknown value

Post DDO

- Experiments, measure
- Do not launch unless measurable benefit

Result

- Fewer failed features
- More successful feature introductions (increased engagement)
- Remove features that do not contribute to metrics

Summary

DDO's

- collect data
- make decisions based on data, not intuition
- use data to drive applications

To be a DDO, you need an efficient way of storing and retrieving data

Reference Model for DDO Solutions

Thanks to Jari Koister
UC Berkeley

Challenge

- A variety of solutions/technologies available
- There is no one solution/technology that solves all possible data analytics problems
- Most solutions solve a range of problems, but are outstanding on a specific type

How to map problems to DDO solutions?

How to compare alternative DDO solutions?

Need for a Reference Model

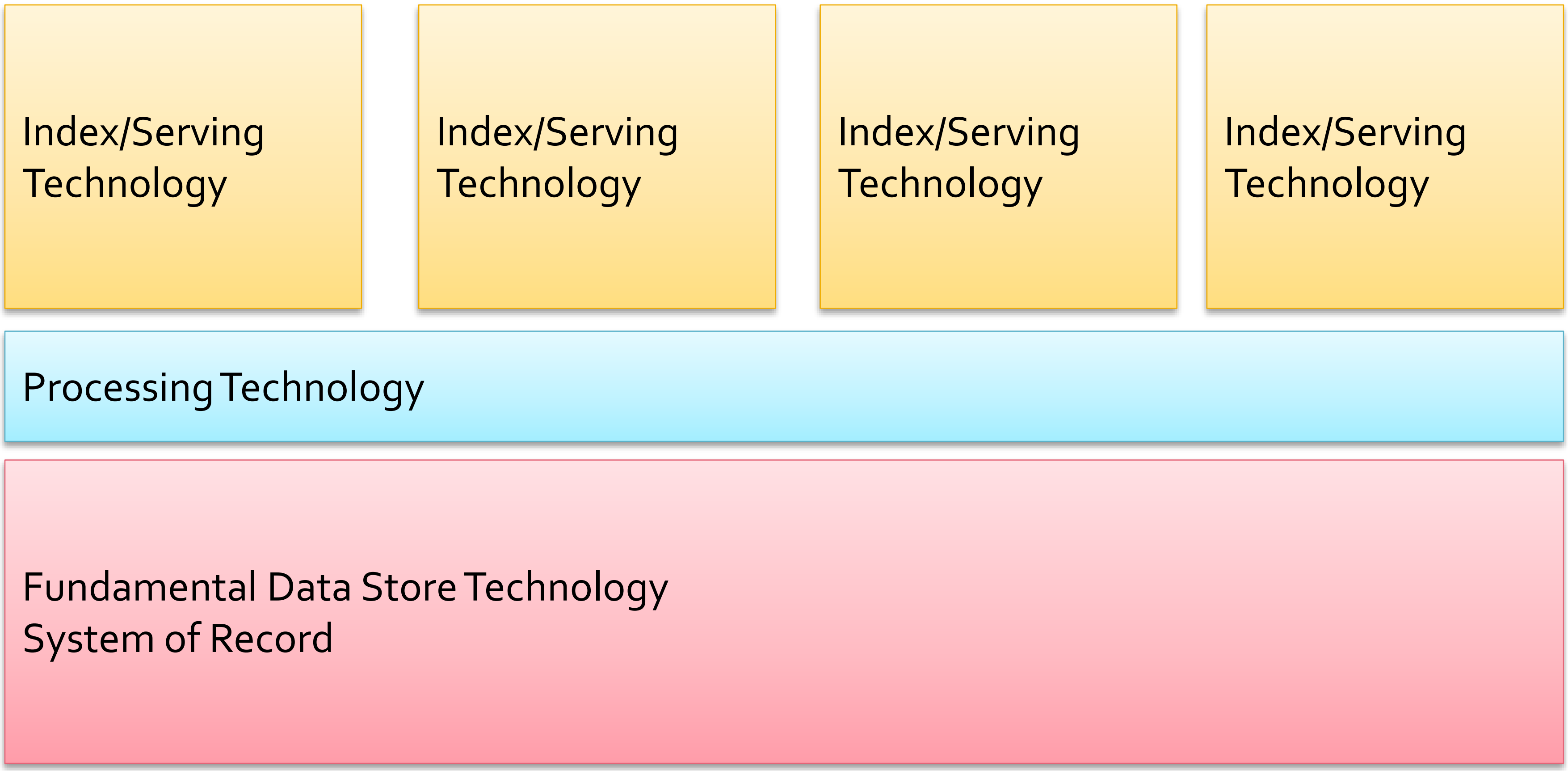
Purpose of the Reference Model

- Provides **a framework** for
  - **understanding** your needs
  - **comparing** solutions
- Not complete, but gives an approach to understanding data analytics systems

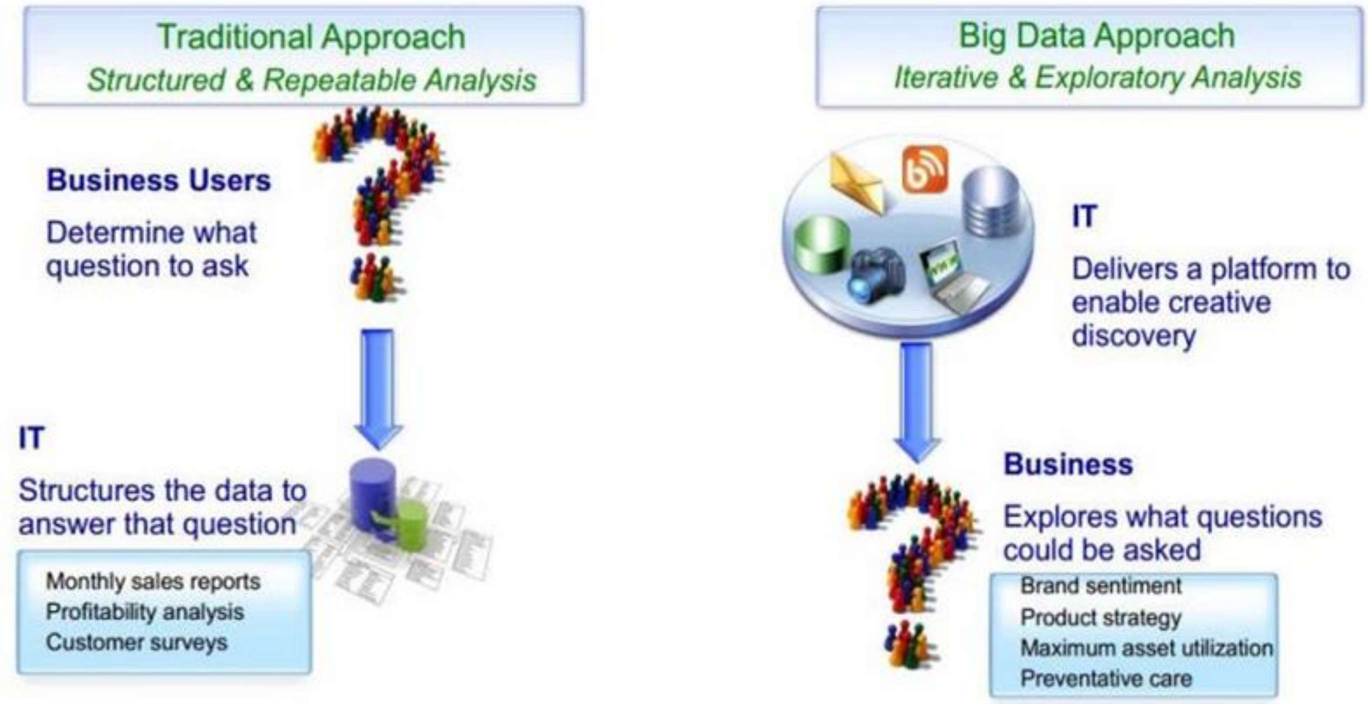
Traditional Structure

- Handles **certain type of data** well
- Handles **certain ranges of data size** well
- Performs **certain types of queries and computations** well

A Big Data Approach



Difference in Approach



Notice the difference!

How to Evaluate a Solution?

To be able to evaluate a solution you need to understand your needs

- What is the structure of your data?
- How big is your data?
- What is the velocity?
- What kind of processing is needed?
- What kind of queries do you want to answer?
- What is the expected latency?
- …

Dimensions

Data

What characteristics should be considered with respect to **data**?

Processing

What characteristics should be considered with respect to **processing**?

Other dimensions (not covered):

cost, implementation complexity

Data Dimension

Data Dimension

Data related characteristics

- Structure
- Size
- Sink Rate
- Source Rate
- Quality
- Completeness

Data Structure

What is the type of the data (Variety)?

Structured: Well defined schema, data types, understandable by machine

Unstructured: Loosely typed (text, pics)

Semi-structured: Mix of structured and unstructured. Ex. Well defined schema, but some attributes are unstructured

Size

What is the size of the data (Volume)?

S: Megabytes

M: Gigabytes

L: Tera Bytes

XL: 100's of Tera Bytes

XXL: Peta Bytes

Sink Rate/Speed

How fast the data are coming in (Velocity)?

Very High: > hundreds of updates per second

High:> tens of updates per hours

Medium: a few updates per hour

Low: Updates daily or less frequently

Source Rate/Speed

How updated is the indexing/speed layer?

High: updated in "real-time" as data arrives
Medium: Updated on an hourly basis
Low: Updates on a daily or less frequently

Quality

How well does the system deal with bad or low quality data (Veracity)?

High: can compensate and handle in an automated fashion

Medium: can handle but results may be unreliable

Low: can not handle bad or low quality data. Will not provide any results

Completeness requirement

How well does the system deal with incomplete data?

Incomplete: can enrich and complete data efficiently

Semi-complete: provides some capabilities for completing and enriching data

Complete: requires data to be complete before processing

Processing Dimension

Processing Dimension

Processing related characteristics

- Query Selectivity
- Query Execution Time
- Aggregation
- Processing Time
- Join
- Precision

Selectivity

Is it better at high or low query selectivity scenarios? (In a High Selectivity scenario a query predicate is more selective, meaning that only small percentage of data rows satisfy the query)

High: expect < 20% of data to be selected
Medium: expect 20-80% of data to be selected
Low: expect > 80% of data to be selected

Query Execution Time

What query response time is the system designed to meet?

Short: milliseconds or less than a few seconds
Medium: speed of thought, ar most 30 seconds
Long: minutes or tens of minutes

Aggregation

What is the level of expressiveness and computational capabilities of *aggregations*?

Advanced: Roll-ups, drill-downs, lattice, cuboids

Medium: Aggregations over multiple dimensions

Basic: simple counters

Processing Time

What processing time is expected for batch jobs? (24 hours is an important limitation for many applications)

Short: < 1 hour
Medium: < 12 hours
Long: > 24 hours

Join

What is the level of expressiveness and computational capabilities of *joins*? (Join is a common operation; there is a variety of joins that are suitable for different data distributions, data sizes etc.)

Advanced: a variety of joins for different functional and optimization scenarios

Basic: limited capability for join

None: No join supported

Precision

What is the expected output precision? (May be impacted by potential loss of data, approximations, sampling, etc.)

Exact: Always exact, includes full data set

Approximate: Approximates result for example through sampling

Lossy: May miss some data for the benefit of speed or scale. Or may count data twice in the event of recovery

Example DDO Solutions

Dimensions: Examples

DDO solutions to investigate

RDBMS: Relational model with powerful querying capabilities

HDFS+M/R: Batch oriented system for processing and storing large data sets

Storm: A stream processing system that can compute in real-time over large streams

BlinkDB: Experimental system for approximate query answering over large data that trade error over response time

RDBMS: Data

Dimension	Characterization	Note
Structure	Structured	Good with structured data. Can store unstructured too
Size	S-->L	Efficiently deals with up to L size data
Sink Rate	High	Depending on the number of records being pushed into a system the ingest capacity will vary. Databases can deal with frequent updates up to a point, but when updates are in hundreds per second the data base will have trouble keeping up
Source Rate	High	Can update results computations quickly and be triggered in real-time
Quality	Medium	Databases in themselves are not good at handling low quality data. But they can be programmed to do cleaning and other tasks to prepare the data
Completeness	Incomplete	Databases can deal with missing values or be used to complete data before processing

RDBMS: Processing

Dimension	Characterization	Note
Query Selectivity	High, Low	Normally databases can deal with both low and high selectivity queries. They have facilities such as indices to optimize for specific use cases
Query Time	Short, Long	Normally intended for quick queries, but also used for long running queries
Aggregation	Advanced	Has advanced facilities for aggregating and grouping data in batch or in realtime
Processing Time	Short, Long	Facilitates both short and long running processes
Join	Advanced	Relational databases normally support a variety of join's for different functional and optimization scenarios
Precision	Exact	Queries and processes are normally over the complete dataset

HDFS + M/R: Data

Dimension	Characterization	Note
Structure	Structured and unstructured	Generally used to handle both structured and unstructured data
Size	XL, XXL	Intended for very large data sets
Sink Rate	very high, high	Can be used to store incoming data at high rate. No ACID properties and immutable data facilitates a fast storage process
Source Rate	medium, low	Updates are not fast due to longer processing cycles
Quality	medium	Can be used to deal with lower quality data
Completeness	incomplete	Can be used to enrich and complete data

HDFS + M/R: Processing

Dimension	Characterization	Note
Query Selectivity	low	general a more efficient method when selectivity is low. But can of course deal with high selectivity as well, has not indices though
Query Time	long	Queries take a long time to execute
Aggregation	medium	almost anything can be done, but certain types of operations may not be as efficient
Processing Time	long, medium	suitable for long and medium length processes
Join	basic	There are many abstractions such as Pig that provide powerful Join capabilities on M/R
Precision	exact	Normally operates on the full data set

Storm: Data

Dimension	Characterization	Note
Structure	Structured	
Size	XL, XXL	Designed to efficiently deal with large sets of streaming data
Sink Rate	Very high	
Source Rate	High	A serving layer can be updated in real-time
Quality	Medium	
Completeness	Complete	Generally expects data to be complete for processing. But it can be augmented

Storm: Processing

Dimension	Characterization	Note
Query Selectivity	high to low	Selectivity is not the major factor. Although high selectivity would result in larger streaming graphs
Query Time	N/A	Is not queried directly, rather results are pushed to a serving component
Aggregation	Medium	Generally better at simpler aggregations over incoming data streams
Processing Time	short	Processing is designed to take place in real-time
Join	basic	Streams can be joined, but there are limitations such as over which datasets joins can be made etc
Precision	lossy	Provides at-least-once semantics

BlinkDB: Data

Dimension	Characterization	Note
Structure	Structured	
Size	XL, XXL	Is designed to handle interactive queries over large datasets. No reason to approximate if datasets or smaller
Sink Rate	N/A	Uses HDFS as underlying storage
Source Rate	N/A	
Quality	low	Designed to process mainly quality data
Completeness	Complete	

BlinkDB: Processing

Dimension	Characterization	Note
Query Selectivity	High, Low	
Query Time	Short	It is designed to give shortest possible response time, but with bounded errors
Aggregation	Medium	Same basic capabilities as Hive and other big data systems
Processing Time	Short, medium	Designed to support shorter processing time over big data sets
Join	Basic	Basic join support as provided by Hive and other systems
Precision	Approximate	Allows errors within bounds by design

Dimensions: Summary

Aspect	Dimension	RDBMS	M/R	Storm	Blink DB
Data	Structure	structured	all	all	structured
	Size	S->L	XL,XXL	S->XXL, streaming	XL,XXL
	Sink Rate	high	very high, high	very high	N/A
	Source Rate	high	medium, low	high	N/A
	Quality	medium	medium	high,low	low
	Completeness	incomplete	Incomplete	complete	complete
Processing	Selectivity	high, low	low	high,low	high,low
	Query Execution time	short,long	long	N/A	medium
	Aggregation	advanced	medium	medium	medium
	Processing time	short, long	long, short	short	short, medium
	Join	advanced	basic	basic	basic
	Precision	exact	exact	lossy	approximate