Transactional Services for NoSQL Big Data Systems

Younas, Muhammad

Oxford Brookes University
Oxford, UK

Outline

- Big Data and its characteristics
- Big Data usage/applications
- Big Data models and NoSQL systems
- NoSQL Systems – Key benefits and issues
- Transactional services for NoSQL systems
  - System design and development
  - Application scenarios and experiments
  - Evaluation: Efficiency and Data Consistency
- Future research Issues

Big Data

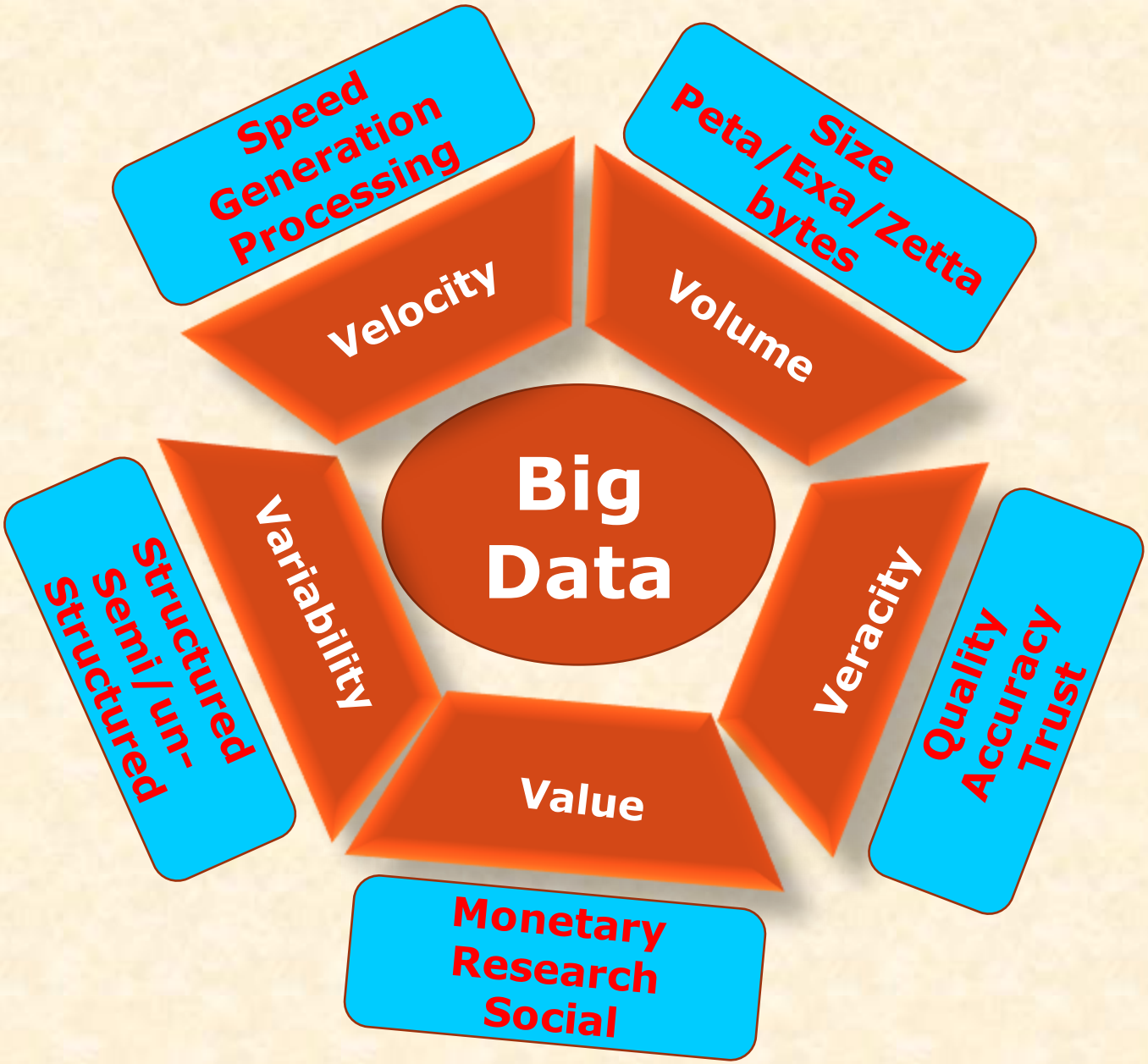
Big data refers to the large volume of complex, structured / unstructured data that is generated in a large size and that arrives (in a system) at a higher speed …

to be analysed for better decision making and strategic organization/business moves.

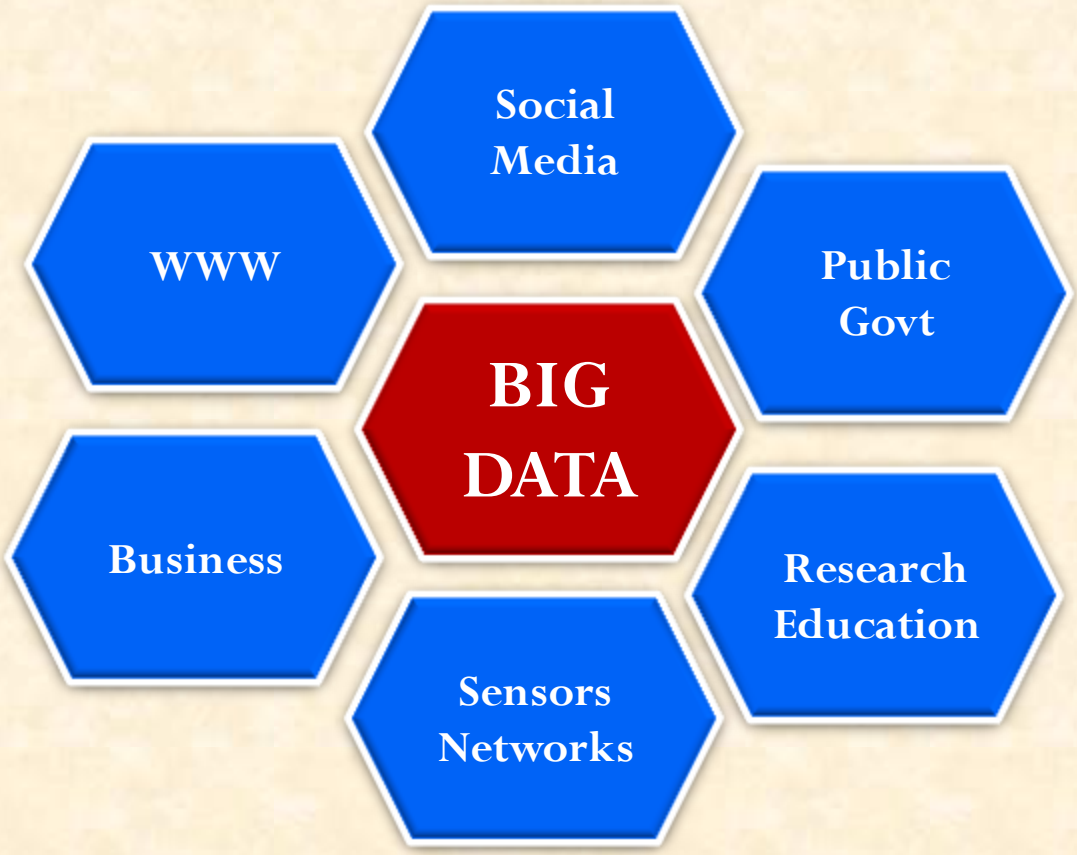
The process of managing large volume of data is not new. e.g., the top-ranking Conference on VLDB: Very Large Databases > 30 years.

But the concept of Big Data gained popularity when the 3Vs or 5Vs characteristics (models) were defined.

Big Data Characteristics

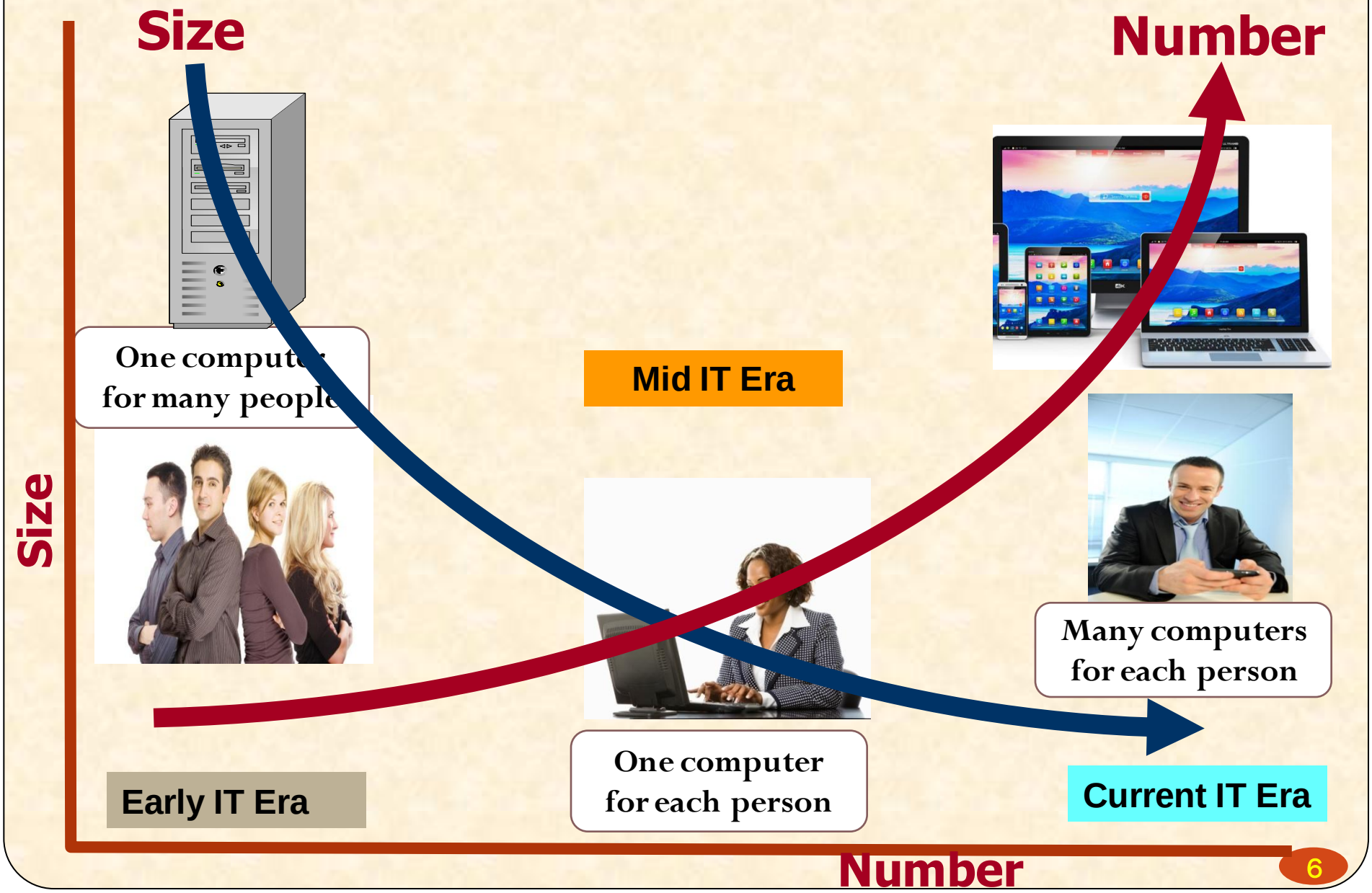


Big Data Usage/Applications



….and many other applications

Big Data Usage/Applications



Big Data Usage/Applications

10 billion mobile devices will be in use by 2020

Trillion of sensors used for monitoring, tracking and communication

Emails sent per day:
294 billion

Google searches:
One billion per day

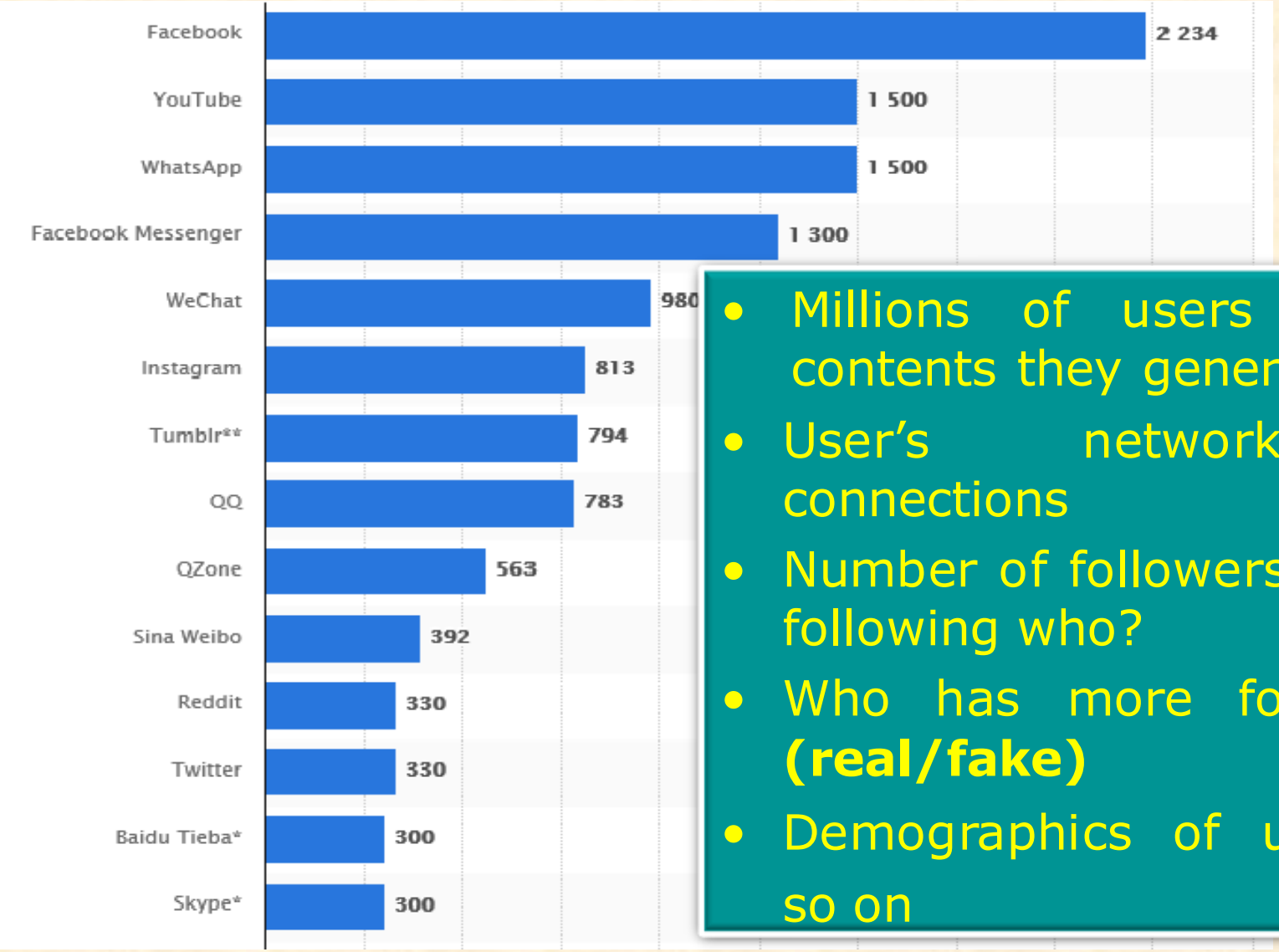
Facebook:
More than 30 petabytes /day

Twitter:
More than 230 million tweets / day

But only 23% organizations have enterprise-wide strategies for Big Data

Source (IBM) Quick Facts and Stats on Big Data. Available online:
http://www.ibmbigdatahub.com/gallery/quick-facts-and-stats-big-data

Social networks April 2018 - number of active users (in millions)



- Millions of users and the contents they generate
- User's network and connections
- Number of followers - Who is following who?
- Who has more followers - **(real/fake)**
- Demographics of users and so on

statista The Statistics Portal
Statistics and Studies from more than 22,500 Sources

Big Data Models

Storing and processing Big Data require new data models and new technologies

Classical Relational model and SQL technologies do not cater for the needs of Big data

Most common Big Data models:

- Document model
- Key-value model
- Column model
- Graph model

Big Data Models & NoSQL Systems

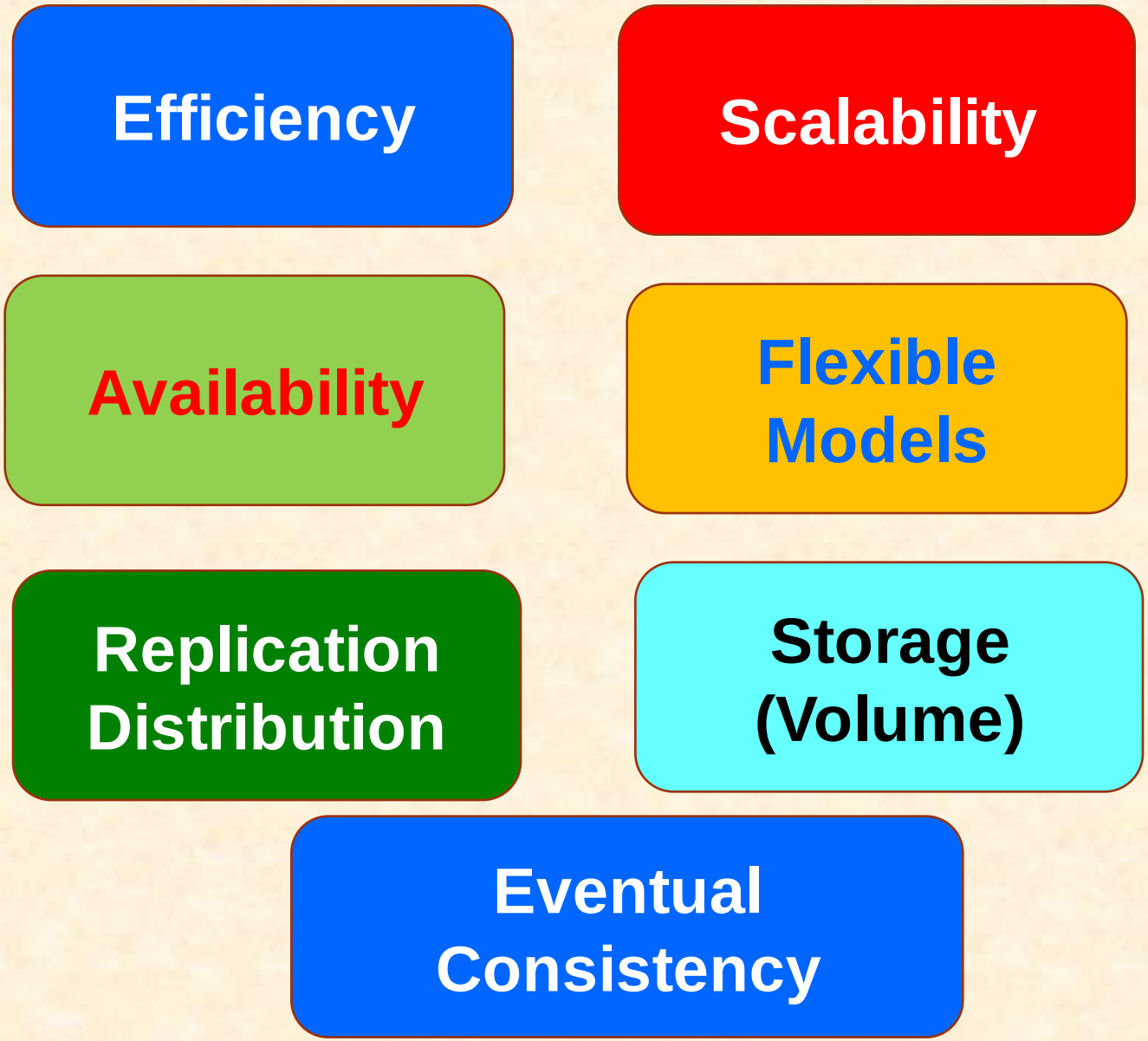
Document DBs

Key-Vale DBs

Column DBs

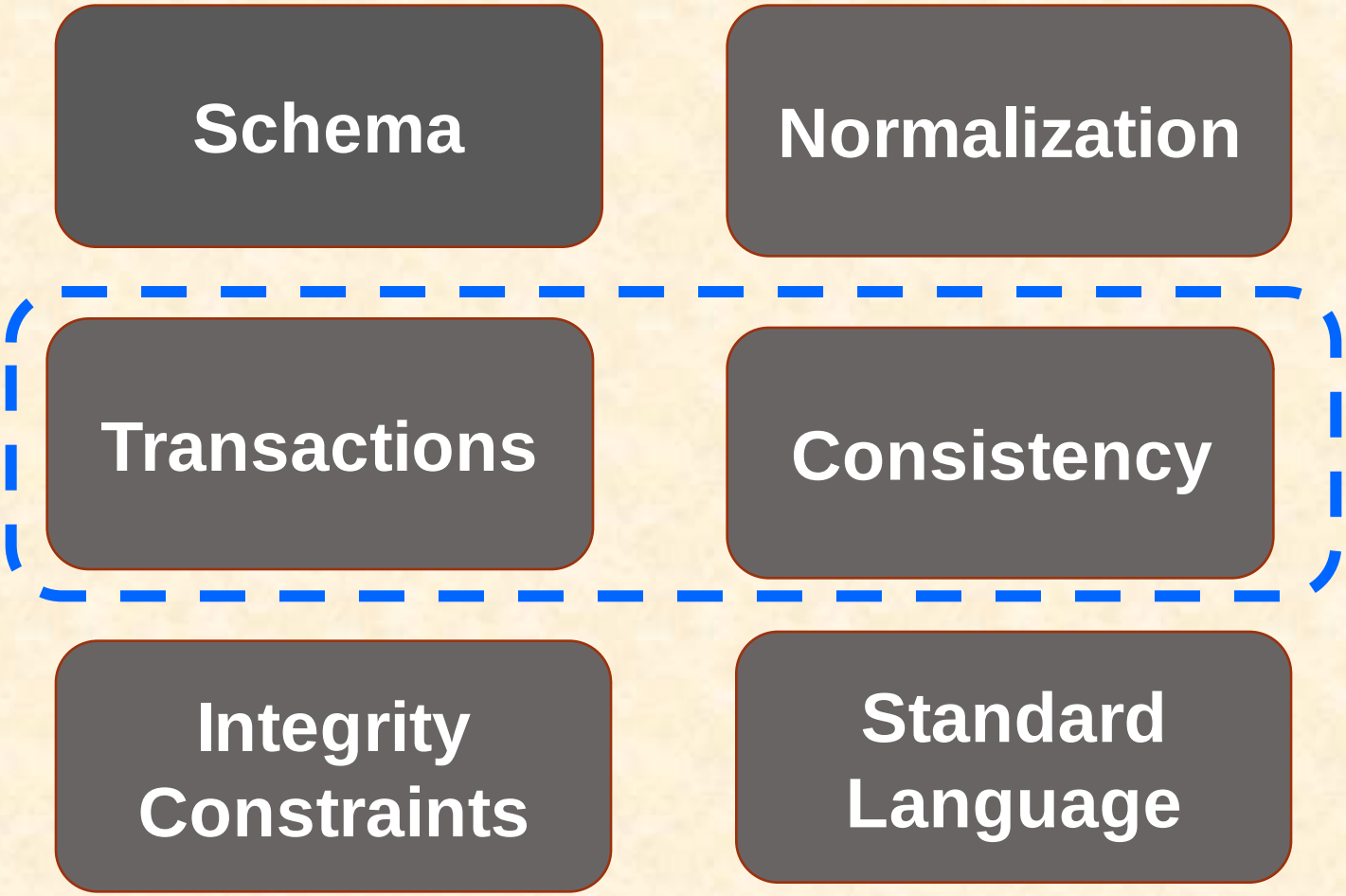
Graph DBs

NoSQL Systems – Key Benefits



NoSQL Systems – Key Issues

Lack of:



NoSQL Systems – Key Issues

Consistency:

- NoSQL databases generally enforce Eventual Consistency that "**all updates will reach all replicas after a certain delay**".
- This works for applications such as social media --- it's acceptable if a new message reach all replicas after certain delay.
- For example, "**parents share photos of life before and after having kids**".

(source Yahoo Style)

(source Yahoo-Style)

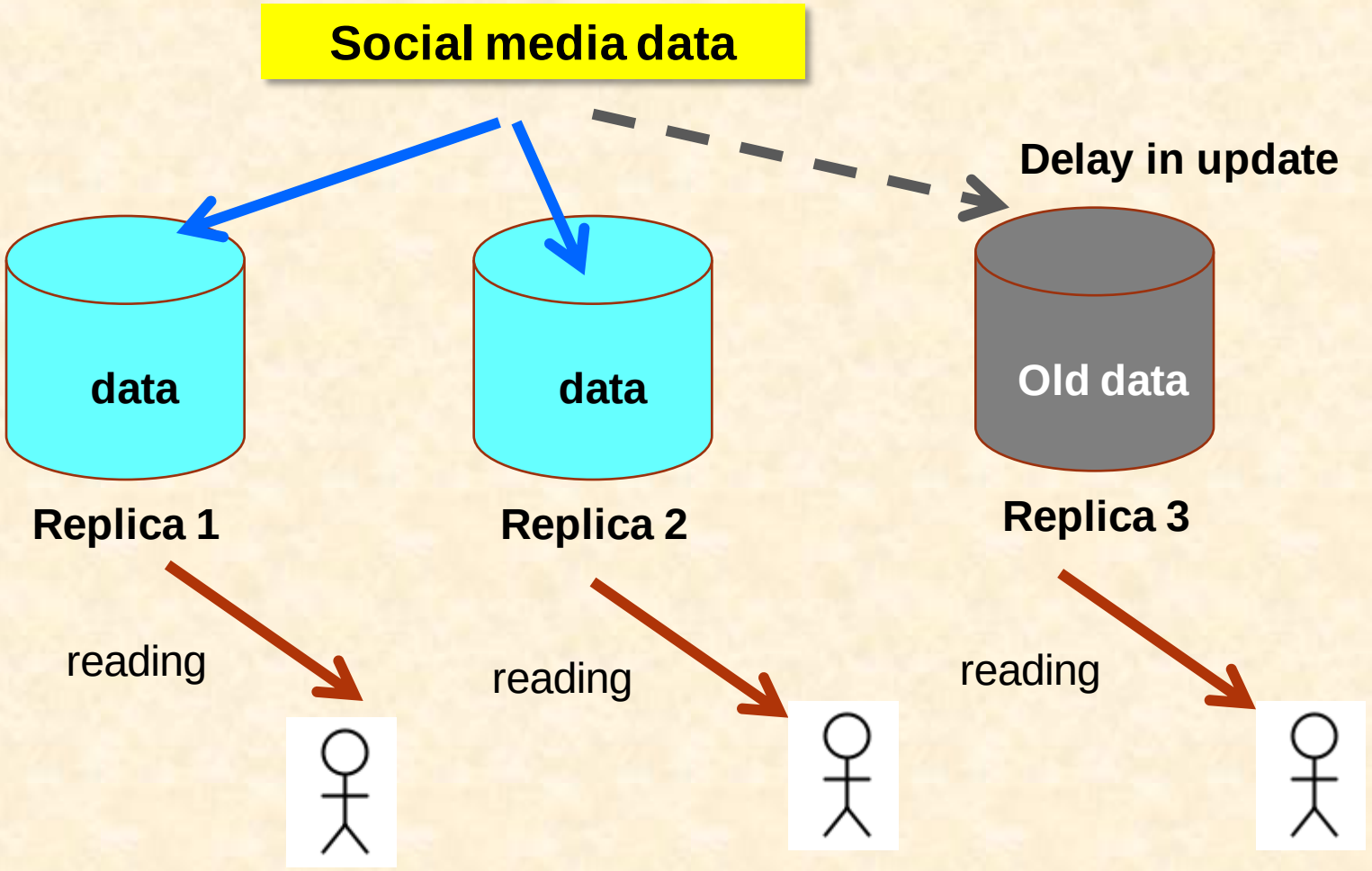
Life before having kids

Life after..

When you've to share food

NoSQL Systems – Key Issues

- Eventual Consistency : updating/reading social media data



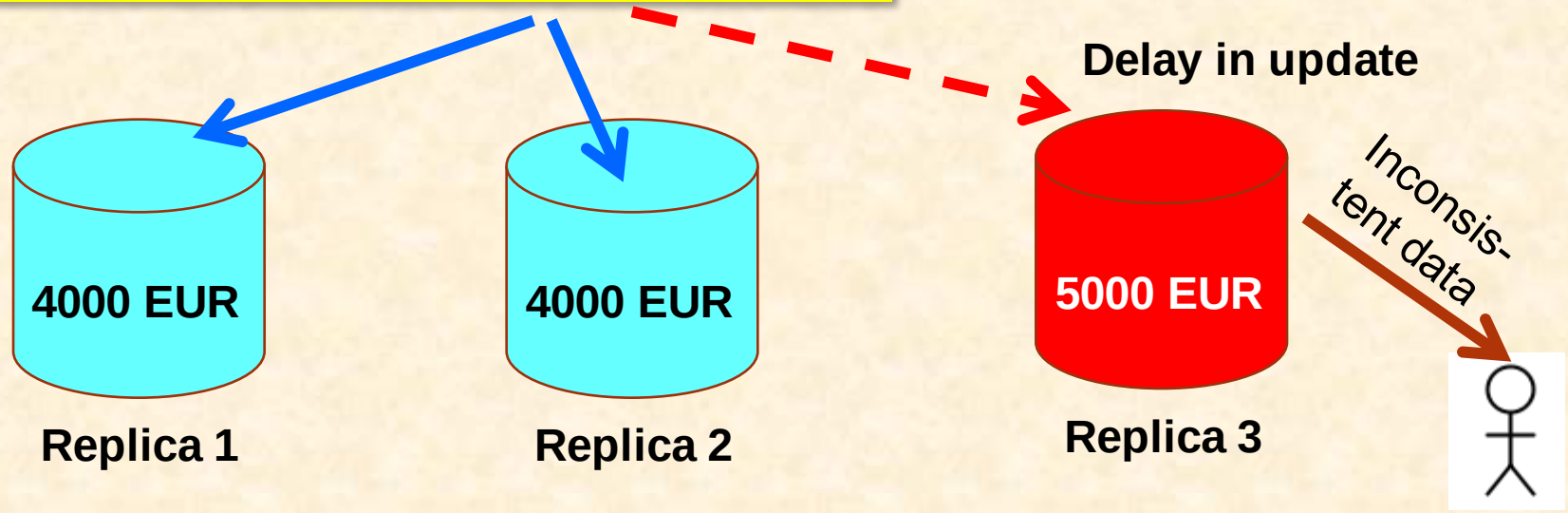
NoSQL Systems – Key Issues

- But Eventual Consistency does not work for applications such as banking transactions that require strict consistency
- For example: Update bank balance data

Balance = 5000 EUR

Pay bill = 1000 EUR (update operation)

Balance = 5000 – 1000 = 4000 EUR



NoSQL Systems – Possible Solutions

Transactional Services: to implement transactions in NoSQL Big Data systems

Ensure stronger consistency or consistency based on the context of an application

Minimise the effects of ensuring stronger consistency on efficiency and availability of NoSQL Big Data systems

ACID Transaction model or other models such as ACID with relaxed atomicity, etc.

The Proposed Solution/Project

Transactional Services for NoSQL Big Data Systems

Joint Project for designing and developing Transactional Services

Oxford Brookes University, UK

University of Oviedo, Spain

Accenture, Madrid, Spain

Also sponsored by the Spanish Ministry of Economy and Competitiveness and the Principality of Asturias, Spain.

Transactional Services for NoSQL Big Data

Objectives are:

- Design and develop new transaction models and protocols
- Ensure consistency in NoSQL Big Data systems
- Test the effects of consistency on efficiency and availability in NoSQL Big Data systems
- Use different application scenarios such as real data from London Bus Services, Yahoo benchmark, etc

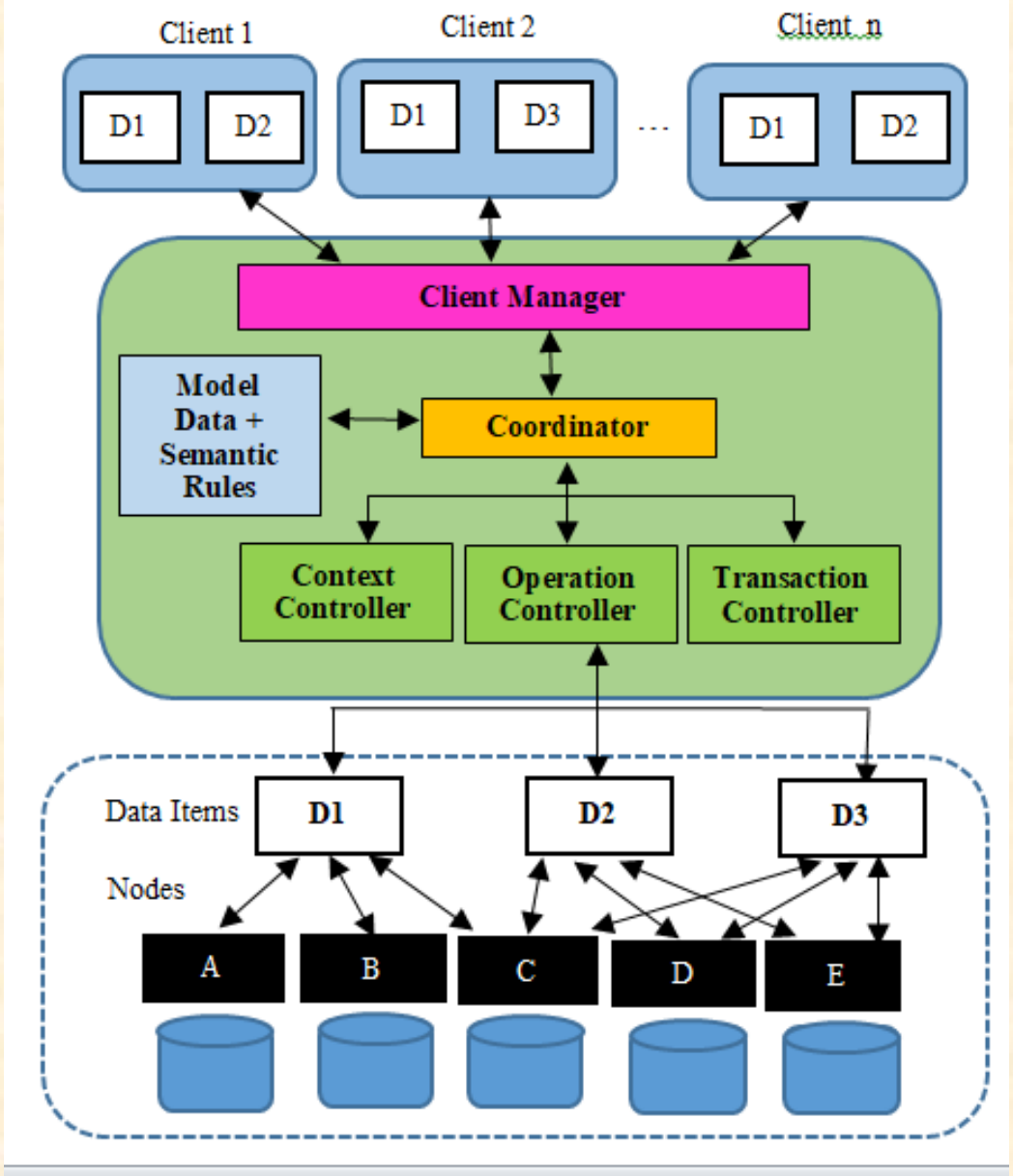
NoSQL Transactions

- A NoSQL transaction (denoted as NST) is defined as the execution of an application which comprises a sequence of operations that provide transitions between consistent states of the NoSQL data.
  - A *NST can be formally defined as a tuple, NST = (OP, PaO), where OP is* a set of operations, *OP = {OPi | i = 1…n}, and PaO is a partial ordering of the operations* which determines their order of execution. For instance, *OPi > OPj represents that Opi* executes before *OPj.*

NoSQL Transactions

- $OP_i^r[DE]$ represents a read operation of *NST*; meaning that *NST* reads a data entity, *DE*, from a NoSQL database.
- $OP_i^w[DE]$ represents a write operation of *NST*; meaning that *NST* writes (updates) a data entity, *DE* , to a NoSQL database.
- Such operations ($OP_i^r[DE]$ and $OP_i^w[DE]$) are used to model the CRUD (Create, Read, Update and Delete) operations which are most commonly implemented in NoSQL systems such as Riak and MongoDB.

Transactional System for NoSQL



NoSQL system, Riak, is used as data store

Transactional System for NoSQL

- The system comprises client layer and a middleware layer. This separates the transactional system from the underlying data store of the NoSQL database, Riak

Client layer:

- It receives requests from the client (user) that needs to be executed.
- The requests are managed by the Client Manager which is part of the Middleware layer

Transactional System for NoSQL

Middleware layer: Comprises different components.

- Coordinator receives requests from the Client Manager
- Coordinator manages the overall execution of CRUD (Create, Read, Update and Delete) operations which are grouped into transactions.
- Every request is then sent to the Transaction Controller (TC).
- TC creates a new transaction and a sequence number for transaction ID. Sequence numbers are assigned to transactions in an increasing order.

Transactional System for NoSQL

Middleware layer (continued):

- Concurrency of multiple transactions, accessing shared data, can be handled using several techniques such as Two-Phased technique or Snapshot isolation.
- Our transactional system uses the latter technique which deals with multi-version concurrency – this is more appropriate for managing multiple replicas of same data in NoSQL databases. Several write or read-write operations can simultaneously manipulate the same data and may result in write-write or read-write conflicts. Coordinator and TC therefore control concurrent data access in order to prevent data constraint violations

Transactional System for NoSQL

Operation Controller:

- It is in charge of the communication with the NoSQL database during execution of transactions.

Context Controller:

- It manages context information during the execution of CRUD operations of a transaction.
- The Coordinator and TC in cooperation with the Context Controller can decide on the outcome of a transaction if it meets the required context.

Transactional System for NoSQL

Context Controller:

- Context refers to a set of NoSQL database internal parameters (N, R, W) which are configured according to level of consistency and availability required by client
- 'N' is the number of replicas of each data item that the NoSQL database will maintain.
- 'R' represents the number of replicas that the application will access when reading the data item.
- 'W' represents the number of replicas of the data item that must be written before the write can complete

Transactional System for NoSQL

Context Controller:

- In order to ensure strong consistency, N = W
- That is, all replicas of the data item need to be updated before the data item can be accessed by an application.
- Weaker level of consistency is achieved by setting W < N, such as W=1 and N=3 (e.g., number of replicas is 3). In this case, write operation has to update one replica of the data item. Other replicas can be updated after certain delay and made consistent (which is the case of eventual consistency).
- **Note: in current NoSQL, strong consistency (with N=W) is ensured when data is processes by a single operation and not a transaction.**

Application Scenario and Test Data

- Prototype transactional systems for NoSQL databases, Riak and MongoDB
- Experiments were conducted using:

A. Data from YCSB+T benchmark, an extension of the Yahoo! Cloud Services Benchmark (YCSB) benchmark

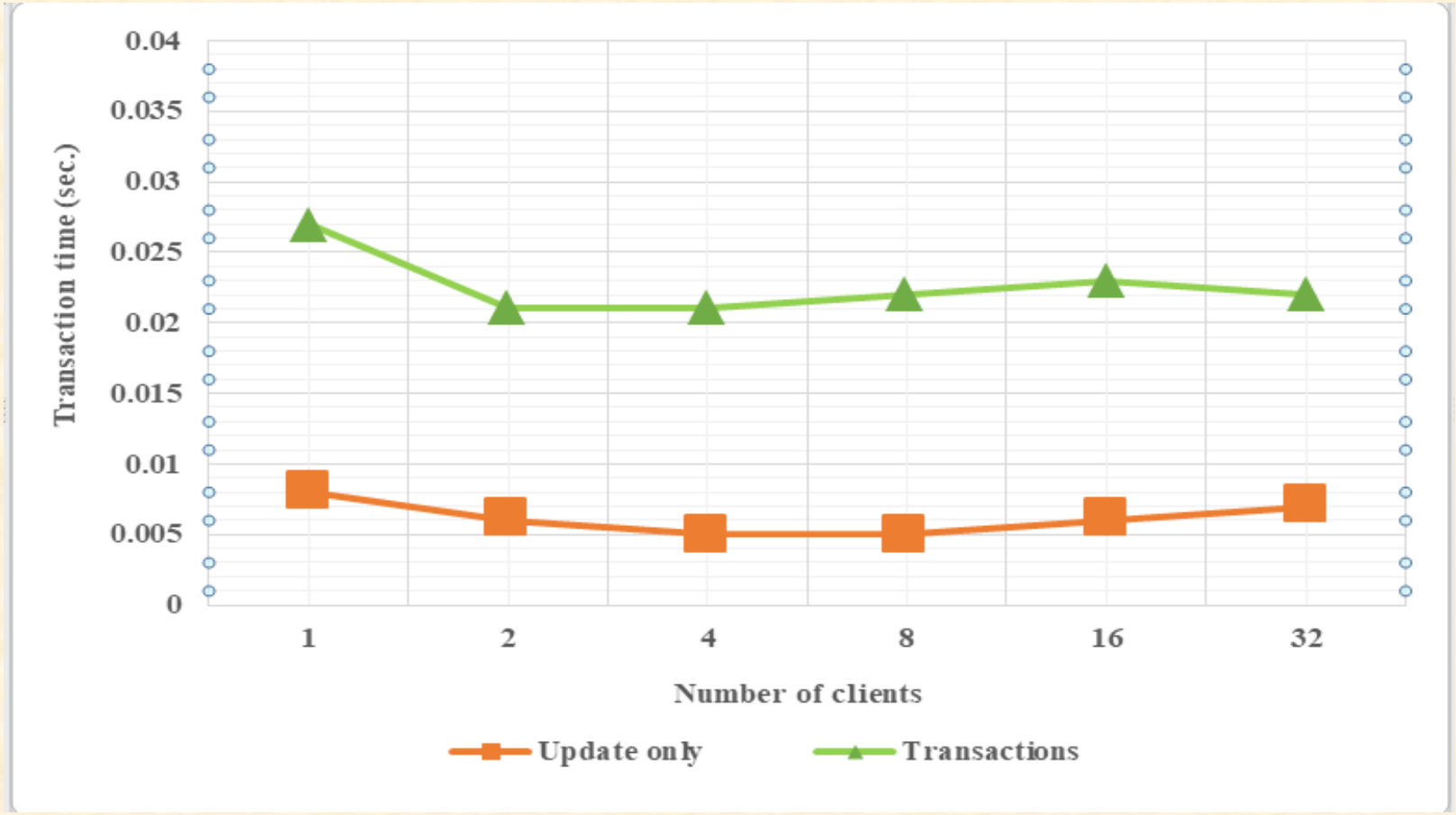
B. Data from the London Datastore -- an open and big about London buses transport fleet, and is data provided by the Greater London Authority, UK

A) YCSB+T benchmark

- **Experimental setup:** To evaluate the efficiency of transactions by taking into account the number of transactions and the time taken to process such transactions in comparison to update-only operations (i.e. non-transactions)
- Number of transactions per client =100.
- The number of clients varies from 1 to 32.
- With one client the system executes 100 transactions/sec. With 32 clients the system executes 3200 transactions/sec
- Two experiments were conducted : One for MongoDB and one for Riak

Experiment 1 (MongoDB)

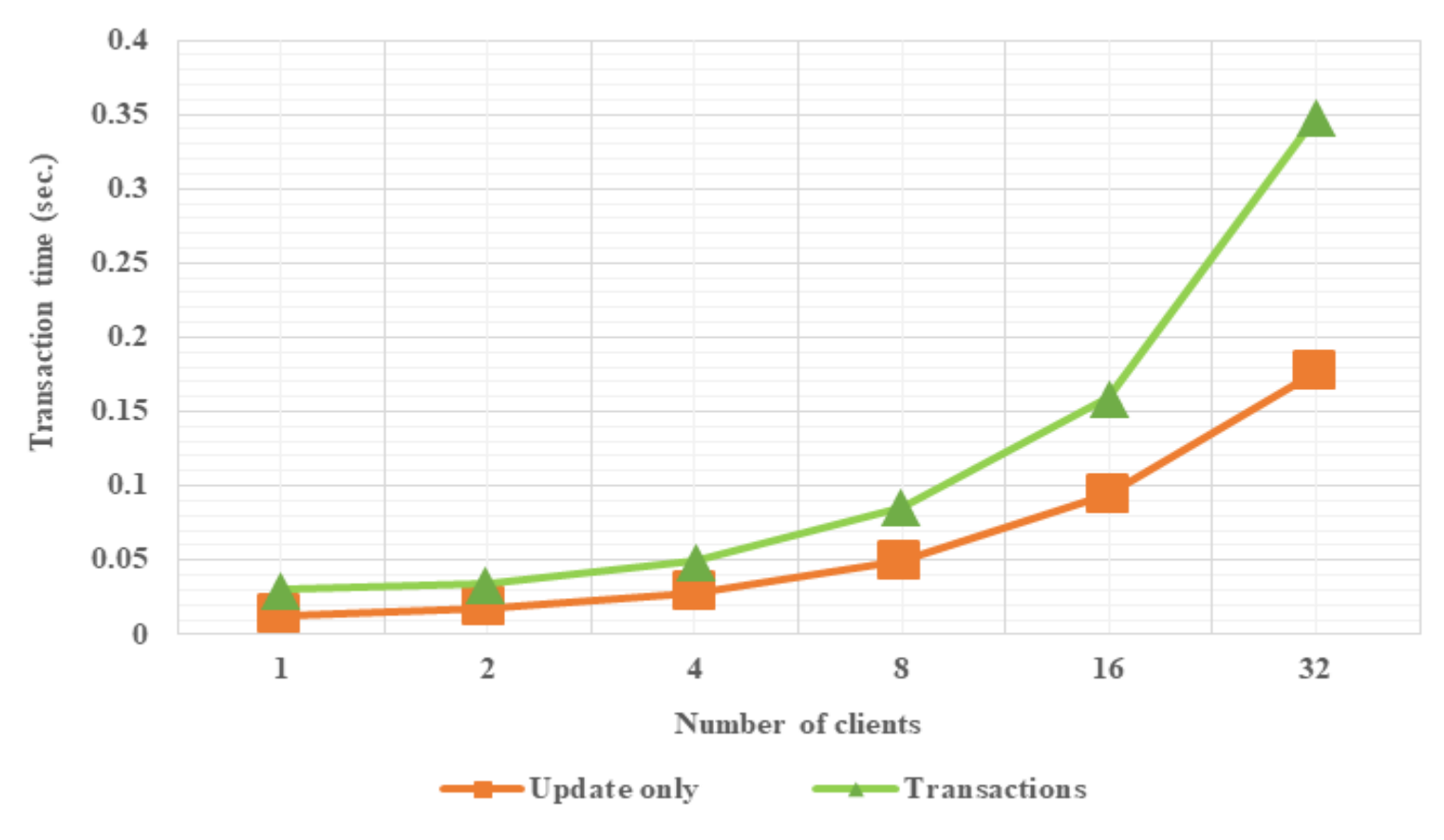
Efficiency of Transaction and Update only (Non-Transaction) operations



Results show that transactions incur extra processing time compared to simple 'Update only' (or Write) operations but transactions enforce 100% consistency

Experiment 2 (Riak)

Efficiency of Transaction and Update only (Non-Transaction) operations



Again results show that transactions incur extra processing time compared to simple 'Update only' (or Write) operations but transactions enforce 100% consistency

Experiments 1 & 2 (Observations)

- The experiments provided some useful insights into the performance and level of consistency.
- Though transaction systems incur performance overhead they are crucial for NoSQL database applications that need strong consistency.
- Focusing only on performance and compromising on consistency can lead to serious issues
- One example is the Flexcoin (a Bitcoin exchange) that was closed down in March 2014 due to hacking. One of the issues was associated with the design of MongoDB which did not have sufficient mechanisms in place for ensuring consistency and concurrency.

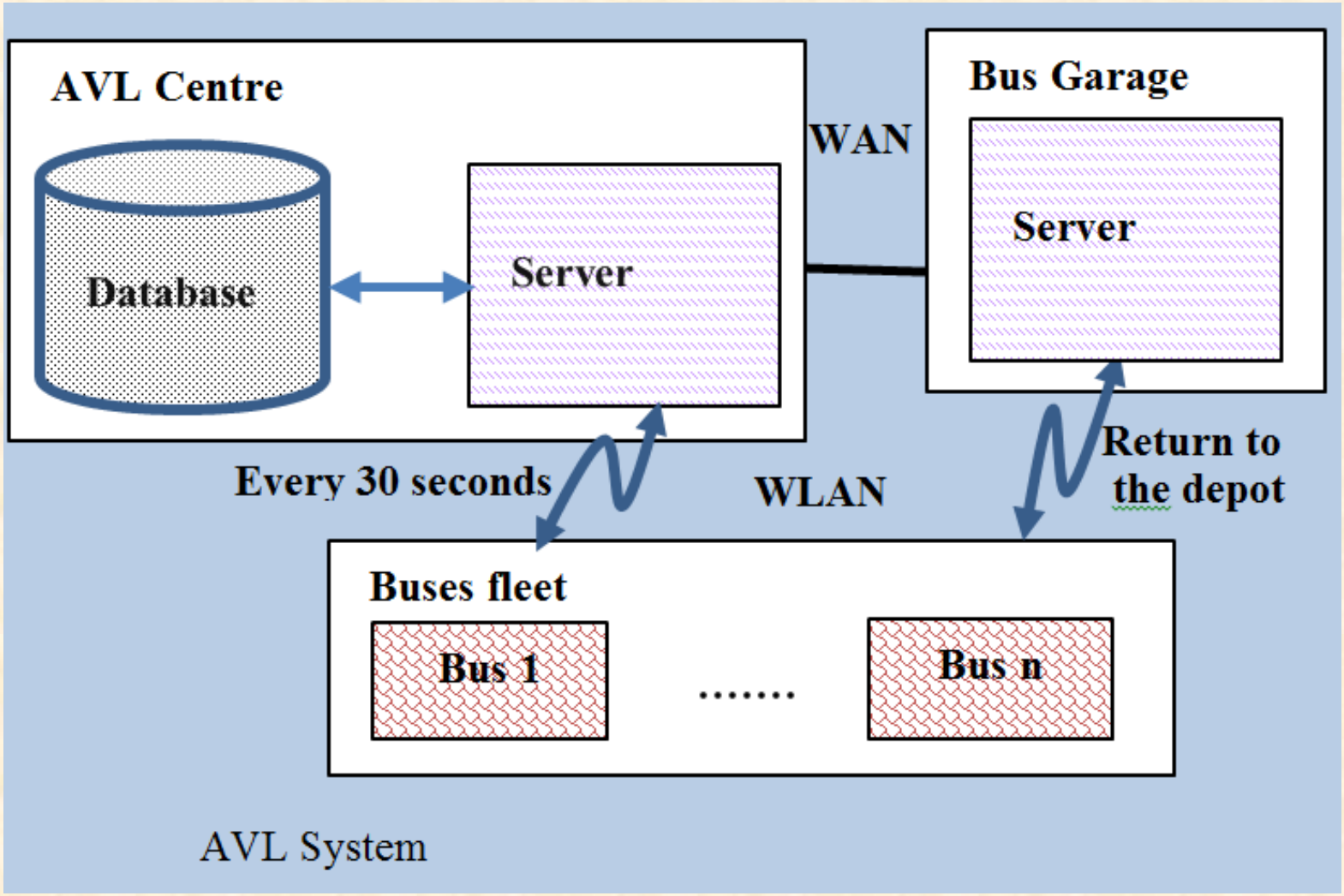
B) London Bus Services

- Some experiments were conducted using data from the London Datastore
- An open and big data about London buses transport fleet, and is provided by the Greater London Authority (in UK)

B) London Bus Services

- The application scenario is built around the iBus system; an Automatic Vehicle Location (AVL) System
- In the iBus system, a special unit OBU (on-board unit) is installed in every bus that stores information received from different sources (GPS, sensors, etc.) in a real time
- Such information is then sent from the OBU to a server in an AVL centre every 30 seconds approximately.
- The data is related to the location of the buses on the road and journey times such as estimated arrival times at destinations.

B) London Bus Services



Automatic Vehicle Location (AVL) System

B) London Bus Services

- The data is sent to and visualized along different electronic screens which are located on the buses and at the bus stops.
- The goal is to keep the passengers informed about the time for the arrival of a specific bus, etc.
- Such data is also published on Internet so that people can see real time bus information on websites.

iBus and AVL Data

- Using the iBus and AVL system, the operation of London buses generates large volume of data which accumulate to big data.
- This data is in line with the 3 or 5Vs characteristics of Big Data and is suitable for testing the proposed system.
- First, such data has a high **Velocity** given that it is sent to the (AVL) system and then distributed to different sources in a real time.
- This is also important for the analysis of the effects of data arrival (with high velocity) on the consistency of the database.

iBus and AVL Data

- Second, due to the nature of the information, different data structures have been used for its representation, thus fulfilling the **Variety** feature of big data.
- Third, the operation of the high number of bus lines and the enormous amount of bus stops generate a large amount of data; **Volume** of big data.
- There are over 19,000 bus stops, 700 bus routes and 8,000 buses.
- The system provides bus arrival information for a period of time of 30 minutes, including destinations, and data refresh every 30 seconds at source.

Experiments – London Bus Service

- Experiments were conducted using data from one of the bus lines with connections in Waterloo (London)
- i.e., bus number 176 from bus stop ‘Tottenham Court Road’ to another bus stop ‘Penge/Pawleyne Arms’.
- It consists of 54 bus stops between the origin and the destination.
- Some of the sample bus stops are shown in the figure.

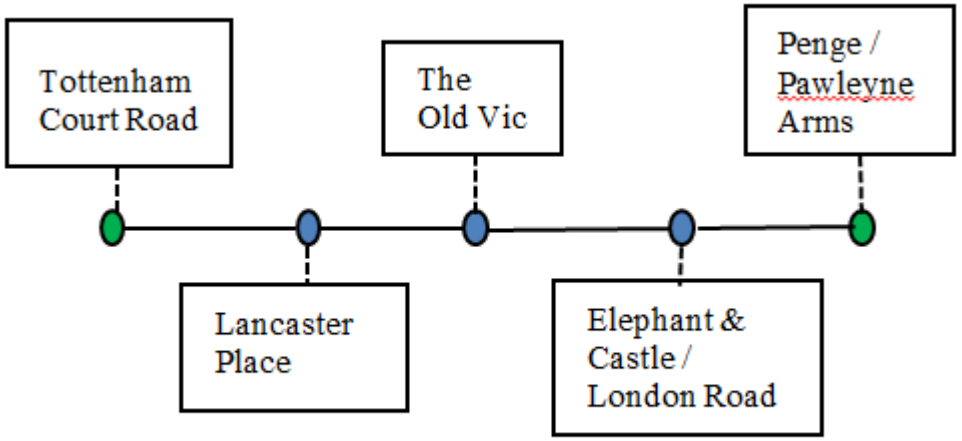
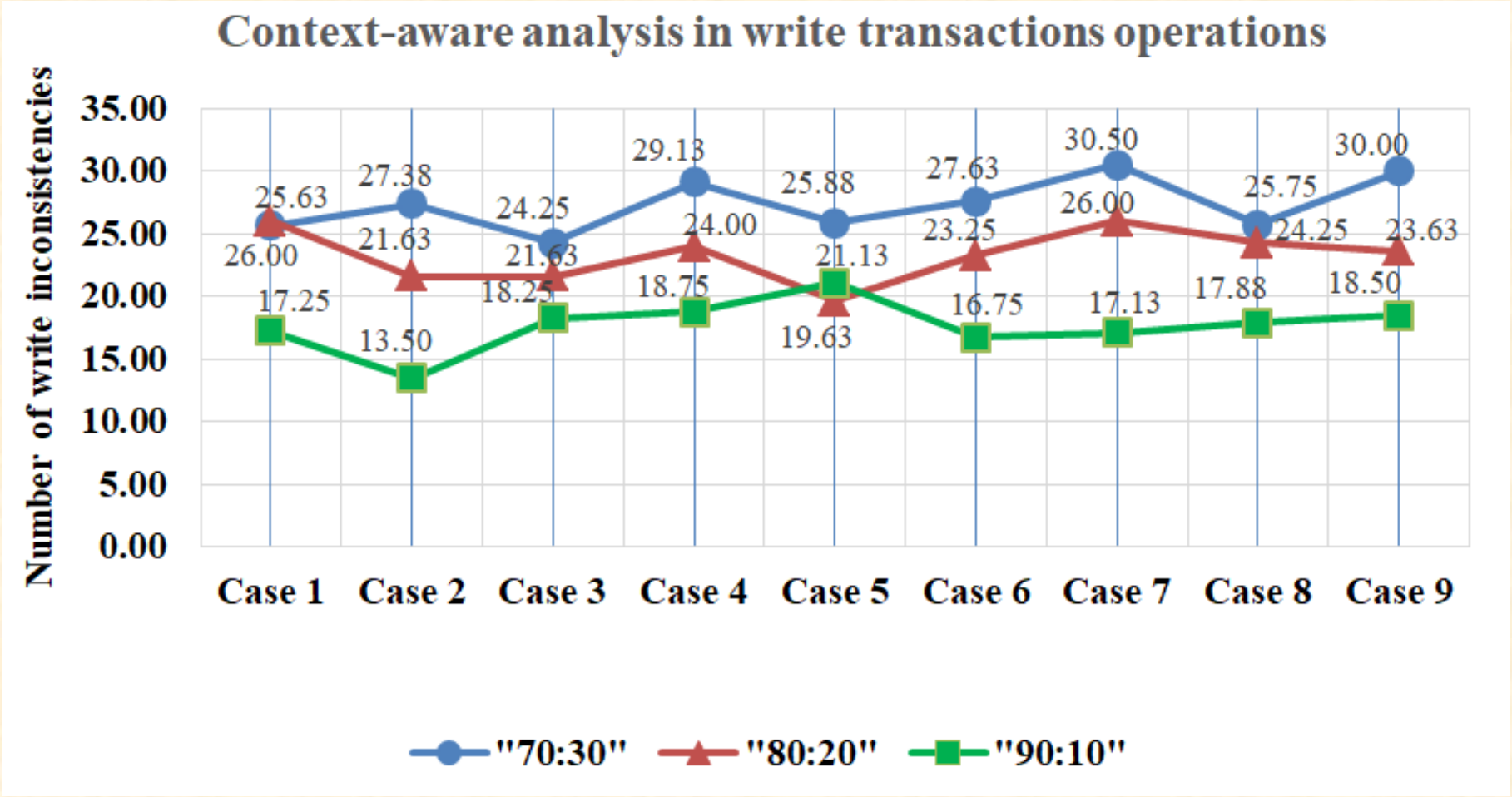


Figure A snippet from the bus line 176 in London city.

Experiment



Ratio of reads to writes operations, for example:
70:30 (70 reads and 30 writes)

Experiments (Observations)

- We measured the influence of two options by taking into account interval of time between write operations (velocity of data updates) and the level of consistency.
- Results show the number of inconsistencies in read and write operations is generally high for different levels of consistency (Case 1 to Case 9), and for different read to write ratio (“70:30”, “80:20” and “90:10”).
- The velocity of write operations does not allow the database to reach a consistent state. It will have an impact on the system results, which will be propagated to the electronic panels for their visualization (screen on bus stops, bus inforation on websites, etc)

Experiments (Observations)

- Our findings are that the velocity of write operations need to be controlled to keep the database in a consistent state
- There should be a trade-off between the interval of time for write operations (velocity) and the number of transactions that read the data
- But this will also have impact on the efficiency and availability of data?
- We have not evaluated the impact on efficiency or availability of data.

Research Issues

- To enhance data consistency in NoSQL Big Data systems so that they can be used in applications that need strict consistency such as
  - Banking transactions
  - Online gaming – consistent data about players and status of game
  - Online shopping (e-commerce, etc) – for example, consistency and availability of shopping carts data.

Research Issues

How to prevent customers from placing order (add to cart) for same product (last remaining products), if the quantity of products is not sufficient

- **Consider the following example where two customers concurrently place order to buy watches – update shared data.**
- **Number of available watches (Quantity) = 5**

Quantity = 5

Watches

Add 4 watches to shopping cart

Customer 1

Add 2 watches to shopping cart

Customer 2

Research Issues

- To minimize the impact of enforcing consistency on efficiency and availability of Big Data
- Other major issues include:
  - Security
  - Trust
  - Privacy
- Ethical aspects of storing and using Big Data
- Enhance users' control over (their own) Big Data

Publications and Sources

- M. T. González-Aparicio, M. Younas, J. Tuya, R. Casado: Testing of transactional services in NoSQL key-value databases. Future Generation Comp. Syst. (2018)
- A. Ogunyadeka, M. Younas, H. Zhu, A. Aldea: A Multi-key Transactions Model for NoSQL Cloud Database Systems. 2nd IEEE International Conference on Big Data Computing Service and Applications, BigDataService 2016, Oxford, United Kingdom, March 29 - April 1, 2016.
- M. T. González-Aparicio, M. Younas, J. Tuya, R. Casado: A New Model for Testing CRUD Operations in a NoSQL Database. 30th IEEE International Conference on Advanced Information Networking and Applications, AINA 2016, Crans-Montana, Switzerland, March, 2016.
- M. T. González-Aparicio, A. Ogunyadeka, M. Younas, J. Tuya, R. Casado: Transaction processing in consistency-aware user's applications deployed on NoSQL databases. Human Centric Computing & Information Sciences (2017)
- Casado R, Younas M (2015) Emerging trends and technologies in big data processing. Concurr. Comput. 27(8):2078–2091
- Some images are taken from Google Images