Providing Wellness Services using Real Time Analytics

Daniel Araújo[1], André Pimenta[1], Davide Carneiro[1,2], and Paulo Novais[1]

[1] Algoritmi Center/Department of Informatics, Minho University
Braga, Portugal
[2] CIICESI, ESTGF, Polytechnic Institute of Porto
Felgueiras, Portugal
daniel.ara.6@gmail.com,{apimenta,dcarneiro,pjon}@di.uminho.pt

Abstract. Data has increased in a large scale in various fields leading to the coin of the term Big Data. Big data is mainly used to describe enormous datasets that typically includes masses of unstructured data that may need real-time analysis. As human behaviour and personality can be captured through human-computer interaction a massive opportunity opens for providing wellness services. Through the use of interaction data, behavioral biometrics can be obtained. The usage of biometrics has increased due to several factors such as the rise of power and availability of computational power. One of the challenges in this kind of approaches has to do with handling the acquired data. The growing volumes, variety and velocity brings challenges in the tasks of pre-processing, storage and providing analytics. In this sense, the problem can be framed as a Big Data problem. In this work it is intended to provide an architecture that accommodates the data pipeline of data generated by human-computer interaction, providing real time data analytics on behavioral biometrics.

Keywords: Real-Time Analytics, Big Data, NoSQL Databases, Behavioral Biometrics

1 Introduction

A large amount of data is created every day by the interactions of billions of people with computers, wearable devices, GPS devices, smart phones, and medical devices. In a broad range of application areas, data is being collected at unprecedented scale [1].

Not only the volume of data is growing, but also the variety (range of data types and sources) and velocity (speed of data in and out) of data being collected and stored. These are known as the 3V's of Big data, enumerated in a research report published by Gartner [2].

Big Data refers to things one can do at a large scale that cannot be done at a smaller one: to extract new insights or create new forms of value, in ways that change markets, organizations, the relationship between citizens and governments, and more [3]. Despite still being somewhat an abstract concept it can

be clearly said that Big Data encomprises the a new generation of technologies and architectures, designed to economically extract value from very large volumes of a wide variety of data, by enabling the high-velocity capture, discovery and analysis [4].

1.1 Human-Computer Interaction

With the advent of the Internet of Things the number of devices that are connected is increasing every. Consequently, the number of interactions that can generate data is growing as well. Humans tend to show their personality or their state through their actions, even in an unconscious way. Facial expressions and body language, for example, have been known as a gateway for feelings that result in intentions. The resultant actions can be traced to a certain behavior. Therefore, it is safe to assume that a human behavior can be outlined even if the person does not want to explicitly share that information.

The interaction with computers and other technological devices can provide dataset containing records that are relative to unconscious behaviors. The rhythm at which a person types on a keyboard or the movement of the mouse changes when the individual becomes fatigued or under severe stress, as it was established in [5, 6]. Moreover, the interaction with smartphones can also provide analogous patterns from diverse sources including a touchscreen (that provides information about touches, their intensities, their area or their duration), gyroscopes, accelerometers, among others.

2 NoSQL for Real Time Analytics

2.1 Real Time Analytics

In the spectrum of analytics two extremes can be identified. On one end of the spectrum there is batch analytical applications, which are used for complex, long-running analyses. Generally, these have slower response times (hours or days) and lower requirements for availability. Hadoop-based workloads are an example of batch analytical applications. On the other end of the spectrum sit real-time analytical applications. Real-time can be considered from the point of view of the data or from the point of view of the end-user. The earlier translates into the ability of processing data as it arrives, making it possible to aggregate data and extract trends about the actual situation of the system (streaming analytics). The former refers to the ability to process data with low latency (processing huge amount of data with the results being available for the end user almost in real-time) making it possible, for example, to to provide recommendations for an user on a website based on its history or to do unpredictable, ad hoc queries against large data sets (online analytics).

Regarding stream processing the main problems are related to: Sampling Filtering, Correlation, Estimating Cardinality, Estimating Quantiles, Estimating Moments, Finding Frequent Elements, Counting Inversions, Finding Subsequences, Path Analysis, Anomaly Detection Temporal Pattern Analysis, Data

Prediction, Clustering, Graph analysis, Basic Counting and Significant Counting. The main applications are A/B testing, set membership, fraud detection, network analysis, traffic analysis, web graph analysis, sensor networks and medical imaging ([7]).

According to [7] these are the most well-known streaming open source tools:

S4 Real-time analytics with a key-value based programming model and support for scheduling/message passing and fault tolerance.

Storm The most popular and widely adopted real-time analytics platform developed at Twitter.

Millwheel Google's proprietary real-time analytics framework that provides exact once semantics.

Samza Framework for topology-less real-time analytics that emphasizes sharing between groups.

Akka Toolkit for writing distributed, concurrent and fault tolerant applications.

Spark Does both offline and online analysis using the same code and same system.

Flink Fuses offline and online analysis using traditional RDBMS techniques.

Pulsar Does real-time analytics using SQL.

Heron Storm re-imagined with emphasis on higher scalability and better debuggability.

Online analytics, on the other hand, are designed to provide lighter-weight analytics very quickly. The requirements of this kind of analytics are low latency and high availability. In the Big Data era, OLAP (online analytical processing [8]) and traditional ETL processes are too expensive. Particularly, the heterogeneity of the data sources makes it difficult the definition of rigid schemas, making model-driven insight difficult hard. In this paradigm analytics are needed in near real time in order to support operational applications and their users. This includes applications from social networking news feeds to analytics, from real-time ad servers to complex CRM applications.

2.2 NoSQL - Not only SQL

Conventional relational databases have proven to be highly efficient, reliable and consistent in terms of storing and processing structured data [9]. However, regarding the 3 V's of big data the relational model has several shortcomings. Companies like Amazon, Facebook and Google started to work on their own data engines in order to deal with their Big Data pipeline, and this trend inspired other vendors and open source communities to do similarly for other use cases. As Stonebraker argues in [10] the main reasons to adopt NoSQL databases are performance (the ability to manage distributed data) and flexibility (to deal with semi-structured or unstructured data that may arise on the web) issues.

A mapping between Big Data characteristics (the 3V's) and NoSQL features can be established. NoSQL data stores can manage large volumes of data by enabling data partitioning across many storage nodes and virtual structures,

overcoming traditional infrastructure constraints (and ensuring basic availability). By compromising on ACID (Atomicity, Consistency, Isolation, Durability ensured by RDBMS in database transactions) properties NoSQL opens the way for less blocking between user queries. The alternative is the BASE system [11] that translates to basic availability, soft state and eventual consistency. By being basically available the system is guaranteed to be mostly available, in terms of the CAP theorem. Eventual consistency indicates that given that the system does not receive input during an interval of time, it will become consistent. The soft state propriety means that the system may change over time even without input.

According to [12], the key characteristics that generally are part of NoSQL systems are, the ability to horizontally scale CRUD operations throughput over many servers, the ability to replicate and to distribute (i.e.,partition or shard) data over many servers, a simple call level interface or protocol (in contrast to a SQL binding), a weaker concurrency model than the ACID transactions of most relational (SQL) database systems, efficient use of distributed indexes and RAM for data storage, and the ability to dynamically add new attributes to data records.

However, the systems differ in many points, as the funcionality ranges from a simple distributed hashing (as supported by memcached[3], an open source cache), to highly scalable partitioned tables (as supported by Google's BigTable [13]). NoSQL data stores come in many flavors, namely data models, and that permits to accommodate the data variety that is present in real problems.

Key-value Stores A Key-value DBMS can only perform two operations: store pairs of keys and values, and retrieve the stored values given a key. These kind of systems are suitable for applications with simple data models that require a resource-efficient data store like, for example, embedded systems or applications that require a high performance in-process database. Redis and Memcahced are popular examples of this kind of database.

Document-oriented Databases These kind of data stores are designed to store and manage documents. Typically, these documents are encoded in standard data exchange (such as XML, JSON, YAML, or BSON). These kind of stores allow nested documents or lists as values as well as scalar values, and the attribute names are dynamically defined for each document at runtime. A single column can hold hundreds of attributes (in an analogy to the relational model), and the number and type of attributes recorded can vary from row to row, since its schema free. Unlike key-value stores, these kind of stores allow the search on both keys and values, support complex keys and secondary indexes. MongoDB and CouchDB are DBMS that function in this paradigm.

Column-oriented Databases Column-oriented databases are the kind of data store that most resembles the relational model on a conceptual level. They retain notions of tables, rows and columns, creating the notion of a schema, explicit from the client's perspective. In this approach, rows are split across nodes through sharding on the primary key. They typically split by range rather

[3] `http://memcached.org`

than a hash function. This means that queries on ranges of values do not have to go to every node. Columns of a table are distributed over multiple nodes by using "column groups". Rows are grouped into collections (tables), and an individual row's attributes can be of any type. For applications that scan a few columns of many rows, they are more efficient, because this kind of operations lead to less loaded data than reading the whole row. Apache Cassandra and Apache HBase are examples of this kind of data store.

Graph-oriented Databases Graph databases are data stores that employ graph theory concepts. In this model, nodes are entities in the data domain and edges are the relationship between two entities. Nodes can have properties or attributes to describe them. These kind of systems are used for implementing graph data modeling requirements without the extra layer of abstraction for graph nodes and edges. This means less overhead for graph-related processing and more flexibility and performance. Neo4J is the most popular graph-oriented database.

Comparative Evaluation of NoSQL Databases As it was presented, there are several options when it comes the time to choose a NoSQL database, and the different categories and architectures serve different purposes. Although four categories were presented, only two of them are adequate for the purposes of this work. Regarding support for complex queries column-oriented and document-oriented data store systems are more adequate than key-value stores (e.g. simple hash tables) and graph databases (which are ideal for situations that are modeled as graph problems). Considering the last presented fact a comparison is presented at [14], where several DBMS are classified in a 5-point scale (Great, good, average, mediocre and bad) regarding a set of quality attributes.

3 Wellness services using Real Time Analytics

Gathering metrics on people's behaviours and providing tools for visualization, particularly real time analytics, enables decision making and data-driven actions concerning well being of individuals. The trend for data collection regarding sensing on humans is growing and the perspective is for this trend to keep strong, giving the expected growth of IoT (Internet of Things).

According to [5], by recording the data from the keyboard and mouse movements it is possible to metrics that enable the prediction of fatigue levels. The captured records contain fifteen values (represented as doubles) that are a result of applying data redundancy techniques (i.e. aggregation of collected data by calculating values such as mean and variance on the very frequently collected values) and additionally contain a timestamp. Therefore, each record needs (15 times 8 bytes, the MongoDB double size plus 8 bytes relative to the timestamp, and 8 bytes relative to two keys that refer the task and user) 136 bytes of storage space. These records are produced every five minutes for each user of the system. As a user is expected to be around eight hours per day (13056 bytes, 12.75 Kbytes) interacting with its desktop/laptop a prediction about the data volumes that need real time processing can be made (see figure 1).

Table 1. Data growth projections.

	1 user	**100 users**	**10000 users**	**1000000 users**
5 minutes	136 bytes	13.28 Kbs	1.297 Mbs	129.7 Mbs
1 day	12.75 Kbs	1.245 Mbs	124.5 Mbs	12.159 Gbs
1 week	89.25 Kbs	8.716 Mbs	871.6 Mbs	85.115
1 month	382.5 Kbs	37.354 Mbs	3.648 Gbs	364.8 Gbs
1 year	4.545 Mbs	454.5 Mbs	44.382 Gbs	4.438 Tbs

As it is shown in figure 1 the architecture of the desired system is divided in three major components. The raw data is generated in the devices, then pre-processed (by redundancy elimination) and stored locally whenever possible (as in smartphones and personal computers) in a SQLite database. Then data is synchronized with the web servers in the cloud. The target database is MongoDB (object-oriented DB). MongoDB[4] is a database that is half way between relational and non-relational systems. It provides indexes on collections, it is lockless and provides a query mechanism. MongoDB provides atomic operations on fields like relational systems MongoDB supports automatic sharding by distributing the load across many nodes with automatic failover and load balancing, on the other hand CouchDB achieves scalability through asynchronous replication. MongoDB supports replication with automatic failover and recovery. The data is stored in a binary JSON-like format called BSON that supports boolean, integer, float, date, string and binary types. The communication is made over a socket connection (in CouchDB it is made over an HTTP REST interface).

MongoDB is actually more than a data storage engine, as it also provides native data processing tools: MapReduce[5] and the Aggregation pipeline[6]. Both the aggregation pipeline and mapreduce can operate on a sharded collection (partitioned over many machines, horizontal scaling). These are powerful tools for performing analytics and statistical analysis in real-time, which is useful for ad-hoc querying, pre-aggregated reports, and more. MongoDB provides a rich set of aggregation operations that process data records and return computed results, using this operations in the data layer simplifies application code and limits resource requirements. The visualization layer (as an web app) is developed on Java technology and uses the D3 library for graphics and diagrams. Regarding fault tolerance MongoDB provides master-slave replication and replica sets[7]. Nowadays, replica sets are recommended for most use cases. The standard (and minimum) number of replicas in a set is three: one being the primary (the only one with writes allowed), and two secondaries (can become the primary in an election), since an odd number of members ensures that the replica set is always able to elect a primary. Another aspect in the data architecture that must be addressed is related to the storage engines. As the there is no longer

[4] `https://www.mongodb.com`
[5] `https://docs.mongodb.org/manual/core/map-reduce/`
[6] `https://docs.mongodb.org/manual/core/aggregation-pipeline/`
[7] `https://docs.mongodb.org/manual/replication/`

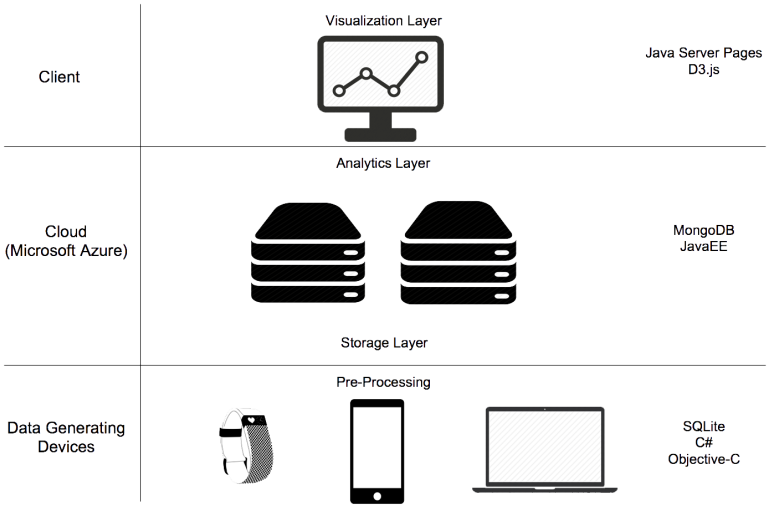

Fig. 1. Architecture of the real time analytics system.

a universal database storage technology capable of powering every type of application built by the business, MongoDB provides pluggable storage engines, namely WiredTiger and MMAPv1. Multiple storage engines can co-exist within a single MongoDB replica set, making it easy to evaluate and migrate engines. Running multiple storage engines within a replica set can also simplify the process of managing the data lifecycle. WiredTiger (default storage engine starting in MongoDB 3.2) will provide significant benefits in the areas of lower storage costs, greater hardware utilization, and more predictable performance [8] and, consequently should be used in this system.

4 Conclusions

In this work we presented an approach for handling the Big Data problems that may arise from a large scale wellness application. The rate of data arriving, the volume of the data and the need for real time aggregations (for data visualization) represent the data requirements. The main challenges are related to the databases architecture, particularly the storage and analytical layers. The database management system and the replica set architecture have been presented. The analytical engine is the Mongo Aggregation Framework, a feature of MongoDB that provides a pipeline for low level analytics operations. The deployment of this data architecture will allow us to test the ability of this kind of services to scale. The door for large scale wellness services is opened, and big data techniques appear as one of the most useful resources for this kind of

[8] `https://docs.mongodb.org/manual/core/storage-engines/`

services to succeed. By aligning this trend with the advances on machine learning distributed systems and Internet of things applications, a future for diverse wellness services can be sighted.

Acknowledgement

This work has been supported by COMPETE: POCI-01-0145-FEDER-007043 and FCT – Fundação para a Ciência e Tecnologia within the Project Scope: UID/CEC/00319/2013. The work of Davide Carneiro is supported by a Post-Doctoral Grant by FCT (SFRH/BPD/109070/2015).

References

1. Bertino, E., Bernstein, P., Agrawal, D., Davidson, S., Dayal, U., Franklin, M., Gehrke, J., Haas, L., Halevy, A., Han, J., et al.: Challenges and opportunities with big data. (2011)
2. Gartner: What is big data? `http://www.gartner.com/it-glossary/big-data` Accessed: 2015-12-20.
3. Mayer-Schönberger, V., Cukier, K.: Big data: A revolution that will transform how we live, work, and think. Houghton Mifflin Harcourt (2013)
4. Gantz, J., Reinsel, D.: Extracting value from chaos. IDC iview (1142) (2011) 9–10
5. Pimenta, A., Carneiro, D., Novais, P., Neves, J.: Monitoring mental fatigue through the analysis of keyboard and mouse interaction patterns. In: Hybrid Artificial Intelligent Systems. Springer (2013) 222–231
6. Carneiro, D., Castillo, J.C., Novais, P., Fernández-Caballero, A., Neves, J.: Multimodal behavioral analysis for non-invasive stress detection. Expert Systems with Applications **39**(18) (2012) 13376–13389
7. Kejariwal, A., Kulkarni, S., Ramasamy, K.: Real time analytics: algorithms and systems. Proceedings of the VLDB Endowment **8**(12) (2015) 2040–2041
8. Chaudhuri, S., Dayal, U.: An overview of data warehousing and olap technology. ACM Sigmod record **26**(1) (1997) 65–74
9. Khazaei, H., Fokaefs, M., Zareian, S., Beigi-Mohammadi, N., Ramprasad, B., Shtern, M., Gaikwad, P., Litoiu, M.: How do i choose the right nosql solution? a comprehensive theoretical and experimental survey. Submitted to Journal of Big Data and Information Analytics (BDIA) (2015)
10. Stonebraker, M.: Sql databases v. nosql databases. Communications of the ACM **53**(4) (2010) 10–11
11. Pritchett, D.: Base: An acid alternative. Queue **6**(3) (2008) 48–55
12. Cattell, R.: Scalable sql and nosql data stores. ACM SIGMOD Record **39**(4) (2011) 12–27
13. Chang, F., Dean, J., Ghemawat, S., Hsieh, W.C., Wallach, D.A., Burrows, M., Chandra, T., Fikes, A., Gruber, R.E.: Bigtable: A distributed storage system for structured data. ACM Transactions on Computer Systems (TOCS) **26**(2) (2008)
14. Lourenço, J.R., Cabral, B., Carreiro, P., Vieira, M., Bernardino, J.: Choosing the right nosql database for the job: a quality attribute evaluation. Journal of Big Data **2**(1) (2015) 1–26