Real Time Analytics for Characterizing the Computer User's State

Davide Carneiro[a,b], Daniel Araújo[b], André Pimenta[b], and Paulo Novais[b]

[a]CIICESI, ESTG, Polytechnic Institute of Porto, Felgueiras, Portugal
[b]Algoritmi Center/Department of Informatics, Minho University, Braga, Portugal
dcarneiro@estg.ipp.pt,pg25331@alunos.uminho.pt,apimenta@di.uminho.pt, pjon@di.uminho.pt

KEYWORD	ABSTRACT
Big Data; Distributed Computing; Behavioural Analysis; Keystroke Dynamics; Mouse Dynamics	*In the last years, the amount of devices that can be connected to a network grew significantly allowing to, among other tasks, collect data about the environment or the people in it in a non-intrusive way. This generated nowadays well-known topics such as Big Data or the Internet of Things. This also opened the door to the development of novel and interesting applications. In this paper we propose a distributed system for acquiring data about the users of technological devices in a non-intrusive way. We describe how this data can be collected and transformed to produce meaningful interaction features, that reveal the state of the individuals. We analyse the requirements of such a system, namely in terms of storage and speed, and describe three prototypes currently being used in three different domains of application.*

1. Introduction

We nowadays live in an information-rich society: we easily create, distribute, use and manipulate information in most of our daily activities. These activities include not only routine and trivial actions such as interacting with social networks but also higher-level social activities at economic, political or cultural levels. This explosion of information profoundly changes our society, providing new possibilities but also raising new challenges.

A significant amount of this information is created from our interaction with technological devices such as computers, wearables, GPS devices, smartphones, medical devices, among others. The immense possibilities created by such devices lead to the collection of data at an unprecedented scale (Bertino et al., 2011) even if not all of it is actually used. The advent of the Internet of Things contributes to this reality with the regular development of new devices, with novel functionalities. Consequently, the number of interactions that can generate data grows as well.

The volume of collected data is thus increasing, which poses a practical challenge. Other two relevant challenges include the variety of the data (with many different data types and sources) and the velocity at which data is collected, stored and accessed. Volume, Variety and Velocity are in fact known as the 3V's of Big Data, depicting three important dimensions whose expansion results challenging.

However, when the aim is to find hidden knowledge in sets of data, large datasets are often necessary. Indeed, the nowadays well-known concept f Big Data refers to things that can be done 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 (Mayer-Schönberger and Cukier, 2013). Despite still being a somewhat abstract concept it can be clearly said that Big Data encomprises 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 (Gantz and Reinsel, 2011).

This paper describes a system for the distributed acquisition of data. Specifically, the system was designed to acquire data non-intrusively from people's interaction with technological devices. This data describes the interaction behaviour, which may be influenced by aspects such as the task being carried out or the state of the user (e.g. stress, attention, performance). In fact, 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 behaviour. Therefore, it is safe to assume that a human behaviour 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 datasets containing records that relate 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 described in (Pimenta et al., 2013; Carneiro et al., 2012).

The interaction with smartphones can provide interesting information as well, from sources as diverse as touchscreens (that provide information about touches, their intensities, their area or their duration), gyroscopes, accelerometers, among others.

In this paper we analyse the requirements of such a system in terms of storage and velocity, especially when the number of users grow. We also describe real scenarios in which this system is being used, to characterize user behaviour in real-time.

The paper is structured as follows. Section 2 addresses real-time analytics and, in particular, the use of NoSQL for this purpose. Section 3 describes a distributed architecture designed for acquiring data describing people's interaction patterns with technological devices. This section also analyses the requirements of such a system and describes the features that can be extracted from it. Section 4 briefly presents three prototypes in three different fields and with different but complementary goals. Finally, Section 5 puts forward the concluding remarks.

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 ((Kejariwal et al., 2015)).

According to (Kejariwal et al., 2015) 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 (Chaudhuri and Dayal, 1997)) 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 (Khazaei et al., 2015). 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 (Stonebraker, 2010) 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 (Pritchett, 2008) 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 (Cattell, 2011), 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[1], an open source cache), to highly scalable partitioned tables (as supported by Google's BigTable (Chang et al., 2006)). 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.

---

[1] http://memcached.org

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 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 (Lourenço et al., 2015), 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).

As described in (Pimenta et al., 2013), by recording the data from the keyboard and mouse movements it is possible to calculate metrics that characterize the user's interaction patterns. 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. These metrics are further described in Section 3.1.

Each record thus needs 15 * 8 bytes (the MongoDB double size) plus 8 bytes for the timestamp and another 8 bytes for the two keys that refer the task and user. Thus amounts to a total of 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[2] 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[3] and the Aggregation pipeline[4]. 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 a 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[5]. 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 a universal database storage technology capable of powering every type of application built by the

---

[2] https://www.mongodb.com

[3] https://docs.mongodb.org/manual/core/map-reduce/

[4] https://docs.mongodb.org/manual/core/aggregation-pipeline/

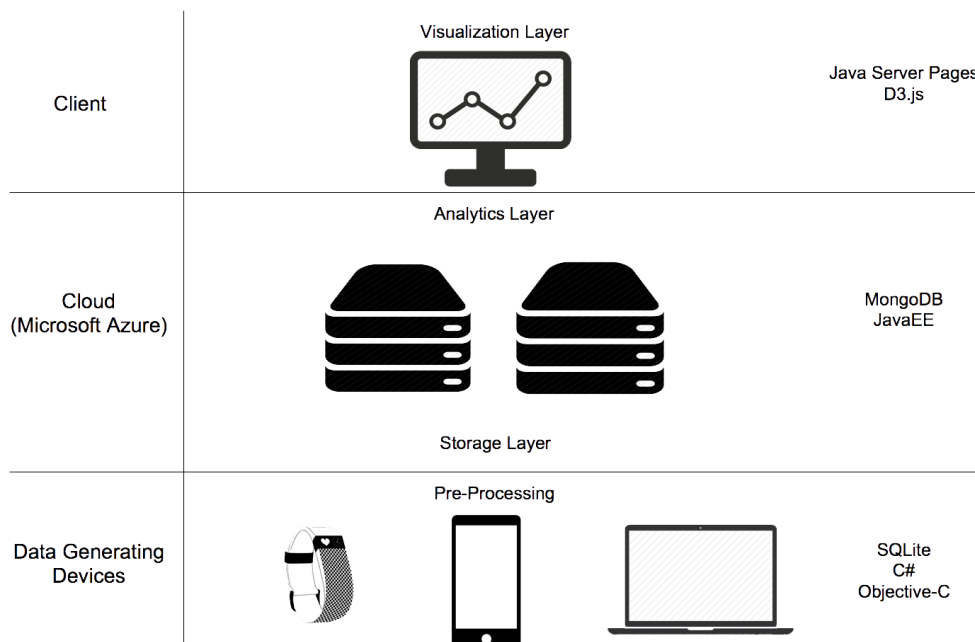[5] https://docs.mongodb.org/manual/replication/

Figure 1: Architecture of the real time analytics system.

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 [6] and, consequently should be used in this system.

3.1 Features

Each tuple of data stored in the database describes, as mentioned previously, the user, the type of task being carried out, a timestamp and the interaction of the user at the time. This interaction is characterized by 15 metrics based on a set of simple events captured form the use of the keyboard and the mouse. The following list describes the events captured by the applications in the data generating devices.

---

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

- MOV, timestamp, posX, posY

  An event describing the movement of the mouse, in a given time, to coordinates (posX, posY) in the screen;

- MOUSE_DOWN, timestamp, [Left|Right], posX, posY

  This event describes the first half of a click (when the mouse button is pressed down), in a given moment. It also describes which of the buttons was pressed (left or right) and the position of the mouse in that instant;

- MOUSE_UP, timestamp, [Left|Right], posX, posY

  An event similar to the previous one but describing the second part of the click, when the mouse button is released;

- MOUSE_WHEEL, timestamp, dif

  This event describes a mouse wheel scroll of amount dif, at a given time;

- KEY_DOWN, timestamp, key

  Identifies a given key from the keyboard being pressed down, at a given time;

- KEY_UP, timestamp, key

  Describes the release of a given key from the keyboard, at a given time.

The following example depicts a brief log that starts with some mouse movement (first two lines), contains a click with a little drag (lines 3-5) and ends with some more movement (last two lines).

```
MOV, 635296941683402953, 451, 195
MOV, 635296941684123025, 451, 197
MOUSE_DOWN, 635296941684443057, Left, 451, 199
MOV, 635296941685273140, 452, 200
MOUSE_UP, 635296941685283141, Left, 452, 200
MOV, 635296941685723185, 452, 203
MOV, 635296941685803193, 454, 205
```

From these events it is possible to extract the previously mentioned fifteen interaction features, twelve of them extracted from the mouse and the remaining three from the keyboard. These features are described next.

CLICK DURATION (CD)
UNITS - milliseconds
Measures the time span between two consecutive MOUSE_DOWN and MOUSE_UP events. The longer the clicks, the less efficient the interaction is.

TIME BETWEEN CLICKS (TBC)
UNITS - milliseconds

The time span between two consecutive MOUSE_UP and MOUSE_DOWN events, i.e., how long did it take the user to perform another click. Similarly to other features, we consider that a smaller time between clicks is representative of a faster working rhythm, thus increased performance.

TIME DOUBLE CLICK (TDC)
UNITS - milliseconds
The time span between two consecutive MOUSE_UP and MOUSE_DOWN events when smaller than 200 ms i.e., the duration of a double click. Similarly to other features, a shorter double click time represents increased interaction performance.

MOUSE VELOCITY (MV)
UNITS - pixels/milliseconds
The distance traveled by the mouse (in pixels) over the time (in milliseconds). The velocity is computed for each interval defined by two consecutive MOUSE_UP and MOUSE_DOWN events. Let us assume two consecutive MOUSE_UP and MOUSE_DOWN events, mup and mdo, respectively in the coordinates $(x1, y1)$ and $(x2, y2)$, that took place respectively in the instants $time_1$ and $time_2$. Let us also assume two vectors $posx$ and $posy$, of size n, holding the coordinates of the consecutive MOUSE_MOV events between mup and mdo. The velocity between the two clicks is given by $r\_dist/(time_2 - time_1)$, in which r_dist represents the distance traveled by the mouse and is given by equation 1. The relationship of this feature with performance is not as straightforward as in previous features. In fact, to a certain extent, a higher velocity may indicate increased performance. However, after a given threshold that is not true as higher velocity will result in less precision and control, which compromises performance. For that reason, we do not consider this feature for performance assessment. Nonetheless, it may still provide interesting insights about changes in the user's behavior.

MOUSE ACCELERATION (MA)
UNITS - pixels/milliseconds^2
The velocity of the mouse (in pixels/milliseconds) over the time (in milliseconds). A value of acceleration is computed for each interval defined by two consecutive MOUSE_UP and MOUSE_DOWN events, using the intervals and data computed for the Velocity. As with mouse velocity, the relationship of mouse acceleration with performance is not straightforward. For this reason, it is analyzed to assess changes in user behavior but is not considered for the purpose of estimating performance.

DISTANCE BETWEEN CLICKS (DBC)
UNITS - pixels
Represents the total distance traveled by the mouse between two consecutive clicks, i.e., between each two consecutive MOUSE_UP and MOUSE_DOWN events. Let us assume two consecutive MOUSE_UP and MOUSE_DOWN events, mup and mdo, respectively in the coordinates $(x1, y1)$ and $(x2, y2)$. Let us also assume two vectors $posx$ and $posy$, of size n, holding the coordinates of the consecutive MOUSE_MOV events between mup and mdo. The total distance traveled by the mouse is given by equation 1. A larger distance between clicks indicates a less efficient movement pattern.

$$r\_dist = \sum_{i=0}^{n-1} \sqrt{(posx_{i+1} - posx_i)^2 + (posy_{i+1} - posy_i)^2} \tag{1}$$

EXCESS OF DISTANCE (ED)
UNITS - pixels
This feature measures the excess of distance that the mouse traveled between each two consecutive MOUSE_UP and MOUSE_DOWN events. Let us assume two consecutive MOUSE_UP and MOUSE_DOWN events, mup and mdo, respectively in the coordinates $(x1, y1)$ and $(x2, y2)$. To compute this feature, first it is measured the distance in straight line between the coordinates of mup and mdo as $s\_dist = \sqrt{(x2 - x1)^2 + (y2 - y1)^2}$. Then, it is measured the distance actually traveled by the mouse by summing the distance between each two consecutive MOUSE_MV events. Let us assume two vectors $posx$ and $posy$, of size n, holding the coordinates of the consecutive MOUSE_MV events between mup and mdo. The distance actually traveled by the mouse, $real\_dist$ is given by equation 1. The Excess of Distance is given by $r\_dist - s\_dist$. A larger excess of distance indicates a lower performance of interaction.

AVERAGE EXCESS OF DISTANCE (AED)
UNITS - pixels
This feature measures the average excess of distance that the mouse traveled between each two consecutive MOUSE_UP and MOUSE_DOWN events. The average excess of distance between the two consecutive clicks (Figure 2 (a) is given by r_dist / s_dist, with r_dist and s_dist computed similarly to the ED feature. Once again, there is an inverse relationship between this feature and performance.

DISTANCE OF THE MOUSE TO THE STRAIGHT LINE (DMSL)
UNITS - pixels
This feature quantifies the sum of the successive distances of the mouse to the straight line defined by two consecutive clicks. Let us assume two consecutive MOUSE_UP and MOUSE_DOWN events, mup and mdo, respectively in the coordinates $(x1, y1)$ and $(x2, y2)$. Let us also assume two vectors $posx$ and $posy$, of size n, holding the coordinates of the consecutive MOUSE_MOV events between mup and mdo. The sum of the distances between each position and the straight line defined by the points $(x1, y1)$ and $(x2, y2)$ is given by 2, in which $ptLineDist$ returns the distance between the specified point and the closest point on the infinitely-extended line defined by $(x1, y1)$ and $(x2, y2)$. It results logical that the relationship of this feature with performance is inverse, i.e., the movement of the mouse is more efficient if it is moving closer to the line.

$$s\_dists = \sum_{i=0}^{n-1} ptLineDist(posx_i, posy_i) \tag{2}$$

AVERAGE DISTANCE OF THE MOUSE TO THE STRAIGHT LINE (ADMSL)
UNITS - pixels
This feature is similar to the previous one in the sense that it will compute the s_dists between two consecutive

MOUSE_UP and MOUSE_DOWN events, mup and mdo, according to equation 2. However, it returns its average rather than its sum during the path. The relationship of this feature with performance is also similar. The average distance of the mouse to the straight (Figure 3 (b)) line defined by two consecutive clicks is thus given by s_dists/n.

ABSOLUTE SUM OF ANGLES (ASA)
UNITS - degrees
This feature seeks to find how much the mouse "turned", independently of the direction to which it turned (Figure 3 (a)). In that sense, it is computed as the absolute of the value returned by function $degree(x1, y1, x2, y2, x3, y3)$, as depicted in equation 3. An efficient mouse movement is characterized by lines of movement that are almost straight. Therefore, higher values of this feature indicate a less efficient movement.

$$rCls\_angle = \sum_{i=0}^{n-2} \mid degree(posx_i, posy_i, posx_{i+1}, posy_{i+1}, posx_{i+2}, posy_{i+2}) \mid \qquad (3)$$

SIGNED SUM OF ANGLES (SSA)
UNITS - degrees
This feature is very similar to the previously mentioned one, with the exception that it measures to which side the mouse turns more. A negative value indicates that, in the overall, the mouse turns more counter-clockwise, while a positive value indicates that the mouse turns more clockwise. Concerning the relationship of this feature with interaction performance, the same considerations of the previous feature apply, i.e., the higher the value the less efficient the movement.

TIME BETWEEN KEYS (TBK)
UNITS - milliseconds
The time span between two consecutive KEY_UP and KEY_DOWN events, i.e., how long did it take the user to press another key after releasing the previous one. The lower the value of this feature the higher the performance.

KEY DOWN TIME (KDT)
UNITS - milliseconds
The time span between two consecutive KEY_DOWN and KEY_UP events, i.e., for how long did the user press a given key. Intuitively, the longer the key press the lower the performance of the writing.

WRITING VELOCITY (WV)
UNITS - keys per minute
The time span between two consecutive KEY_UP and KEY_DOWN events, i.e., how long did it take the user to press another key. The lower the value of this feature the higher the performance.
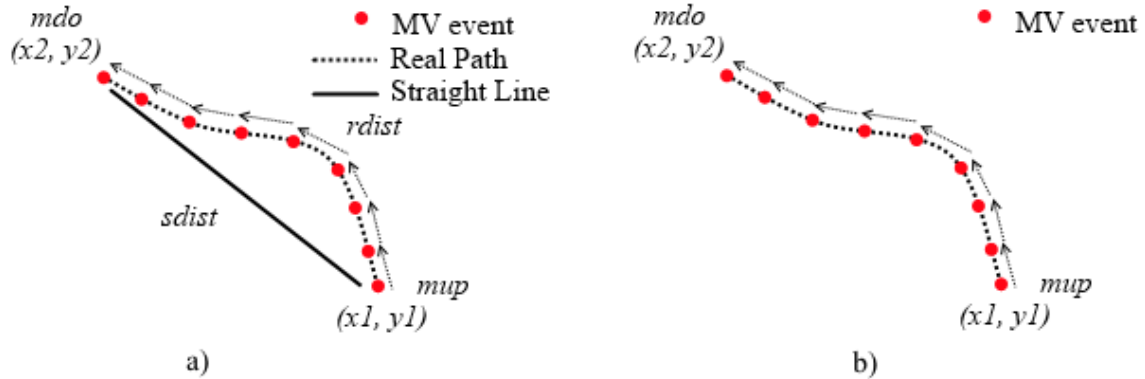
Figure 2: (a) A series of MOV events, between two consecutive clicks of the mouse. The difference between the shortest distance (sdist) and distance actually traveled by the mouse (rdist) is depicted. (b) The real distance traveled by the mouse between each two consecutive clicks is given by summing the distances between each two consecutive MOV events.
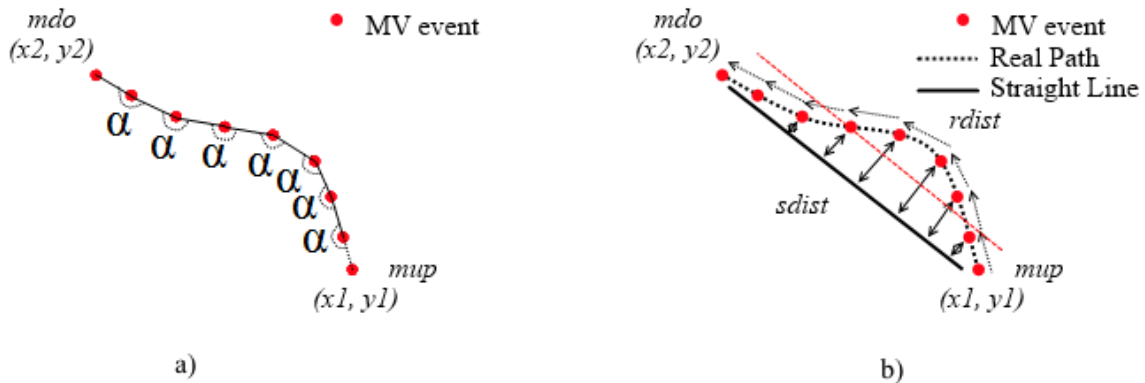

Figure 3: (a) The sum of the angles of the mouse's movement is given by summing all the angles between each two consecutive movement vectors. (b) The average distance at which the mouse is from the shortest line between two clicks is depicted by the straight dashed line.

4. Applications

The proposed system is being used in different real-life scenarios, each one with different goals. These scenarios are described individually in the following three subsections.

4.1 Stress Detection

Modern life can be often assume a frenetic rhythm, caused by competitiveness, social judgment, productivity demands, information overload and many other modern sources of pressure. This exerts a significant and constant pressure on individuals, driving them to a constant attempt to perform more and better. There are environments which constitute particularly "good" examples of this reality. The classroom, for one, is a milieu in which individuals, from early in their lives, are confronted with frequent evaluations of their performance and the pressure that stems from its impact on their future and from the social judgment of their peers (Dyrbye et al., 2006).

Higher education, in particular, is a period of the individual's educational path that is especially prone to result in added pressure (Goebert et al., 2009). It is so because it constitutes a transition period before students reach the working environment, combining the fears and the pressure of both environments. Students are subjected to increasing periods of work with a progressive focus on autonomy and continuous assessment as mandated by current educational policies. The increasing workload is perceived as stressful and commonly leads to mental disorders and perception that their cognitive performance is bellow their expected standards (Strous et al., 2012). This is corroborated by the high prevalence of anxiety disorders among higher education students.

Assessment is a fundamental phase in the training and certification process that a higher education student is submitted to. It is also one of the strongest stress factors due to the high-stake implications in the academic progress and self-perceived image. Stress is a risk factor for anxiety and may lead to worsening of performance in assessment tasks (McEwen, 2012; Soares et al., 2012).

The proposed system is being used in the Secondary School of Caldas das Taipas and in the School of Medicine of the University of Minho, both located in Northern Portugal. In the former, students are being followed during the whole term, which allows to analyze their evolution and, especially, to compare classes with different characteristics (e.g. assessment and regular class). In the latter, students are being monitored in assessment moments which, at this point in their lives, constitute a particularly stressful event.

Figure 4 details the evolution of the student's interaction patterns during the exam. In the three cases an increase in performance is visible through a decrease in the feature's values, i.e., a smaller click duration, for instance, shows a faster (thus more efficient) click.

Figure 5 shows, in more detail, one particular student, following the same improving trend during the exam. However, not all students behave alike. Moreover, this trend of improving performance may not happen in longer exams, in which the duration and the ongoing pressure wear the student out. Also, the same student may behave differently on different days, depending on many external factors.

For these reasons, it is important to follow students throughout their academic path. This system may provide important insights for teachers and other school staff (e.g. psychologists) on their mission to prepare future medical professional to efficiently deal with stress. Indeed, the aim of the approach is to identify those students
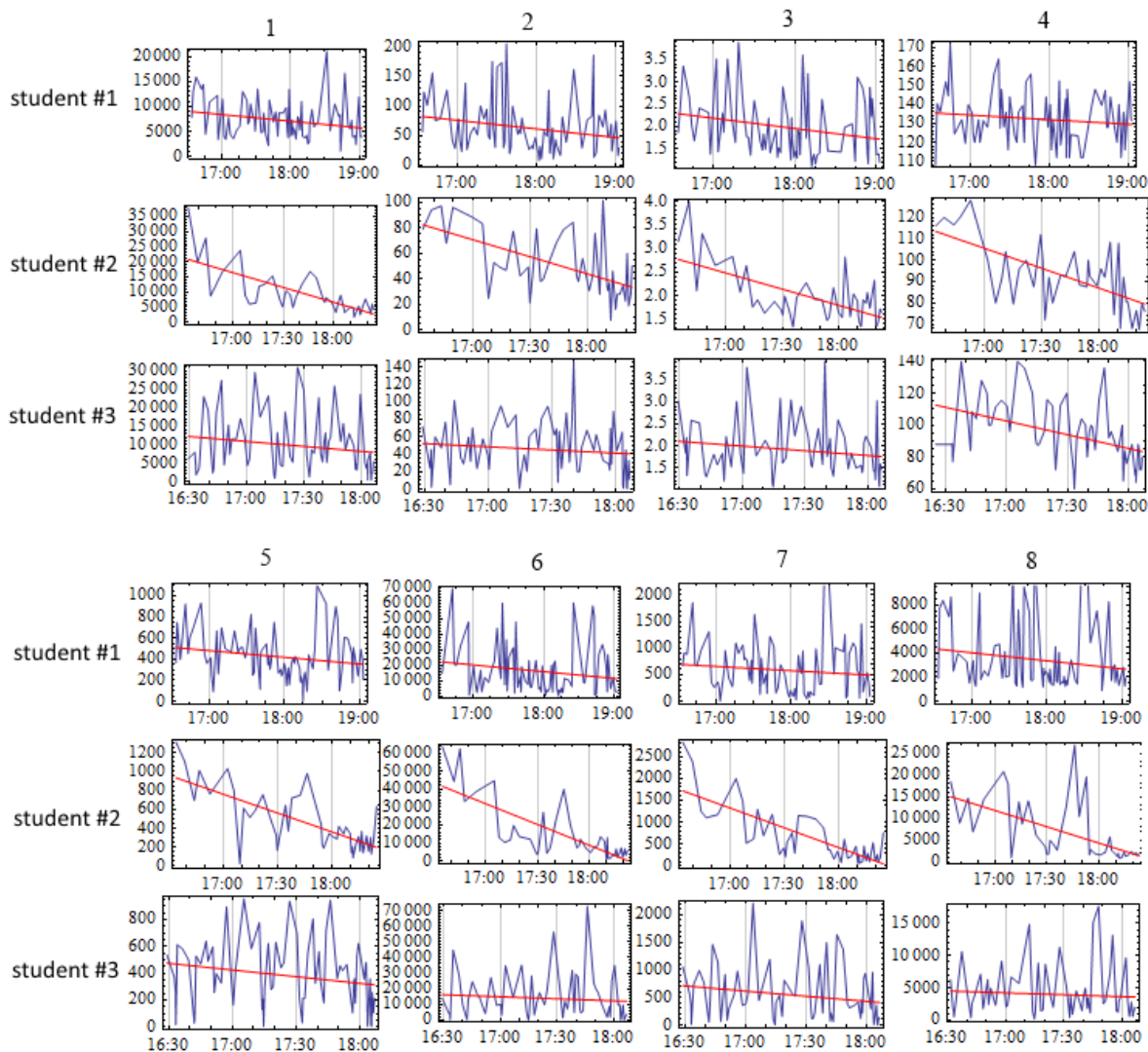
Figure 4: Time plot of the features for three arbitrary students. The negative correlation with time is visible for all features. Lines depict three different students. Columns depict the following eight features: 1 - ASA, 2 - ADMSL, 3 - AED, 4 - CD, 5 - DBC, 6 - DMSL, 7 - ED, 8 - TBC.

whose stress coping mechanisms are less efficient and provide guidance so as to prepare them to better cope with future stressful events, both academic or professional.
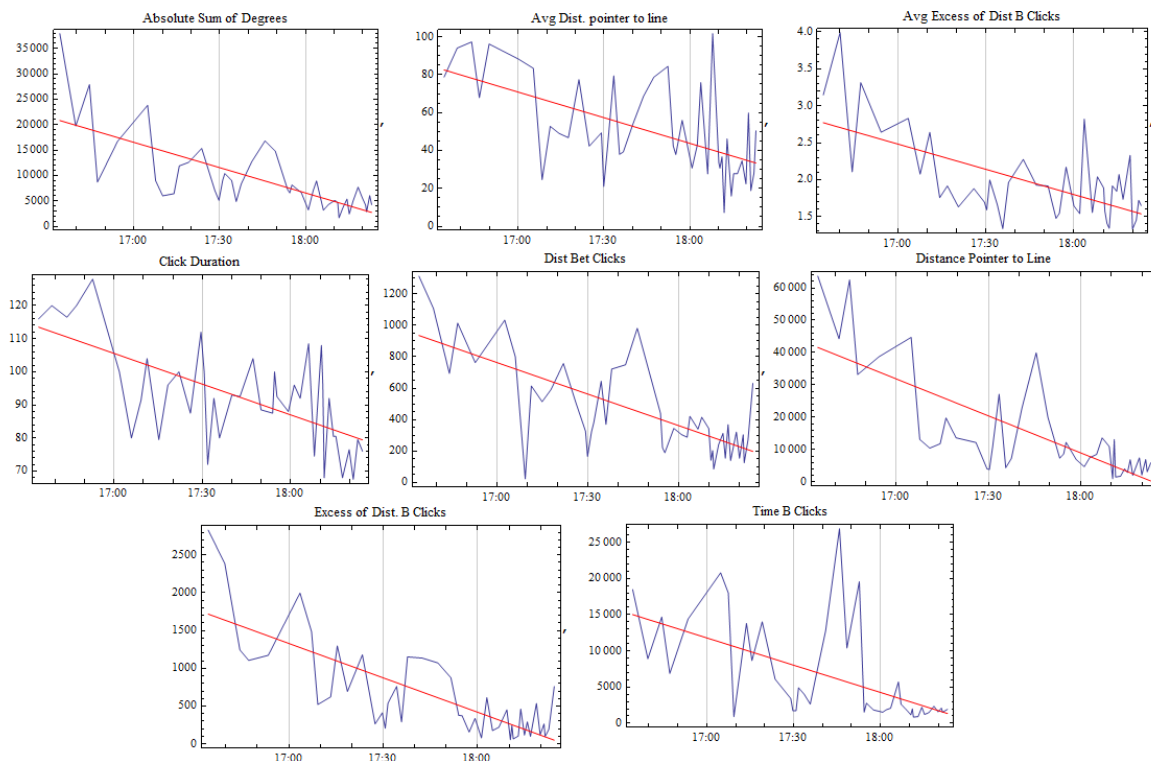

Figure 5: Plot of the data of all features of a particular student. Negative correlations are visible. Features, from bottom to top and left to right: 1 - ASA, 2 - ADMSL, 3 - AED, 4 - CD, 5 - DBC, 6 - DMSL, 7 - ED, 8 - TBC.

4.2 Performance Assessment

Our performance while at work, as well as in other domains, may be influenced by many different factors, including the circadian rhythm, our level of fatigue, our motivation or the type of task. Traditionally, performance is assessed periodically (e.g. monthly, weekly, daily) through productivity-based indicators. This kind of performance monitoring is often stressful for workers as it constitutes an ongoing source of pressure, which sometimes even decreases productivity (Aiello and Kolb, 1995).

There is thus the need to improve performance or productivity by other means that do not bring along such negative effects. To achieve this aim an environment was developed, based on the presented system, that takes as

input individuals' behavioral cues. Indeed, Humans tend to show their personality or their state through their actions, whether in a conscious or unconscious way. The way individuals interact with technological devices is influenced by such factors as well and can thus be analyzed to assess them. Some of these effects, analyzed in preliminary studies, show that our performance tends to decrease with the onset of mental fatigue (Pimenta et al., 2013) or that the way we touch or handle a smartphone changes with our level of stress (Carneiro et al., 2012).

This means that the performance of each individual in a group (and consequently of the group) can be assessed in real time and continuously, instead of being assessed on a regular basis as happens with traditional approaches. Moreover, workers are more likely to accept this kind of performance monitoring as opposed to traditional measures based on productivity indicators. The privacy of the worker is also safeguarded.

This approach also translates into a range of other advantages. Namely, it is non-invasive and non-intrusive as it requires no specific interactions by the workers. In that sense, the approach is also transparent.

The developed system allows a real-time analysis of the performance of the members of a team. The valuable information made available through this approach can be used by the organization leaders to improve their human resources management. specifically, leaders will know not only the state of each worker in real time (making it possible to advise a pause or to better distribute work) as well as the best working rhythms for each worker. Figure 6 shows some of the information compiled that details the evolution of the performance of an individual throughout the day.
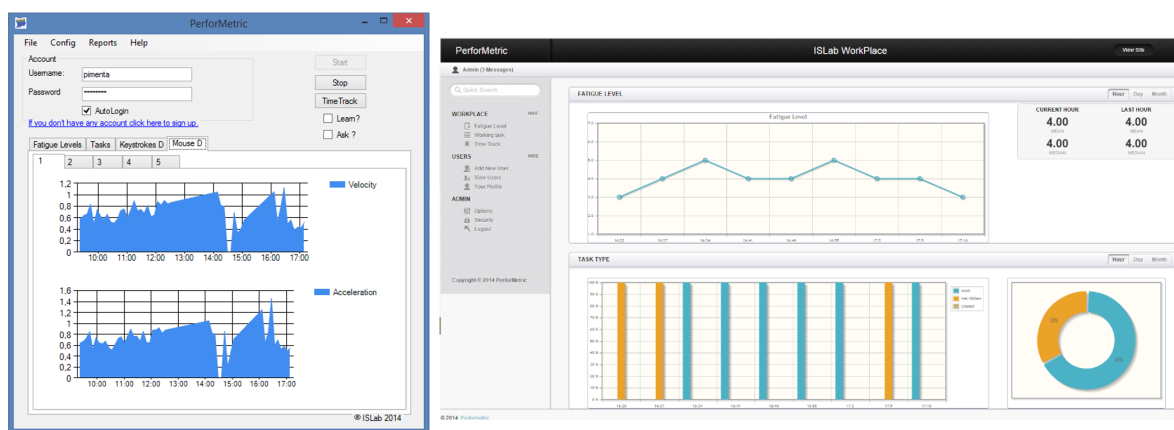

Figure 6: User Interface showing some of the raw data collected by the tool (left) and web interface showing compiled high-level information (right).

4.3 Attentive Behaviour

Attention is one of our most important mechanisms of our development as individuals influencing, among many others, learning, work quality or performance. Concerning the academic environment, attention during class is

recognizably one of the most important factors determining the success of learning (Hwang and Yang, 2009; Pimenta et al., 2015).

However, it is nowadays increasingly difficult for students to maintain a suitable degree of attention during classes. Among many other possible reasons, one of the most common is the availability of devices such as computers and smartphones and the facilitated access that they provide to distracting sources such as social networks or games.

Nonetheless, the same technology that distracts students can also be used to monitor their level of attention to task. It is, in our opinion, the responsibility of the school to develop means for monitoring and improving attention during class and thus improving learning success. The teacher, especially, may better adapt teaching strategies if she/he has access to knowledge regarding the evolution of attention during the class, taking into account specific events that happened during the class, the contents being taught or the time and duration of the class.

The proposed system is being used with several classes and teachers of the Secondary School of Caldas das Taipas, Portugal, to monitor the student's attention during class. Teachers establish tasks and applications to be used for those tasks. The system monitors application usage and computes a degree of attention based on the analysis of the amount of time spent by the students interacting with task-related applications. The level of activity is also quantified, based on the amount of interactions between user and computer per unit of time.

With this information classifiers can be trained to assign a level of attention to each student, in real time. Attention can be computed individually, so that the teacher can identify particularly problematic students. Asides from a summary of the student's level of attention (Figure 7) the teacher has also detailed access to an historic of the usage of each application throughout the class.

Alternatively, the teacher can also perform a group analysis, which will reveal the overall state of the class allowing to identify the onset of inattention in the group. This visual representation (Figure 8), achieved by calculating a moving average, may reveal interesting aspects such as contents that are less motivating for students, or times of the day or of the class in which attention tends to worsen. Moreover, it may also reveal how different classes react to the same contents. This is done in Figure 8, in which two different classes (vocational class and belles-letters class) show different degrees of attention to the same contents: the vocational class starts with a higher degree of attention with a tendency to decrease while the belles-letters class starts with a very low degree of attention, with a tendency to increase.

The proposed system is integrated in a TEL (Technology-enhanced Learning) environment, being used in the previously mentioned secondary school. The final goal is to dynamically provide recommendations to the teacher in order to improve student-teacher relationship and learning performance.

5. Conclusions

In this paper we described a distributed system for analysing people's interaction with technological devices, in real-time and in a non-intrusive way. The main aim of this system is to allow different kinds of team managers to take better decisions, through the provision of relevant knowledge about the people they are responsible for.

Evidently, such an approach poses challenges of its own, namely in what concerns the acquisition and storage

Date	% Work	% Others
📅 Mon 7 Dec 2015 08:58:06 GMT	87.4223	12.5777
📅 Mon 7 Dec 2015 09:03:24 GMT	34.9715	65.0285
📅 Mon 7 Dec 2015 09:16:05 GMT	73.1357	26.8643
📅 Mon 7 Dec 2015 09:21:07 GMT	77.2261	22.7739
📅 Mon 7 Dec 2015 09:26:38 GMT	93.2369	6.7631
📅 Mon 7 Dec 2015 09:33:26 GMT	100.	0.
📅 Mon 7 Dec 2015 09:43:10 GMT	100.	0.
📅 Mon 7 Dec 2015 09:48:10 GMT	99.3287	0.671286
📅 Mon 7 Dec 2015 09:53:23 GMT	100.	0.
📅 Mon 7 Dec 2015 09:56:10 GMT	100.	0.

Figure 7: Evolution of attention of a specific student, at regular intervals.
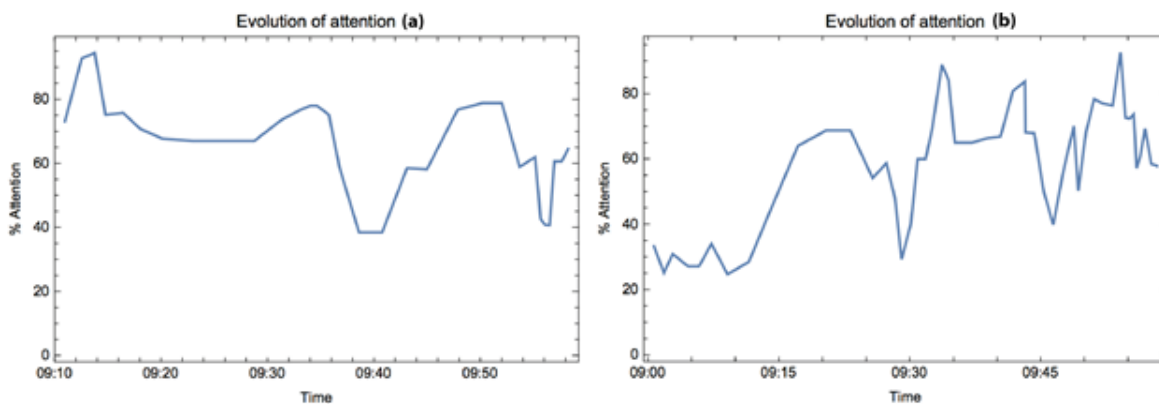

Figure 8: Temporal evolution in two class in the same situations. (a) vocational class 12I and (b) belles-letters class 12F.

of the data. Thus, in this paper we also presented an approach for handling the Big Data problems that may arise

from such large scale wellness applications. 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 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.

Specifically, we presented three domains of application in which we currently have deployed prototypes. The first is aimed at stress detection, especially in the context of academic assessment, and allows teachers to perceive how students are affected and cope with stress, allowing to define better stress coping strategies to prepare them not only for future academic assessments but also for future stressful events in their professional context. This is especially important in the case of medicine students, as is the case.

The second prototype is designed for the professional context, aiming at revealing how the performance of each worker varies throughout the day. It's main aim is to allow a more efficient management of human resources, while taking into account each worker's best working rhythm and state. Moreover, it does not rely on productivity indicators but rather on real-time performance assessment.

Finally, the third prototype aims to monitor task attention. While this specific prototype is being validated in an academic context, it could also be used in workplaces in which people interact with computers. The goal is, in any case, to perceive the attention and the motivation of each individual to perform the assigned task, in real-time.

Concluding, this kind of approach may reveal very interesting insights into people's behaviours, with benefits for both team managers and team members.

Acknowledgement

6. References

Aiello, J. R. and Kolb, K. J., 1995. Electronic performance monitoring and social context: impact on productivity and stress. *Journal of Applied Psychology*, 80(3):339.

Bertino, E., Bernstein, P., Agrawal, D., Davidson, S., Dayal, U., Franklin, M., Gehrke, J., Haas, L., Halevy, A., Han, J. et al., 2011. Challenges and Opportunities with Big Data. doi:10.14778/2367502.2367572.

Carneiro, D., Castillo, J. C., Novais, P., Fernández-Caballero, A., and Neves, J., 2012. Multimodal behavioral analysis for non-invasive stress detection. *Expert Systems with Applications*, 39(18):13376–13389. doi: 10.1016/j.eswa.2012.05.065.

Cattell, R., 2011. Scalable SQL and NoSQL data stores. *ACM SIGMOD Record*, 39(4):12–27. doi:10.1145/1978915.1978919.

Chang, F., Dean, J., Ghemawat, S., Hsieh, W. C., Wallach, D. A., Burrows, M., Chandra, T., Fikes, A., and Gruber, R. E., 2006. Bigtable: A Distributed Storage System for Structured Data. In *Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation - Volume 7*, OSDI '06, pages 15–15. USENIX Association, Berkeley, CA, USA.

Chaudhuri, S. and Dayal, U., 1997. An overview of data warehousing and OLAP technology. *ACM Sigmod record*, 26(1):65–74. doi:10.1145/248603.248616.

Dyrbye, L. N., Thomas, M. R., and Shanafelt, T. D., 2006. Systematic review of depression, anxiety, and other indicators of psychological distress among US and Canadian medical students. *Academic Medicine*, 81(4):354–373.

Gantz, J. and Reinsel, D., 2011. Extracting value from chaos. *IDC iview*, (1142):9–10.

Goebert, D., Thompson, D., Takeshita, J., Beach, C., Bryson, P., Ephgrave, K., Kent, A., Kunkel, M., Schechter, J., and Tate, J., 2009. Depressive symptoms in medical students and residents: a multischool study. *Academic Medicine*, 84(2):236–241. doi:10.1097/ACM.0b013e31819391bb.

Hwang, K.-A. and Yang, C.-H., 2009. Automated Inattention and Fatigue Detection System in Distance Education for Elementary School Students. *Educational Technology & Society*, 12(2):22–35.

Kejariwal, A., Kulkarni, S., and Ramasamy, K., 2015. Real time analytics: algorithms and systems. *Proceedings of the VLDB Endowment*, 8(12):2040–2041. doi:10.14778/2824032.2824132.

Khazaei, H., Fokaefs, M., Zareian, S., Beigi-Mohammadi, N., Ramprasad, B., Shtern, M., Gaikwad, P., and Litoiu, M., 2015. How do I choose the right NoSQL solution? A comprehensive theoretical and experimental survey. *Submitted to Journal of Big Data and Information Analytics (BDIA)*. doi:10.3934/bdia.2016004.

Lourenço, J. R., Cabral, B., Carreiro, P., Vieira, M., and Bernardino, J., 2015. Choosing the right NoSQL database for the job: a quality attribute evaluation. *Journal of Big Data*, 2(1):1–26. doi:10.1186/s40537-015-0025-0.

Mayer-Schönberger, V. and Cukier, K., 2013. *Big data: A revolution that will transform how we live, work, and think*. Houghton Mifflin Harcourt. ISBN 978-0544227750.

McEwen, B. S., 2012. Brain on stress: how the social environment gets under the skin. *Proceedings of the National Academy of Sciences*, 109(Supplement 2):17180–17185. doi:10.1073/pnas.1121254109.

Pimenta, A., Carneiro, D., Novais, P., and Neves, J., 2013. Monitoring mental fatigue through the analysis of keyboard and mouse interaction patterns. In *Hybrid Artificial Intelligent Systems*, pages 222–231. Springer. doi:10.1007/978-3-642-40846-5_23.

Pimenta, A., Gonçalves, S., Carneiro, D., Fde-Riverola, F., Neves, J., and Novais, P., 2015. Mental workload management as a tool in e-learning scenarios. In *Pervasive and Embedded Computing and Communication Systems (PECCS), 2015 International Conference on*, pages 25–32. SCITEPRESS. doi:10.5220/0005237700250032.

Pritchett, D., 2008. Base: An acid alternative. *Queue*, 6(3):48–55.

Soares, J. M., Sampaio, A., Ferreira, L. M., Santos, N., Marques, F., Palha, J. A., Cerqueira, J., and Sousa, N., 2012. Stress-induced changes in human decision-making are reversible. *Translational psychiatry*, 2(7):e131. doi:10.1038/tp.2012.59.

Stonebraker, M., 2010. SQL databases v. NoSQL databases. *Communications of the ACM*, 53(4):10–11.

doi:10.1145/1721654.1721659.

Strous, R. D., Shoenfeld, N., Lehman, A., Wolf, A., Snyder, L., and Barzilai, O., 2012. Medical students'
self-report of mental health conditions. *International journal of medical education*, 3:1. doi:10.5116/ijme.
4ed1.d1e0.