Journal of Ambient Intelligence and Humanized Computing
https://doi.org/10.1007/s12652-018-0845-9

ORIGINAL RESEARCH

# Cloud services integration for farm animals' behavior studies based on smartphones as activity sensors

**Olivier Debauche[1,2] · Saïd Mahmoudi[1] · Andriamasinoro Lalaina Herinaina Andriamandroso[2,3] · Pierre Manneback[1] · Jérôme Bindelle[3] · Frédéric Lebeau[2]**

Received: 15 February 2018 / Accepted: 15 April 2018


**Abstract**
Smartphones, particularly iPhone, can be relevant instruments for researchers in animal behavior because they are readily available on the planet, contain many sensors and require no hardware development. They are equipped with high performance Inertial Measurement Units (IMU) and absolute positioning systems analyzing users' movements, but they can easily be diverted to analyze likewise the behaviors of domestic animals such as cattle. The study of animal behavior using smartphones requires the storage of many high frequency variables from a large number of individuals and their processing through various relevant variables combinations for modeling and decision-making. Transferring, storing, treating and sharing such an amount of data is a big challenge. In this paper, a lambda cloud architecture innovatively coupled to a scientific sharing platform used to archive, and process high-frequency data are proposed to integrate future developments of the Internet of Things applied to the monitoring of domestic animals. An application to the study of cattle behavior on pasture based on the data recorded with the IMU of iPhone 4s is exemplified. Performances comparison between iPhone 4s and iPhone 5s is also achieved. The package comes also with a web interface to encode the actual behavior observed on videos and to synchronize observations with the sensor signals. Finally, the use of Edge computing on the iPhone reduced by 43.5% on average the size of the raw data by eliminating redundancies. The limitation of the number of digits on individual variable can reduce data redundancy up to 98.5%.

**Keywords** Animals' behavior · Smart agriculture · IMU · iPhone · Lambda architecture · Precision livestock farming

## 1 Introduction

The use of sensors in agriculture, particularly in livestock farming is becoming widespread, especially in dairy cattle operations. Among the different parameters that can be monitored on the animals themselves, behavior is probably the most critical as it provides essential information on their health or reproductive status. Also, when no particular health or reproduction-related issue is at stake, it allows understanding how well an animal is performing in the farming environment. The feeding behavior is such a key feature. Three main components are required to analyze the behavior of animals: (1) the location obtained by radio frequency triangulation or by Global Positioning System (GPS), (2) the low frequency component of behavior as posture of the animal (e.g.: position of the head, tilt of the neck, etc.), and (3) the high frequency component of behavior (e.g. movement of the jaws) (Andriamandroso 2016). Recently, the use of smartphones, particularly iPhone, was suggested for

Data from this paper were partially presented and published in the proceedings of the 14th International Conference on Mobile Systems and Pervasive Computing (MobiSPC 2017) (Debauche et al. 2017).

✉ Olivier Debauche
Olivier.debauche@umons.ac.be

[1] Computer Science Unit, Faculty of Engineering, University of Mons, 20 Place du Parc, 7000 Mons, Belgium

[2] Biosystems Dynamics and Exchanges Axis, Biosystem Engineering Department, ULg-Gembloux Agro-Bio Tech, University of Liège, Passage des Déportés 2, 5030 Gembloux, Belgium

[3] TERRA Teaching and Research Centre, AgricultureIsLife/EnvironmentIsLife and Precision Livestock and Nutrition, AgroBioChem, Gembloux Agro-Bio Tech, University of Liège, Passage des Déportés 2, 5030 Gembloux, Belgium

Published online: 14 May 2018

this purpose (Andriamasinoro et al. 2015) as they are readily available on the planet, contain the relevant sensors and require no hardware development. They are equipped with high performance Inertial Measurement Units (IMU) and absolute positioning systems that can easily be diverted to investigate the feeding behaviors of cows under a range of environmental conditions. Such researches aiming to the development of Precision Livestock Farming (PLF) applications, i.e. shifting from the management of a herd to the individual management of the animals in the herd, require the collection of many data in real or near real time and the setting up of a dedicated computer infrastructure. This infrastructure, in addition to data collection, should allow researchers to share their datasets and models. Most methods suggested, to treat the data collected to analyze the feeding behavior of cows, use automatically calibrated classification models based on complex mathematical calculations (Andriamasinoro et al. 2015). Although they reach accuracies as high as 90%, these methods are of limited use outside the context they have been developed for and the "black-box" approach they use does not allow collective improvement of the classification algorithms.

The collection of data from a large number of cows of different kinds, reared in different environments around the world is a corner stone in the development of new models and their validation on large data sets. Building such a large set of data will open new fields of research. Hence, a new open-source observer-based classification algorithm has been suggested based on I-phones IMU that can be used and shared by various researchers around the globe. To fulfill this goal, the experimental data must be processed and verified in order to guarantee on one hand its consistency; and on other hand, the structuration of data to facilitate the exchange between research groups. Using data collectively will enable the development and the validation of new models from larger datasets and provide new research opportunities for animal feeding behavior and health as well as pasture management. This will also pave the way to PLF by allowing parameter identification or combination and the sampling frequencies that are required to accurately detect specific behaviors, and the development of specific, accurate and reliable connected sensors. Indeed, the Internet of The Things (IoT) will offer tremendous opportunities in PLF by making it possible to know at any moment the health status of the animals and to detect problems before they become worse. From massively collected data, the most appropriated variables and the frequency of sampling adapted for each variable must be identified in the aim to optimize the quantity of data to collect and store on the device. The accuracy (pertinent number of digits) needed for each variable must also be evaluated. Moreover, the redundancy of data in the local storage must also be eliminated in order to limit de size of data to store without loss. Thus, using connected sensors in association with cloud computing technologies for research and later routinely in the farms poses a triple challenge. Firstly, at the level of the sensors, data must be collected at a high-frequency (up to 100 Hz) and processed to eliminate redundancy. But the compression still must be reversible with ideally no loss of data to allow the preservation of the full extent of the original data to enable the use of all any recorded signals in the further improvement of the behavior classification algorithms or in the development of new applications. For these raisons data size reduction is essential to reduce bandwidth requirements for transmission and improve battery life. Sensors are generally powered by external batteries whose main sources of consumption are the network transmissions. Moreover, in order to allow farmers to benefit in the future from these new technologies, the development of a new set of microcontrollers sensors will be necessary. Such sensors will be optimized based on the results of each type of collective research that will generate essential information like what is the best frequency to identify behavior with high accuracy. New sensors need also to be optimized to reduce power consumption through local processing of information and a limit transmission of information through the network. The second challenge lies on the storage and the processing of large amounts of data per animal per day (from several Tera to several Peta bytes) arriving at high speed at the cloud level. Finally, the third challenge is the matching of data with complementary data such as those provided by milking robots, environmental data, etc. in order to refine the models and support the elaboration of decision-support tool to help farmers to optimize dairy cattle operation and improve efficiency (Frost et al. 1997). No forgetting that data related to breeding conditions, animal performances and health may be sensitive and require protective measures, by anonymizing and controlling their use.

In this paper, we present a chain of tools fulfilling the above-mentioned criteria for data related to cattle behavior measured by means of the IMU signals of iPhone worn by the animals on halters (Andriamandroso et al. 2017). We propose and describe a new infrastructure allowing to collect, store, treat and share information between scientists. The sharing of important amount of data is important to create more robust models and re-validate existing models. The proposed lambda architecture brings benefits in storage, real-time processing and abilities for large scale data storage and analytics.

## 2 Optimizing smartphones as behavior sensors

Different types of sensors are used to monitor the behavior of Humans and animals such as microphones, pressure sensors, electromyography, location sensors and accelerometers (Andriamandroso et al. 2016). Acceleration sensors have

**Table 1** List of signals captured by IMU of iPhone 4s/5s using sensor data application

| Sensors | Measured signals | Unit |
|---|---|---|
| Accelerometer | Acceleration on x, y, z | g |
| Gyroscope | Euler angles (pitch, roll, yaw) | Radian |
| | Attitude quaternion on *x, y, z* | Radian |
| | Rotation matrix (3×3) | |
| | Gravitational component of acceleration | g |
| | User component of acceleration | g |
| | Rotation rate | rad $s^{-1}$ |
| Magnetometer | Magnetic data | μT |
| | Magnetic and true heading | ° |
| Location | Latitude and longitude | ° |
| | Altitude and accuracies | m |
| | Course | m |
| | Speed | m $s^{-1}$ |
| | Proximity sensor | [0,1] |

also been successfully used to recognize human behavior (Chikhaoui et al. 2017). Similarly, using a GPS and an accelerometer implanted on the neck of a cow allowed to reach 90% accuracy in behavior classification (González et al. 2015). As stated before, smartphones are generally equipped with an Inertial Measurement Units (IMU) which contains motion and location sensors able to record signals at high rate (Andriamandroso et al. 2017). They are widely available, easy to use and they save long hardware developments. IMU generally contains a 3-axis accelerometer, a 3-axis gyroscope and in recent version a magnetometer (digital triaxial compass) (see Table 1). The accelerometer is used to measure inertial acceleration. The gyroscope measures angular rotation. The magnetometer improves the precision of the gyroscopic measurements by correcting the drift of the magnetic pole. In this field, several works have been published using the IMU of iPhone. iPhone have been used to measure human posture and movement for upper arm in Yang et al. (2017b), human body position (Milani 2014), and sports monitoring (McNab et al. 2011; Rowlands and James 2011). Smartphones have also been used to detect falls of elderly people (Miccuci 2017).

## 2.1 iPhone and amounts of data

In the framework of the platform that is developed in this manuscript, factory-calibrated IMU in an iPhone 4s/5s mounted on a halter were used as sensors and an appropriate behavior classification algorithm was developed (Andriamandroso et al. 2017).

Various raw and combined signals are measured by the iPhone and sampled at the frequency of 100 Hz (obtained with iOS operating system). Those signals are recovered using the Data Sensor application v1.26[1] that is installed on the phone. This application stores data locally in csv files or use UDP protocol over IEEE 802.11 g to stream data to the gateway. This protocol is acceptable for short range such as an experimental pasture or farm, reliable connections with no expected loss of information packets with direct wireless connection ad hoc (Rowlands and James 2011).

## 2.2 Testing replicability

The precision of both iPhone was evaluated by attaching an iPhone 4s with an iPhone 5s using elastics. The two-solidarized iPhone were placed on an animal in order to compare possible variations in the IMU measurements. The accuracy of the accelerometer, gyroscope and location sensor was evaluated with a displacements table and a checker. These instruments allow respectively mastered movements and 3D - acceleration and 3D - rotation.

## 2.3 Testing battery life

The autonomy of the battery of new iPhone 4s and 5s was evaluated at different sampling frequencies: 1, 2, 3, 5, 10, 20, 30, 50, 100 Hz with all the 41 parameters including 15 truly measured parameters and 26 calculated ones and replicated 5 times. Those older models were tested because they are available at a low price on the second-hand market making them a readily available tool for researchers. The lowest frequencies (1–5 Hz) should cover most kinds of movements imprinted by an animal. The frequencies 10, 20, 30 and 50 Hz are a tenfold increase in previous sampling rates allowing a good representation of phenomenon as used in most animal behavior applications although in theory according Shannon/Nyquist the double of the frequency of the studied phenomenon should be enough. Finally, 100 Hz frequency is the maximum rate supported by the device. In a second time, the same measurements were redone with measured parameters only and replicated 3 times. The replication of the measurements allowed to evaluate standard deviation.

## 2.4 Testing data compression

An application in Xamarin[2] was developed to measure the compressibility of data by reducing the precision. In Sensor Data, all raw data are logged with a decimal precision of 6 digits. The compression of raw data has been performed in two wats: (1) by eliminating redundancies, i.e. replacement of redundant data by a time interval during which the value

[1] edited by Wavefront Labs and available in the Apple App store: https://itunes.apple.com/us/app/sensor-data/id397619802?mt=8.

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

**Table 2** Compression rate obtained for each category of data collected on 24 h

| Type of data | Variable number | Data treated | Data size (Mb) | Comp rate [%] |
|---|---|---|---|---|
| Acceleration in the X, Y, Z axis | 3 | 25,920,000 | 98.87 | 4.26 |
| Euler angles of the device | 3 | 25,920,000 | 98.87 | 24.13 |
| Attitude quaternion | 4 | 34,560,000 | 131.84 | 24.13 |
| Rotation matrix (3 × 3) | 9 | 77,760,000 | 296.63 | 24.13 |
| Gravitational component of 3D acceleration | 3 | 25,920,000 | 98.87 | 24.13 |
| User acceleration component of 3D acceleration | 3 | 25,920,000 | 98.87 | 24.13 |
| Rotation rate | 3 | 25,920,000 | 98.87 | 24.13 |
| Magnetic and true heading | 3 | 25,920,000 | 98.87 | 0.01 |
| Magnetometer data | 3 | 25,920,000 | 98.87 | 60.79 |
| Position (latitude, longitude) | 3 | 25,920,000 | 98.87 | 0.01 |
| Course and speed | 2 | 17,280,000 | 95.92 | 99.75 |
| Altitude | 1 | 8,640,000 | 32.96 | 99.99 |

remains constant was applied to preserve data integrity, and (2) by truncating data to 3, 4 and 5 decimal digits. For the 3 above-mentioned tests, results were obtained for 24 h of data collected locally and stored on the iPhone in offline mode and post treated on a Laptop with Intel core i7-6700HQ processor 2.6 Ghz with 16 GB RAM. Computing time related to 24 h of data (1,879,596 KB) is 72.29 s.

### 2.5 Results

Both iPhone 4s and 5s solidarized and placed on animal gave similar results in terms of accelerometric, gyroscopic and magnetic measurements. These measurements were confirmed by imposing controlled movements and vibrations on both iPhone. The both iPhone measured correctly controlled movements with a precision of $10^{-3}$.

Table 2 shows the different compression rates obtained on raw data by redundancy elimination without any loss of data.

Acceleration, magnetic/true heading and position data are weakly compressible by removal of redundancies as opposed to magnetometer data, course and speed, altitude that are extremely compressible. Other data have the same compressing rate of 24.13% because they are linked data or calculated valued.

The compression of the data allowed a reduction of the bandwidth consumed for their transmission to the gateway by 42% on average. The transmission of the float data at 100 Hz corresponds to 16,000 bytes of data. By eliminating redundancies, the amount of data transmitted every second was reduced to 9290 bytes per second.

The truncation of raw data respectively to 5, 4 and 3 digits and the elimination of redundancies reduces up to 44.5% on average the size of data to store. The small decrease on average in the size of data with 6, 5, 4 and decimals can be explained by the combination of parameters inside groups that limits the possibilities of compression.

Table 3 shows the mean autonomy time of batteries for two models of iPhone obtained with different sampling frequency for 15 and 41 parameters, respectively. The 15 parameters are those actually measured by the IMU. The 41 parameters are measured and calculated parameters on basis of the 15 measured parameters.

Sensor Data can be used to collect 41 parameters from the IMU of the iPhone at a frequency up to 100 Hz. However, it is not possible to switch off the screen during the acquisition phase, autonomy of the iPhone. The use of alternative software such as Power Sense which operates in the background allowing putting the screen to sleep will undoubtedly significantly increase the acquisition time of the data.

The evolution of the compression rate with a reduction in the number of decimals is not sensible when variables are treated in groups as opposed to individual data. (Table 4). The combination of parameters inside groups limits indeed the possibilities of compression. The compressibility of individual parameters is much more variable because it depends on one hand on the precision of sensors that are contained in the IMU, and on the other hand of the activity of the animal. Furthermore, the acquisition rates of sensors are different and therefore the variability in data also. For example, positioning systems update information every second while acceleration sensors display a potential new value every 0.01 s.

## 3 Novel architectures options for cloud storage platforms

Once reduced in volume, the data collected on the phones must be stored and treated appropriately on the cloud.

Large-scale data collection and sharing requires indeed a cloud storage platform to standardize, store and exchange data. Several options are suggested in the literature to fulfill

**Table 3** Autonomy in hours for different sampling frequency of all 41 parameters are measured and calculated parameters on basis of the 15 measured parameters

| Frequency (Hz) | 41 parameters | | 15 parameters | |
|---|---|---|---|---|
| | iPhone 4s | iPhone 5s | iPhone 4s | iPhone 5s |
| 1 | 5.47 (±1.72) | 7.68 (±1.83) | 5.10 (±1.90) | 7.95 (±1.89) |
| 2 | 6.37 (±2.20) | 10.30 (±2.29) | 7.50 (±3.16) | 9.47 (±2.38) |
| 3 | 7.21 (±2.62) | 9.16 (±1.86) | 6.58 (±2.49) | 9.32 (±2.00) |
| 5 | 5.84 (±2.22) | 8.44 (±2.23) | 5.95 (±2.32) | 8.48 (±2.03) |
| 10 | 5.26 (±1.72) | 7.75 (±1.74) | 5.26 (±1.82) | 7.80 (±1.78) |
| 20 | 5.24 (±1.76) | 7.59 (±1.87) | 5.24 (±1.82) | 7.87 (±1.78) |
| 30 | 5.15 (±1.80) | 7.68 (±1.82) | 5.23 (±1.78) | 7.80 (±1.84) |
| 50 | 5.09 (±1.81) | 7.80 (±1.95) | 5.24 (±1.83) | 7.47 (±1.74) |
| 100 | 5.15 (±1.78) | 7.65 (±1.77) | 5.11 (±1.73) | 7.56 (±1.80) |

this goal such as Hadoop, Apache Spark, Apache Storm, Apache Spark Streaming, Apache Samza and Druid.

Hadoop is a highly available open-source software framework dedicated to store and provide access to large amounts of data. Hadoop is composed of a distributed file system (HDFS), an application framework (MapReduce) and a resource manager (YARN). However, it does not offer any performances guarantee on how quickly that data can be accessed. The performances decrease under heavy load. Furthermore, Hadoop is unable to provide the sub-second data ingestion latencies. Finally, it is not optimized to store and make data immediately readable (Yang et al. 2014, 2017a).

A solution to reduce disk latency is to keep in memory data to reuse for multiples tasks. Apache Spark processes a large amount of data with low latency and includes fault tolerance by introducing a novel resilient distributed dataset abstraction. However, data sharing application must be written in external storage, such as Apache Cassandra, Apache Hive, Apache Pig, Apache Hbase, Apache Chukwa, Amazon S3 and HDFS (Diáz et al. 2016).

Stream processing frameworks like Apache Storm, Apache Spark Streaming and Apache Samza[3] offer low-latency model to ingest and process stream at near real-time speed. Apache Samza is a distributed stream processing framework which treats stream coming from Apache Kafka which is a distributed streaming platform. Apache Hadoop YARN is used to provide fault tolerance. However, these stream processing frameworks generally do not provide the same guarantees as batch processing frameworks in matter of correctness (Yang et al. 2017a). Also, the processing may suffer from duplicated events and other problems of accuracy in data.

The speed of data availability depends of how data are stored in the database. Opensource Relational Data Management systems and NoSQL key/value stores are unable to provide a low latency data storing. Furthermore, it is also not possible to provide query platform for interactive applications[4]. Raw data must be transformed or cleaned before their use (Yang et al. 2017a). As consequence, the process of data loading and batch processing can take a long time (several hours).

Druid presented in Yang et al. 2014 is a distributed column-oriented fault-tolerant presenting real-time analytical data store. This platform powers high performance application with low query latencies. Druid is designed to solve problems around ingesting and exploring large quantities of times series data. The unit of storage in Druid is called "segment". Each segment is composed of 5–10 million times-stamped events that covers one period of time. Segments can be compressed by LZ4[5] by default or LZF[6] algorithm and can also be stored in a column orientation database. Druid cluster is composed of 4 kinds of nodes: real-time, historical, broker and coordinator.

Druid uses two external dependencies. The first one is MySQL, PostgreSQL or SqlServer database in order to store metadata of segments. The second is Zookeeper that monitors the four kinds of nodes present in the cluster. These four nodes coordinate, broke, store in real-time or archive data on a distributed storage system. Druid is able to import data from Kafka, Stream data or files data (TSV, CSV and Json). Druid can use local storage or external service to deep store old segments: Amazon S3, HDFS, Microsoft Azure, Google Cloud Storage and Apache Cassandra.

Numerous platforms for sharing scientific data exist such as Mezuri (Kipf et al. 2016), DHIS2 (Manya et al. 2012), Open Foris (Miceli et al. 2011), Conveyor (Linke et al.

[3] Apache Samza (2017) http://samza.apache.org/. Accessed 5 April 2017.

[4] Tschetter E (2011) Introducing druid: Real-time analytics at a billion rows per second. http://druid.io/blog/2011/04/30/introducing-druid.html, Accessed 4 April 2017.

[5] LZ4. http://www.lz4.org, Accessed 4 April 2017.

[6] Liblzf (2013) http://freecode.com/projects/liblzf. Accessed 4 April 2017.

**Table 4** Compression rate obtain after truncation of raw data

| Individual data | Compression rate [%] on individual data | | | | Mean compression rate [%] by group | | | |
|---|---|---|---|---|---|---|---|---|
| | 6 decimal | 5 decimal | 4 decimal | 3 decimal | 6 decimal | 5 decimal | 4 decimal | 3 decimal |
| Acceleration in the X axis | 46.02 | 46.02 | 83.56 | 96.16 | 4.26 | 4.26 | 4.26 | 4.27 |
| Acceleration in the Y axis | 55.81 | 55.81 | 86.93 | 97.64 | | | | |
| Acceleration in the Z axis | 61.76 | 61.76 | 86.93 | 92.77 | | | | |
| Euler angles of the device (Roll) | 30.38 | 61.28 | 90.44 | 98.49 | 24.13 | 24.13 | 24.13 | 24.13 |
| Euler angles of the device (Pitch) | 27.10 | 46.62 | 86.10 | 97.84 | | | | |
| Euler angles of the device (Yaw) | 24.64 | 28.93 | 57.16 | 93.75 | | | | |
| Attitude quaternion in the X axis | 27.85 | 51.73 | 91.22 | 98.97 | 24.13 | 24.13 | 24.13 | 24.47 |
| Attitude quaternion in the Y axis | 27.77 | 51.34 | 90.77 | 98.99 | | | | |
| Attitude quaternion in the Z axis | 26.06 | 39.83 | 81.40 | 98.01 | | | | |
| Attitude quaternion in the W axis | 25.94 | 39.54 | 81.10 | 98.01 | | | | |
| Rotation matrix (element 11) | 25.66 | 37.49 | 80.69 | 98.01 | 24.13 | 24.13 | 24.13 | 24.15 |
| Rotation matrix (element 12) | 25.73 | 37.94 | 80.90 | 98.01 | | | | |
| Rotation matrix (element 13) | 27.83 | 50.20 | 87.98 | 98.57 | | | | |
| Rotation matrix (element 21) | 26.12 | 40.43 | 81.29 | 98.01 | | | | |
| Rotation matrix (element 22) | 25.89 | 39.10 | 81.00 | 98.01 | | | | |
| Rotation matrix (element 23) | 30.52 | 61.69 | 90.68 | 98.60 | | | | |
| Rotation matrix (element 31) | 26.09 | 40.98 | 84.59 | 98.26 | | | | |
| Rotation matrix (element 32) | 26.11 | 40.98 | 84.72 | 98.31 | | | | |
| Rotation matrix (element 33) | 31.43 | 66.12 | 94.94 | 99.37 | | | | |
| 3D Gravitational acceleration (X) | 27.79 | 50.20 | 87.98 | 98.57 | 24.13 | 24.13 | 24.23 | 32.02 |
| 3D Gravitational acceleration (Y) | 30.49 | 40.43 | 90.68 | 98.60 | | | | |
| 3D Gravitational acceleration (Z) | 31.40 | 39.10 | 94.94 | 99.37 | | | | |
| 3D User acceleration (X) | 31.29 | 62.49 | 90.49 | 98.10 | 24.13 | 24.13 | 24.13 | 24.25 |
| 3D User acceleration (Y) | 29.62 | 58.60 | 90.09 | 98.19 | | | | |
| 3D User acceleration (Z) | 30.90 | 62.41 | 91.73 | 98.56 | | | | |
| Rotation rate in the X axis | 25.70 | 62.02 | 76.10 | 91.65 | 24.13 | 24.13 | 24.13 | 24.13 |
| Rotation rate in the Y axis | 25.62 | 36.76 | 73.97 | 89.75 | | | | |
| Rotation rate in the Z axis | 25.53 | 35.90 | 69.75 | 84.94 | | | | |
| Magnetic Heading | 68.40 | 68.43 | 68.63 | 70.87 | 0.01 | 0.01 | 0.01 | 0.01 |
| True Heading | 68.25 | 68.28 | 68.51 | 70.78 | | | | |
| Heading Accuracy | 100.00 | 100.00 | 100.00 | 100.00 | | | | |
| Magnetometer data in the X axis | 79.31 | 82.22 | 94.79 | 97.86 | 60.82 | 60.83 | 60.84 | 60.86 |
| Magnetometer data in the Z axis | 98.22 | 98.22 | 98.22 | 98.22 | | | | |
| Magnetometer data in the Z axis | 72.40 | 72.40 | 75.67 | 88.68 | | | | |
| Latitude | 99.93 | 99.99 | 100.00 | 100.00 | 99.85 | 99.94 | 99.99 | 99.99 |
| Longitude | 99.92 | 99.98 | 100.00 | 100.00 | | | | |
| Position Accuracy | 100.00 | 100.00 | 100.00 | 100.00 | | | | |
| Course | 99.96 | 99.96 | 99.96 | 99.96 | 99.75 | 99.75 | 99.75 | 99.75 |
| Speed | 99.85 | 99.85 | 99.85 | 99.87 | | | | |
| Altitude | 99.99 | 99.99 | 99.99 | 99.99 | 99.99 | 99.99 | 99.99 | 99.99 |

2011), Mobyle (N'eron et al. 2009), Taverna (Hull et al. 2006), Kepler (Altintas et al. 2004). Those systems authorize the provision of previously anonymized data and with the prior consent of the owner of the data. Many of these platforms are specific to a domain and/or only support certain programming languages (Kipf et al. 2016). Among this myriad of platforms, the GIFT-Cloud sharing medical imaging platform for the research proposes an exchange of data between producers of data and researchers. This platform provides encryption and an onsite anonymization before the storage of data, a direct web-based data access and a REST API for integrate tiers software (Doel et al. 2017). In the present case, the principles of GIFT-Cloud were adapted to the data coming from the Internet of Things. Moreover,

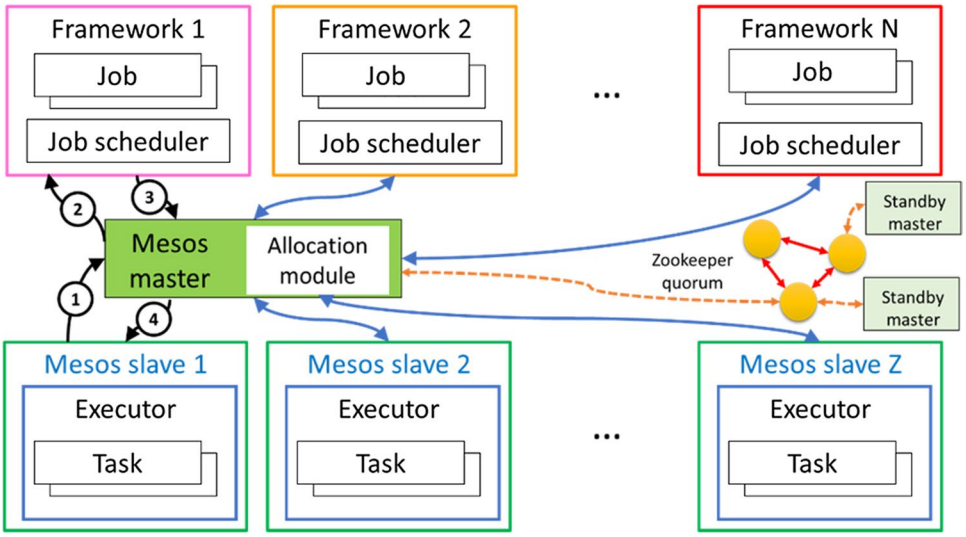


**Fig. 1** Apache mesos architecture

the hosting and the isolation of software and models which access directly and more rapidly to data were added to the platform, turning it into a more complete and versatile tool.

Apache Mesos is a fault-tolerant and highly available sharing layer that provides a framework common interface allowing a fine-grained sharing across diverse cluster computing frameworks. Fault tolerance is ensured by Apache Zookeper.[7] Mesos offers a scalable and resilient core for enabling various frameworks. This is particularly important to share efficiently clusters. A master node manages slave daemons running on each node in the cluster (Hindman et al. 2011).

Each framework that run on the top of Mesos use a job scheduler registered to the master node and ask resources while an executor process is on slave nodes to run tasks of the framework (See Fig. 1).

Slaves nodes report to the master nodes available resources (number of CPU and amount of memory) (1). Then, the master node invokes the allocation policy module and determines the amount of resources to be allocated to each framework, and the scheduler selects each nodes of the offered resources to assign to the framework (2). At this step, the framework can reject the offered resources if they do not satisfy its constraints and wait another offer. If the framework accepts the offered resources (3), it sends to the master node a description of the tasks to launch on offered resources by nodes slave. A framework may specify a whitelist of nodes with which it can run and avoid node with which it always have offers reject. The master node sends the task to the slave node which allocate resources to the framework executor (4) (Hindman et al. 2011).

Allocation of resources is performed by two modules. The first performs fair sharing between resources, and the second implements strict priorities. Frameworks executor on slave nodes are isolated by leveraging existing OS isolation. Resource offers are scalable and robust through three mechanisms: filers to the master node, the count of resources, the re-offers of resources. Filters avoid communication by providing filters to master node for frameworks which always reject certain resources. Mesos counts resources offered to a framework in its allocation of the cluster. When a framework does not respond quickly enough to an offer, Mesos can re-offer the resources to another framework. Fault tolerance uses ZooKeeper to run multiple masters in a hot-standby configuration (Hindman et al. 2011).

Mesos provides also three containerization modes. Mesos containerizing allows using runtime environment, operating system control and additional resources like disk usage limit. Mesos allows also Docker containerizing[8] in order to use tools coming with Docker package. The composing of both containerization technology allows to test different types of resources isolations. Mesos is a good solution to implement multiple framework and share fine-grained resource of the cluster. This solution allows hosting applications for multiple use case such as cattle behavior.

A large amount of applications developed on different frameworks are needed to treat stored data in the cloud. Nowadays, to share a cluster, we have two main solutions. On one hand, we can run one framework on one partition of the cluster. On the other hand, the solution consists of allocating a set of virtual machines to each framework. However, theses solution cannot allow high use and efficient data sharing.

In this field, several works have been published, for example: a lambda architecture to treat all kinds of data was

[7] Apache Zookeeper (2017) http://hadoop.apache.orf/zookeper, Accessed 10 April 2017.

[8] Merkel D (2014) Docker: lightweight Linux containers for consistent development and deployment. Linux Journal 239, Article n°2.

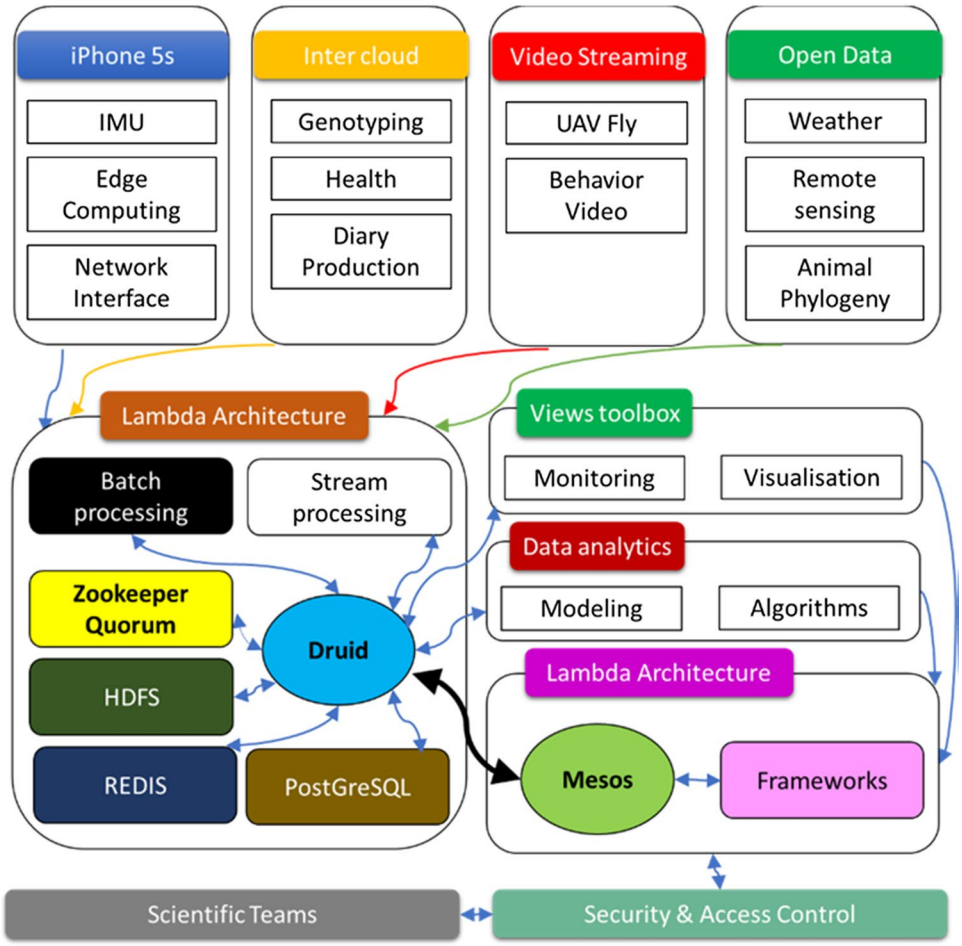


**Fig. 2** Lambda architecture with scientific sharing platform

proposed in Diáz et al. (2016). Marioti et al. (2017) propose a novel database management system (DBMS)-based system for the integration of Grids and Clouds. Kozhirbayev and Sinnott (2017) compared virtualization technologies with containers-based technologies and showed that the virtualization is utilized in data centers, for server consolidation and elastic scaling. But, they are overheads technologies avoided in high performance Computing (HPC) environments and limited in Input/Output (I/O), while Docker is horizontal scalable and low cost. They also compared performances on CPU, disk I/O and memory of container-based technologies and show that Docker can be faster compared to Flockport (LXC) if it uses multi-layer unification file system without using of network translation module.

## 4 Platform architecture

Lambda architectures are designed to handle large amounts of data in conjunction with both batch and stream processing methods (Veith et al. 2016) in combination with a serving layer (Diáz et al. 2016; Yang et al. 2017a). The particularity of lambda cloud architecture is that it is easily adaptable, and it is compatible with a wide range of use cases. Lambda cloud architecture is able to treat all kinds of data e.g. images, video, temporal data, event data or classic data. This paradigm allows the processing of real time data from stream and enables the rapid use of stored data (Debauche et al. 2018a, b, c). The no-prior or punctual data are processed in batch processing. The use of container technology makes it easy to deploy with different versions of the same model. This technology also allows a continuous integration of changes made on the model and rapid deployment. Virtual machines are also implemented for the elastic scaling of the infrastructure with the load.

In this paper, we propose an architecture structure, shown in Fig. 2, which combines on one hand a lambda architecture and on the other hand a hosting and sharing platform. In the proposed lambda architecture, data are separated into video streams treated by streaming processing and time/event - related data. Streaming processing treats video coming from behavior observations which are periodically acquired. Time-related and event related data

are treated by batch processing which consists of data verification (complete data). If data are not complete, they can be corrected, and missing data must be interpolated. Each interpolated data is specifically tagged as "generated data" in order to differentiate it from the original data. The batch processing treats also data recovered from private clouds such as genotyping, health information, production data, of each animal and open to external data information such as weather, remote sensing (e.g through UAV flights), animal phylogeny data. All this information is subsequently stored in a distributed database.

The behavior of the cows is studied using sensors placed on the animal. The information produced by the iPhone has been sent in online or offline mode. In the online mode, the data measured by the sensors can be used to calculate other derived parameters that are sent by UDP protocol over Wi-Fi to the Cloud platform. In the offline mode, the iPhone uses local storage to back up measured and calculated data. The data is saved locally in a CSV file, which is then transmitted and processed on a gateway before being transmitted to the cloud platform. The implementation of fog computing on the iPhone 4s/5s allows the reduction in the amount of data that must be transmitted to the gateway and the bandwidth used. It also improves storage efficiency in the distributed database hosted in the cloud. These sensors measure the movement of the cow in its environment. Videos of the behavior of the cows are also carried out and then synchronized with the data measured on the animal. Videos can also be streamed live or sent periodically. Such combination of videos and sensors signals must be treated on the platform to allow creating models of behaviors classification.

Streaming processing achieve video annotation and features calculation. The results of both operations are stored in the distributed database. Video and other data are synchronized manually to associate tagged behaviors on video with variations of sensors measures. Toolbox View allows to visualize the real-time data sent to the database. A monitoring system sends alerts when the data exceeds previously determined thresholds. This toolkit is available for the platform for sharing scientific data. Data analytic toolbox contains tools for modeling, and algorithm creation that can be deployed continuously on virtual machines or containers of the scientific sharing platform. Several versions of the same software can be maintained in the platform by using virtual machines or containers and versioning system.

The proposed scientific sharing platform is a major novelty of our lambda architecture specific for Internet of Things in conjunction with a hosting and scientific sharing platform. The lambda architecture collects process and stores data. In this hosting and sharing scientific platform, a web service takes care of the anonymization and securitized the access to the data. This platform allows to host models and software of researchers by mean of containerization technologies. The security module provides authentication and authorization that enables setting up role based on permission granted per privacy policies. Permissions to access the data and the different applications are proposed by the research teams. The rights can be given at the individual level, for a group of users or to a fixed IP address.

Our cluster is built with Apache Kafka, Apache Samza, Apache Hadoop, Druid, PostgreSQL, a Zookeeper quorum and Redis, (See Fig. 3).

As shown on Fig. 3, the stream processing is treated by the combination Kafka, Samza, Yarn, Apache Kafka provides a message between data producers and Apache Samza. YARN containers run Apache Samza to clean up false data, performs lookups and events. Batch processing uses MapReduce, Yarn, Hadoop to treat non-critical data. The treatment in the batch processing can take several hours. Then, Druid's real-time nodes ingest data by event reading. In this configuration, Druid cluster is able to consume 1,50,000 events by second (Yang et al. 2014).

Druid is composed of different types of nodes: Real-times, Historical, Brokers and Coordinators Nodes. Real-times nodes provide functionality to ingest, query, index event streams for small time range. The maintaining of Indexes in-memory allows a direct access of data. Next, immutable blocks appealed "segments" and merged indexes are created from data ingested by real-time node by a background Segments are uploaded to a HDFS (Shvachko et al. 2010) permanent backup storage. HDFS is a distributed file system for storing distributed and replicated data in a cluster of server (Diáz et al. 2016). Historical Nodes contain functionalities to load and serve the immutable blocks of data created by real-time nodes. Brokers nodes route incoming queries to historical or real-times node and return a final consolidated result to the applicant. Brokers nodes contain a caching system with a LRU (Lee et al. 2001) invalidation strategy and using Redis (Zawodny 2009) to store key/value. Finally, Coordinators nodes are in charge of data management and distribution on historical nodes: loading, dropping replication and moving of data. A PostgreSQL database connected to coordinators nodes store operational parameters, configurations. This database contains also the list of all segments that can be served by historical nodes and rules to create, destroy and replicate blocks of data in the cluster. The database can be updated by any service that creates persistent blocks of data. Batch data process event from static files in JSON or CSV format one at a time and produce segments directly uploaded in the deep storage. Batch data processing may take several hours by opposition of real-time where data are treated in sub second time (Yang et al. 2017a).

We use a share and hosting platform to treat and explore data from the IoT Lambda Architecture. This platform uses Apache Mesos and Docker containers to isolate and host applications. We can notice that Mesos isolation is

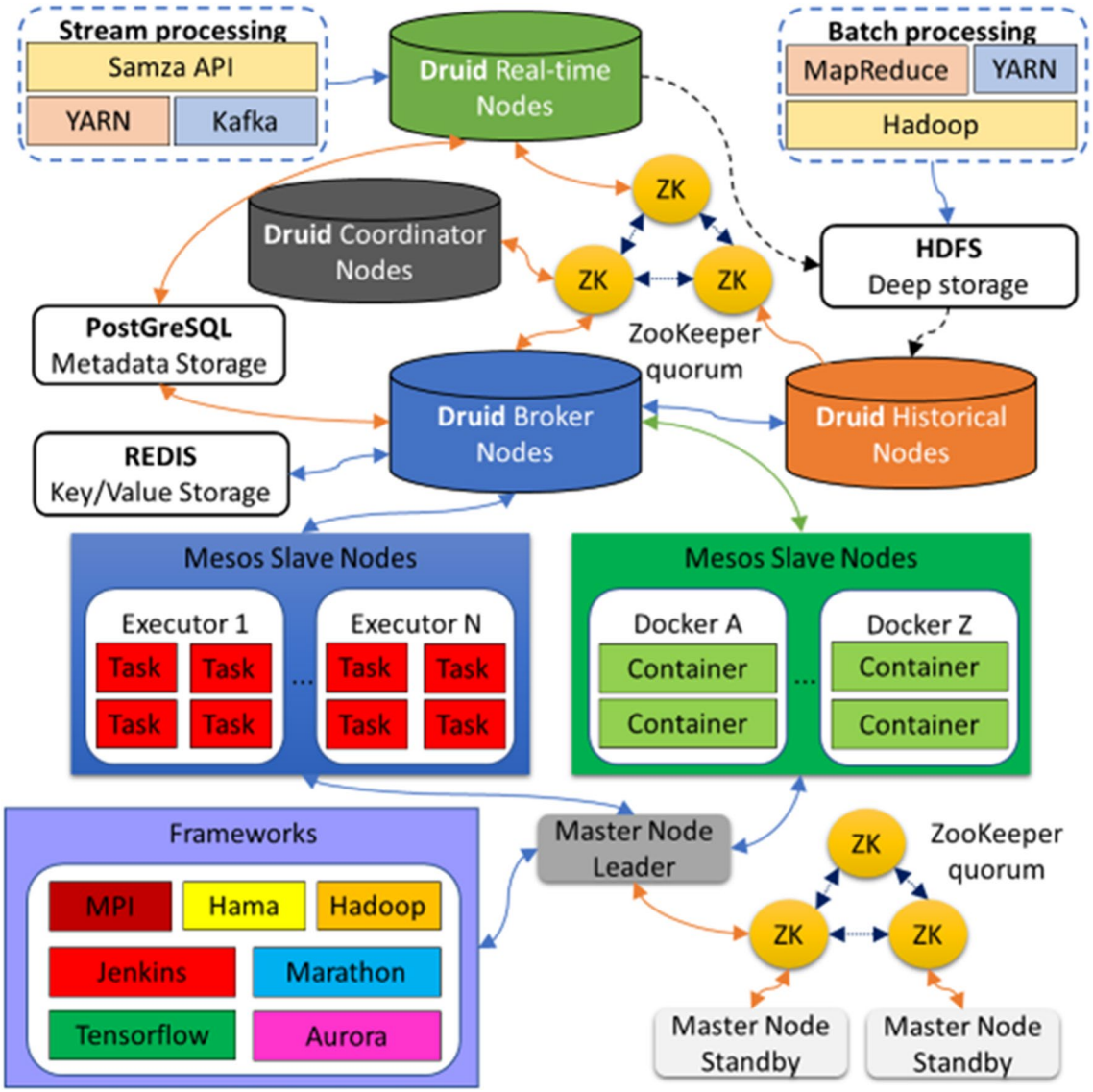


**Fig. 3** Components of the proposed architecture

better than Docker, that is why we have mixed these both containerizing methods for compatibility reasons (Fig. 1). The application sharing platform use a quorum of 3 nodes Zookeeper: one master node and two master standby nodes. These two standby nodes ensure fault tolerance in the cluster.

Mesos offers several pluggable frameworks. Each framework sends tasks to the master node which transfer them to a slave node available to execute the task. When the task is executed the result is send to node master which forward them to the framework. Docker slave node can host external application which don't initially be developed to work on frameworks plugged on Mesos. They can nevertheless be hosted with container technology offered by Docker.

Six frameworks plugin are installed on our application sharing platform. Jenkins framework allows continuous integration and dynamic launch of workers depending on the workload. Jenkins allows to researcher to develop algorithm and test them on the cluster. Tensor Flow (Abadi et al. 2016) enables to run distributed machine learning tasks with GPU. Tensor flow allows researchers to experiment machine learning algorithms on a set of images acquire by 3D cameras. Marathon is a Private as a Service (PaaS) which ensures that an application is always "on". It automatically handles hardware or software failures and guarantee the availability of paying. services. MPI (Gropp 1996) is a message-passing system to function on parallel computers. MPI allows to accelerate application by starting parallel jobs. It has been plugged for compatibility reasons for some algorithms and

models. Apache Hama (Apache Foundation 2017a) is used for distributed computing for massive scientific computations and big data analysis based on Bulk Synchronous Parallel (BSP) computing techniques. It provides also vertex and neuron centric programming models (Apache Foundation 2017a). Hama is principally uses in data analysis and model elaboration. Apache Aurora (Apache Foundation 2017b) is a service scheduler used to run long-running services while benefiting of scalability, isolation and fault-tolerance of Mesos. Aurora is used to develop applications to treat raw data from the lambda architecture and execute cron jobs. Finally, Hadoop framework distributes MapReduce on the cluster which is used for cloud computing.

## 5 Conclusion

In this paper, a new data storage architecture dedicated to scientific research has been proposed. This lambda architecture is able to collect data at high frequency and is adaptable easily to many cases. The main originality of the architecture lies on its ability to share the data and applications created by the different teams of scientists from a common database. This architecture is also able to integrate complementary data such UAV images, and external data from other cloud platforms such as health and production data. The iPhone is an inexpensive means of measuring cow behavior. I-Phone 5SE, 6S, 7S and 8S are equipped with a new factory-calibrated IMU. This new IMU is not much evaluated in scientific literature (Yang 2017a, b). Currently the data is transmitted from the iPhone to the gateway by using the UDP protocol on WIFI, but this protocol may cause a data packet loss problem when it is necessary to collect data from several iPhone simultaneously. The compressibility of data massively acquired can be reduced by 43.5% on average and this can hardly be improved any further. By opposition, we have shown that individual parameters can be highly compressible. In the future when the most explicative parameters will be selected for a given research application, the compressibility of data will be improved. Finally, applying edge computing during the massive collection of data is not interesting. The future development of microcontrollers which acquire pertinent parameters at specific sampling rate and use low throughput LoRa will allow to resolve this issue. Indeed, using acknowledgment in the protocol LoRA guarantee the correct reception of payload transmitted. The lambda architecture proposed for collecting, storing, processing and sharing data between research teams is flexible enough to be used for other uses than cow behavior provided such that different teams contribute to the system. Other data compression algorithms must be considered to optimize the energy consumption of the battery. The sharing platform will integrate a billing system to valuate on one hand the using of data and multi-tenancy of software on the other hand.

**Acknowledgements** We would like to thank our colleagues from the CARE *AgricultureIsLife* (TERRA Teaching and Research Unit, Gembloux Agro-Bio Tech) and the Precision Livestock and Nutrition Axis, without whom this work would not have been possible. We would also like to thank Prof. Ghalem Belalem for the architecture brainstorming.

## References

Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G. Isard M, Kudlur M, Levenberg J, Monga R, Moore S, Murray DG, Steiner B, Tucker P, Vasudevan V, Warden P, Wicke M, Yu Y, Zheng X, Google Brain (2016) TensorFlow: a system for large-scale machine learning. In: 12th USENIX Symposium on Operating Sytemps Design and Implementation (OSDI'16). Novembre 2–4, 2016, ISBN: 978-1-931971-33-11

Altintas I, Berkley C, Jaeger E, Jones M, Ludascher B, Mock S (2004) Kepler: an extensible system for design and execution of scientific workflows. In: Scientific and Statistical Database Management. In: Proceedins of the 16th International Conference on, IEEE, pp 423–424. https://doi.org/10.1109/SSDM.2004.1311241

Andriamandroso ALH, Bindelle J, Mercatoris B, Lebeau F (2016) Review on the use of sensors for jaw movements' detection. Biotechnol Agron Soc Environ 20(S1):273–286

Andriamandroso ALH, Lebeau F, Beckers Y, Froidmond E, Dufrasne I, Heinesch B, Dumortier P, Blanchy G, Blaise Y, Bindelle J (2017) Development of an open-source algorithm based on inertial measurement units (IMU) of a smartphone to detect cattle grass intake and ruminating behaviors. Comput Electron Agric 139:126–137. https://doi.org/10.1016/j.compag.2017.05.020

Andriamasinoro ALH, Lebeau F, Bindelle J (2015) Changes in biting characteristics recorded using the inertial measurement unit of a smartphone reflect differences in sward attributes. Precision Livestock Farming 15:283–289

Apache Foundation (2017a) Apache Hama. https://hama.apache.org. Accessed 7 June 2017

Apache Foundation (2017b) Aurora is a Mesos framework for long-running services and cron jobs. http://aurora.apache.org. Accessed 7 June 2017

Chikhaoui B, Ye B, Mihailidis A (2017) Aggressive and agitated behavior recognition from accelerometer data using non-negative matrix factorization. J Ambient Intell Human Comput. https://doi.org/10.1007/s12652-017-0537-x

Debauche O, Mahmoudi S, Adriamandroso ALH, Manneback P, Bindelle J, Lebeau F (2017) Web-based cattle behavior service for researches based on the smartphone inertial central. Proc Comput Sci 110:110–116. https://doi.org/10.1016/j.procs.2017.06.127

Debauche O, El Moulat M, Mahmoudi S, Manneback P, Lebeau F (2018a) Irrigation pivot-center connected at low cost for the reduction of crop water requirements. In: International Conference on Advanced Communication Technologies and Networking (CommNet 2018). IEEE. ISBN: 978-1-5386-4609 **(in press)**

Debauche O, El Moulat M, Mahmoudi S, Boukraa S, Manneback P, Lebeau F (2018b) Web monitoring of bee health for researchers and beekeepers based on the internet of things. Proc Comput Sci **(in press)**

Debauche O, Mahmoudi S, Manneback P, Massinon M, Tadrist N, Lebeau F, Mahmoudi SA (2018c) Cloud architecture for digital phenotyping and automation. In: The 3rd International Conference on Cloud Computing Technologies and Applications—Cloud-Tech'17. https://doi.org/10.1109/CloudTech.2017.8284718

Diáz M, Martin C, Rubio B (2016) State-of-the art, challenges, and open issues in the integration of Internet of things and cloud computing. J Netw Comput Appl 67:99–117. https://doi.org/10.1016/j.jnca.2016.01.010

Doel T, Shakir DI, Pratt R, Aertsen M, Moggridge J, Bellon E, David AL, Deprest J, Vercauteren T, Ourselin S (2017) GIF-Cloud: a data sharing and collaboration platform for medical imaging research. Comput Methods Programs Biomed 139:181–190. https://doi.org/10.1016/j.cmpb.2016.11.004

Frost AG, Schofield CP, Beaulah SA, Mottram TT, Lines JA, Wathes CM (1997) A review of livestock monitoring and the need for integration systems. Comput Electron Agric 7(2):139–159. https://doi.org/10.1016/S0168-1699(96)01301-4

González LA, Bishop-Hurley GJ, Handcock RN, Crossman C (2015) Behavioral classification of data from collars containing motion sensors in grazing cattle. Comput Electron Agric 110:91–102. https://doi.org/10.1016/j.compag.2014.10.018

Gropp W, Lusk E, Doss N, Skjellum A (1996) A high-performance, portable implementation of the MPI message passing interface standard. Parellel Comput 22(6):789–828. https://doi.org/10.1016/0167-8191(96)00024-5

Hindman B, Konwinski A, Zaharia M, Ghodsi A, Joseph AD, Katz R, Shenker S, Stoica I (2011) Mesos: a platform for fine-grained resource sharing in the data center. NSDI'11, pp 1–14. https://www.usenix.org/legacy/events/nsdi11/tech/full_papers/Hindman_new.pdf

Hull D, Wolstencroft K, Stevens R, Goble C, Pocock MR, Li P, Oinn T (2006) Taverna: a tool for building and runing workflows of services. Nucleic Acids Res 34(2):W729-W732. https://doi.org/10.1093/nar/gkl320

Kipf A, Brunette W, Kellerstrass J, Podolsky M, Rosa J, Sundt M, Wilson D, Borriello G, Brewer E, Thomas E (2016) A proposed integrated data collection, analysis and sharing platform for impact evalution. Dev Eng 1:36–44. https://doi.org/10.1016/j.deveng.2015.12.002

Kozhirbayev Z, Sinnott RO (2017) A performance comparison of container-based technologies for the Cloud. Future Gen Comput Syst 68:175–182. https://doi.org/10.1016/j.future.2016.08.025

Lee D, Choi J, Kim J-H, Noh SH, Min SL, Cho Y, Kim CS (2001) LRFU: a spectrum of policies that subsumes the least recently used and least frequently used policies. IEEE Trans Comput 50(12):1352–1361

Linke B, Giegerich R, Goesmann A (2011) Conveyor: a workflow engine for bioinformatic analyses. Bioinformatics 27(1):903–911. https://doi.org/10.1093/bioinformatics/btr040

Manya A, Braa J, Øverland L, Titlestad O, Mumo J, Nzioka C (2012) National roll out of district health information software (DHIS2) in kenya, 2011—central server and cloud based infrastructure. In: IST-Africa 2012 Conference Proceedings. ISBN: 978-1-905824-34-2

Mariotti M, Gervasi O, Vella F, Cuzzocrea A, Costantini (2017) Strategies and systems towards grid and clouds integration: a DBMS-based solution. Future Gen Comput Syst. https://doi.org/10.1016/j.future.2017.02.047

McNab T, James DA, Rowlands D (2011) Iphone sensor plaforms: applications to sports monitoring. 5th Asia-Pacific Congress on Sport Technology (APCST). Proc Eng 13:507–512. https://doi.org/10.1016/j.proeng.2011.05.122

Miceli G, Pekkarinen A, Leppanen M (2011) Open foris initiative—tools for forest monitoring and reporting. Concept Note

Micucci D, Mobilio M, Napoletano P, Tisato F (2017) Falls as anomalies? An experimental evaluation using smartphone accelerometer data. J Ambient Intell Human Comput 8:87–99. https://doi.org/10.1007/s12652-015-0337-0

Milani P, Coccetta CA, Rabini A, Sciarra T, Massazza G, Ferriero G (2014) Mobile smartphone application for body position measurement in reabilitation: a review of goniometric tools. PM R 2014 6:1038–1043. https://doi.org/10.1016/j.pmrj.2014.05.003

Néron B, Ménager H, Maufrais C, Joly N, Maupetit J, Letort S, Carrere S, Tuffery P, Letondal C (2009) Mobyle: a new full web bioinformatics framework. Bioinformatics 25(22):3005–3011. https://doi.org/10.1093/bioinformatics/btp493

Rowlands D, James D (2011) Real time data streaming from smart phones. Procedia. 5th Asia–Pacific Congress on Sports Technology. (APCST) Eng 13:464–469. https://doi.org/10.1016/j.proeng.2011.05.115

Shvachko K, Kuang H, Radia S, Chansler R (2010) The hadoop distributed file system. In Mass Storage Systems and Tdchnologies. IEEE 26th Symposium on pp 1–10

Veith AS, Anjos JCS, de Freitas EP, Lampoltshammer TJ, Geyer CF (2016) Strategies for big data analytics through lambda architectures in volatile environments. IFAC-PapersOnLine 49–50:114–119. https://doi.org/10.1016/j.ifacol.2016.11.138

Yang F, Tschetter E, Léauté X, Ray N, Merlino G, Ganguli D (2014) A real-time analytical data store. SIGMOD'14, June 22–27, 2014, Snowbird, UT, USA. ACM 978-1-4503-2376-5$414/06. https://doi.org/10.1145/2588555.2595631

Yang F, Merlino G, Ray N, Léauté X, Gupta H, Tschetter E (2017a) The RADStack: open source lambda architecture for interactive analytics. In: Proceedings of the 50th Hawaii International Conference on System Sciences, pp 1703–1712

Yang L, Grooten WJA, Forsman M (2017b) An iPhone application for upper arm posture and movement measurements. Appl Ergon 65:492–500. https://doi.org/10.1016/j.apergo.2017.02.012

Zawodny J (2009) Redis: lightweight key/value store that goes the extra mile. Linux Magazin 31 August, 2009