Cloud Architecture For Digital Phenotyping And Automation

Olivier Debauche*
Computer Science Unit
FPMs, UMons
Mons, Belgium
olivier.debauche@umons.ac.be

Saïd Mahmoudi
Computer Science Unit
FPMs, UMons
Mons, Belgium
said.mahmoudi@umons.ac.be

Pierre Manneback
Computer Science Unit
FPMs, UMons
Mons, Belgium
pierre.manneback@umons.ac.be

Mathieu Massinon
BioDynE Axis
TERRA, GxABT - ULiège
Gembloux, Belgium
m.massinon@ulg.ac.be

Nassima Tadrist
BioDynE Axis
TERRA, GxABT - ULiège
Gembloux, Belgium
nassima.tadrist@doct.ulg.ac.be

Frédéric Lebeau
BioDynE Axis
TERRA, GxABT - ULiège
Gembloux, Belgium
f.lebeau@ulg.ac.be

Sidi Ahmed Mahmoudi
Computer Science Unit,
FPMs, UMons
Mons, Belgium
sidi.mahmoudi@umons.ac.be

***Abstract*—Digital phenotyping presents a very important tool for scientists to measure with high accuracy the effects of external phenomena on plant development. Plant phenotyping is mainly based on imaging techniques. However, the number of images and parameters used to store and treat these parameters are continuously growing. Consequently, the high-throughput of data and the need of specific treatment in real or near real-time requires a large quantity of resources. Moreover, the increasing amount of particular phenotyping case studies needs the development of specific application. Cloud architectures offers means to store a wide range of numerous data and host a large quantity of specific software to process these data. In this paper, we propose a new approach that shows how logic synthesis works to match digital phenotyping need and cloud possibilities in a lambda cloud architecture in order to store and treat this important amount of data in real time. We also suggest a data platform allowing to host applications and access to the stored data within the lambda architecture. The present application platform allows to use several frameworks with a fine-grained resource use of the cluster. Finally, we develop a case study in a controlled environment system (growth chamber) where we grow basil plants.**

Keywords—cloud; lambda architecture; digital phenotyping; 3D plant model; phytotron; application platform.

I. Introduction

Digital phenotyping (DP) is a combination of novel technologies: non-destructive imaging (thermal infrared, fluorescence, 3D and tomographic imaging) [1], spectroscopy (multispectral and hyperspectral remote sensing) [1], image analysis, High Performance Computing (HPC) and robotic [2]. Several image analysis tools for studying plant biology have been developed. DP can be defined as a set of methodologies and methods using non-destructive systems in high-throughput to accurately analyze traits of plant at several scales. It enables scientists to measure with high accuracy the effects of the environment on plant development. It also informs scientists on genomic functionality. DP covers a wide range of applications like for instance: breeding, developing treatments for cultivation, environmental monitoring and precision agriculture, etc. Principal applications where DP is actually used are: ecotoxicology, field phenotyping, functional genomics, plant physiology, high throughput plant phenotyping, high content plant phenomics, plant growth, root phenomics and stress response.

Ecotoxicology uses phenotyping to quantify non-mobile organisms. Its colors, area size may also be measured. From these parameters, it is possible to have a quantitative response of toxicant. With mobile organisms, screening tests are performed to identify the present species or provide movement quantification in very confined vessels. These measures can allow for example growth rates of organism population, inhibitions values, etc. Images can be acquired in various condition of lighting: top light, back light, dark field illumination, custom light sources. These various conditions of lighting apply also flexibility in software systems.

In the field of phenotyping, breeders and scientist search to quantify yield, plant development and physiology parameters, ecological biotic growth factors and stress response of crops under real climate conditions. Precise screening using nondestructive methods is achieved on punctual sampling. Image-based measurement is challenging in outdoor condition because light exposure evolved in quantity and quality during the day.

Functional genomics combine high-throughput genotyping and gene sequencing. 3D high resolution phenotyping data is used to find similarities or differences within or between the phenome of different genotypes [3]. In this case, storage size is important. Authors of [4] reconstruct plant from a cloud of 1.2 to 1.7 million of 3D points. The computing time with a recent CPU processor (Inteli7 with 3.6 Ghz and 8GB of RAM) is 9000s.

In the plant physiology field, digital phenotyping allows to see the impact of climatic parameters, water and nutrient usage efficiency, ability to grow on various substrates, such as degraded soils. Methods, like hyperspectral imagery, are widely used to extract information from images made under highly controlled light condition in greenhouse or growth chamber.

As shown in Table 1, High Throughput Plant Phenotyping aims to characterize plant complex traits such as growth, development, tolerance, resistance, architecture, physiology, ecology and yield. These complex traits are determined from the measure of a wide range of individual quantitative parameters that are image-based. In this application, the interest is focused on the number of parameters to monitor [2] and [1] (Table 1).

TABLE I. EXAMPLES OF PLANTS TRAITS

Level	***Parameters***
Plant	volume, biomass
Main-stem	height, size, inclination
Petiole	length, Initiation angle
Leaf	Width, Length, Inclination, thickness, area, curvature, shape
Root	Morphology
Fruit	fruit characteristic

High-Content Screening is used to discriminate pharmaceutical tests with read-out parameters. Multiple read must be done one by parameter on each sample.

Plant growth rate is an important parameter to describe development over time. Growth pattern obtained from multiple images are used to compare growth rate between plants.

Roots phenotyping requires 3D scanning to acquire the root development over the time. NIR (Near Infrared) imaging made at short time allows to evaluate soil water content and distribution. Water-Use-Efficiency (WUE) can also be measured to determine the volume of water used to produce biomass. The impact of waterlogging can also be evaluated by digital phenotyping.

Finally, the stress response phenomics describes the influence of environmental stress on the growth of plants and the reduced yield in comparison with optimal growth conditions. NIR imagery allows the monitoring of leaf water content, and IR (Infrared) imagery is used to monitor water content for seeds and Leaf Area Index (LAI)[1], while VIS (Visible Spectrum) imagery provides information about biomass development, plant architecture and leaf reactions to climate change or stress.

The remainder of this paper is organized as follows. Related works are illustrated in Section 2. In Section 3, we present our scientific phytotron in container. Then, we describe the digital phenotyping and the environmental monitoring parameter of our case study. In Section 4, we describe the proposed cloud architecture and application platform. We show the application of our solution to our phytotron. We explain how we automated this growth chamber by controlling light, atmospheric parameters and nutriment. In section 5, we present conclusions and future works.

II. RELATED WORK

The main applications of digital phenotyping show that requirements for information processing are very different and depend on the aim of the phenotyping. For instance, measuring phenomena, such as foliar reactions, requires a rapid treatment of many images. In other cases, such as pattern of growth, there are many images over a long time that must be archived and prepared for further post-treatment. On one hand, phenotyping requires both rapid processing of large number of images and related data. On the other hand, it requires the massive storage of very large amounts of data and the processing capacities of all this mass of information.

Large scale data storage and multiple treatment application of these data require a cloud architecture platform. In digital phenotyping, wide range of images of different kind must be acquired and stored. These images must be completed with information from other sensors before being processed and that actions may be taken.

Data analysis in the field of Smart Agriculture, is growing rapidly. However, in parallel with the increasing amount of data to be processed, processing systems fails to process information in short delays. Hadoop ecosystem has proved its efficiency to overcome this problem in a wide range of use case.

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 [5] and [6].

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 Cassandra, Hive, Pig, Hbase, Chukwa, S3 and HDFS [5].

Stream processing frameworks like Apache Storm, Apache Spark Streaming and Apache Samza[2] 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

[1] The one-sided green leaf area per unit ground surface area.
[2] Apache Zookeeper. http://hadoop.apache.orf/zookeper.

streaming platform). Apache Hadoop YARN is used to provide fault tolerance. However, theses stream processing frameworks generally do not provide the same guarantees as batch processing frameworks in matter of correctness [6]. However, the processing may suffer from duplicated events and other problems of accuracy in data.

The speed of data availability depends on 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 [7]. First, raw data must be transformed or cleaned before their use [6]. Hence, the process of data loading and batch processing can take a long time (several hours).

Lambda architectures are designed to handle large amounts of data in conjunction with both batch and stream processing methods[3] in combination with a serving layer [8] and [6]. The particularity of this cloud architecture is its compatibility with different cases. Lambda cloud architecture can treat all kinds of data e.g. images, video, temporal data, event data or classic data. This paradigm allows processing at real time data from stream and enables using rapidly data stored. The not priority or punctual data are processed in batch processing.

The aggregation of real-time and batch processed data is in the serving layer.

Druid presented in [5] that 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 to 10 million times-stamped events that covers one period of time. Segments can be compressed by LZ4[4] by default or LZF[5] algorithm, and can also be stored in a column orientation database. Druid cluster is composed of 4 kinds of nodes.

Druid uses two external dependencies. The first one is MySQL, PosGreSQL 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: S3, HDFS, Microsoft Azure, Google Cloud Storage and Apache Cassandra.

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

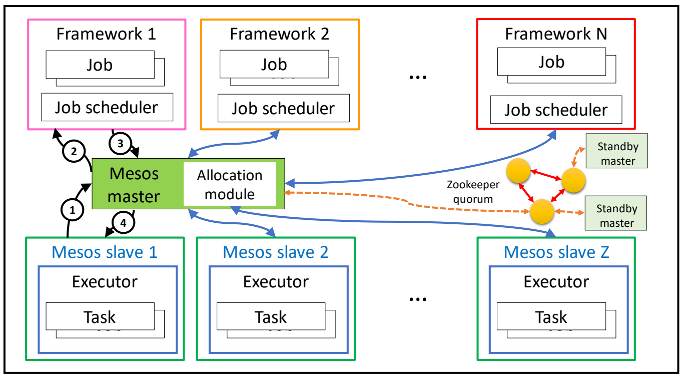

Fig 1. Apache Mesos Architecture

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 [9]. 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.

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.

Allocation of resources is performed by two modules. The first performs fair sharing between resources, and the second implements strict priorities. Frameworks executor on nodes slave 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 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 [10]. Mesos provides also three containerization modes. Mesos

[3] Liblzf, http://freecode.com/projects/liblzf , March 2013.

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

[5] Apache Samza. http://samza.apache.org/ , April 2017.

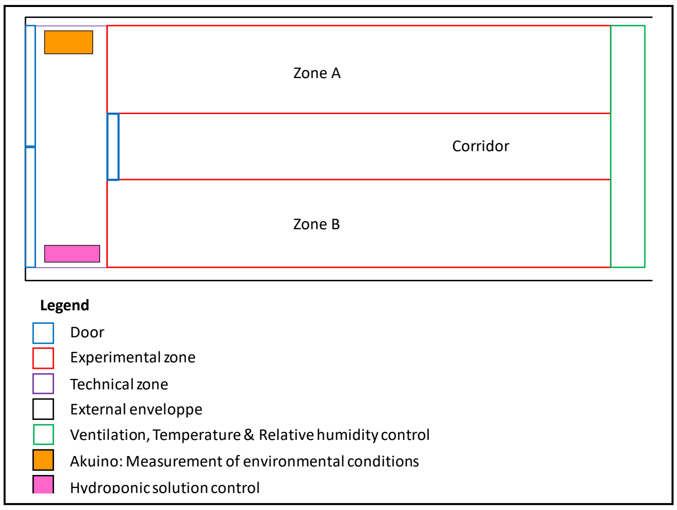
Zone A
Corridor
Zone B
Legend
Door
Experimental zone
Technical zone
External enveloppe
Ventilation, Temperature & Relative humidity control
Akuino: Measurement of environmental conditions
Hydroponic solution control

Fig 2. Phytotron Container Organization

containerizing allows to use runtime environment, operating system control and additional resources like disk usage limit. Mesos allows also Docker containerizing [11]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 of digital phenotyping. Our contribution consists of highlighting the important diversity of needs faced by researchers in digital phenotyping. In addition to that, we exhibit the potential of the cloud to meet these needs. Thus, we contribute by proposing a cloud lambda architecture allowing to store, analyze and host applications for plant phenotyping. This architecture provides strategic direction and guidance solution in order to process images from digital phenotyping and complementary sensor data.

III. MATERIAL

Our scientific phytotron (Fig. 2) is a closed enclosure located in 20' container Advanced Fresh Air Management Plus (AFAM+) from Thermo King. The container is divided into two experimentation zones. Each one is composed of a shelf with 3 levels. Each level of the shelf can store up to 100 basil plants in hydronic system.

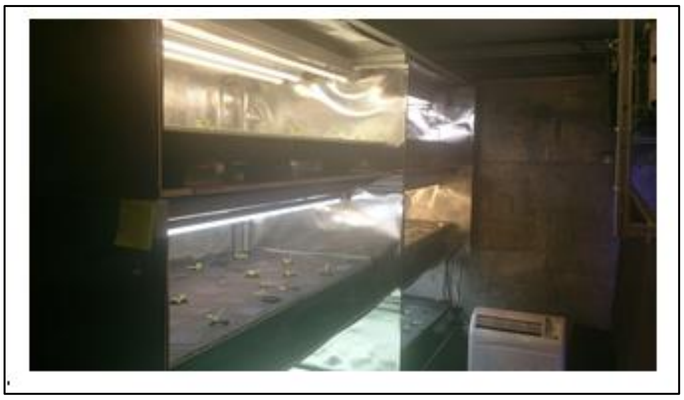

Fig 4. Shelfs of Zone A

Principal parameters monitored in the phytotron installation are aerial, root and light parameters. The aerial parameters include air temperature, air moisture, carbon dioxide concentration, light spectrum and intensity. The root parameters in hydroponic solution are solution composition including electrical conductivity, nutrient concentration, pH, Oxygen and NO_3^- concentration [12]. NO_3^- is acquired by a sensor Probe Pro and a Reference Probe Pro Libelium connected on a shield Smart Ion Board mount on a Waspmote 1.5 A supplementary module Wifi Pro Libelium is charged of the transmission of data to Cloud Architecture. An Akuino[6] equipped with a UPS with an autonomy of 6 hours acquires all data in the container (see Fig 3). Akuino is an open source solution based on Raspberry Pi and Arduino adapted to agriculture production context. An Arduino Uno connected to the container regulation system controls the temperature in a day/night cycle and the relative humidity. The pH and concentration of nutriment in the hydroponic solution is maintained by an automatic system provided of peristaltic pump. (see Fig 5).

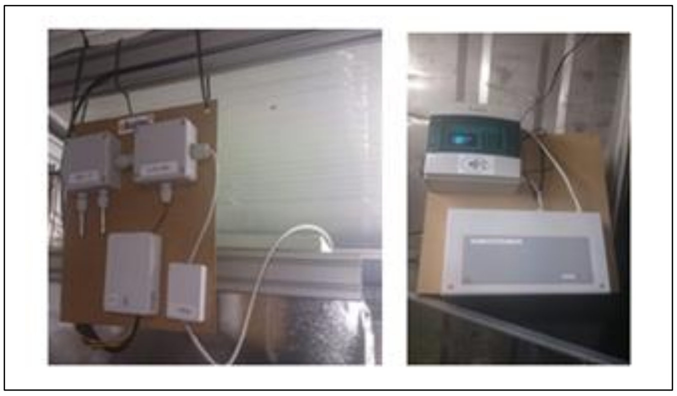

Fig 3. Measurement of environmental parameters

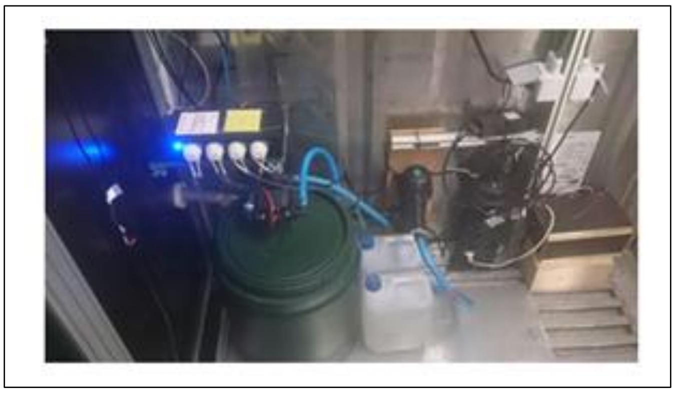

Fig 5. Hydroponic control solution

[6] Akuino, Online: http://www.akuino.net , June 2017.

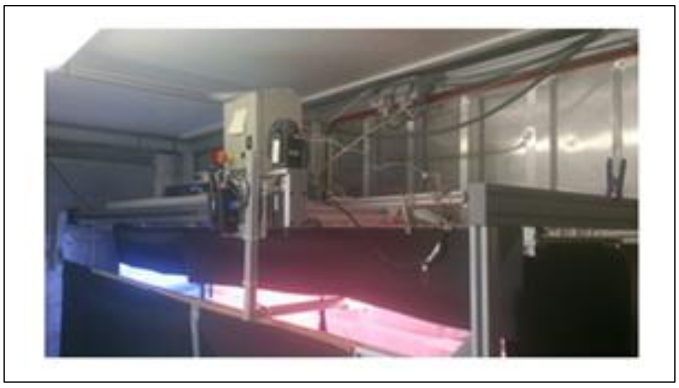

Fig 6. Camera position control and acquirement

One of the most important parameter that must be optimized is the light spectrum. This parameter influences directly the growth rate and phenotypic development of plants. The light plays a role in the life of plants in terms of growth, informative and biosynthetic. The light spectral composition particularly in red and blue regions influence directly growth and biosynthetic functions. Otherwise, the use of pulsed light can also improve plant productivity compared to continuous light [1]. Spectral measurement should be made with a maximal bandwidth of 20 nm in the photomorphogenic radiation range (300-800nm) [12].

Nowadays, many controllable LED lightings are available and offer interesting photosynthetically active radiation (PAR). However, they do not allow to reproduce all the required light spectra finely enough for a scientific phytotron. That is why; our research installation use ArtNet to control each channel of led placed above each shelf of plants. ArtNet is a protocol over UDP using Ethernet or Wi-Fi to transmit Digital Multiplexing (DMX512) frames. Each shelf in the phytotron contains a variable number of channels (N). The number of channels is generally from 8 to 64 channels depending of the kind of spectrum required. Each channel of the DMX512 system is decomposed into 256 levels (8 bits) of intensity. This number of levels is largely greater than the 100 levels proposed by

authors of [1]. The number of possible combination is 256*N. Considering the large number of possible combinations, it is not possible to systematically examine all combinations in a reasonable time with conventional means. The spectral measurement of each LED is operated at different levels of intensity by spectrophotometer.

For each level of each shelf, a 3D camera Intel® RealSense™ SR300 is mounted on a translation stage to move the camera above the plants, both controlled by an Intel® Up board (see Fig 6). This setup allows to retrieve the 3D structure of the plants independently. Each 3D camera RealSense SR-300 take one 2,1-mega pixel photo each hour. As a 20' container can contain 600 plants, in one day, 14 400 photos must be stored. In one year, up to 5.256 million of photos are made in the container. Six Intel® Up board control movements and acquisition of each of six cameras above the center of the pot. Images can be acquired in visible spectrum or in the near infrared spectrum. The Camera is also able to measure distance with the plant and evaluate the height of the plant. Then, photos are dispatched to the cloud where they are stored and are quickly made available to be treated by an application hosted in the application platform. Our scientific phytotron is controlled by a cloud algorithm, which monitor, alert and take decision on basis of data collected and stored in our cloud solution.

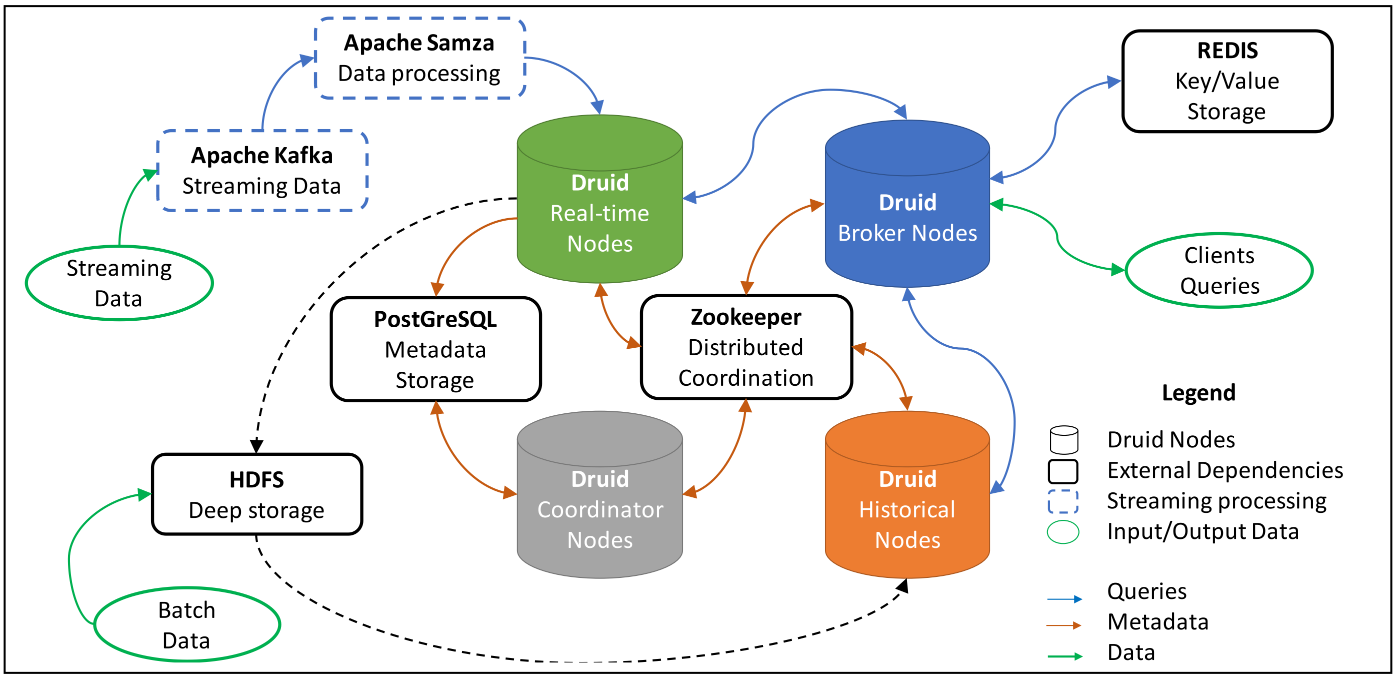

Fig 7. Lambda Architecture Proposed

IV. METHODOLOGY

The literature review has shown that a lambda architecture is able to collect and store the wide range of data from phenotyping and environmental parameters such as temperature, relative humidity, cation-exchange capacity (CEC), NO_3, etc. The application hosting using Mesos keeps the structure easily adaptable to various framework while proposing isolation and fine-grained resources of the cluster.

Phenotyping in growth chamber faces several major limitations. The first one is the limited space in the phytotron, which make more complex the process of data acquisition at advanced growth stages. Furthermore, artificial lighting can alter pattern of plant growth and development [9].

A. *Cloud architecture*

We propose a cloud solution based on a lambda architecture to collect, store and treat data from sensors placed in the phytotron. The architecture is designed to host various applications and allows to use them with others phytotron commercial phytotron than those for which they were originally conceived. Various kinds of data must be stored: 3D picture of growth, temporal and event data. Our cluster is built with Apache Kafka, Apache Samza, Apache Hadoop, Druid, PostGreSQL, Zookeeper and Redis, (Fig 7).

Apache Kafka provides a message bus between producers and Apache Samza. YARN containers run Apache Samza to clean up faults in data, performs lookups and performs events. Then, Druid's real-time nodes ingest data by event reading. In this configuration, Druid cluster is able to consume 150 000 events by second [5]. Druid is composed of four 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. Indexes are maintained in-memory to be directly queryable. A background task merges indexes together and build immutable blocks from data ingested by real-time nodes. Segments are uploaded to a HDFS [13] permanent backup storage. HDFS is a distributed file system for storing distributed and replicated data in a cluster of server [8]. 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 [14] invalidation strategy and using Redis [15] 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 block 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 [6].

B. *Application Sharing Platform*

We use a share and hosting platform to treat and explore data from the IoT Lambda Architecture. This platform uses

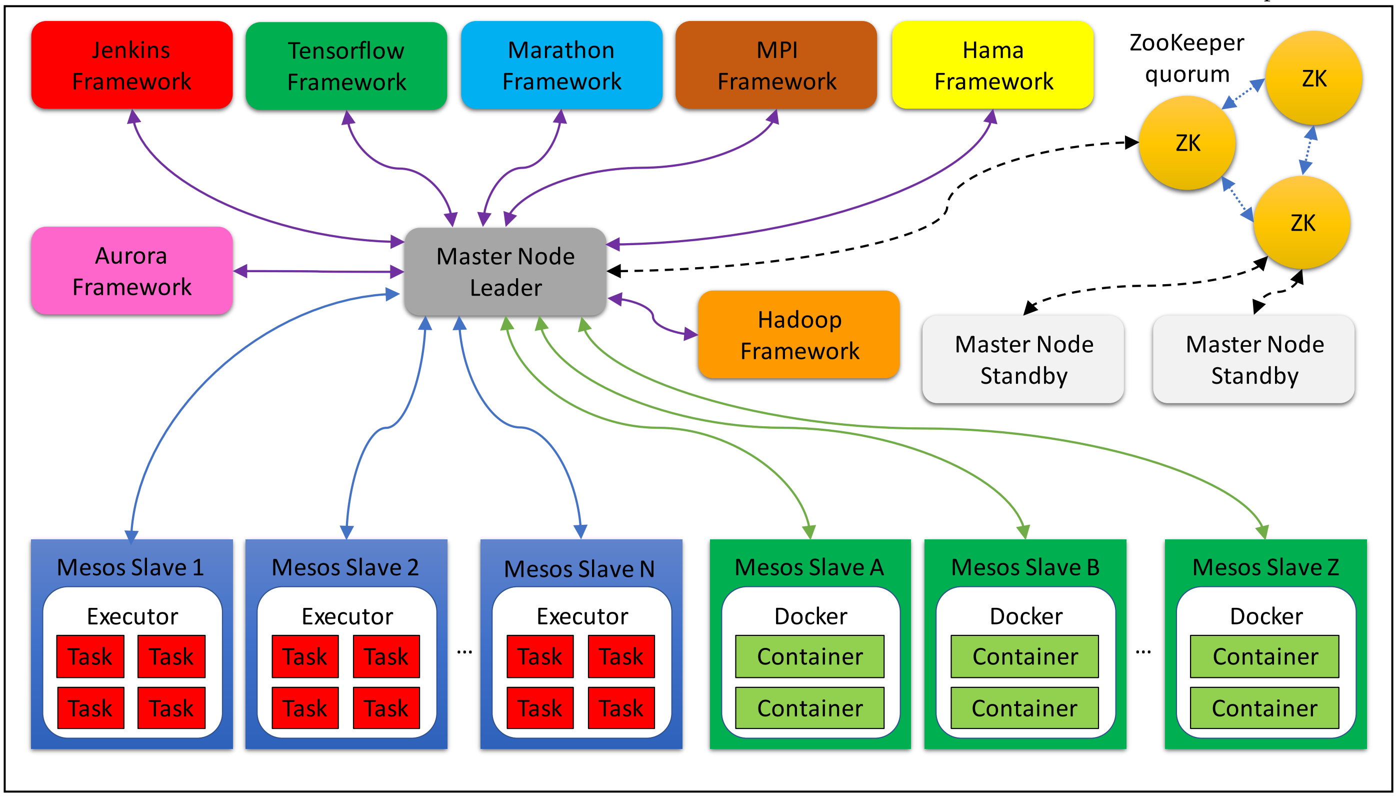

Fig 8. Application Platform Proposed

Apache Mesos and Docker containers to isolate and host applications. We can notice that Mesos isolation is better than Docker, that is why; we have mixed these both containerizing methods for compatibility reasons (Fig 8). 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 slave node available which 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. TensorFlow [16] enables to run distributed machine learning tasks with GPU. Tensor flow allows to researcher to experiment machine learning on set images acquire by 3D camera. Marathon is a Private as a Service (PaaS) ensures that an application is always "on". It automatically handles hardware or software failures and guarantee the availability of paying services MPI [17] 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[7] is for distributed computing for massive scientific computations and big data analyses based on Bulk Synchronous Parallel (BSP) computing techniques. It provides also vertex and neuron centric programming models. Hama is principally uses in data analyses and model elaboration. Apache Aurora [18] is a service scheduler 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.

C. *Light optimization algorithm*

The search for an optimal spectral combination of several light sources is a challenge in this work. Indeed, when the number of intensity levels and the light sources are important, it is difficult to find a good solution in reasonable time. Authors of [19] have shown that more of 100 steps of intensity are necessary to achieve a good accuracy in terms of spectral combination. These researchers have proposed different approaches to optimize LED lighting and obtain specific spectra.

Authors of [20] have proposed a novel LED-based tunable light source which comprises 31 spectral bands to reconstruct spectrum lighting. An algorithm is used to calculate the weight of each channel. This algorithm uses an interior-point method to minimize the distance between resulting spectrum and targeted spectrum. Authors of [21] propose a system based on 24 LEDs and covering a spectrum between 700 nm to 1070 nm. The intensity of each LED is modulable with 4000 levels. Therefore, the possible combinations number is 4000^{24}. They use an iterative algorithm based on a mobile window and a progressive reduction of the variation values of each led to fix weights.

In our work, we use the MapReduce paradigm with data stored in the lambda architecture to calculate a good and acceptable solution by mean of a Particle Swarm Optimization (PSO) algorithm [22]. This meta heuristic intend for simulating social behavior to improve by iteratively trying a candidate solution. A candidate solution is composed of a combination of intensity of each light source. The evaluation is achieved by comparison with target spectrum by calculation of difference between both spectrum for each group of wave length of 10 nm. The calculation must be reproduced for each level of each shelf to take into consideration the aging of LEDs and to avoid possible failures. Indeed, each shelf can be equipped by different combination of LED which are not known by the system. The algorithm must find the best combination with the minimal distance between resulting and targeted spectrum. The optimization is obtained under conditions of non-negative weighting. The convolution integration is calculated on the range 400 to 700 nm by 10 nm step. The spectra of seven CIE standard illuminants (D65, D50, A, E, F2, F11 and HP1) were used to validate the quality of spectra reconstructed.

V. CONCLUSIONS AND FUTURE WORKS

The wide range of applications of digital phenotyping shows that it is not possible to develop all applications on only one framework and need a set of frameworks on the same cluster. The optimal solution is to implement Apache Mesos, which allows a fine-grained use of resource of the cluster without drawback of multiple Virtual Machine (VM) or portioning of cluster.

We have proposed a better solution than using traditional ones like Hadoop. Our solution consists of a new lambda architecture based on Druid and an application sharing platform based on Mesos.

Druid allows to treat in quasi real time incoming data and make them available under second time. Our architecture can provide large range of data mixing old data archived in deep storage (HDFS) and recent data which are just treated but not yet archived. This aspect is particularly important for critical data which needs rapid processing and eventually reactions. The Lambda Architecture proposed can ingest a large panel of data such as time series, images, video, etc. This lambda Architecture is able to adapt to significant variation of quantity of data to treat at real time. Moreover, Druid's data ingestion latency is directly dependent of the complexity of the data set ingested [23].

The application sharing data proposed allows to host applications developed by scientist for specific application and to share them with the community on one hand, and on the other hand to test these applications with other data. The platform proposed is able to receive various kind of applications to exploit

[7] Apache Foundation, "Apache Hama", https://hama.apache.org June2017.

heterogenous data. Our application sharing platform uses Mesos which achieve a fine-grained allocation of resources in the cluster. Mesos allocates resources of available slave nodes in function of the requested tasks.

European legislation that will be translated into national law will require us to ensure the safety and traceability of the use of the data. We will need to develop a means to guarantee the security and confidentiality of the data exchanged between the lambda architecture and the application platform. It will be necessary to ensure that the data transmitted can only be used for the concurrent license duration and for the authorized applications.

VI. Acknowledgment

We would like to thank our colleagues from Biosystems Dynamics and Exchanges Axis, Biosystem Engineering Department, Gembloux Agro Bio-Tech (ULiège) without whom this work would not have been possible. We would especially like to thank Mr Rudy Schartz for his technical support and for setting up all the electronic systems necessary for carrying out this research.

VII. References

[1] L. Lei, Q. Zhang and D. Huang, "A Review of Imaging Techniques for Plant Phenotyping," *Sensor,* pp. 20075-20111, 2014.

[2] F. Golbach, G. Koostra, S. Damjanovic, G. Otten and R. Van de Zedde, "Validation of plant part measurements using a 3D reconstruction method suitable for high-throughput seedling phenotyping," *Machine Vision and Application,* pp. 663-680, 2016.

[3] M. P. Pound, A. P. French, J. A. Fozard and E. H. Murchie, "A patch-based approach to 3D plant shoot phenotyping," *Machine Vision and Application,* pp. 767-769, 2016.

[4] J. Mack, C. Lenz, J. Teutrine and V. Steinhage, "High-precision 3D detection and reconstruction of grapes from laster range data for efficient phenotyping base ond supervised learning," *Computers and Eelectronics in Agriculture,* pp. 300-311, 2017.

[5] F. Yang, E. Tschetter, X. Léauté, N. Ray and G. Merlin, "Druid. A Real-time Analytical Data Store," in *SIGMOD*, Snowbird, 2014.

[6] F. Yang, G. Merlino, N. Ray, X. Léauté, H. Gupta, and E. Tschetter, "The RADStack: Open Source Lambda Architecture for Interactive Analytics," in *The 50th Hawaii International Conference on System Sciences,* Hilton Waikoloa Village, 2017.

[7] E. Tschetter, "Introducing Druid: Real-Time Analytics at a Billion Rows Per Second," 30 April 2011. [Online]. Available: http://druid.io/blog/2011/04/30/introducing-druid.html.

[8] M. Díaz, C. Martín and B. Rubio, "State-of-the-art, challenges, and open issues in the integration of Internet of things and cloud computing," *Journal of Network and Computer Applications,* vol. 67, pp. 99-117, 2016.

[9] G. Bai, Y. Ge, W. Hussain, P. S. Baenziger and G. Grae, "A multi-sensor system for high throughput field phenotyping in soybean and wheat breeding," *Computers and Electronics in Agriculture,* pp. 181-192, 2016.

[10] B. Hindman, A. Konwinski, M. Zaharia, A. Ghodsi, A. D. Joseph, R. Katz, S. Shenker and I. Stoica, "Konwinski, M. Zaharia, A. Ghodsi, A.D. Joseph, R. Katz, S. Shenker, and I. Stoica," *Mesos: A platform for Fine-Grained Resource Sharing in the Data Center,* pp. 1-14, 2011.

[11] D. Merkel, "Docker: lightweight Linux containers for consistent development and deployment," *Linux Journal,* 2014.

[12] D. K. Krizek, J. C. Sager and T. W. Tibbitts, "Chapter 15 - Guidelines for Measurement and Reporting of Environmental Conditions," Iowa State University of Science and Technology, Ames, 2016.

[13] K. Shvachko, H. Kuang, S. Radia and R. Chansler, "The hadoop distributed file system," *IEEE,* pp. 1-10, 2010.

[14] C. S. K. Lrfu, "A spectrum of policies that subsumes the least recently used and least frequently used policies," *IEEE,* pp. 1352-1361, 2001.

[15] J. Zawodny, "Redis: Lightweight key/value Store That Goes the Extra Mile," Monday, 31st August 2009. [Online]. Available: http://www.linux-mag.com/id/7496/.

[16] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, I. M, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng and Google Brain, "TensorFlow: A system for large-scale machine learning," in *12th USENIX Symposium on Operating Sytemps Design and Implementation (OSDI'16)*, Savannah, 2013.

[17] W. Gropp, E. Lusk, N. Doss and A. Skjellum, "A high-performance, portable implementation of the MPI message passing interface standard," *Parellel Computing,* pp. 789-828, 1996.

[18] G. Lobet, X. Draye and C. Périlleux, "An online database for plant image analysis software tools," *Plant Methods,* pp. 1746-4811, 2013.

[19] F. J. Burgos, M. Vilaseca, E. Perales, J. A. Herrera-Ramírez, F. M. Martínez-Verdú and J. Pujol, "Reconstruction of CIE standard illuminants with an LED-based spectrally tuneable light source," in *Proceedings of the 12th International AIC Congress*, Newcastle-Gateshead, 2013.

[20] F. J. Burgos-Fernández, M. Vilaseca, E. Perales, J. A. Herrera-Ramírez, J. A. Martínez-Verdú and J. Pujol, "Spectrally tunable light source based on light-emitting diodes for custom lighting solutions," *Optica Applicata,* vol. XLVI N°1, pp. 117 - 129, 2016.

[21] M. Lukovic, V. Lukovic, I. Belca, B. Kasalica, I. Stanimirovic and M. Vicic, "LED-based Vis-NIR spectrally tunable light source – the optimization algorithm," *Journal of the European Optical Society-Rapid Publications,* vol. 19, pp. 1-12, 2016.

[22] M. R. Bonyadi and Z. Michalewicz, "Particle swarm optimization for single objective continuous space problems: a review," *Evolutionary Computation,* pp. 1-54, 2017.

[23] A. S. Veith, J. C. S. Anjos, E. P. Freitas, T. J. Lampoltshammer and C. F. Geyer, "Strategies for Big Data Analytics through Lambda Architectures in Volatile Environments," *IFAC-PapersOnLine,* p. 2016, 114-119.

[24] Apache Foundation, "Aurora is a Mesos framework for long-running services and cron jobs," 7th June 2017. [Online]. Available: http://aurora.apache.org.