Horizon 2020 Program (2014-2020)

Big data PPP

Research addressing main technology challenges of the data economy

Industrial-Driven Big Data as a Self-Service Solution

D5.6: Big-Data-as-a-Self-Service Test and Integration Report v3[†]

Abstract: The current deliverable presents the final version of the Big-Data-as-a-Self-Service Test and Integration Report. The work gives important perception towards the release of the envisioned I-BiDaaS solution, ensuring a smooth and effective integration of the separate I-BiDaaS components. It also presents the revised integration plan, which details how and when the individual technical elements of the I-BiDaaS solution are adapted and integrated in a common solution.

Contractual Date of Delivery	31/12/2020
Actual Date of Delivery	31/12/2020
Deliverable Security Class	Public
Editor	*Vassilis Chatzigiannakis (ITML)*
Contributors	ITML, UNSPMF, BSC, AEGIS, ATOS, FORTH, SAG
Quality Assurance	*Enric Pages (ATOS)* *Raül Sirvent (BSC)* *Kostas Lampropoulos (FORTH)*

[†] *The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 780787.*

The *I-BiDaaS* Consortium

Foundation for Research and Technology – Hellas (FORTH)	Coordinator	Greece
Barcelona Supercomputing Center (BSC)	Principal Contractor	Spain
IBM Israel – Science and Technology LTD (IBM)	Principal Contractor	Israel
Centro Ricerche FIAT (FCA/CRF)	Principal Contractor	Italy
Software AG (SAG)	Principal Contractor	Germany
Caixabank S.A. (CAIXA)	Principal Contractor	Spain
University of Manchester (UNIMAN)	Principal Contractor	United Kingdom
Ecole Nationale des Ponts et Chaussees (ENPC)	Principal Contractor	France
ATOS Spain S.A. (ATOS)	Principal Contractor	Spain
Aegis IT Research LTD (AEGIS)	Principal Contractor	United Kingdom
Information Technology for Market Leadership (ITML)	Principal Contractor	Greece
University of Novi Sad Faculty of Sciences (UNSPMF)	Principal Contractor	Serbia
Telefonica Investigation y Desarrollo S.A. (TID)	Principal Contractor	Spain

Document Revisions & Quality Assurance

Internal Reviewers

1. *Enric Pages, ATOS*
2. *Raül Sirvent, BSC*
3. *Kostas Lampropoulos, FORTH*

Revisions

Version	Date	By	Overview
0.0.1	9/9/2020	Editor	TOC
0.1	17/11/2019	Contributions by FORTH, BSC, AEGIS, UNSPMF, ATOS	1st draft
0.5	30/11/2020	Editor	1st draft
0.7	22/12/2020	Internal review	2nd draft
1.0	31/12/2020	Editor	Final

Table of Contents

List of Abbreviations

ADMM: Alternating Directions Method of Multipliers
AVT: Advanced Visualisation Toolkit
BDVA: Big Data Value Association
CI: Continuous Integration
CPU: Central Processing Unit
CRM: Customer Relationship Management
CVS: Concurrent Versions System
E2E: End-to-End
GPU: Graphics Processing Unit
GUI: Graphical User Interface
ISTQB: International Software Testing Qualifications Board
IOPs: Input/output operations per second
IoT: Internet of Things
ITU: International Telecommunication Union
JSON: JavaScript Object Notation
KPI: Key Performance Indicator
MES: Manufacturing Execution System
MIT: Massachusetts Institute of Technology
ML: Machine Learning
MQTT: Message Queuing Telemetry Transport
MVP: Minimum Viable Product
NIST: National Institute of Standards and Technology
OCCI: Open Cloud Computing Interface
OEE: Overall Equipment Effectiveness
PoCs: Proof-of-concepts
QA: Quality Assurance
REST: Representational State Transfer
RTC: Real-time Computing
SCA: Strong Customer Authentication
SCADA: Supervisory Control and Data Acquisition
SCM: Source Control Management
SOI: Software Oriented Infrastructure
SQL: Structured Query Language
SVN: Subversion
SW: Software
TDF: Test Data Fabrication
TOSCA: Topology and Orchestration Specification for Cloud Applications
UAT: User Acceptance Testing
UM: Universal Messaging

List of Figures

List of Tables

Executive Summary

The current deliverable presents the third and final version of the Big-Data-as-a-Self-Service Test and Integration Report. The work that corresponds to tasks T5.3, and T5.4, is the result of the continuous integration process that was described in the previous releases of the I-BiDaaS solution, ensuring a smooth and effective integration of the separate I-BiDaaS components, taking into consideration their availability, interoperability potential, scalability, and performance requirements.

Following a brief introduction, the second part of this document (Section 2) is dedicated to the delivery of the 2nd integrated I-BiDaaS platform. The 2nd integrated solution comprises an updated architecture with respect to the one presented within WP1, and a number of complementary use cases. To that end, Section 2 describes, in detail, the architecture and the updates with respect to the deployment of the I-BiDaaS generic use case; it also summarizes the I-BiDaaS components and technologies that have been integrated in the unified solution.

The integrated version comprises three (3) modes of operation with different functionalities, depending on the end user:

1. The Self-service mode allows industry in-house personnel that has domain knowledge and some insights about data analysis (non-experts), to easily construct Big Data pipelines in a user-friendly way, selecting a pre-defined data analytics algorithm from an available list.
2. The Expert mode allows experts (developers) to upload their own data analytics code based on the available I-BiDaaS highly reusable templates.
3. The Co-Develop mode corresponds to an end-to-end solution for a given industry project developed by the I-BiDaaS team (the I-BiDaaS use cases).

The use cases implemented, on top of the generic one, comprise: three use cases provided by CAIXA (enhanced control of customers to online banking, analysis of relationships through IP addresses and advanced analysis of bank transfer payment in financial terminal); two use cases provided by CRF (production process of aluminum die-casting and maintenance and monitoring of production assets); and three use cases designed by TID (accurate location prediction with high traffic and visibility, optimization of placement of telecommunication equipment, quality of service in call centres). All these use cases and their implementation in the integrated platform are described in detail in Section 3.

Section 4 presents the final testing and integration procedures, the main results of the integration testing of the I-BiDaaS platform. All tests have been passed and the integration has been successfully completed for all use cases. Section 5 presents the way I-BiDaaS technically approaches the sustainability of the platform, and Section 6 concludes this deliverable.

1 Introduction

The current deliverable presents the third and final version of the Big-Data-as-a-Self-Service Test and Integration Report. The work follows the previous release of the integrated I-BiDaaS solution based on a smooth and effective integration of the separate I-BiDaaS components, taking into consideration their availability, interoperability potential, scalability and performance requirements. It also provides insights about the final integration procedures followed and describes in detail the use cases implemented in this final integrated version and the exploitation of the I-BiDaaS technologies and components.

1.1 Relation to other Tasks and Work Packages

In general, this work provides important insights regarding the implementation and deployment of the I-BiDaaS' integrated framework, which is a large-scale computational infrastructure that allows real-time decision-making supporting analytics.

Besides, there is a close interrelation between this deliverable and Task 1.3 (Conceptual architecture specification) and Task 1.4 (Experimental protocol specification) in WP1 (Setting the scene: Baseline framework). It has been initially stated that the outcomes of Task 1.3 will be the starting point for Task 5.3 (Integration towards Big-Data-as-a-Self-Service). Particularly, in Task 1.3, a thorough understanding of the challenges, technologies and the state-of-the-art in terms of data economy, big data processing and end users' requirements have been successfully addressed. Thus, D5.6 has been structured based on some important insights taken from T1.3 (Conceptual architecture specification), as it has been the case for D5.4 as well.

Furthermore, as the final release of the envisioned I-BiDaaS solution had to fulfil all the key quality tests with respect to the industrial-validated benchmarks defined in Task 1.4, D5.6 is closely connected to Task 1.4. Finally, the I-BiDaaS evaluation and impact analysis (Task 6.3) is strongly aligned with Task 5.3; D6.1, D6.2 and D6.3 and have been following the I-BiDaaS prototype releases (D5.2, D5.4 and D5.6).

1.2 Contribution to the Scientific and Business Objectives

Having a central and core position within the whole concept of the I-BiDaaS solution, D5.6 and the related versions contribute towards the following I-BiDaaS objectives:

Objective 1: Develop, **validate, demonstrate**, and support, a complete and solid big data solution that can be easily configured and adopted by practitioners.

Objective 2: Construct a safe environment for methodological big data experimentation, for the development of new products, services and tools.

Objective 3: Develop data processing tools and techniques applicable in real-world settings and **demonstrate** significant increase of data speed throughput and access.

1.3 Structure of the Document

This deliverable is divided into the following sections: **Section 2** is dedicated to the description of the updates in the I-BiDaaS generic use case, the presentation of the updated / final architecture used in the integrated solution, and a brief description of the tools, technologies and components that were integrated in the I-BiDaaS platform. **Section 3** presents a detailed

description of the use cases provided by the I-BiDaaS end users that were implemented in the integrated platform, focusing on the enhancements and updates with respect to the previous integrated version. **Section 4** provides the final testing and integration procedures, and the results of the integration testing, which were all successful. **Section 5** presents the way I-BiDaaS technically approaches the sustainability of the platform. Finally, **Section 6** concludes this deliverable.

2 The 2nd I-BiDaaS Integrated version

2.1 2nd integrated version generic use case description

The functionality of the I-BiDaaS integrated platform can be divided into three categories with the following characteristics:

- The Expert mode allows experts (python developers with ML expertise) to upload their own data analytics code based on the available I-BiDaaS highly reusable templates.
- The Self-service mode allows industry in-house personnel with domain knowledge and some insights about data analysis (non-experts) to easily run Big Data analytics tasks in a user-friendly way, selecting a pre-defined ML algorithm from an available list.
- The Co-Develop mode corresponds to an end-to-end solution for a given industry project developed by the I-BiDaaS team (the I-BiDaaS use cases). The integration of these use cases is presented in Section 3 of this document, so it won't be shown in this Section.

2.1.1 Expert Mode

In the Expert mode, users can create new projects with their own code and run their own experiments, using the I-BiDaaS infrastructure. The high-level sequence of the expert-mode is the following:

- A registered user logs in to the web application and navigates to the Expert mode.
- A user creates a new custom project. A name, a description, and an input method (file, Cassandra keyspace[1] or no input) are required in order to create a custom project. Furthermore, the user must choose whether they will use the default I-BiDaaS docker image or another one from a public docker registry such as a Docker hub. The user must upload a zip file with the python code that will be used for this project.
- Custom projects belong to the user who created them and only they can run experiments on them.
- Running custom project experiments requires providing a name for that experiment, the command that should be used to run the custom code, and the input, if it is necessary for the specific project. Optionally, the user can select a pre-loaded custom visualisation, to be loaded after the completion of the experiment, provided in the form of a Jupiter notebook[2].
- These custom experiments comply with the following rules/functionality:
    - When running a new experiment, the user may select the resources to be utilized in the experiment, in terms of number of CPUs and available memory per CPU.
    - Experiments that have already taken place are available after selecting a specific project.
    - They can be deleted.
    - Their logs can be downloaded.

[1] https://docs.datastax.com/en/dse/5.1/cql/cql/cql_reference/cqlsh_commands/cqlshDescribeKeyspace.html

[2] https://jupyter.org/

To aid the users in providing compatible code, two code templates have been created and are provided as downloadable zip files during the initialization of a custom project. The code templates are depicted in the following table:

Table 1: Code templates for Expert mode

Name	Description	Executable script	Command to run
Random forest	A random forest implementation based on dislib[3], the example code loads a predefined dataset stored within the library. It prints the predicted values in the stdout (application_log.out).	main.py	main.py
ADMM Logistic Regression	A logistic regression algorithm. The example code runs with 2 workers and the dataset provided with the code. It prints the dimensions of the dataset and the total elapsed time.	Main_LogReg_ADMM.py	Main_LogReg_ADMM.py 2

2.1.2 Self-service Mode

In the Self-service mode, the users can run experiments in predefined projects with already implemented algorithms (*e.g.*, K-means). Then, they create and execute their own experiment(s), based on their data and preferences. The high-level sequence of the self-service-mode is the following:

- The user logs in to the web application and navigates to the Self-service mode.
- In the Self-service mode, algorithm/project selection is assisted by a wizard, in the form of a multi-step questionnaire. It is designed to provide users with a recommendation on which algorithm to use based on their dataset.
- A user selects a predefined project and chooses to run a new experiment. Then, the user is required to provide the parameters, which vary depending on the project. Once these parameters are provided, the experiment starts. Although these projects are accessible by every user, the experiments and their results belong to the user who initiated them and are not visible to other users.
- The user can choose the input for each experiment either by selecting a file, which has been pre-uploaded to the server, or a Cassandra keyspace. The available files and keyspaces are provided by a dropdown list.
- While the experiment is active, its status remains on "running". Once it has finished, the user can download the logs of the experiment, including any output file(s).
- The results of the experiments are also represented visually. Visualisation differs depending on the project.

The predefined projects in the Self-service mode are depicted in the following table:

[3] https://www.bsc.es/research-and-development/software-and-apps/software-list/dislib

Table 2: Self-service projects/algorithms

Name	Description	Input type	Visualisation
Random forest	Random forest is an ensemble learning method for classification and regression. The code is based on dislib, requires a labelled dataset provided by the user, creates an ensemble and prints its evaluation (predictions vs actual values on the provided dataset).	Labelled csv file with scalar fields. Last field is the objective field containing the labels.	Prints prediction values.
Decision Tree	A decision tree is a decision support tool that uses a tree-like model of decisions and their possible consequences, including chance event outcomes, resource costs, and utility. The code is based on dislib, requires a labelled dataset provided by the user, creates an ensemble and prints its evaluation (predictions vs actual values on the provided dataset).	Labelled csv file with scalar fields. Last field is the objective field containing the labels	Prints prediction values.
K-means Prediction	K-means Prediction algorithm can be used for clustering a dataset with scalar attributes.	A csv file with scalar attributes.	A 2D Visualisation of the clusters and their centroids.
K-means Evaluation	K-means Evaluation algorithm can be used for training a K-means classifier, using a labelled dataset of scalar attributes.	A labelled csv file with two attributes and the cluster id.	A 2D Visualisation of the clusters and their centroids.

2.2 2nd integrated version components and technologies

2.2.1 I-BiDaaS Architecture

The I-BiDaaS 2nd integrated version includes the modules identified in the complete I-BiDaaS architecture (Figure 1). The description of the final state of the Batch Processing module, the Streaming Analytics module, the Visualisation module and the Orchestrator module that binds them together are presented in the following sub-sections.

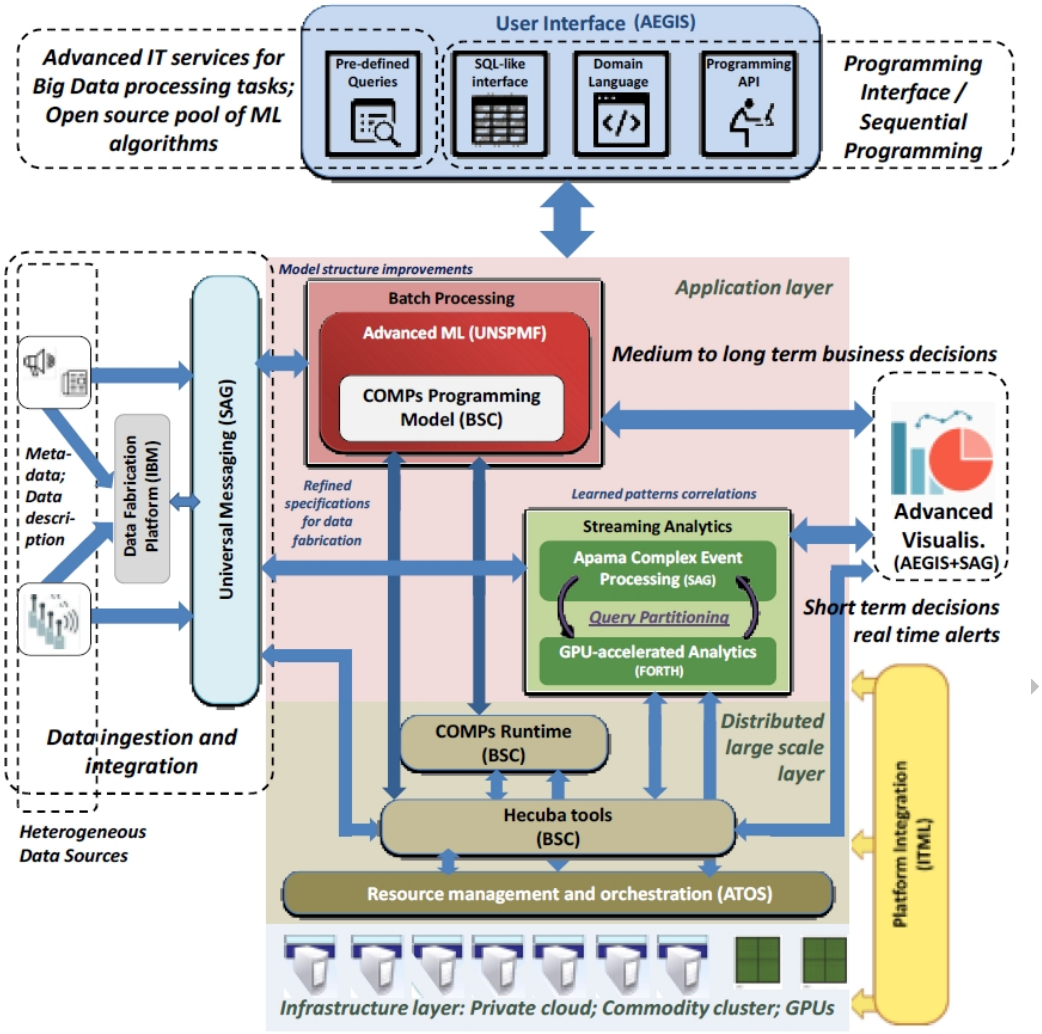

Figure 1. The I-BiDaaS architecture

2.2.2 Batch Processing in the 2nd integrated version

The Batch Processing Module of the I-BiDaaS architecture has been developed in the context of WP3, and its main developments for the second integrated version can be found in detail in *D3.3: Batch Processing Analytics module implementation final report*. A summary of the new developments and innovations is included in this section.

As depicted in Figure 1, the Batch Processing Module is one of the core pieces of the Application Layer and includes a set of components and algorithms that will be described in later paragraphs. Besides, as one of the central components in the architecture, its connections to the different layers are important, such as:

- The distributed large-scale layer: where storage takes place
- The user interface: a portal for end users' interaction
- The data ingestion and integration subsystem: how data is ingested in the system and fabricated
- The streaming analytics module: a real-time processing module

The Batch Processing Module is composed of two main parts:

- A set of Algorithms that the users can leverage to conduct their data analyses. They are based on Machine Learning techniques, and can leverage different algorithms from already existing libraries, such as dislib or scikit-learn[4].
- A group of software components that enable the development and execution of the aforementioned algorithms. Those components are:
    - COMPSs[5]: A parallel programming model with a sequential programming paradigm, able to transparently run applications in distributed environments.
    - Hecuba[6]: An easy interface to a variety of database storage solutions.
    - Qbeast[7]: A Multidimensional Storage with Efficient Sampling (MuSES) patented technology that includes faster access to data, together with interactive data exploration.
    - TDF[8]: The Test Data Fabrication component, able to automatically generate synthetic data from a set of rules, which can be used for testing purposes.

For each of the parts included in the Batch Processing Module, different novelties have been implemented. The following paragraphs will give a brief overview of "*what's new*" in this second version of the Module.

New Machine Learning Algorithms – The set of data analytics algorithms provided in the platform has continued growing by either developing algorithms from scratch or leveraging existing ones, even modifying them to achieve improvements. The new set of algorithms included is:

- ADMM based K-means
- ADMM based logistic regression
- Stochastic gradient-based logistic regression
- ADMM based Ridge and ElasticNet regression
- Differentially private K-means
- Differentially private Random Forest

This list enlarges the already available list of algorithms:

- K-means (dislib[9])
- DBSCAN (dislib)
- t-SNE (I-BiDaaS Knowledge Repository[10])
- PCA (scikit-learn)
- K-modes (K-modes library)
- Elbow method
- Time-series based on Prophet (I-BiDaaS Knowledge Repository)
- Random Forest (dislib)
- XGBoost
- CATBoost

[4] https://scikit-learn.org/stable/

[5] https://www.bsc.es/research-and-development/software-and-apps/software-list/comp-superscalar

[6] https://github.com/bsc-dd/hecuba

[7] https://qbeast.io/

[8] https://www.ibm.com/il-en/marketplace/infosphere-optim-test-data-fabrication

[9] https://www.bsc.es/research-and-development/software-and-apps/software-list/dislib

[10] https://github.com/ibidaas/knowledge_repository

- Weighted Random Forest (I-BiDaaS Knowledge Repository)
- Deep Neural Network (I-BiDaaS Knowledge Repository)
- DenseNet-201
- Interquartile Range Test (IQR)

Hecuba integrated into dislib – the dislib library works with the numpy *ndarray* data structure, and Hecuba provides storage capacities for numpy *ndarrays* with a structure called *StorageNumpy*. The integration allows the dislib algorithms to remain unchanged while transparently exploiting the Hecuba storage enhancements with Cassandra, such as saving disk space by using references to the database.

Qbeast visualisation tool – with this visualisation tool, an interactive analysis workflow is achieved (i.e. users can provide feedback to perform new analyses, leading to a tuning of algorithms and models used). The tool is based on the Jupyter ecosystem[11] (Voila and Widgets), together with additional JavaScript frameworks. The users invoke the interactive visualisation through a web browser, and when they interact with the visual component, the application status is updated (triggering new computation when needed). Figure 2 shows the architecture of the visualiser, which also leverages COMPSs, dislib and Hecuba.

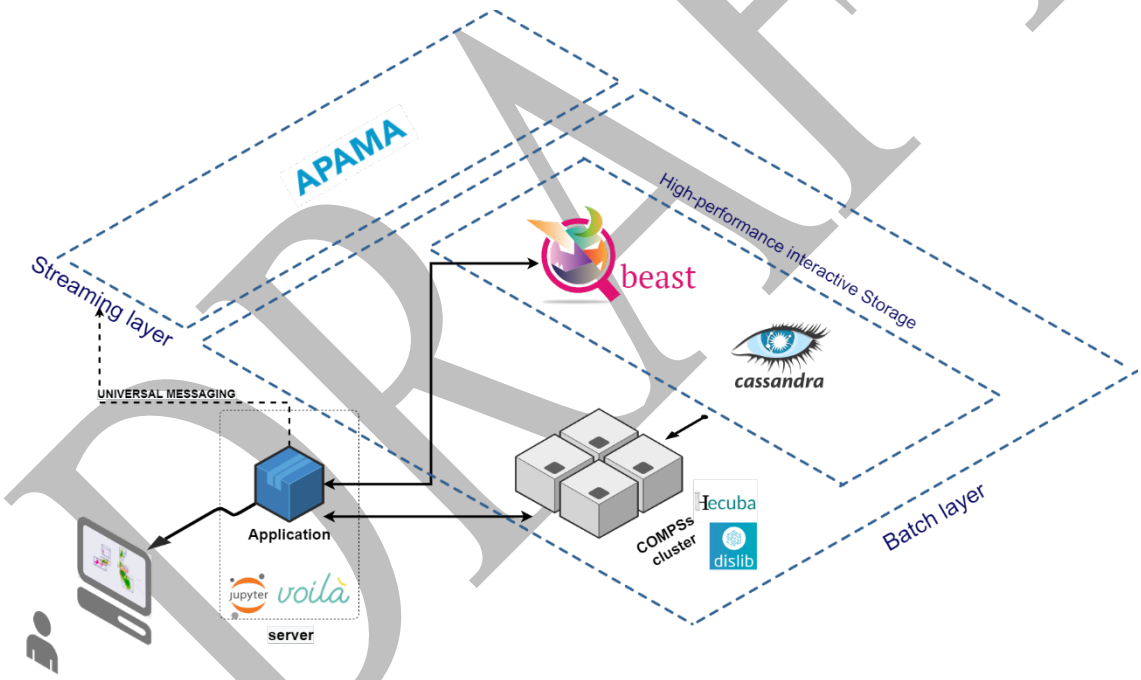

Figure 2. I-BiDaaS interactive analysis architecture

Data statistics fed to TDF – the data fabrication feature provided in I-BiDaaS from the Test Data Fabrication (TDF) component has been enhanced through a preliminary data characterization step, where different descriptive statistics of the real set of data are collected. Now, TDF is able to process these statistics to automatically generate a number of constraint rules that can be used for the automatic generation of synthetic data.

Integration between Batch Processing and Streaming Analytics – while in the first version of the I-BiDaaS Toolbox, the Batch Processing and Streaming Analytics modules worked

[11] https://jupyter.org/

independently, in this second version, we have enabled the communication between these two modules to provide extra functionalities in terms of being able to store data coming from a stream for analysing it later in a batch manner. A connector module listens to the MQTT queue and stores the data in the selected database (i.e. Cassandra). This has been done using the Hecuba framework for an easy access to the storage system.

2.2.3 Streaming analytics in the 2nd integrated version

The Streaming Analytics Module of the I-BiDaaS architecture has been developed in the context of WP4, and its main developments for the second integrated version are described in detail in Deliverable "D4.3: Streaming Analytics and Predictions". In the current section, we present the summary of the new developments and innovations.

The architecture of Streaming analytics use cases is depicted in Figure 3. Input streams from external sources insert data into the I-BiDaaS platform, in the form of JSON messages that are published in the Universal Messaging (UM). The Universal Messaging[12] is a mature and stable commercial product, part of SAG's Enterprise Service Bus that provides a message broker supporting multiple message-passing standards like JMS, AMQP and especially MQTT, an ISO standard. For the I-BiDaaS project, the MQTT has been chosen as a message passing standard. It can be operated in a cluster to provide high availability and reliability. The UM servers contain common messaging resources, such as channels/topics or queues.

For each use case, a pair of input and output topics are created. The external source publishes messages to the input topic. Streaming analytics subscribes to the input topic and waits for new messages. For each message it receives, it performs the inference (e.g., prediction/classification) depending on the use case. The inference output is then published to the output topic of the use case. The Advanced Visualisation Toolkit (AVT) subscribes to the output topic, receives the inference data and visualises it.

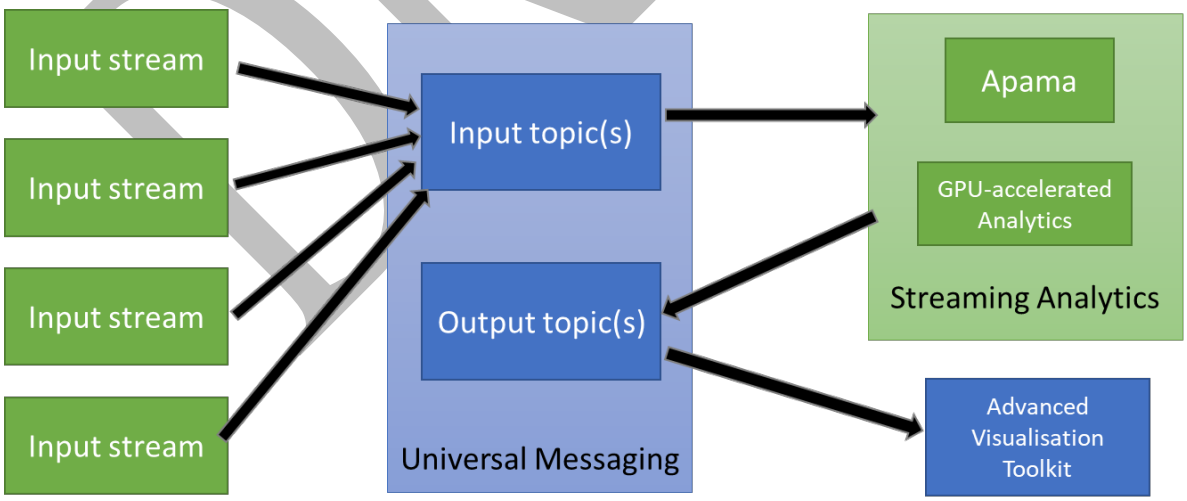

Figure 3. Integration of Streaming analytics

[12] http://www.universalmessaging.org

2.2.4 Visualisation in the 2nd integrated version

The Visualisations of the final integrated version of the platform have accommodated all the feedback received during the implementation of the experiments by the pilots (as reported in D6.4) and the online workshops that were held for all the use cases in the domains of I-BiDaaS. Based on the concept of setting up a project, running multiple experiments and visualizing the results that were implemented on the first integrated version, the platform interface was extended with the visualisation elements required to cover the needs of all use cases.

Figure 4 depicts the final internal architecture of AVT and the communication channels with other components of the platform. It must be noted that the modularity of the component allowed its deployment in both the environments of the general cloud-based I-BiDaaS platform as well as the internal infrastructure of TID (for the Quality of Service in Call Centres use case).

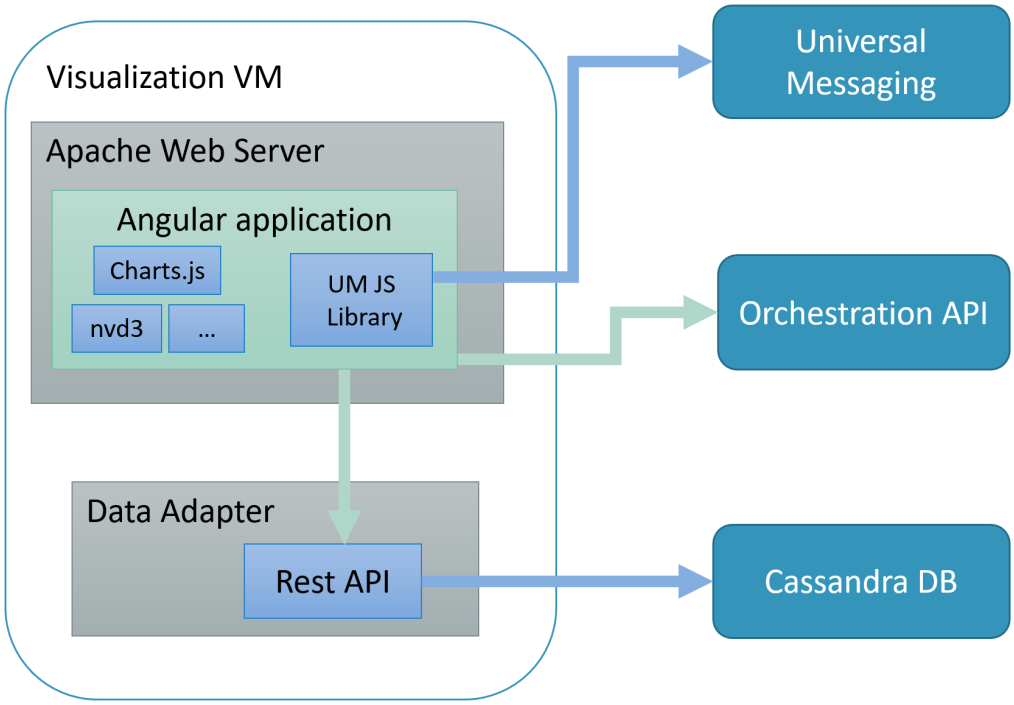

Figure 4. AVT Internal Architecture

The following paragraphs provide an overview of the basic functionalities of the Visualisation component (already reported in detail in D6.4) for completeness and mainly describe final additions that were introduced to address feedback received after the final execution of experiments. The updates based on the three operation modes, namely Expert, Self-Service and Co-develop are depicted in Figure 5 and described in the following subsections.

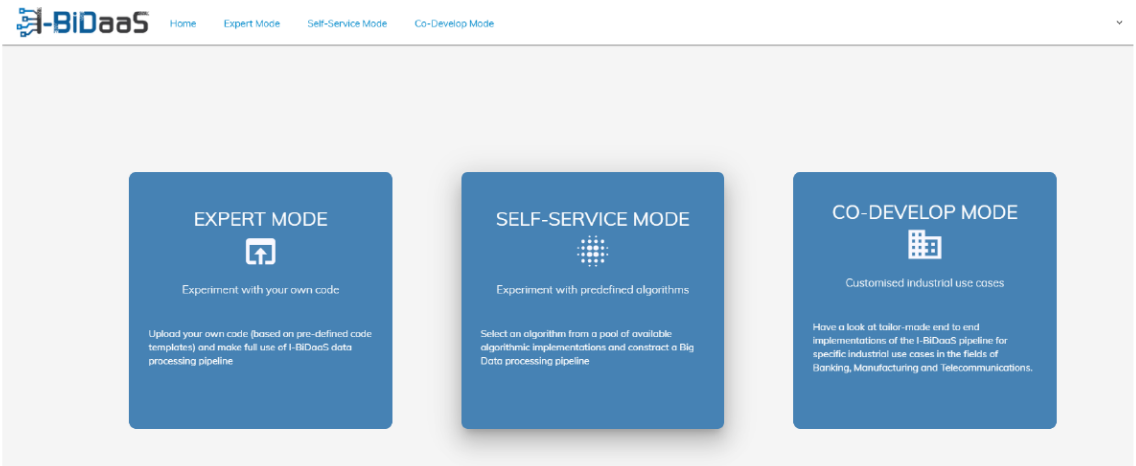

Figure 5. Operation Modes

2.2.4.1 Expert Mode

The Expert mode allows the experts (developers) to upload their own data analytics code based on the available I-BiDaaS highly reusable templates. They can point to an existing public docker image or use the default one as the execution environment and furthermore define the data source for the experiments. One of the latest added features is the capability to link existing Jupiter Notebooks and visualize the results in Voila dashboards (Figure 6). Finally, the complete execution results (logs, statistics and output) are available for every experiment.

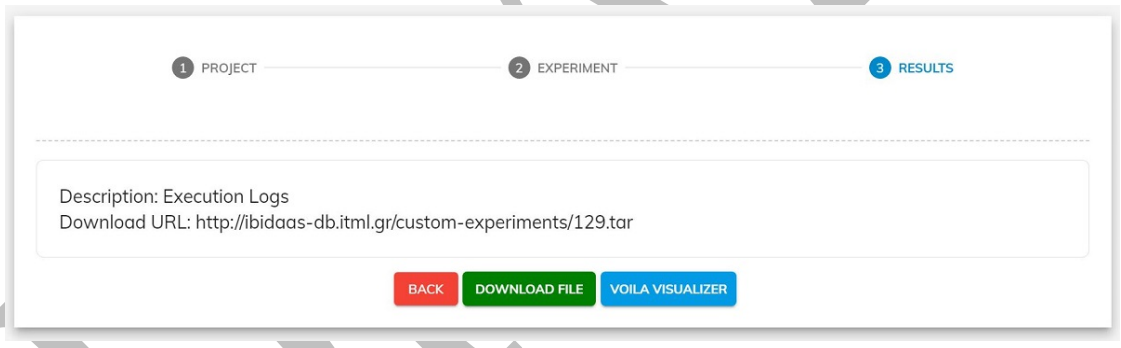

Figure 6. Jupiter Notebooks in Expert Mode

2.2.4.2 Self-Service Mode

The Self-service mode allows users having the relevant domain knowledge and some experience about data analysis (non-experts), to easily construct Big Data pipelines in a user-friendly way, selecting a pre-defined data analytics algorithm from the available list. The final list contains five implemented algorithms (Figure 7) that can be used to execute experiments on datasets defined by the users. For every algorithm, the users can also configure the available parameters, visualise the results in appropriate visualisations and finally download results for further processing outside I-BiDaaS (Figure 8).

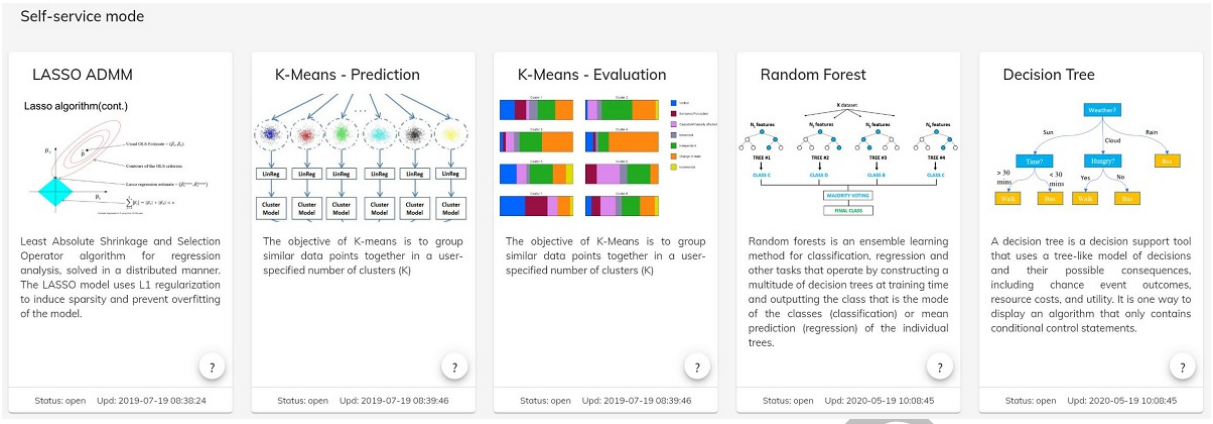

Figure 7. Self-service algorithms

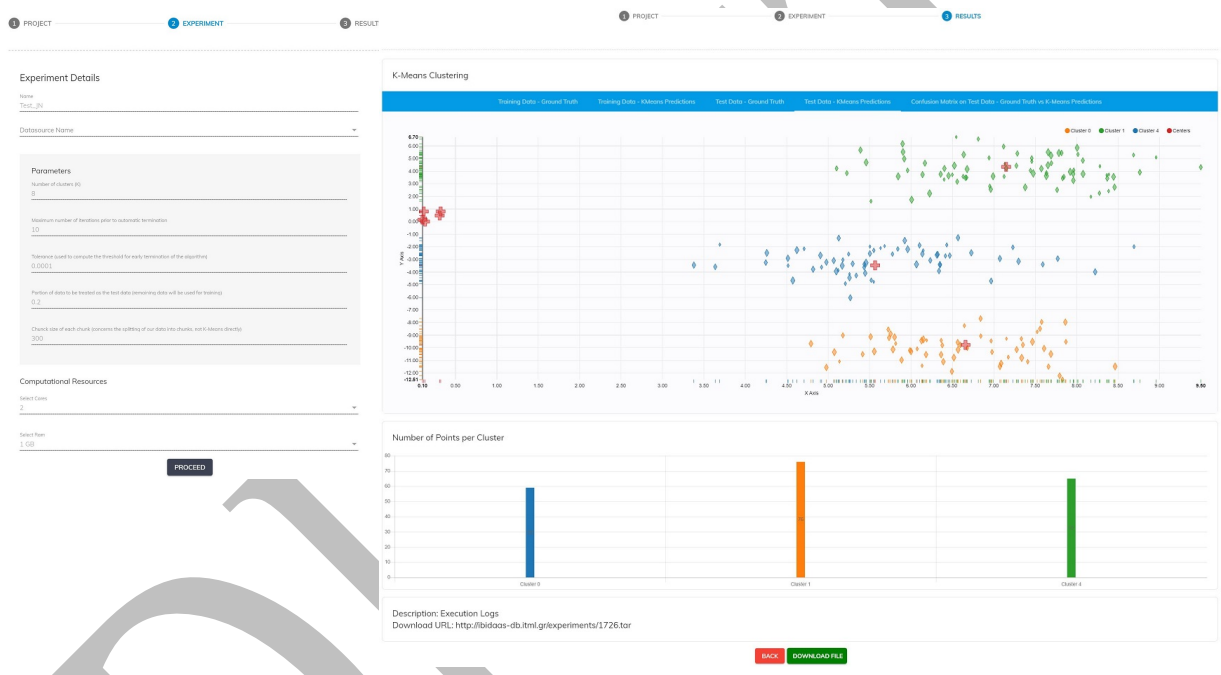

Figure 8. Algorithm Setup and Results

2.2.4.3 Co-Develop Mode

The Co-Develop mode corresponds to an end-to-end solution for a given industry project developed by the I-BiDaaS team (the I-BiDaaS use cases). The final integrated prototype includes all the use cases provided by CAIXA, CRF and TID (Figure 9). Visualisations for all these use cases have been tailored to suit their requirements and include a variety of visualisation elements such as network graphs, real-time updated line charts and pie charts, spatial representations and interactive maps. The following figures present characteristic visualisations for all the I-BiDaaS domains.

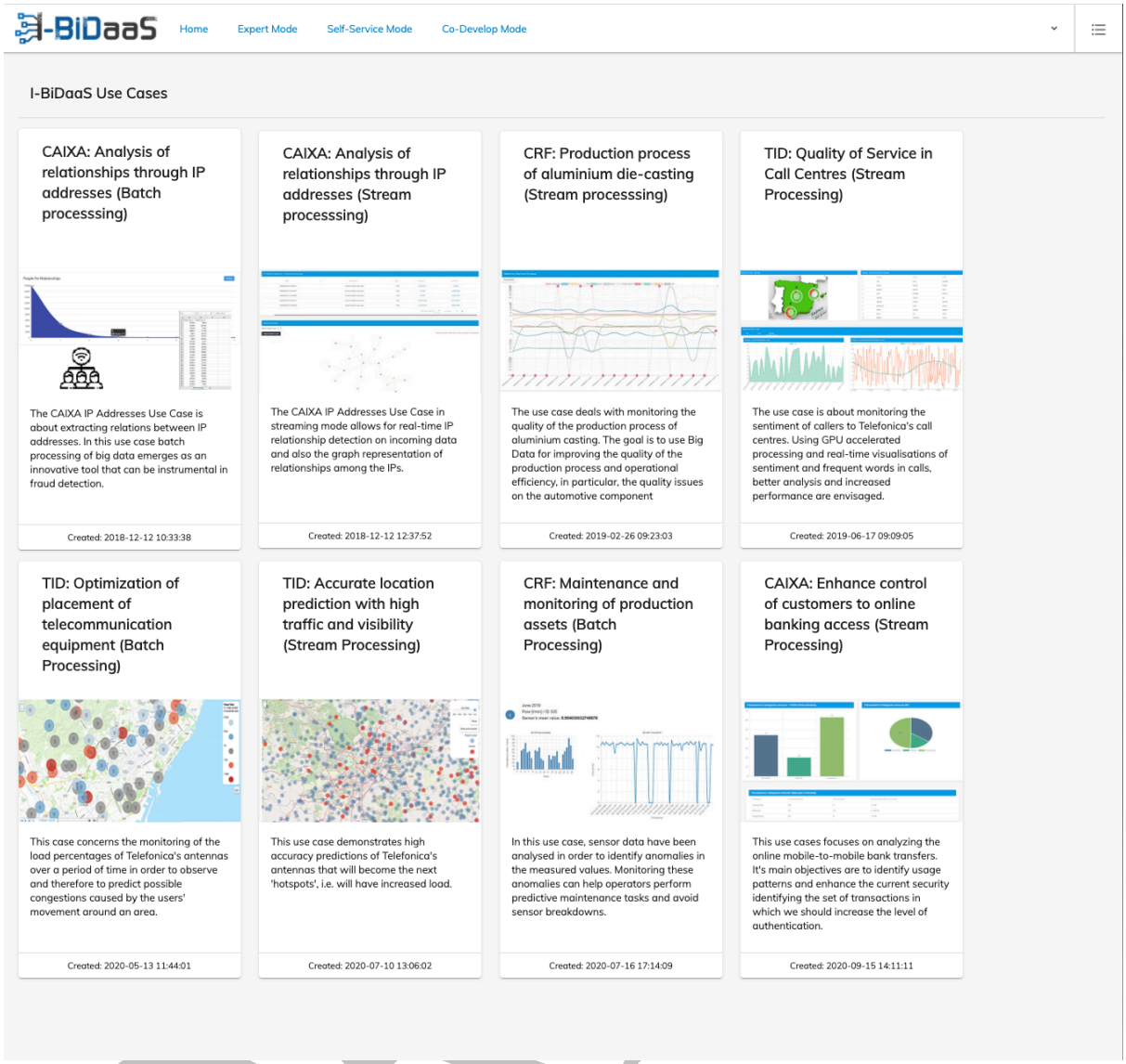

Figure 9. Co-Develop Use Cases

Figure 10. Visualisations in Financial domain

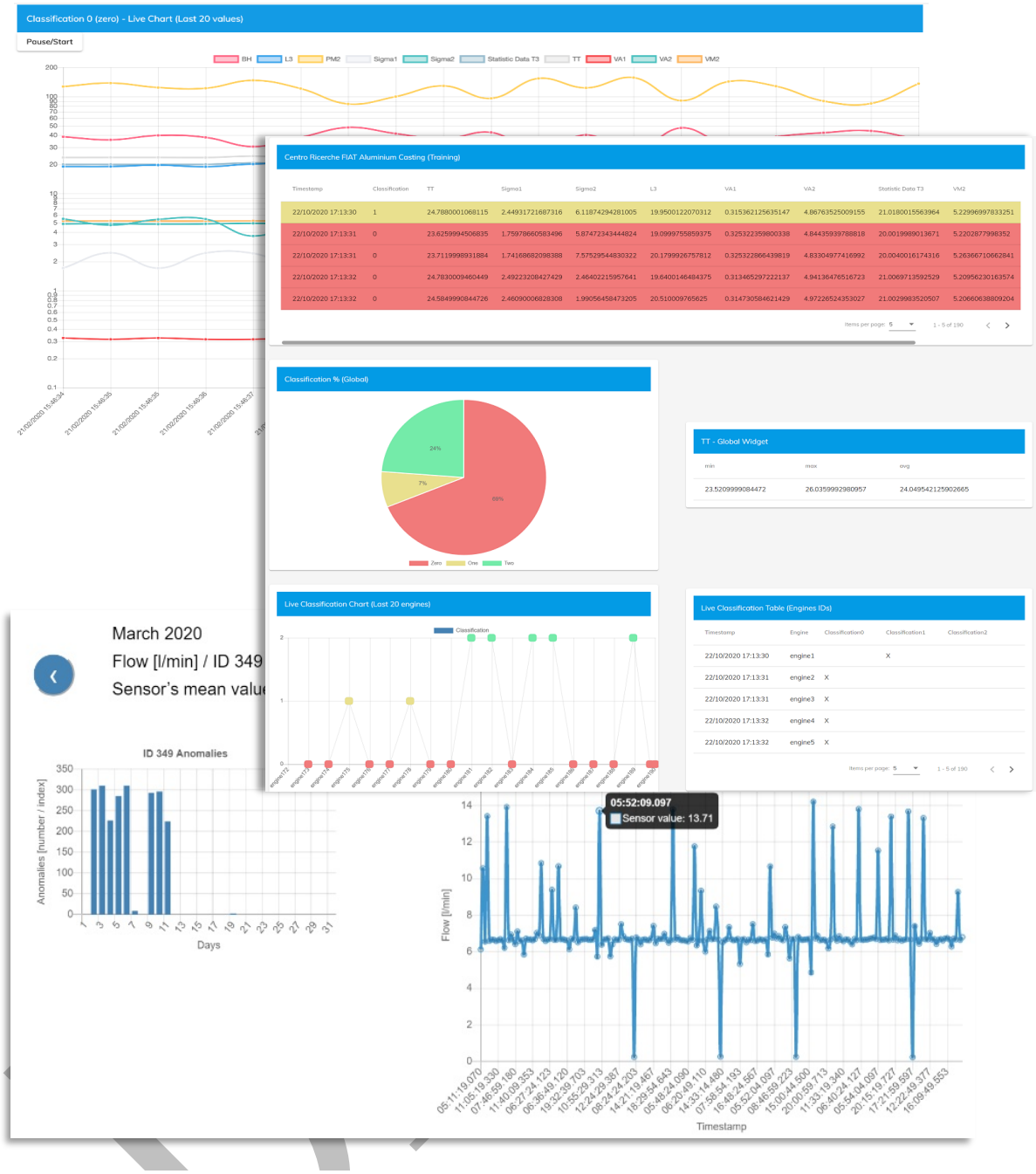

Figure 11. Visualisation in Manufacturing domain

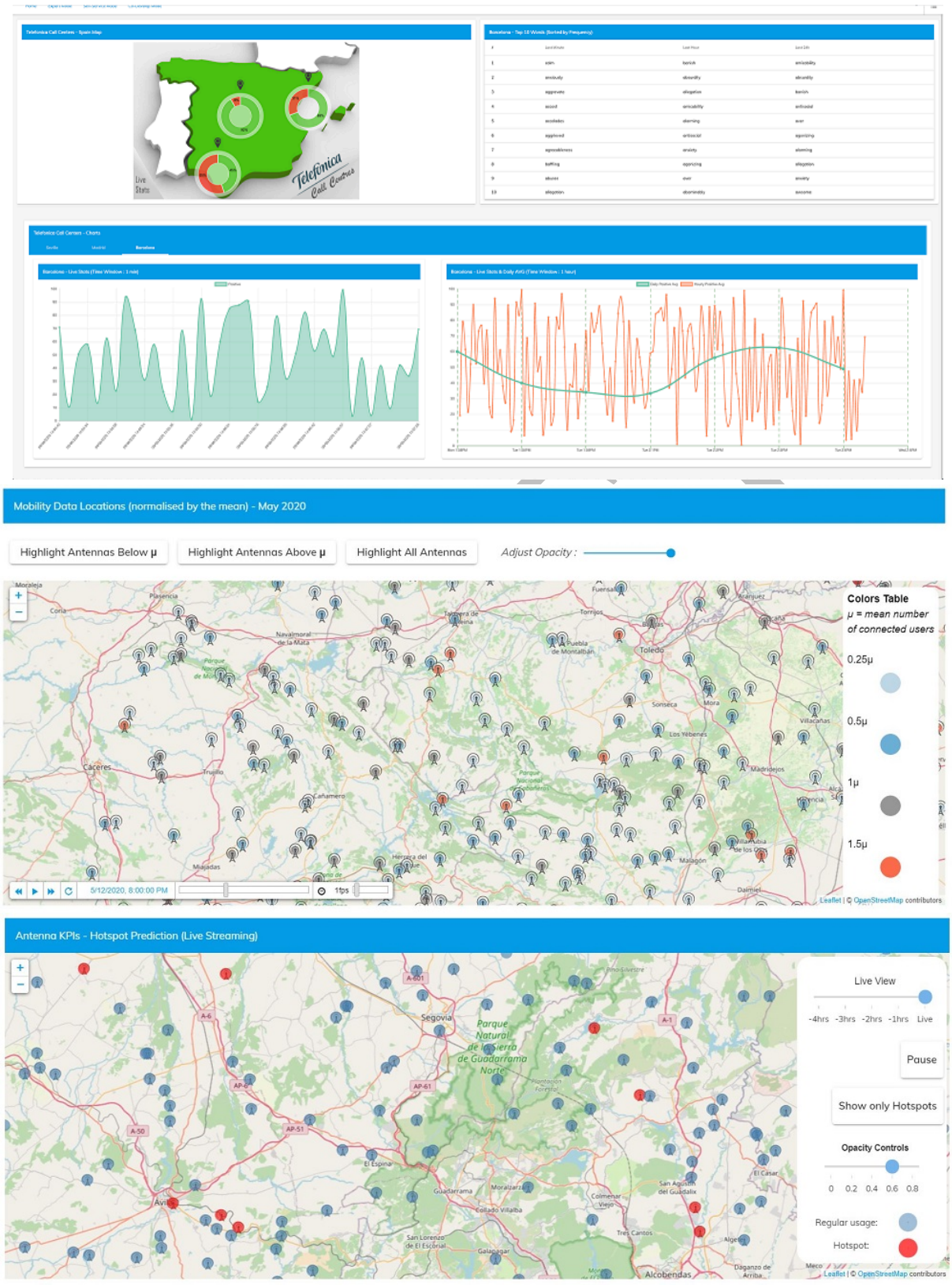

Figure 12. Visualisations in Telecommunications Domain

2.2.4.4 Utilisation of SAG's Mashzone for visualisation purposes

SAG's MashZone[13] tool offers the functionality of connecting arbitrary data sources and from a set of existing visualisation widgets collecting the ones that can better visualize the linked data. During the last period of the project, AEGIS and SAG experimented with how data utilized in AVT could be introduced to MashZone. We have conducted a number of tests to see if there is an effective way of letting the user pass the results obtained via the analysis with the I-BiDaaS platform to MashZone's editors. A direct link between the two tools was not feasible to be established. Therefore, in order to use resulting data in MashZone a user would have to obtain the canonical URL of a resulting dataset from AVT, manually open the so-called 'Data Feed Editor' of MashZone and then import the feed using the relevant function. This course of actions can enable cooperation of the two tools, however, feedback from pilot experiments and users on the ways that I-BiDaaS-based data analysis results can be utilized in business pipelines led us to the conclusion that no further action should be taken at this stage of the project.

This decision was also backed by the fact that the expert users (who would be the main target group of such functionality) would mostly benefit from the existence of community-based tools of higher adoption rate in the data analytics domain, such as the Jupyter Notebooks. Therefore, it was decided to integrate exactly these tools and accompanying visualisation capabilities rather than further pursuing a tighter integration with MashZone.

2.2.5 Orchestration in the 2nd integrated version

The orchestrator is a middleware between the AVT and the rest of the components. It exposes a REST API that can be used to create new projects and experiments in all I-BiDaaS modes (Expert, Self-Service and Co-Develop mode):

The Orchestrator provides:

- **Creation and configuration of experiments**: Code uploading, data selection, storing and validating configuration per experiment)
- **Scalability**:
  - Configuration of nodes/core number, RAM selection per experiment
  - Capability of adding/removing nodes to the cluster, without user intervention
- **Security/isolation**: Every experiment runs in docker containers that are destroyed after experiment completion
- **Self-service-ness**:
  - Users do not need to maintain a COMPSs environment in the machines
  - Users can experiment on ready-to-use docker images available both in the I-BiDaaS docker registry and in the docker hub[14] with templates/examples
- **CRM Features**
  - User provisioning
  - Authentication, Authorization, Accounting
  - History (Custom project management, Experiment history)

[13] https://www.ariscommunity.com/university/downloads/aris-mashzone

[14] https://hub.docker.com/repository/docker/vchatzi/ibidaas-universal

The REST API is implemented in PHP Laravel[15], supported by Mysql RDBMS[16]. Its functionality, in terms of the available endpoints, is provided in Appendix 1.

Focusing explicitly on batch processing, the system architecture is shown in Figure 13. The cluster that runs the batch processing jobs is based on docker swarm[17]. The swarm consists of a manager node and a set of worker nodes. The orchestrator dynamically assigns a set of workers from the set of available nodes in docker swarm, based on the user's preferences and set-up inserted through the AVT. These workers exploit the Cassandra DB and the shared filesystem in order to complete the requested job.

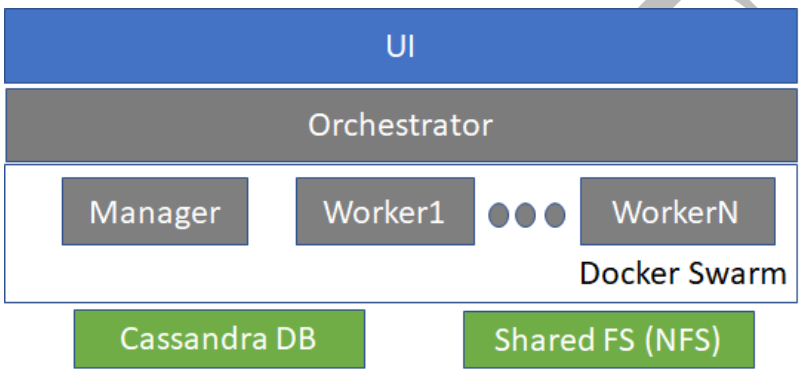

Figure 13. Batch processing architecture: The orchestrator and the docker swarm

The Orchestrator provides integration between AVT and the docker swarm executing the requested experiments. The sequence diagram for running a new batch experiment is depicted in Figure 14.

[15] https://laravel.com/

[16] https://www.mysql.com/

[17] https://docs.docker.com/engine/swarm/

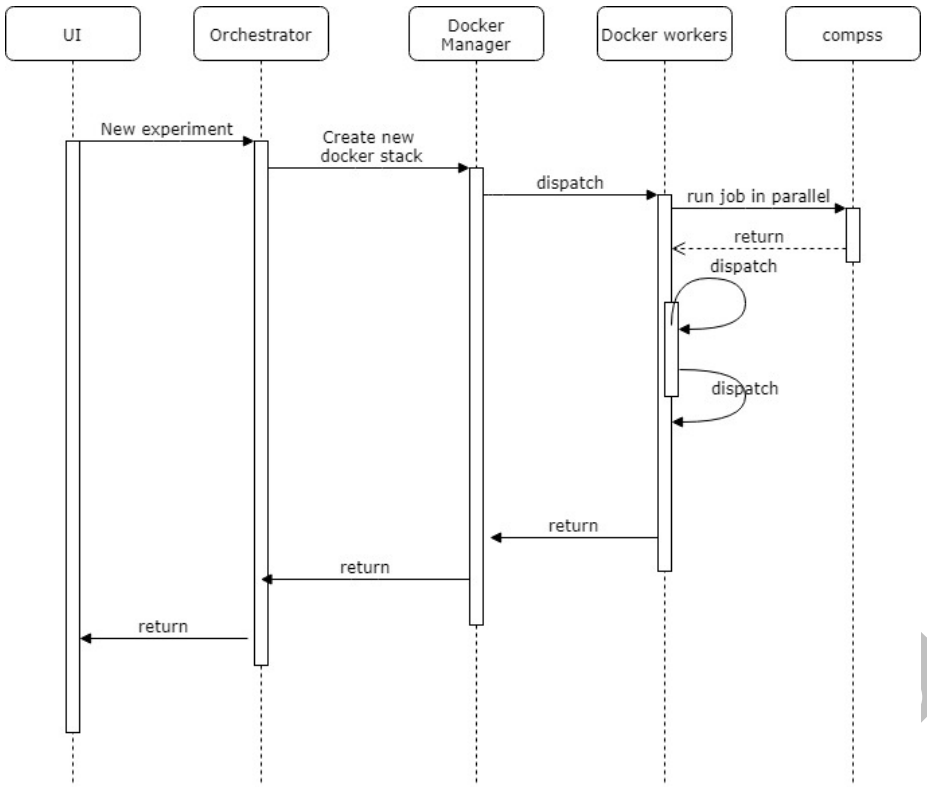

Figure 14. The Generic use case process sequence diagram

On the other hand, Figure 15 demonstrates the case of stream processing; here, the analysis is carried out through APAMA[18] (or sometimes custom python scripts in cases where using APAMA is not technically feasible) in collaboration with the UM tool, rather than via docker swarm.

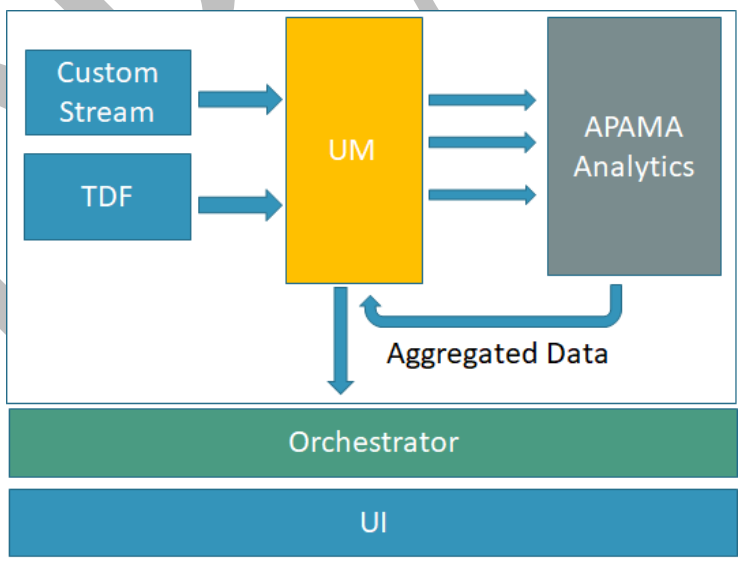

Figure 15. Streaming processing architecture: The orchestrator and APAMA analytics

The relational model of the MySQL database supporting the REST API is depicted in Figure 16. The projects that each user can initiate have a specific code (an implementation of the I-

[18] http://www.apamacommunity.com/

BiDaaS use case) or a generic algorithm implementation, *e.g.*, K-means. In Custom projects, users can upload and run their own code (Expert mode).

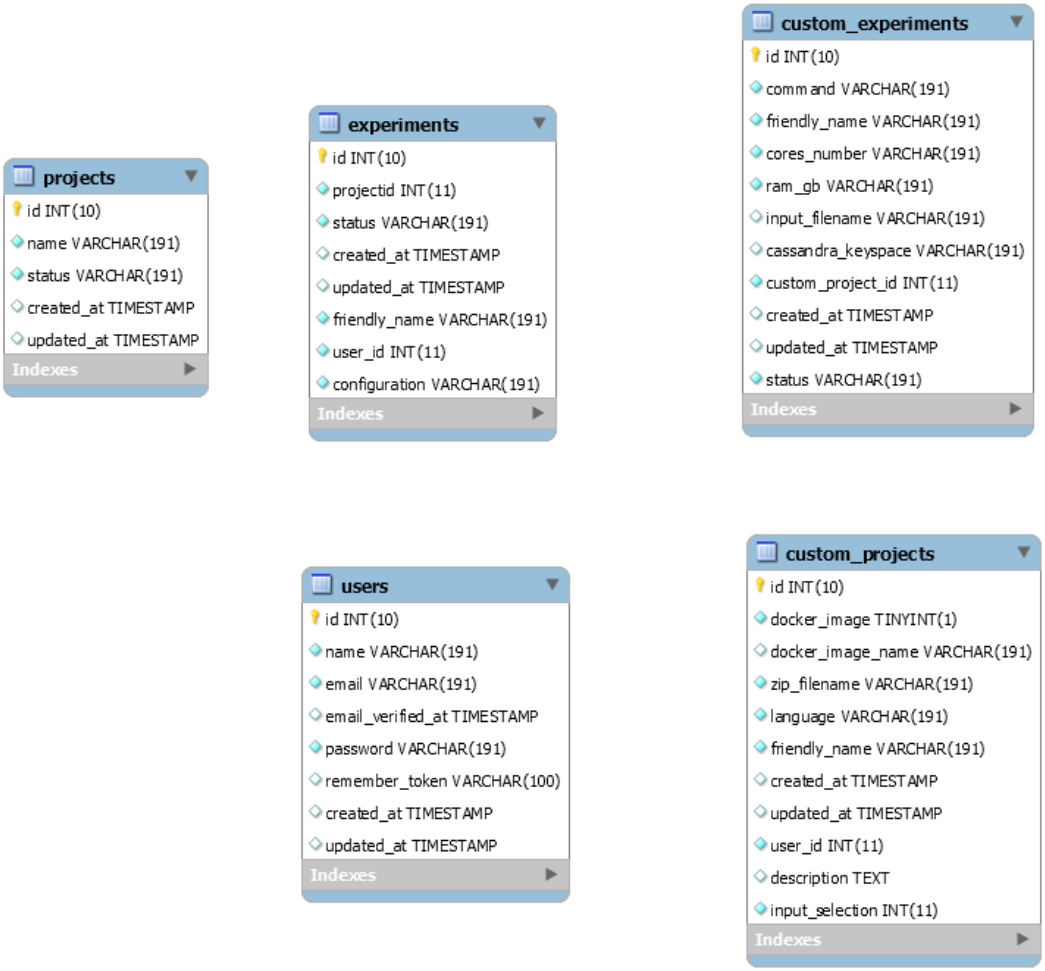

Figure 16. The I-BiDaaS generic use case database structure

3 Report on the 2nd integrated version per Use Case

3.1 Accurate location prediction with high traffic and visibility

This use case, which was included for the first time in the 2nd Integrated version of the I-BiDaaS Platform, aims to analyse the behaviour of local and non-local customers over various periods of time (e.g., holidays), and extract insights on the behavioural patterns of groups of people, enabling them to optimize their value propositions. When users travel around the city, they create traffic congestions in the network, so it would be useful to immediately forecast next events to anticipate movements at scale, improve the routing and placement of the telecommunication equipment that is already in place, or to arrange accordingly the new equipment obtained.

The input data used for the use case include both synthetic and an anonymized, encrypted real data set containing more than 120,000 samples. Each sample represents a time series for an antenna for a period of one month. The points in the series are divided into periods of four hours, resulting in 186 points in total. The values inside the time series represent the number of users on a particular antenna at that point of time. The input data format is JSON, where each antenna represents a JSON object. A point of interest is to track the number of users on the network, as well as to predict the movements of the users, i.e., to predict the number of users that will appear on various network parts. Some of the important challenges regarding this use case are: the interpolation of missing events, the minimization of the processing time with respect to growing data size and the process of maintaining real-time delivery of results.

A time series model was trained on these data. There were two possible approaches: to train the model per antenna or per time series. For this purpose, the state-of-the art time series model tool Prophet[19] (the Python version), developed by Facebook AI, was used. The tool represents a procedure for time series data forecasting, where the process is based on an additive model. The data pre-processing was conducted using the Python library pandas[20]. The pre-processing only implied joining all the data into one large table. The rows in the table represented the antennas, and the columns represented the time points. The time series were split so that the last five points were used for validation, where the rest of the data were used for training purposes. After fitting and testing with Prophet, the forecast values were compared to the actual ones, and the mean absolute error was obtained. The training process was parallelized by using the Python library joblib[21]. A process-based parallelization was used, where a CPU core was training the model for one antenna. Regarding the volume of data, 120,000 time series needed to be trained. This was achieved in two hours on the TID server. The mean average error served as a basis for selecting the best 1000 models that could be used for visualisation and prediction. A baseline metric of the average mean absolute error equals to 1.2565 was obtained. The predictions deviated by 1 – 2 users compared to the real values. This way, it is possible to predict whether an antenna will experience a rise of users that are connected to it. However, these predictions cannot be directly related to particular events or to the rise of users on the neighbouring antenna. This possibility remains open for future work. Although the achieved results are satisfactory, a more accurate model could probably be obtained by additional pre-processing steps applied.

[19] http://facebook.github.io/prophet/

[20] https://pandas.pydata.org/

[21] https://github.com/joblib/joblib/

This use case is implemented as a batch use case, integrated in the Co-Develop mode and available in the I-BiDaaS platform instance located within Telefonica infrastructure for privacy reasons. It was not available in the 1st integrated version of the I-BiDaaS Platform. Figure 17 shows the visualisation of the results on the I-BiDaaS platform, utilised in the 'Co-Develop mode', with a mean number of connected users highlighted in different colours.

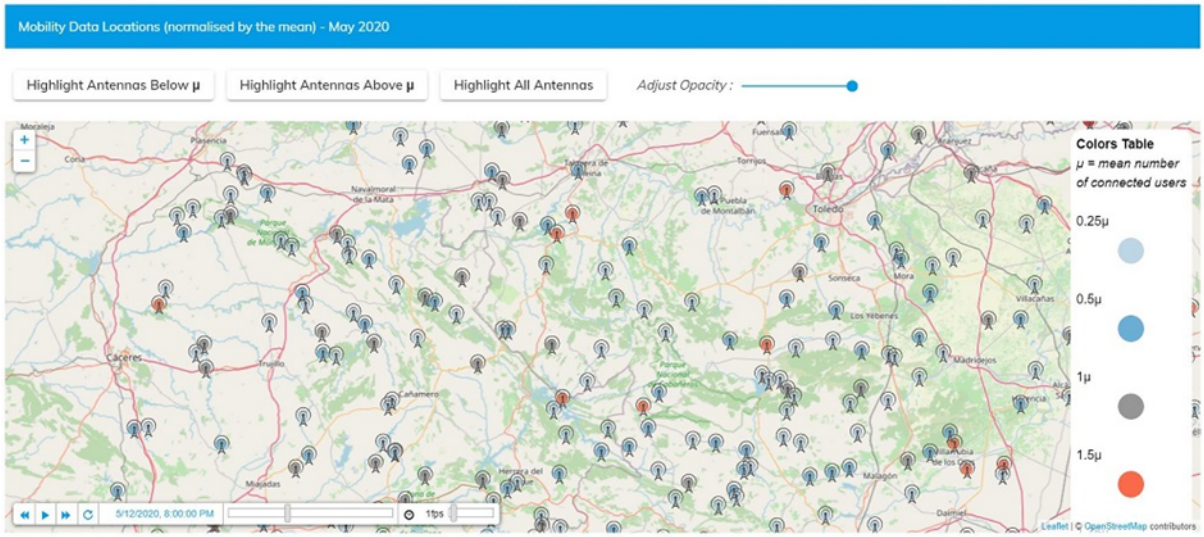

Figure 17. Location prediction Visualisation

3.2 Optimization of Placement of Telecommunication Equipment

This use case, which was included for the first time in the 2nd Integrated version of the I-BiDaaS Platform, aims to optimise the network operations by providing caches and identifying optimal antenna locations, given the provided data from customer usage. The important challenges were to (i) analyse streaming data for improving the routing and placement of the telecommunication equipment that is available or arranging for new equipment to be obtained; (ii) to study the spatio-temporal patterns and (iii) provide insights on the dynamics of cellular sectors.

The dataset used in this use case contains anonymized and encrypted data for a period of 24 hours. It holds data for over 40K antennas in a city, where a data sample contains 16 antenna KPIs. The name of the city and of the KPIs are also anonymized. An aspect of interest for this use case is the optimization of network operations. This can be achieved by providing caches and finding optimal antenna locations based on customer usage data. The challenges that arise here include: analysing the streaming data for telecommunication equipment placement, routing improvement or new equipment obtaining; studying the spatio-temporal patterns to gain some conclusions on the dynamics of cellular sectors.

The provided KPIs serve as a basis for applying a modelling approach that could predict whether the cell sites will become a hot spot. This can be achieved by a supervised binary classification approach. First, the antennas were grouped by their IDs, and then 80% of the groups were used as a train set, where 20% were used for testing. In such way, an antenna belongs either to the train or to the test set. The goal was to find a splitting that yields as a similar percentage of the positive class as possible in the train and test sets. The classification algorithms used here include Random Forest, XGBoost[3] and CatBoost [4]. The Random

Forest algorithm was implemented both in Scikit Learn[22] and also in PyCOMPSs [2]. On the other hand, CatBoost and XGBoost represent gradient boosting algorithms. All the mentioned models showed satisfactory results: the best accuracy equals to 0.999 and precision and recall equal to 0.998 and were obtained using XGBoost. By analysing the hotspot prediction task, it was previously observed that the forecasting can be also performed for sectors with non-regular behaviour. Besides, it turned out that the time of the forecast does not influence the results significantly and that a forecast accuracy plateau is reached in the cases when at least one week of past information is considered. As the high accuracy and throughput of the mentioned models show, the achieved classification is of high quality. Based on this, one can conclude that the I-BiDaaS solution can be extremely useful in understanding network performance.

This is a streaming use case that was not available in the 1st Integrated version of the I-BiDaaS platform. A stream of JSON data is published periodically to UM topic /ibidaas/tid/antenna_in. Each message in this stream contains an array of antenna status updates. An example of this message is shown in Figure 18.


```
{
    "date": "2020-12-09 08:56:05.987329",
    "antennas": {
        "9d475e38cfa623cadc9feb80463765b0": {
            "features": [
                1.0,
                1.001125,
                0.0,
                0.0,
                0.0,
                0.0,
                0.001898,
                0.996008,
                0.9761549999999999,
                1.5504229999999999,
                781502.0,
                39.116659999999996,
                0.0,
                21.0,
                1.313991
            ],
            "hotspot": 0
        },
        "efabb4cba6768a15dfa5a64ade4dafdf": {
            "features": [
                1.0,
                1.0,
                0.0,
                0.0,
                0.0,
                0.0,
                0.0,
                1.0,
                1.0,
                1.260226,
                689650.0,
                38.381929,
                0.0,
                10.0,
                0.341884
            ],
            "hotspot": 0
    },
    …
```


[22] Scikit-learn: machine learning in python. https://scikit-learn.org/stable/

```
    ]
}
```

Figure 18. Json messages containing antenna information

The Streaming Analytics module consumes these messages, performs classification using a the CatBoost model created offline and publishes the original data with a binary classification (hotspot or not) to a /ibidaas/tid/antenna_out message in UM. Finally, AVT consumes these messages and visualises the hotspot prediction.

Figure 19 shows the visualisation of the hotspot prediction on the I-BiDaaS platform, implemented in the 'Co-Develop mode', for the Optimization of Placement of Telecommunication Equipment.

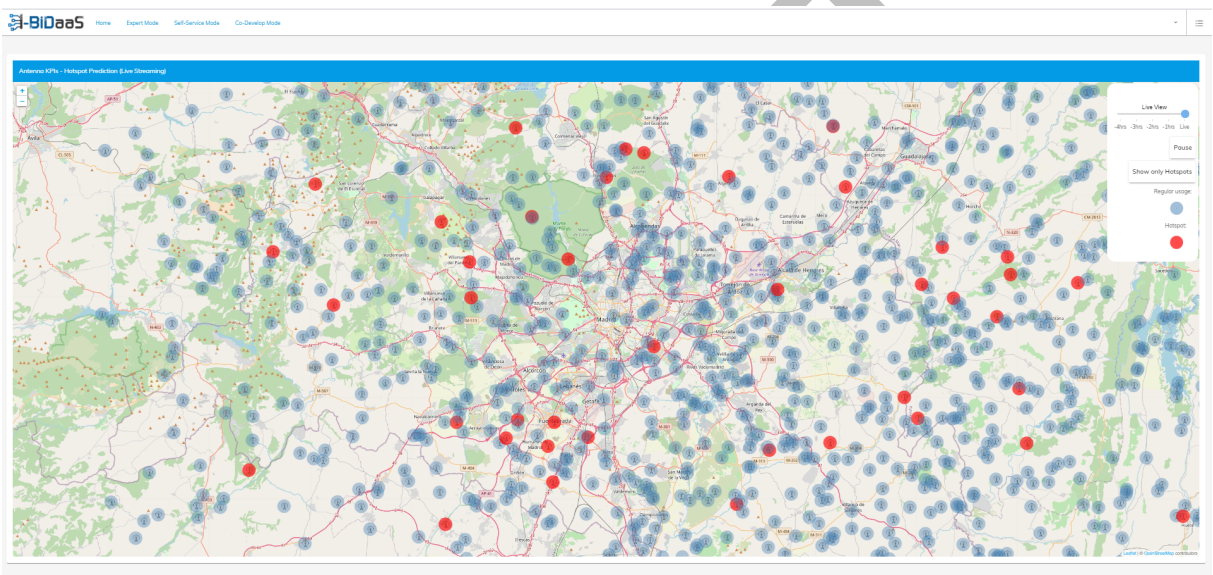

Figure 19. Hotspot Prediction Visualisation

3.3 Quality of Service in Call Centres

This use case addresses the challenge of developing speech technologies that transform audio calls into relevant information for the Call Centre, which can be used to assess its performance and/or to automatically screen phone calls. By facilitating the results of the project, TID plans to improve the number of audio calls that can be processed per time unit.

There is a wide variety of customer calls with respect to its nature: to ask for service and product information, report technical problems, to follow-up with a purchase, to provide feedback, etc. The I-BiDaaS solution allows to quickly get familiar and understand customer's perspective and main interests, and to facilitate a fast response and improve customer service by using big data speech and language analytics. This is achieved by shortening the call durations, waiting time and First Call Resolution (FCR) time by anticipating customer's situation based on previous insights, e.g., using the aggregation of previous analytics by Call Centres or regions.

A demo-version of this use case was introduced in the 1st Integrated version of the I-BiDaaS platform and was presented in Deliverable "D5.4 Big-Data-as-a-Self-Service Test and Integration Report v2". The demonstrator simulated the aggregation of data streams originating from three different Call Centres in Spain. The I-BiDaaS-based final solution for computing

streaming analytics in call centres and the corresponding algorithms are described in Deliverable "D4.3 Streaming analytics and prediction". In essence, the solution is able to process the recorded text transcripts, which can originate from different call centres in real-time. Given the vast amount of data that needs to be processed, I-BiDaaS is able to demonstrate faster end-to-end text analytics with GPU acceleration and provide different types of rolling analytics (such as sentiment score, word frequencies, and most frequent words) for different time intervals (e.g., last minute, last hour, and last day). To overcome the large memory consumptions of rolling windows, we use the notion of time-decayed counters where the value of a counter automatically decays over time using a mathematical formula. These analytics can be used for real-time predictions, such as QoS, and help a company to take appropriate actions in order to better understand or control a given situation.

Figure 20 and Figure 21 show the results of the analysis provided to the end-user via AVT on the I-BiDaaS platform, utilised in the 'Co-Develop mode', that provides an easy tool for the end-user inspection and with valuable insights about real-time operations of the CC.

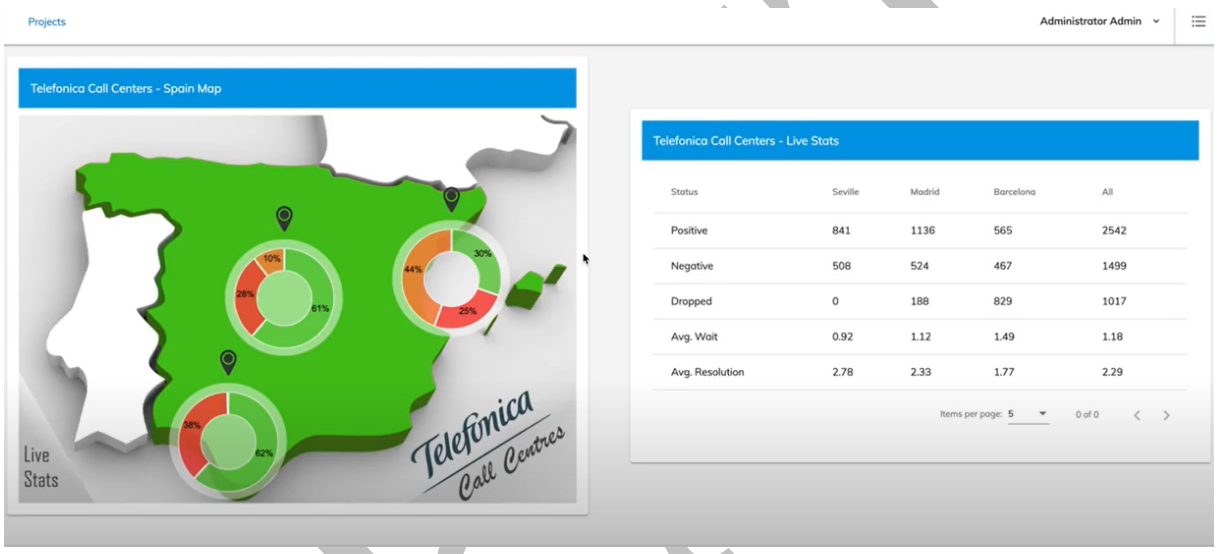

Figure 20. Advanced Visualisation Toolkit (AVT) supporting scalable data visualisation.

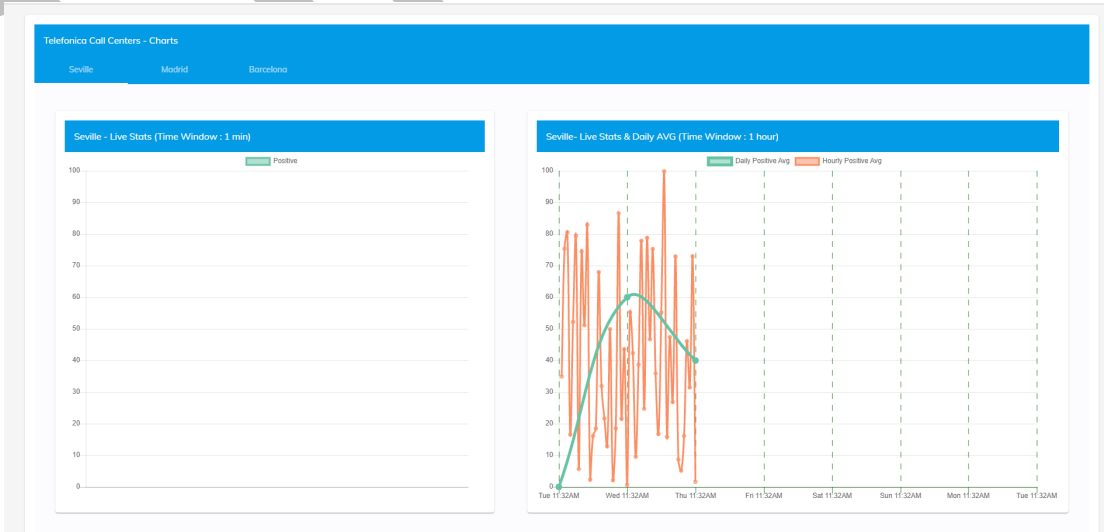

Figure 21. Example of Live Stats & Daily AVG (Time Window: 1 hour)

3.4 Enhanced control of customers to online banking

In this use case, which was included for the first time in the 2nd Integrated version of the I-BiDaaS Platform, we focused on analysing the mobile-to-mobile bank transfers ordered through online banking (web and application). It focuses on assessing that the controls applied to authenticate the user are applied adequately (e.g., Strong Customer Authentication -SCA- by means of second-factor authentication) according to PSD2 regulation and depending on the context of the bank transfer.

Initially, a K-means algorithm was considered for the analysis of the data. However, even applying transformations in categorical variables, the result was not satisfactory. Thus, a new way to apply clustering was found by using the K-modes algorithm[23]. This algorithm is similar to K-means but has the precise modifications that enable it to work with categorical variables.

The optimal number of clusters to be used with K-modes was determined with the elbow method [9], a heuristic technique commonly used for this purpose. Intuitively, this method finds the point in the curve where adding more clusters does not significantly improve the modelling of the data.

Once clusters are generated, a final visual analysis from the expert is needed to determine if anomalies can be found in the data. More details on the algorithm used in this case are available in Deliverable "D3.3 Batch Processing Analytics module implementation final report". A visualisation of the clustering results can be found in Figure 22.

Figure 22. Sample of the 'Enhanced Control of customers to Online Banking' use case clustering results in the I-BiDaaS platform

This use case has also a streaming version. By using the centroids created in the batch version of the use case, the user can create rules for classifying streaming data; each centroid used defines a category/rule. Using the coordinates of the incoming data, and calculating the distance from each cluster centroid, each incoming record is classified to the category that corresponds to the centroid with the minimum distance.

More specifically, the steps for classifying a new record are the following:

[23] https://pypi.org/project/kmodes/

1) New rules are sent to the *ibidaas/caixa/bizum_centroids* channel in the UM, they are shown in Figure 23:

```
{
   "id": "rule-1",
   "update_time":"2020-12-02",
   "domain":[ { "x":0.0, "y":-3.0, "z":0.0 }, { "x":1.0, "y":3.0, "z":2.0} ],
   "centroids":[
      {
         "label":"normal-1",
         "values":[ "1", "1", "-1", "OK", "-2", "OK", "ACCP", "-1", "EMIT",
"LA", "ko", "1" ]
      },
      {
         "label":"strange-1",
         "values":[ "0", "1", "-1", "MOT", "MOT", "OK", "ACCP", "-1", "EMIT",
"LA", "ko", "1" ]
      }
   ]
}
```

Figure 23. Centroid related rules

2) These rules are stored for later reference; if a rule with the same id is detected, it overwrites the previous rule with the same name and the counter is reset to 0.
3) Apama listens on the *ibidaas/caixa/bizum_in* channel for new requests to be evaluated, they have the form shown in Figure 24.

```
{
  "pk_anyomes": 202001,
  "pk_anyomesdia": 20200114,
  "pk_ide_usuclo_orig": 8190758100,
  "pk_idenumsesion": "jimr311",
  "pk_orden": 0,
  "pk_referencia": "m74ap2xdzj",
  "importe_operacion": 7.45,
  "ind_autenticacion_reforzada": 0,
  "tipo_autenticacion": "ok",
  "autenticacion_esperada": "-2",
  "estado_op_dec": "ok",
  "estado_op_dec_ltx": "accp",
  "motivo_estado": "-1",
  "estado_op_trf": "emit",
  "num_particion": 1,
  "fk_numperso_tit_loe": 27513972,
  "cod_tipo_terminal": "la",
  "terminal": 9725091907581,
  "empresa": 1,
  "motivo_anulacion": -1,
  "hora_confirmacion": "14/01/20 21:56:34,095000000",
  "fecha_carga": "44077",
  "x": null,
  "y": null,
  "z": null
}
```

Figure 24. Records sent to UM for evaluation

4) For every record it receives, Apama checks whether it lies within the domain of one of the rules (by calculating the distance to each rule/centroid). The counter of the matched centroid is increased by 1.

5) Every X minutes, all rules are sent back one by one to the *ibidaas/caixa/bizum_out* channel. The values of the counters of the centroids are displayed in the UI.

Figure 25 shows the results of the analysis provided to the end-user via AVT on the I-BiDaaS platform, utilised in 'Co-Develop mode'.

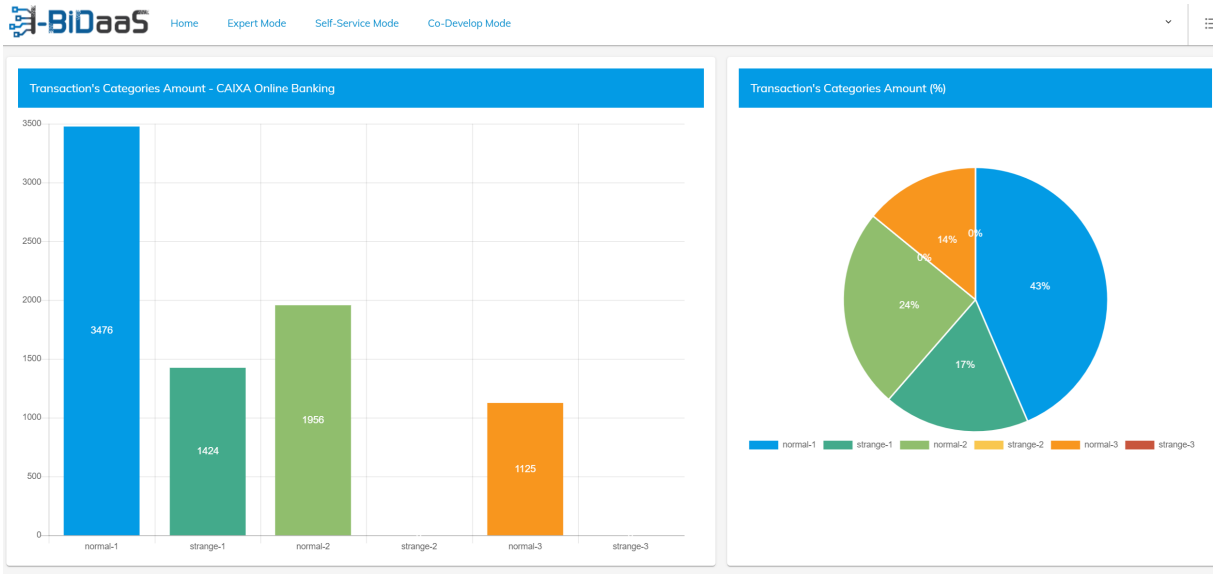

Figure 25. Sample of the 'Enhanced Control of customers to Online Banking' use case streaming results in the I-BiDaaS platform

3.5 Advanced analysis of bank transfer payment in financial terminal

This use case aims to detect the differences between reliable transfers and possible fraudulent cases. The goal of this experiment is to test the efficiency of the I-BiDaaS solution in the context of anomaly detection in bank transfers from employees' workstations (*financial terminal*).

The main idea of analysing bank transfers executed on financial terminals by the bank employees is to detect both possible anomalies different from regular working procedures or possible fraud. The details of the use case have been widely explained in Deliverable D3.3, Section 3.1, where we can see that data is encrypted using Dice Coefficient (although other encryption types are available), and the dataset is made of attributes that describe the steps the bank employee made, together with client information, account, amount of money, etc.

The analysis conducted has two steps. First of all, the input data is processed with a PCA algorithm to transform the data in order to apply unsupervised clustering. After that, DBSCAN is used to determine the optimal number of clusters, and K-means is feed with that number. Finally, from 2D and 3D plots generated from K-means result, a visual analysis can be done to identify values far away from the cluster centre (therefore, possible anomalies). All the algorithms are provided by the dislib library, and execution results can be seen with much more detail in Deliverable D3.3 'Batch Processing Analytics module implementation final report'.

The use case has been deployed in the Expert Mode of the I-BiDaaS platform, as the "Caixa Advanced Bank Transfers" and was available from the 1st version of the Integrated I-BiDaaS Platform. A custom project has been defined, which uses a tokenized dataset, using the Batch Processing module as the Data Processing Mode, and a Cassandra Keyspace as the way inputs are provided to the algorithm. The default Docker image provided by the platform is used, where the analysis code, together with all required tools (e.g. dislib and Hecuba) are included,

and were uploaded previously. The only interaction that users can do in this case is to create Experiments, as seen in the screenshot, in Figure 26.

To support the graphical interface of this use case, the Voila visualiser was implemented and integrated in the 2nd version of the Integrated I-BiDaaS Platform as part of the Expert mode. It allows the user to upload and run custom visualisations per project (use case) in the form of Jupyter notebooks. More information can be found in section 2.2.2.

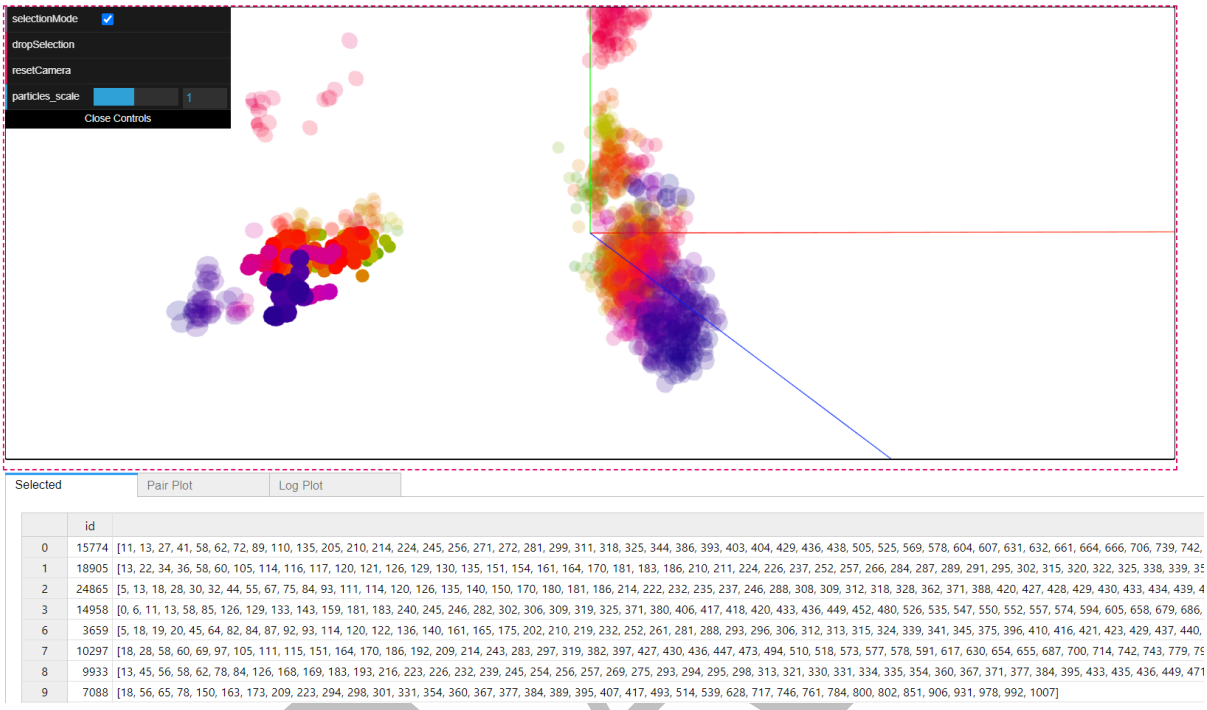

Figure 26. Sample of the 'Advanced analysis of bank transfer payment in financial terminal' use case in the I-BiDaas Expert Mode visualisation

3.6 Analysis of relationships through IP addresses

In this use case, CAIXA aims to validate the usage of synthetic data and the usage of external big data analytics platforms. It is deployed in the context of identifying relationships between customers that use the same IP address in their connections to online banking. CAIXA stores information about their customers and the operations they perform (bank transfer, check their accounts, etc.), using channels such as mobile apps or online banking, and they afterwards use this data for security and fraud prevention processes. One of the processes is to identify relationships between customers and use them to verify posterior bank transfers between linked customers. Such operations are considered with lower possibility to be fraudulent transactions. It allows CaixaBank's Security Operation Centre (SOC) to directly discard those bank transfers in their revision processes. The goal of this experiment is to validate the use of synthetic data for analysis, if the rules act in the same situations as with the real data and test the time efficiency of the I-BiDaaS solution.

This use case was finalized in the 1st version of the I-BiDaaS platform and has not changed since. More information about this use case can be found in Deliverable "D5.4 Big-Data-as-a-Self-Service Test and Integration Report v2", section 3.1.

3.7 Maintenance and monitoring of production assets

This use case, which was included for the first time in the 2nd Integrated version of the I-BiDaaS Platform, has been selected to use the data to optimise a real industrial process and to set a predictive maintenance procedure in order to prevent faults before they happen by doing maintenance at the right time (not too late or too early, to avoid inefficiencies). Different types of sensors are installed on the production line and acquire different data information (e.g. acceleration, velocity, pressure, temperature, etc.). All the sensors record their perception of the surroundings, and upload and transfer this information to a server that manages the obtained data. For example, accelerometers are used for measuring vibration and shock on machines and basically anything that moves. Therefore, the monitoring of vibrations is important to check the status of a machine, and the analysis of the trend of vibrations over time allows to predict the onset of deterioration and to intervene in time before the failure. The continuous and periodic control of the service conditions of a machine is known as Predictive Maintenance. The input data for this use case implies two different sets, the SCADA and the MES datasets. The SCADA dataset consists of daily vehicle production data, such as production, process and control parameter values. The MES dataset is related to data describing the details of the type of the vehicle that is being produced. The main goals here are: to optimize an industrial process; and to set a predictive maintenance procedure.

The data pre-processing subsumed the process of transforming the data into separate time series, where one time series corresponds to one sensor. An outlier detection analysis was performed on each sensor, and the time stamps of the detected anomalous measurements were compared across different sensors. The analysis was performed using a modified interquartile range[24] (IQR) test on a subset of 16 sensors, on a single GPU (NVIDIA RTX2070), without parallelization. It was evident that the sensors mostly have around 500 different days with those anomalous measurements. Also, in the majority of cases, i.e., more than 90%, those measurements were common to different sensors. This means that sensors have a large percentage of shared days when anomalies occur. It was also observed that there exist differences between the number of days with anomalous behaviour for different sensor, but this could be due to the fluctuation of sensors measurements, as they are used for different vehicle types. The efficiency and accuracy of the I-BiDaaS model with respect to internal CRF analyses were a suitable basis for visualising the results on the I-BiDaaS platform in 'Co-Develop Mode'. This also served as a foundation for creating a structured foundational database to be easily utilized, in order to check outliers. This results in improving the efficiency of manufacturing plants, by production loss reduction, and also in achieving greater competitiveness of the company by an increase of 0.05 % the current Overall Equipment Effectiveness (OEE) and a decrease of 50 % in maintenance costs.

[24] https://en.wikipedia.org/wiki/Interquartile_range

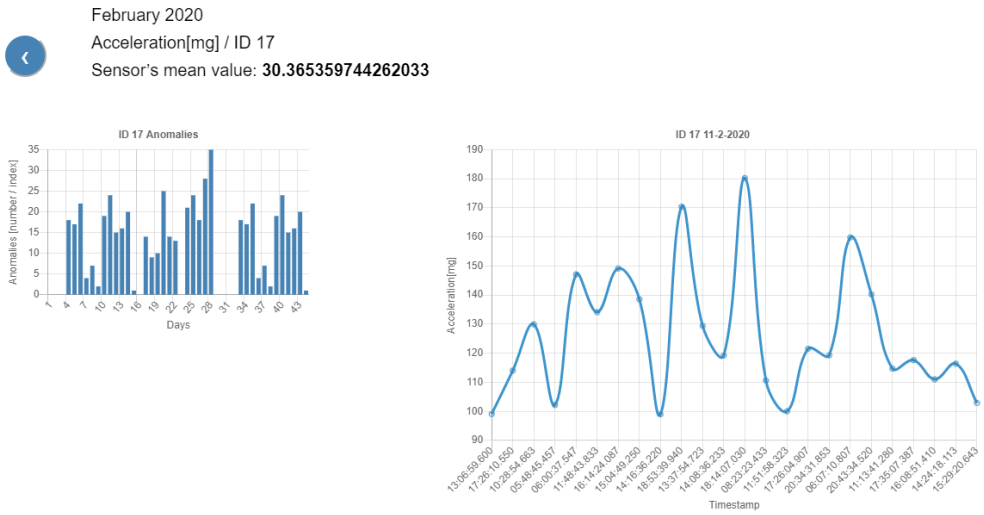

Figure 27. Sensor's mean value and anomalous trend for the selected sensor and day

3.8 Production process of aluminium Die-casting

This use case aims to improve the quality of the production process of the engine blocks. During the die-casting process, molten aluminium is injected into a die cavity, where it quickly solidifies. The process is complex, and it is important to not only carefully design parameters and temperatures but also to control them because they have a direct impact on the quality of the casting. Big Data analysis aims to improve the quality of the process, with the aim of finding the most significant parameters to monitor and control. The goal of this experiment is to test the efficiency of the I-BiDaaS solution in the context of correlating defects with the production process parameters and resetting these to prevent repairs and reprocessing of the engine blocks.

This use case was initially integrated in the 1st version of the Integrated I-BiDaaS Platform. The initial version included only sensor data, whereas in the second version, images of thermal cameras, as depicted in Figure 28, are provided in order to check if there is a correlation between temperature and the other parameters in relation to the quality levels.

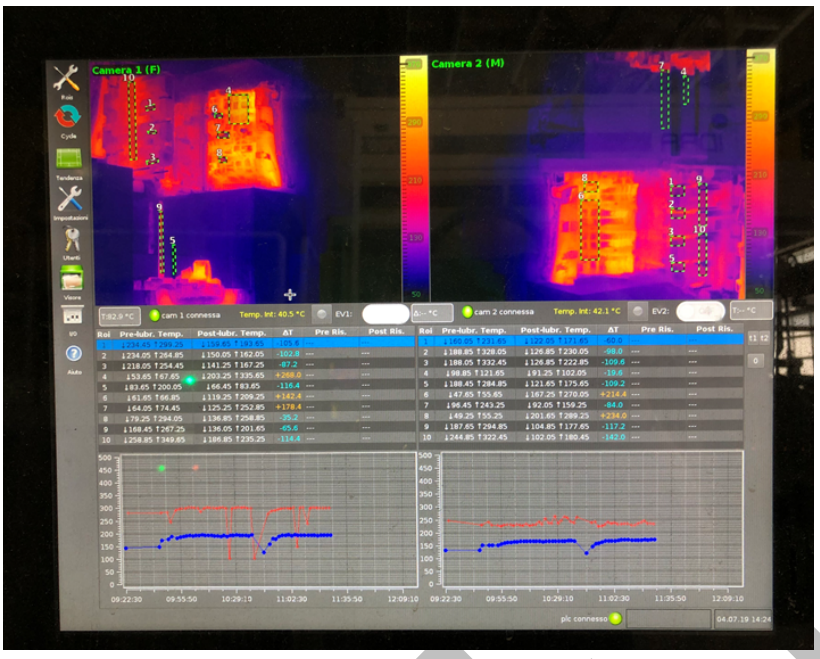

Figure 28. Frame by thermal camera

The input dataset contains a collection of casting process parameters, monitored by sensors. Additionally, a large dataset of annotated thermal images of an engine block casting process is available. The assumption is that there exists a correlation between sensor data, thermal data and the process outcome. One of the main goals here is to test the efficiency of the I-BiDaaS solution regarding the correlation between defects and production process parameters in order to enable resetting to prevent repairs and reprocessing.

Generally, a supervised learning M-ary classification problem matches the requirements of the use case. Different models have been developed, starting from sensor-only models, to models that utilize both sensor and thermal images data.

Random Forest was used for classification of the sensor data, where the accent is on classification of engine blocks according to their control classes, as well as on assessing the feature importance and identifying the most important parameters for the classification. As it turned out that the use case corresponds to an imbalanced problem, an extension of the basic Random Forest, namely weighted Random Forest, was used to overcome this issue. Observing this problem as a binary classification problem, a binary classification algorithm was applied. Besides that, the distributed alternating directions method of multipliers (ADMM) algorithm was also used to perform binary classification on the data set and was developed in PyCOMPSs.

A neural network model was developed based on thermal images only. The pre-processing of data subsumed cropping the images, in order to remove non-useful parts, as well as resizing of the images in order to better fit the models. The images were also converted to grayscale in order to induce faster convergence, and finally transformed to tensors of appropriate dimension. The ResNet18 network [5] was used, as the experimentation showed that simple shallow neural network models do not perform satisfactorily on the data. The ResNet18 network was modified to accept the image dimensions and to work with the appropriate number and format for the output classes. Both the pre-trained (trained on the ImageNet data set [8]) and non-trained models behaved similarly regarding accuracy. The dataset was split in order to use 80% for training and 20% for validation purposes. The training process was done for 300 epochs, using a learning rate of 10^{-3} and cross-entropy as the loss function. The implementation is written in

PyTorch [6]. The results for training showed an average loss of 0.0618, with an accuracy of 88%. For validation, the average loss was 0.5642, with an accuracy of 57%.

A joint model for thermal images and sensor data, which used an even deeper neural network for thermal images and a fully-connected network for sensor data, was also considered. The pre-processing of sensor data was performed in the same manner as for the random forest model. It subsumes removing NaN values, column-wise normalization and time-stamp conversion. Additionally, in order to fasten the convergence, standard image channel normalization was also introduced. The DenseNet201[25] neural network model was adapted and used here. In particular, two different models were developed: a model that uses the full imbalanced dataset; and a model that uses a random under-sampling technique to produce a balanced dataset. The DenseNet201 model, with a 7-layer fully connected neural network for sensor data, was applied. Finally, both mentioned models converged fully and quickly on the training set, and also demonstrated satisfactory accuracy values. The validation accuracy is 73.54% for the first model, and 54.06% for the second model.

In this case, batch analytics were used to develop the high-level algorithms in order to identify and select the critical parameters by providing a 'Co-Develop-Mode' to timely check the status of the process and classify the quality levels that we have identified as KPIs. Subsequently, CRF copies data from its internal server to the I-BiDaaS platform. The stream of data consists of pairs of sensor readings and an image of the thermal scan of the die-cast. When a new pair of data is copied from the CRF premises to the I-BiDaaS infrastructure, a message is published periodically to UM topic /ibidaas/crf/data_in. Each message contains the file path of the image and the sensor reading in JSON format.

The Streaming Analytics module consumes these messages, performs classification using the neural network model created offline and publishes the original data with a classification to a topic called /ibidaas/crf/data_out in UM. Finally, the AVT consumes these messages and visualises the classification level (Figure 29).

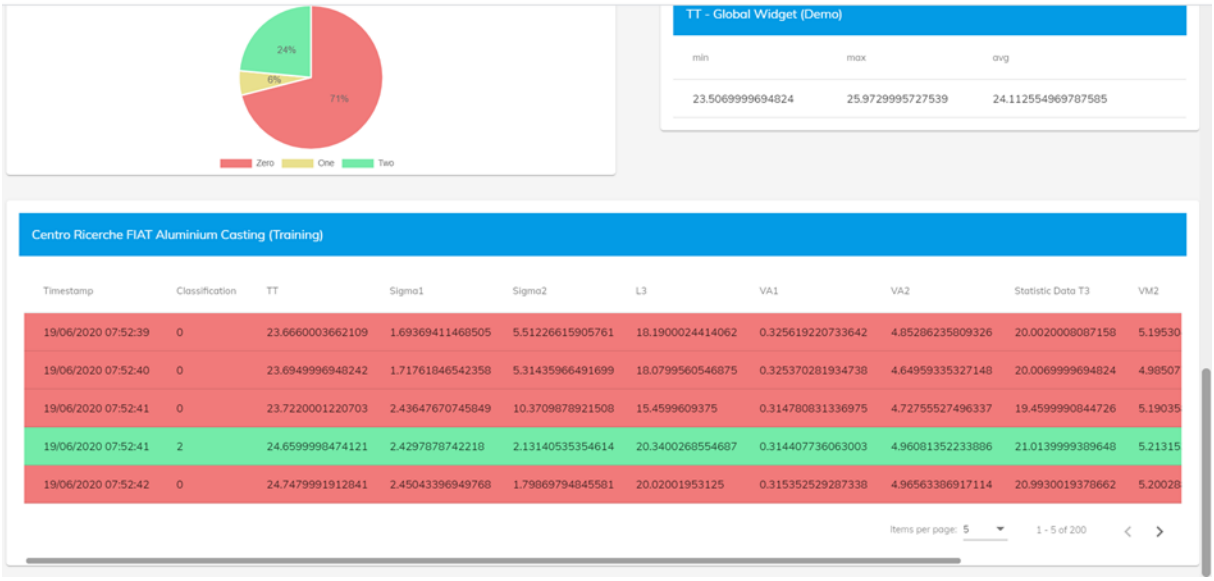

Figure 29. Classification level

[25] https://pytorch.org/hub/pytorch_vision_densenet/

4 Testing and Integration

In this section, an integration plan for WP2 – WP5 is presented. Integration and Testing is explicitly decoupled from the "development of the architecture components": that is, the development of each software component (Batch processing, Streaming analytics, Resource management, Visualisation, *etc*.) follows its own methodology. Some of them (especially the ones that are not open-source) can be considered as “black boxes” that will be integrated into the I-BiDaaS platform.

This section is a revised version of the Testing and Integration Plan provided in Deliverable “D5.4 Big-Data-as-a-Self-Service Test and Integration Report v2”, submitted in M24. Moreover, information about the tools and methodologies used in this integration can be found in Deliverable “D5.2: Big-Data-as-a-Self-Service Test and Integration Report (first version)”, submitted in M12.

4.1 I-BiDaaS actors and their integration activities

The I-BiDaaS actors, integration activities, together with integrated components, are listed in Table 3.

Table 3: I-BiDaaS actors and their integration activities

Actor	Integration Activities	Integrated Components
Foundation for Research and Technology – Hellas (FORTH)	Development of GPU-accelerated Analytics	Streaming Analytics
Barcelona Supercomputing Center (BSC)	Development of Hecuba Tools and Qbeast	Batch Processing Module
IBM Israel – Science and Technology LTD (IBM)	Development of Test Data Fabrication	Data Fabrication Tool
Centro Ricerche FIAT (FCA/CRF)	Pilot leader / Data provider	N/A
Software AG (SAG)	Development of Streaming analytics & Messaging	• APAMA Server • Universal Messaging
CaixaBank S.A. (CAIXA)	Pilot leader / Data provider	N/A
University of Manchester (UNIMAN)	Resource Management and Orchestration	N/A
ATOS Spain S.A. (ATOS)	Resource Management and Orchestration	Resource Management Orchestrator
Aegis IT Research LTD (AEGIS)	Development of Visualisation	I-BiDaaS Dashboard

Information Technology for Market Leadership (ITML)	• Development of Orchestrator • Platform Integration • QA	Orchestrator
University of Novi Sad Faculty of Sciences (UNSPMF)	Development of Batch Processing algorithms	Batch Processing Module
Telefonica Investigación y Desarrollo S.A. (TID)	Pilot leader / Use case and Data provider	N/A

4.2 Testing

In this Section, a list of different types of tests conducted in the I-BiDaaS platform is described: Unit Testing, Universal Messaging Testing, End-to-End Testing and Code Quality Testing. The actual test results for the 2nd version of the integrated I-BiDaaS platform are presented in Section 4.4.

4.2.1 Unit Testing

Each team is responsible for the unit testing of the component they are developing. Unit testing will verify the functions created to implement the business logic for each component, verifying the results. Although Unit Testing results per component are not in the scope of this document, specifically for the orchestrator module that is responsible for the coordination of the Platform, Unit Testing after the integration of the rest of the modules is more important, as it is equivalent for testing the whole platform (apart from the UI/Visualisation Toolkit). For that reason, in Section 4.3, we will specifically present Unit Testing designed and conducted for the Orchestrator module.

4.2.2 UM Testing

The UM handles communication between the major components of the I-BiDaaS platform for streaming the use cases. This simplifies the integration testing up to a point, since even if a component is not ready, the associated message queue receiving messages for that component will be available, thus allowing at least to test the messages' sending. The communication concerning the UM must be tested for three important aspects:

- Each queue/topic must correctly deliver the messages to the correct components.
- Each component must have a worker retrieving the messages from the message queue/topic that is assigned to that component. The use of the message after it is retrieved does not belong to the scope of this part of testing.
- Each component must correctly implement the logic that sends the messages to the correct queues when attempting to communicate with another component.

This will verify that all messages will be delivered, that each component will be able to receive messages and that each component will be able to send messages.

Once the communication with the UM has been established and tested, the integration testing proceeds with testing the communication between pairs of components (where available) through the UM in order to verify that each request receives the appropriate response. The purpose is not to check the validity of the responses but to ensure that they trigger the correct chain of messages between the various components since, in many cases, a request to one component will be processed through another component, which will deliver the final response.

4.2.3 End to End Testing

The purpose of End-to-End (E2E) testing is to verify the desired functionality across the whole platform, offering insights on the performance and, in this case, of a multi-component system, demonstrating the intended communication between these components.

Successful completion of this test confirms that the I-BiDaaS platform is working as intended both in software and hardware terms. Each test case will remain valid for as long as the tested functionality remains in the application and should be repeated after each update of the application in the future.

E2E tests are performed by ITML, using the test cases described in the next section. Any case that leads to non-intended results or disruption of the application flow should be documented as detailed as possible, including information about the process that led to it, the conditions, and the platform. An attempt to recreate the error should also be made.

4.3 Benchmarking tests

For benchmarking the I-BiDaaS platform, it was decided to measure the throughput and latency, using two Machine Learning algorithms available in the Self-Service mode: K-means and Random forest. These two are two of the most common examples of unsupervised and supervised learning. Both algorithms are implemented in PyComps, using the Dislib library.

For the experiments, the I-BiDaaS platform was installed in a public cloud provider (DigitalOcean[26]), in a cluster of 6 nodes with 8 CPUs and 16GB of RAM each. The cluster was initiated using the I-BiDaaS open-source installer, described in Section 5.2. The benchmarking results are presented in "D6.5: Assessment Report and Impact Analysis" , Section 3.2.

4.4 Integration Test Results

The list of integration and quality assurance tests that were performed for the 2st version of the I-BiDaaS Platform is summarized in Table 4. All the tests in Table 4 have already been conducted and have passed.

The list of tests is divided into the following types:

- Orchestrator Unit Testing (OUT): Tests for verifying Orchestrator functionality. Since the orchestrator is a Rest API (described in Section 2.2.5), Unit tests are tests designed for each endpoint of the API. These tests are automated; check is done through validating the API response (in JSON format).
- Use Case Testing (UCT): E2E test for verifying a Use Case specific functionality. These tests correspond to the functionalities of the integrated use cases that are presented in Section 3. The tester performs a specific task in the UI and verifies its validity by checking (i) the Visualisation Toolkit output, (ii) the database of the orchestrator and (iii) the log files and output produced by relevant modules (*e.g.*, docker-compss).
- Generic Use Case Testing (GUCT): E2E tests for verifying the functionalities of the Generic Use Case, as presented in Section 2.1.1. The tester validates the task in the same way as with the Use Case Testing.

[26] https://www.digitalocean.com/

- Universal Messaging Testing (UMT): Tests for verifying intercommunication in the streaming use cases. Testing is done by validating the contents of MQTT messages exchanged.
- Generic Advanced Visualisation Toolkit Tests (AVT): Generic tests of the toolkit, such as login, logout functionalities.

Table 4: Integration and quality assurance tests

Test Name	Test Type	Description	Preconditions	Output
Update-experiment	OUT	Set status of experiment to terminated. Used internally when COMPSs terminates	• Specified experiment exists • Specified experiment has status: running	• Specified experiment status is set to "terminated"
Get-projects	OUT	Get the list of projects (predefined projects)	• User is logged in	• A valid array of the predefined projects
Get-project-experiments	OUT	Get the list of project experiments owned by the current user	• User is logged in • Specified project exists	• A valid array of experiments belonging to the specified project • Returned experiments belong to current user
Get-project-details	OUT	Get the details of a project	• User is logged in • Specified project exists	• A valid JSON representation of the specified project
New-experiment	OUT	Start a new experiment for a specified project	• User is logged in • Specified project exists	• A new experiment is created in the DB • A new docker stack is created in docker swarm • The experiment is running in the infrastructure
Get-experiment	OUT	Get the details of experiment, specified by id	• User is logged in • Specified experiment exists • Specified experiment belongs to user	• A valid JSON representation of the specified experiment
Stop-experiment	OUT	Request the termination of an experiment	• User is logged in • Specified experiment exists	• The status of the new experiment is set to terminated

Test Name	Test Type	Description	Preconditions	Output
			• Specified experiment belongs to user • Specified experiment has status "Running"	• Docker stack belonging to this experiment is terminated
Download-experiment	OUT	Return a package (tarball) of the results of an experiment, specified by id	• User is logged in • Specified experiment exists • Specified experiment belongs to user • Specified experiment has status "Terminated"	• A tarball containing the files created by docker/COMPSs
Delete-Experiment	OUT	Delete a specified experiment	• User is logged in • Specified experiment exists • Specified experiment belongs to user	• The experiment is deleted from the database • Docker/COMPSs Output files are deleted
Get-experiment-output	OUT	Return the results of an experiment, in JSON format	• User is logged in • Specified experiment exists • Specified experiment belongs to user • Specified experiment has status "Terminated	• Returns a valid JSON with the output of the experiment (use case specific)
Get-input-dirs	OUT	Return a JSON representation of the available directories to be used as input, for the current user	• User is logged in • User has access to the returned directories	• Returns a valid JSON with an array of directories
Get-input-files	OUT	Return a JSON representation of the available files to be used as input, for the current user	• User is logged in	• Returns a valid JSON with an array of all files that the user has access to

Test Name	Test Type	Description	Preconditions	Output
Get-keyspaces	OUT	Return a JSON representation of the available Cassandra keyspaces to be used as input, for the current user	• User is logged in	• Returns a valid JSON with an array of all keyspaces that the user has access to
Create-custom-project	OUT	Create a new project, owned by the current user	• User is logged in	• A new project is created in the database • Code is uploaded in user space
Get-custom-projects	OUT	Get the list of custom projects of the current user	• User is logged in	• A valid array of the custom projects belonging to the user
Get-custom-project	OUT	Get the details of a custom project	• User is logged in • Specified project exists • Custom Project belongs to current user	• A valid JSON representation of the specified project
Update-custom-project	OUT	Update the parameters of a custom project	• User is logged in • Specified project exists • Custom Project belongs to current user	• The specified project is updated in the DB
Delete-custom-project	OUT	Delete a specified custom project	• User is logged in • Specified project exists • Custom project belongs to current user	• All experiments belonging to the specified project are deleted from the DB • The specified project is deleted from the DB
New-custom-experiment	OUT	Start a new experiment for a specific custom project	• User is logged in • Specified project exists • Custom Project belongs to current user	• A new experiment is created in the DB • A new docker stack is created in docker swarm

Test Name	Test Type	Description	Preconditions	Output
				• Custom docker image is created (if specified by user during project creation) • Docker-compss logs created (showing that COMPSs is running in worker nodes)
Delete-custom-experiment	OUT	Delete the specified experiment, belonging to a custom project	• User is logged in • Specified project exists • Experiment belongs to current user	• The experiment is deleted from the database • Docker/COMPSs Output files are deleted
Get-custom-experiments	OUT	Return a list of experiments, belonging to a custom project	• User is logged in • Specified project exists • Custom Project belongs to current user	• A valid array of experiments belonging to the specified custom project • Returned experiments that belong to current user
Stop-custom-experiment	OUT	Request the termination of a specified experiment, belonging to a custom project	• User is logged in • Specified experiment exists • Specified experiment belongs to user • Specified experiment has status “Running”	• The status of the new experiment is set to terminated • Docker stack belonging to this experiment is terminated
Download-custom-experiment	OUT	Return a package (tarball) of the results of the specified custom project experiment	• User is logged in • Specified experiment exists • Specified experiment belongs to user • Specified experiment has status “Terminated”	• A tarball containing the files created by docker/COMPSs

Test Name	Test Type	Description	Preconditions	Output
User login	AVT	User can successfully login to the Advanced Visualisation Toolkit	• User is already registered	• User is redirected to the Advanced Visualisation Toolkit Dashboard page
User logout	AVT	User can successfully logout from the Advanced Visualisation Toolkit	• User is already logged in	• User is redirected to the Advanced Visualisation Toolkit login page
Create-custom-project	GUCT	Create a new project, owned by the current user	• User is logged in	• UI validates user input • UI returns a new project • A new custom project is created in the DB, with the same properties as the one defined in the UI
Get-custom-projects	GUCT	Get the list of custom projects of the current user	• User is logged in	• UI returns all custom projects belonging to the user
Get-custom-project	GUCT	Get the details of a custom project	• User is logged in • Specified project exists • Custom Project belongs to current user	• A valid UI representation of the specified project, as stored in the DB
Update-custom-project	GUCT	Update the parameters of a custom project	• User is logged in • Specified project exists • Custom Project belongs to current user	• The specified project is updated in the DB • A custom project is updated in the DB, with the same properties as the one defined in the UI

Test Name	Test Type	Description	Preconditions	Output
Delete-custom-project	GUCT	Delete a specified custom project	• User is logged in • Specified project exists • Custom project belongs to current user	• The specified project is deleted from the DB
New-custom-experiment	GUCT	Start a new experiment for a specific custom project	• User is logged in • Specified project exists • Custom Project belongs to current user	• A new experiment is created in the DB • The UI returns the current experiment with running status
Delete-custom-experiment	GUCT	Delete the specified experiment, belonging to a custom project	• User is logged in • Specified project exists • Experiment belongs to current user	• The experiment is deleted from the database • The UI returns to the Dashboard page
Get-custom-experiments	GUCT	Return a list of experiments, belonging to a custom project	• User is logged in • Specified project exists • Custom Project belongs to current user	• A valid Visualisation of experiments belonging to the specified custom project • Returned experiments belong to current user
Stop-custom-experiment	GUCT	Request the termination of a specified experiment, belonging to a custom project	• User is logged in • Specified experiment exists • Specified experiment belongs to user • Specified experiment has status “Running”	• The status of the new experiment is set to terminated
Download-custom-experiment	GUCT	Return a package (tarball) of the results of the specified custom project experiment	• User is logged in • Specified experiment exists • Specified experiment belongs to user	• User successfully downloads a tarball containing the files created by docker/COMPSs

Test Name	Test Type	Description	Preconditions	Output
			• Specified experiment has status "Terminated"	
New-K Means experiment	UCT	Start a new experiment for Kmeans (Self-service mode)	• User is logged in	• A new experiment is created in the DB • The UI returns the current experiment with running status
View- K Means - experiment	UCT	View the results of experiment for K means (Self-service mode)	• User is logged in • Specified experiment exists • Specified experiment belongs to current user	• The UI parses the JSON output of the results and visualises it correctly
New-Random-forest experiment	UCT	Start a new experiment for random forest (Self-service mode)	• User is logged in	• A new experiment is created in the DB • The UI returns the current experiment with running status
Download-Random-forest - experiment	UCT	Download the results of experiment for K means (Self-service mode)	• User is logged in • Specified experiment exists • Specified experiment belongs to current user	• The UI returns a valid link with experiment results
New-Decision-Tree experiment	UCT	Start a new experiment for random forest (Self-service mode)	• User is logged in	• A new experiment is created in the DB • The UI returns the current experiment with running status
Download-Decision-tree -experiment	UCT	Download the results of experiment for K means (Self-service mode)	• User is logged in • Specified experiment exists • Specified experiment belongs to current user	• The UI returns a valid link with experiment results

Test Name	Test Type	Description	Preconditions	Output
New-IP Address Relations Streaming-experiment	UCT	Start a new experiment for IP Address Relations Streaming (Co-develop mode)	• User is logged in	• A new experiment is created in the DB • The UI returns the current experiment with running status
View- TID-Quality of service in call Centres-experiment	UCT	View the results of experiment Quality of service in call Centres (Co-develop mode)	• User is logged in • Specified experiment exists • Specified experiment belongs to current user	• The UI parses the JSON output of the results and visualises it correctly
View- TID-Optimisation of placement of telecommunication equipment-experiment	UCT	View the results of experiment Optimisation of placement of telecommunication equipment (Co-develop mode)	• User is logged in • Specified experiment exists • Specified experiment belongs to current user	• The UI parses the JSON output of the results and visualises it correctly
View- TID-Accurate location prediction with tragic and visibility -experiment	UCT	View the results of experiment Accurate location prediction with tragic and visibility (Co-develop mode)	• User is logged in • Specified experiment exists • Specified experiment belongs to current user	• The UI parses the JSON output of the results and visualises it correctly
View- CRF Casting - experiment	UCT	View the results of experiment for CRF Casting (Co-develop mode)	• User is logged in • Specified experiment exists	The UI parses the JSON output of the results and visualises it correctly

Test Name	Test Type	Description	Preconditions	Output
			• Specified experiment belongs to current user	
View-CRF-Maintenance and monitoring of production assets	UCT	View the results of experiment for CRF Casting (Co-develop mode)	• User is logged in • Specified experiment exists • Specified experiment belongs to current user	The UI parses the JSON output of the results and visualises it correctly
Download - experiment	UCT	Return a package (tarball) of the results of a self-service-mode or a co-develop mode project (only for batch processing projects)	• User is logged in • Specified experiment exists • Specified experiment belongs to user • Specified experiment has status "Terminated"	• User successfully downloads a tarball containing the files created by docker/COMPSs
IP Address Relations Streaming-Messages	UMT	Check the messages exchanged in UM for IP Address Relations Streaming	• Experiment has started	• Messages describing new transactions are published in topic #caixa_in • Messages for transactions between members of the same group are published in UM topic #caixa_out
TID-antennas-KPI-Messages	UMT	Check the messages exchanged in UM for Antennas KPI use case	• Experiment has started	• Messages describing are published in topic /ibidaas/tid/antennas_kpi
CRF Casting - Messages	UMT	Check the messages exchanged in UM for CRF Casting	• Experiment has started	• Messages describing new engine data are published in topic #tecsid_in

Test Name	Test Type	Description	Preconditions	Output
				• Messages for engines with classification results are published in UM topic #tecsid_out

5 From the prototype to the final I-BiDaaS solution

5.1 Future Sustainability

The I-BiDaaS solution has been designed, implemented, integrated, tested and validated in the scope of a 3 year RIA EU project, and during the project lifetime one of the main goals was research on identifying technical debts to create new opportunities for self-service analytics with the final aim of achieving a unified platform to boost the access to Big Data analytics paradigm to non-experts, in order to easily take advantages of Big-Data.

The final solution, released in three iterative cycles, offers three different modes (Expert, Self-Service, Co-Develop), which has been demonstrated in three relevant sectors such as Financial, Telecommunications and Manufacturing, each of them providing success stories which could engage new opportunities for other companies or data scientist willing to embrace Big-Data as a self-service.

The platform has been designed and developed keeping in mind that there are many factors that influence the reliability, robustness and maturity of a software platform and its associated technologies, especially in a complex distributed platform where several software modules work in a coordinated manner to process and analyse large volumes of complex data sets. The way I-BiDaaS technically approaches the sustainability of the platform was looking to the future from different perspectives trying to mitigate the barriers that could affect the proper evolution of future developments. Here is a breakdown summary of the different approaches to achieving software sustainability.

Techno-centric

The I-BiDaaS architecture has been designed as a Software Oriented Infrastructure (SOI) platform composed of several software modules that could be deployed isolated and distributed, facilitating; the dynamic allocation of infrastructure's resources needed to operate, the reuse of these modules out of the scope of the functionalities of the platform or even the replacement of one or more of these software modules.

In this sense, the platform is designed to operate at large-scale thanks to the ability of the software modules in charge of the data-processing and resource management allocation to be deployed as a cluster, capable of accommodating more resources in an elastic manner when they are required.

It is well known that software is reliant on hardware, with the aforementioned statement in mind, the platform software modules have been virtualized and containerized to facilitate their deployment across different computing environments while providing a ready to use environment for future developments or bug fixes.

We are aware that vendor lock-in is one of the major barriers to the adoption and sustainability of the cloud computing solutions. In order to tackle the vendor lock-in problem, there are technology-oriented approaches to mitigate this risk, and our designed solution provides an abstraction layer on top of different Cloud Service Providers (CSPs) capable of orchestrating the computing resources needed by the platform to operate across the computing continuum (multi cloud-edge).

In addition to the architectural considerations, in order to maximize the platform capabilities on its development sustainability, most of the software modules follow an Open Source Software (OSS) development approach allowing the software community to be involved in future

enhancements of the platform or even adapt the current solution for different contexts. The set of software modules which do not follow OSS strategy provides a community edition covering the main functionalities required. Moreover, an open-source version of the I-BiDaaS platform has been released and is described in Section 5.2.

Data-centric

Since the beginning of the project, we have been focused on two main pillars to develop our solution, on one hand how to operate in an easy way big data pipelines as well as on the other hand how to break the data silos existing within the companies that operate or analyse big volume of data.

In order to break data silos, we were working closely with our pilot partners for each of the sectors involved to understand the kind of obfuscation or anonymization required to the sensitive data in order to be shared or analysed outside the data owner premises. As a result of the work carried out, I-BiDaaS was able to share several datasets relevant to financial, manufacturing and telecommunications sectors in Zenodo[27], which is a general-purpose open-access repository that allows researchers to deposit papers, datasets, software or any other research related digital artifacts. They are all referenced on the I-BiDaaS web site[28].

As an outcome of the project, together with the platform, we are able to provide to the community sample datasets and applications ready to be used, in addition to configuration templates and containerized artifacts that are accompanying the different getting-started guides documented for the platform modes.

Functionality-centric

The I-BiDaaS platform provides three modes to accommodate different users' profiles according to their knowledge or/and technical expertise. Firstly, the Expert mode enables IT experts to develop their own analytics algorithms giving them entire freedom to choose which is the software that needs to be executed for performing the data analysis. Secondly, the Self-service mode offers a more guided experience for data practitioners who do not necessarily have a deep know-how on big-data related technologies. Last but not least, the platform offers the Co-Developed mode, which is more business-oriented, where all involved actors work together to build a solution capable of being used by final users transparently without the need to be aware of the data process and data analysis, which is done behind the scene by the platform.

Another particularity of the I-BiDaaS solution is that we divided the batch and stream process pipelines, the software components involved in each workflow can be instantiated without the need of deploying the other modules of the platform.

Process-centric

I-BiDaaS, as a research project, has been continuously looking at the state-of-art in order to advance based on the current needs of the market. Towards this goal, we strongly involved, during the project life-time, a community of experts on the field to monitor our progress. The I-BiDaaS External Advisory Board, together with the EU reviewer experts, helped us to track the drawbacks that the platform had for every iteration and rapidly incorporate their suggestions in our development process. In parallel, we established Info Days, Workshops, Hackathons, either physical or virtual with possible target customers or end-users in order to get early

[27] https://zenodo.org/

[28] https://www.ibidaas.eu/tools/

feedback and recommendations from their side. Detailed information about our collaborations and early feedback obtained from the different events that has been organised is documented within the WP6 deliverables (in "D6.4 Experiments implementation", Sections 4 and 5 and in "D6.5 Assessment Report and Impact Analysis", Sections 3, 4 and 5).

Another important activity to highlight is the collaboration with other projects under the BDVA[29] umbrella to build a common framework of understanding about the nowadays challenges on Big Data. As a result of this process, we ensure that the data processing pipeline selected in our solution is aligned or complementary with other projects working on the same topic, promoting collaboration within the solutions in the future. For example, we performed periodical meetings with the DataBench[30] project, which helped us to improve the validation phase of I-BiDaaS according to the market relevant benchmark KPIs.

During the project lifetime, we encouraged the use of standards and best practices towards the future sustainability and compatibility of our solution, the list of current standards will be provided in an annex in D6.5. From the architectural point of view, our proposed architecture is layered aligned with the Cloud computing - Functional architecture of Big Data as a Service defined by ITU[31] and Cloud Federation Reference architecture described by NIST[32]. The COMPSs runtime environment is compliant with the Open Cloud Computing Interface (OCCI[33]) promoted by the OpenGridForum[34] and our deployment can be described using Topology and Orchestration Specification for Cloud Applications (TOSCA[35]) defined by OASIS[36].

Knowledge-centric

In order to disseminate the project results and achievements together with the platform modules and the project public reports, I-BiDaaS releases the Platform Documentation Guidelines and Best Practices that will be published on the project website. The aforementioned guidelines together with the open-source version of the platform boost future sustainability for organisation not involved directly within the I-BiDaaS consortium.

Additionally, towards the future exploitation of the I-BiDaaS results, one of the business models proposed includes Training and Learning activities that the academy partners could exploit in the future. Additionally, the pilot partners involved in the project can be considered early adopters through the experimental training environments deployed in their premises. CaixaBank, CRF and TiD used these training environments to demonstrate the platform to external audience during the project workshops and hackathons.

Further information about the future exploitation of the platform and its associated business models and activities can be found in Deliverable "D7.8 Third report on Exploitation Strategy and activities".

[29] https://www.bdva.eu/

[30] https://www.databench.eu/

[31] https://www.itu.int/rec/dologin_pub.asp?lang=e&id=T-REC-Y.3519-201812-I!!PDF-E&type=items

[32] nist.gov/system/files/documents/2019/07/09/nist_cfra_20190709_draft_v1.0.pdf

[33] https://occi-wg.org/

[34] https://www.ogf.org

[35] https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=tosca

[36] https://www.oasis-open.org/

5.2 I-BiDaaS platform open-source version

5.2.1 Rationale

To support the sustainability of the I-BiDaaS technical achievements and disseminate the expertise that the consortium gathered during the project, it was decided to create and publish an open-source version of the I-BiDaaS platform. Regardless of the possible commercial exploitation of I-BiDaaS by the consortium, the open-source version can be used by the community as a Big-Data analytics platform.

The architecture of this open-source version is depicted in Figure 30.

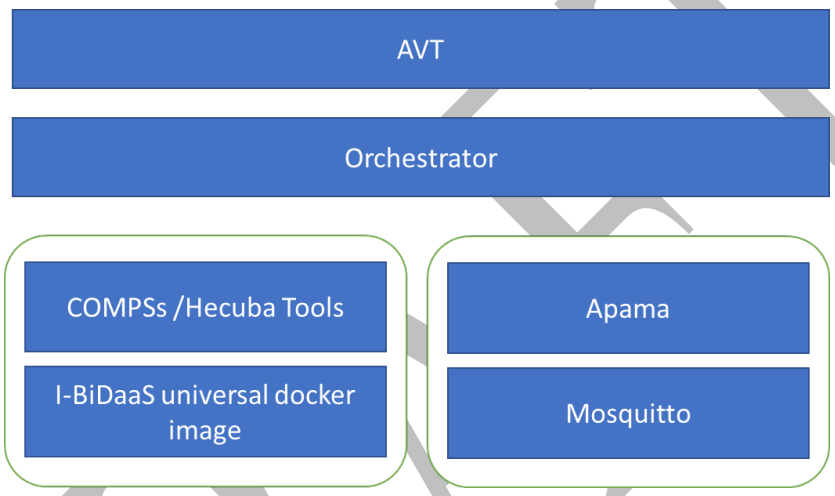

Figure 30. I-BiDaaS open-source components

The components from the I-BiDaaS platform that are included in the open-source version are:

- The Advanced Visualisation Toolkit (AVT) by AEGIS
- The Orchestrator by ITML
- The COMPSs/PyCOMPSs programming model and the Hecuba Tools by BSC
- The docker image[37] that contains code and libraries developed by UNSPMF and BSC is used for running the batch processing experiments
- The APAMA community version by SAG. In order to create and run a streaming use case in the open-source version, UM that is a commercial product can be replaced by an open-source MQTT compatible broker like Mosquitto[38]

5.2.2 Installing the I-BiDaaS platform Open-source version

The installer, available in the I-BiDaaS knowledge repository[39] creates a docker swarm with one node acting as a manager, also running the orchestrator and the I-BiDaaS AVT, and one or more workers used for running the PyCOMPSs jobs.

[37] https://hub.docker.com/repository/docker/vchatzi/ibidaas-universal

[38] https://mosquitto.org/

[39] https://github.com/ibidaas/knowledge_repository/tree/master/tools_technologies/ibidaas_installer/ibidaas_installer

To run the installer, there must be at least two (three recommended) available CentOS[40] hosts (or virtual machines), with SSH and public key authentication enabled.

Pre-Installation Requirements

One host running ansible[41] V2.7 or higher, access (via SSH with public key authentication) to two or more CentOS nodes. A two-node installation will result in installing the software shown in Figure 31.

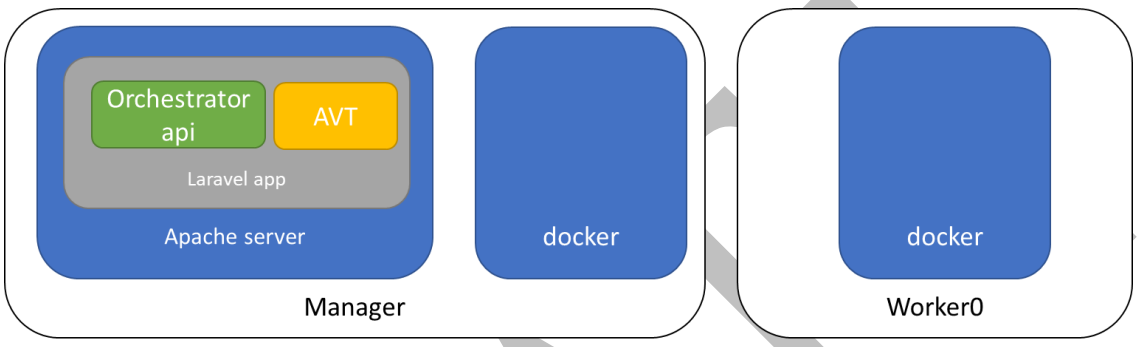

Figure 31. Installed software schema

Installation and Configuration Procedure

Auto deployment of Docker Swarm (multimanager) cluster (including docker installation)

- Edge version of docker will be used for cluster deployment
- Docker cluster will be deployed with N-managers, N-workers
- Hosts are used in swarm-cluster already will be skipped

Swarm hosts requirements:

- CentOS \ RedHat
- Internet connection (to get packages from repos)

Ansible host requirements:

- Ansible v2.7 (or newer)
- Jinja[42] 2.9 (or newer)
- Ssh-keys should be copied to all hosts of your inventory (ssh-copy-id can be used)

Cloning repo from gitlab

git clone https://github.com/ibidaas/installer

[40] https://www.centos.org/

[41] https://www.ansible.com/

[42] https://jinja.palletsprojects.com/en/2.11.x/

Edit hosts file

In this file, we define the hosts to be used. An example configuration with one manager and two workers is the following:

```
manager ansible_ssh_host=192.168.15.100
worker1 ansible_ssh_host=192.168.15.101
worker2 ansible_ssh_host=192.168.15.102


[docker-swarm-manager]
manager

[docker-swarm-node]
worker1
worker2

[docker-swarm:children]
docker-swarm-manager
docker-swarm-node
```

Running ANSIBLE-PLAYBOOK

ansible-playbook -h hosts playbook swarm-cluster.yml

ansible-playbook -h hosts playbook site.yml

ansible-playbook -h hosts playbook manager.yml

6 Conclusions

According to the DoA, for WP5 the "*the primary objective is to provide the distributed large-scale framework that permits powerful and scalable Data processing on top of heterogeneous and federated infrastructures.*"

The current deliverable presents the third version of the "Big-Data-as-a-Self-Service Test and Integration Report". The work that corresponds to tasks T5.3, and T5.4, is the result of the continuous integration process that was described in the previous releases of the I-BiDaaS solution, ensuring a smooth and effective integration of the separate I-BiDaaS components, taking into consideration their availability, interoperability potential, scalability and performance requirements.

The list of actions performed to overcome the WP challenges are summarized in the table below:

Table 5. Challenges of WP5 Tasks 5.3, 5.4 and counter-actions.

Challenge	Summary of actions to M36
Implementation and deployment of I-BiDaaS's integrated framework (T5.3)	• We have successfully released 3 versions of the I-BiDaaS platform. • The I-BiDaaS platform evolved in a holistic solution that can be used by different stakeholders, because it provides 3 complementary modes of operation (Expert mode, Self-Service mode and Co-Develop mode) (see Section 2).
Definition of test cases and quality tests with respect to industrial-validated benchmarks (T5.3)	• In cooperation with WP6, all I-BiDaaS use cases have been integrated and validated (see Section 3, Section 4.3 and D6.5 Section 3.3). • Moreover, benchmarking of the I-BiDaaS end-to-end solution was performed (see Section 4.2 and D6.5 Section 3.2)
Provision of different instantiations of the platform, with different data sources (T5.4)	• The use cases integrated in the Co-Develop mode showcase how the I-BiDaaS platform can be utilized in different sectors and with heterogeneous data sources (see Section 3). • Different instances of the I-BiDaaS platform have been installed in the main I-BiDaaS infrastructure provided by ATOS, in TID premises and in a public cloud provider.
Achieving future sustainability (T5.4)	• An analysis of the future sustainability of the I-BiDaaS platform. Four different approaches (techno-centric, data-centric, functionality-centric and process-centric) for achieving software sustainability were presented in Section 5.1. • An open-source version of the I-BiDaaS platform, provided in an installation package was developed and presented in Section 5.2

References

[1] Taylor, Sean J., and Benjamin Letham. "Forecasting at scale", The American Statistician 72.1 (2018): 37-45.

[2] "PyCOMPSs: Parallel computational workflows in Python", Enric Tejedor, Yolanda Becerra, Guillem Alomar, Anna Queralt, Rosa M. Badia, Jordi Torres, Toni Cortes, Jesús Labarta, IJHPCA 31(1): 66-82 (2017), DOI: 10.1177/1094342015594678

[3] T. Chen, C. Guestrin, "XGBoost: A Scalable Tree Boosting System", KDD '16: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 2016 Pages 785–794.

[4] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, Andrey Gulin, "CatBoost: unbiased boosting with categorical features", NIPS'18: Proceedings of the 32nd International Conference on Neural Information Processing Systems, December 2018, Pages 6639–6649.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, "Deep Residual Learning for Image Recognition", 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

[6] Paszke et al., "PyTorch: An Imperative Style, High-Performance Deep Learning Library", 2019, available at: https://arxiv.org/abs/1912.01703

[7] C. Bishop, "Pattern Recognition and Machine Learning", Springer, 2006.

[8] Russakovsky, Olga, et al. "Imagenet large scale visual recognition challenge", International journal of computer vision 115.3 (2015): 211-252.

[9] Robert L. Thorndike (December 1953). "Who Belongs in the Family?". Psychometrika. 18 (4): 267–276. doi:10.1007/BF02289263.

Appendix 1 – Orchestrator REST API

Route	Type	Description
experiment/update-status/{id}	GET	Sets status of experiment to terminated. Used internally when compss terminates
custom-experiment/update-status/{id}	GET	Sets status of experiment to terminated. Used internally when compss terminates
project	GET	Gets the list of projects (predefined projects)
project/experiments/{id}	GET	Gets the list of project experiments owned by the current user, project is specified by project_id
project/{id}	GET	Gets the details of a project, specified by project_id
experiment/run/{project_id}	POST	Starts a new experiment for project, specified by project_id
experiment/{id}	GET	Gets the details of experiment, specified by id
experiment/stop/{id}	GET	Requests the termination of an experiment, specified by id
experiment/download/{id}	GET	Returns a package (tarball) of the results of an experiment, specified by id
experiment/{id}	DELETE	Deletes the specified experiment
experiment/output/{id}	GET	Returns the results of an experiment, in JSON format
input-dirs	GET	Returns a JSON representation of the available directories to be used as input, for the current user
input-files	GET	Returns a JSON representation of the available files to be used as input, for the current user
keyspaces	GET	Returns a JSON representation of the available Cassandra keyspaces to be used as input, for the current user
custom-project	POST	Creates new project, owned by the current user
custom-project	GET	Gets the list of custom projects of the current user
custom-project/{custom_project_id}	GET	Gets the details of a project, specified by project_id

custom-project-update	POST	Updates the parameters of a project, specified by project_id
custom-project/{custom_project_id}	DELETE	Deletes the specified project
custom-experiment/run/{custom_project_id}	POST	Starts a new experiment for project, specified by project_id
custom-experiment/{custom_experiment_id}	DELETE	Deletes the specified experiment
custom-project/experiments/{custom_project_id}	GET	Returns a list of experiments, belonging to a custom project, specified by custom_project_id
custom-experiment/stop/{custom_experiment_id}	GET	Requests the termination of an experiment, specified by custom_project_id
custom-experiment/download/{id}	GET	Returns a package (tarball) of the results of the specified experiment