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

Industrial-Driven Big Data as a Self-Service Solution

D2.5: The Data Fabrication Platform (DFP, Final Version)[†]

Abstract: This report contains details about the I-BiDaaS automated methodology for fabrication of synthetic and realistic data sets. Various data analyses results are ingested by test data fabrication platform and are used to create a set of constraint rules that model the learned data. Using this model, the user can fabricate large data sets that exhibit similar properties as the real data, thus making it realistic. The work described in this document is the implementation of an end to end automation process where metadata combined with data analysis results are fed into the platform to automatically fabricate complete realistic data sets.

Contractual Date of Delivery	31/12/2019
Actual Date of Delivery	30/12/2019
Deliverable Security Class	Public
Editor	*Omer Boehm (IBM)*
Contributors	IBM, FORTH, SAG, BSC, CAIXA, TID, CRF
Quality Assurance	*Dr. Gerald Ristow (SAG)* *Dr. Ramon Martin de Pozuelo Genis (CAIXA)* *Dr. 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. *Dr. Gerald Ristow,* SAG
2. *Dr. Ramon Martin de Pozuelo Genis,* CAIXA
3. *Dr. Kostas Lampropoulos, (FORTH)*

Revisions

Version	Date	By	Overview
0.0.8	19/12/2019	Internal reviewers	Final review and approval
0.07	17/12/2019	Editor	Incorporation of the final comments
0.0.6	11/12/2019	CAIXA + SAG first review	Comments on the final draft
0.0.5	09/12/2019	Editor	Final draft
0.0.4	21/11/2019	Partners Contributions	CAIXA and SAG inputs
0.0.3	05/11/2019	Editor	Final ToC
0.0.2	02/11/2019	Quality Assurance, Internal reviewers	Comments on the first draft
0.0.1	22/10/2019	Editor	First Draft ToC

Table of Contents

List of Figures

List of Abbreviations

CSP – Constraint Satisfaction Problem

CSV – Comma Separated Value

DB – Database

EMD – Earth Mover's Distance

FK – Foreign Key

JSON – JavaScript Object Notation

KL – Kullback-Leibler

PK – Primary Key

MQTT – Message Queuing Telemetry Transport

MVP – Minimum Viable Product

TDF – Test Data Fabrication

UM – Universal Messaging

UML – Unified Modeling Language

XML – Extensible Mark-up Language

XLS – Microsoft Excel File

Executive Summary

In this document, we will provide a complete automated methodology for the fabrication of synthetic and realistic data sets. The generated synthetic data will:

(i) solve the problem of security and leakage of sensitive data to unauthorized third parties, and

(ii) provide data for early exploration and development phases of the applications in cases where real data does not exist or is not available yet.

Our goal in this task is to complete the work to extend IBM's Test Data Fabrication (TDF) [1] in order to fabricate data that is as realistic as possible and should have all the characteristics of the underlying real data that are important for processing and providing analytics. We implemented the interfaces and the support to ingest the extracted data and metadata logic combined with real data analysis to automatically create fabrication constraint rules that the platform provides for data modeling.

Data analysis can be performed using machine-learning algorithms for meta-data and data analytics developed in WP3 and also by standard data analysis tools. Automated data analysis significantly eases the complex and time-consuming process of manual data characterization and search for relationships.

This deliverable, being the final TDF deliverable, also provides the latest status update of the datasets that were generated for the various use cases as well as a quality evaluation methodology for them.

1 Introduction

This document is the second and final deliverable related to Test Data Fabrication (TDF) in WP2, which addresses data curation ingestion and pre-processing. It covers the user requirements and the Minimum Viable Product (MVP) needs and interactions expected between the I-BiDaaS actors and the offered environment. The work presented in this deliverable also relates to the work performed in WP3 regarding the evaluation and validation of the platform through real-life industrial experiments.

The purpose of this deliverable is to provide an automated methodology for the fabrication of synthetic and realistic data sets using IBM's technology.

Constructing a data model is achieved via a set of constraint rules. The rules characterize every aspect of the data, and only a solution that satisfies all the constraints is considered valid data.

Proper modeling of complex data is typically a very difficult and time-consuming task.

The data modeler must be highly familiar with the underlaying data set and the assumptions applications make when using it. The modeler must also be familiar with the solver language and the available constraints in order to capture the true nature of the data.

This deliverable, being the final TDF deliverable, also includes the latest status update of the datasets that were generated for the various I-BiDaaS use cases as well as a quality evaluation methodology for it.

The structure of the deliverable is as follows. Chapters 2 and 3 includes an overview of the data analyses and resulting formats, that can be ingested into the TDF platform, followed by details about the modeling automation component which enables the platform to create appropriate constraint rules. Next, Chapters 4 details about the status of the datasets that were fabricated for the I-BiDaaS use cases and their challenges. Chapter 5 provides details about the position of the TDF module with respect to the overall I-BiDaaS platform, as depicted in Figure 1. The module interacts directly with the Universal Messaging bus and Hecuba and COMPS. Finally, Chapter 6 provides conclusions and outlines next steps in development.

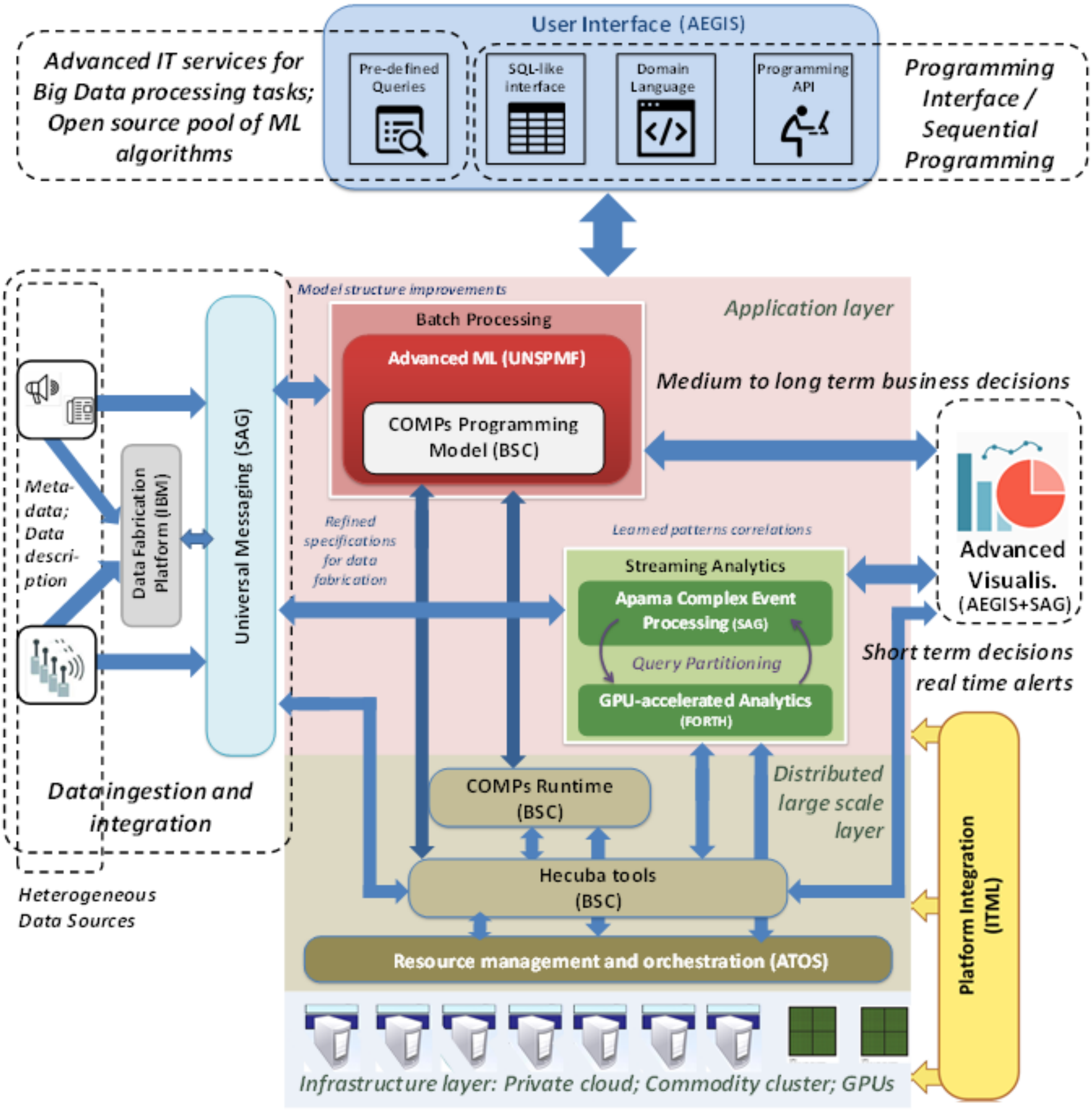

User Interface (AEGIS)
Pre-defined Queries
SQL-like interface
Domain Language
Programming API
Advanced IT services for Big Data processing tasks; Open source pool of ML algorithms
Programming Interface / Sequential Programming
Model structure improvements
Application layer
Batch Processing
Advanced ML (UNSPMF)
COMPs Programming Model (BSC)
Medium to long term business decisions
Advanced Visualis. (AEGIS+SAG)
Meta-data; Data descri-ption
Data Fabrication Platform (IBM)
Universal Messaging (SAG)
Refined specifications for data fabrication
Learned patterns correlations
Streaming Analytics
Apama Complex Event Processing (SAG)
Query Partitioning
GPU-accelerated Analytics (FORTH)
Short term decisions real time alerts
Data ingestion and integration
Heterogeneous Data Sources
COMPs Runtime (BSC)
Distributed large scale layer
Hecuba tools (BSC)
Resource management and orchestration (ATOS)
Platform Integration (ITML)
Infrastructure layer: Private cloud; Commodity cluster; GPUs

Figure 1. The I-BiDaaS platform

2 Data analysis

Data analysis is an attempt to construct a simple yet general data model, which can then be used to fabricate synthetic realistic workloads at will, possibly with slight (but well controlled) modifications. The goal here is to be able to create datasets that can be used for application testing, performance evaluations, data augmentation or evaluation studies. The synthetic dataset is supposed to be 'similar' to those that occur in practice on real world use cases.

The essence of the modeling, as opposed to just observing and recording, is one of abstraction. This means two things: generalization and simplification. The most common approach used in descriptive modeling (often called exploratory data analysis) is to create a statistical summary of the observed dataset. (see - **Figure 2**). This can be further extended with the identification of relationships, patterns, types, formats etc. found in the data.

Within the class of descriptive models, typically one would look for the simplest abstract mathematical model that has certain properties that are considered as most important. A synthetic workload can be generated for such model by sampling from the distributions that constitute the model. The model can also be used directly to parameterize a mathematical analysis.

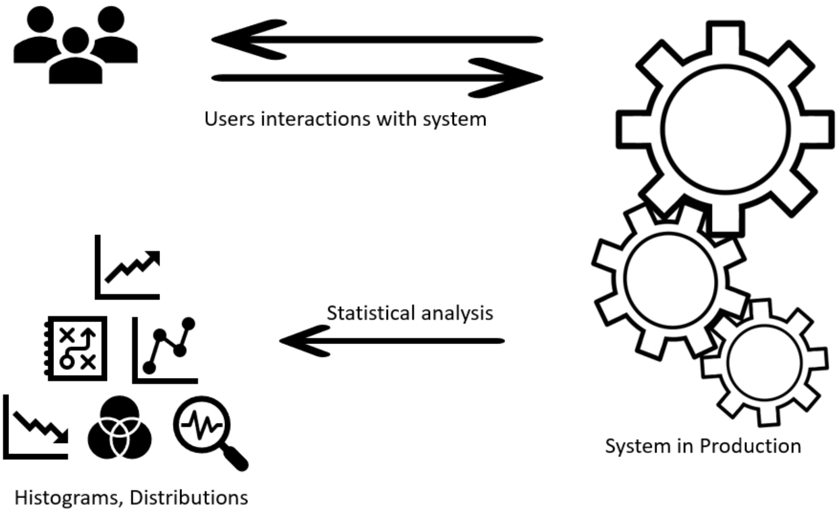


Figure 2. Dataset modeling based on statistical summary

Data analysis of structured data typically includes specific types or classes. E.g., data class, numeric, categorical, text and date, and can be performed as a ***single attribute*** level or at a ***multiple attribute*** level. Generic data classes could be defined as follows:
Code – System specific data values from a value domain. Each of the values has a specific meaning. E.g., product status codes.
Date - Data values which are specific date, time, or duration references. E.g., product order date.
Identifier - Data values that are typically unique and are used to reference a specific entity. E.g., product number.
Indicator - Code values that have only two mutually-exclusive values in the domain set. E.g., product Make / Buy indicator. They are often called Flags.
Quantity - Numerical data values that could be used in a computation. E.g., product price.

Text - Free-form alphanumeric data values from an unlimited domain set. E.g., product description.

Each of these could naturally be further refined into a more specific class. E.g. ‘birthdate’ that falls under **date** or ‘social security number’ that falls under **identifier** etc.

In addition to the data classes, data analysis can infer properties of attributes and provide an overview of properties for a wide variety of attributes in a data set. Analysis of single attribute numeric types may infer specific types (e.g. short, int, long, decimal, big int, boolean, etc. possibly along with length, precision, scale additionally inferred), uniqueness, distinctiveness, the option to have null values, distribution, min, max and other statistical properties. Analysis of single attribute categorical types should infer the value-list of the categories along with their frequency. Analysis of single attribute text types can infer interesting patterns that may be predefined e.g., credit card number, zip code, country code etc. or just frequent formats ’99.9’, ‘AAB’, ‘Yes’. Dates can be analyzed for starting date, end date, time ranges, distributions, date formats, etc.

Analysis of multiple attribute types is much wider and more complex. This analysis can infer important value correlations between various attributes. Below are a few examples.

- Numeric to numeric polynomial or piecewise curve fitting relation.
- Category to category frequency distribution relation.
- Category to numeric infers a per category statistical distribution relation.
- Date to date infers time difference relation.
- Category to numeric to numeric infers category wise polynomial curve fitting relation

Within I-BiDaaS, the data analysis is performed by Hecuba and COMPS. (Further details are given at ‘Integration with Hecuba and COMPS’ section). The dataset is analysed by Hecuba and produces various descriptive statistics metrics for Dates, Integers, Floats, Decimals, Longs, and Bigints. The analysis is performed per column, and produces a JSON file written in an agreed schema, containing its count, mean, stdev, variance, max value, min value, count of null, distinct and empty values, length, etc.

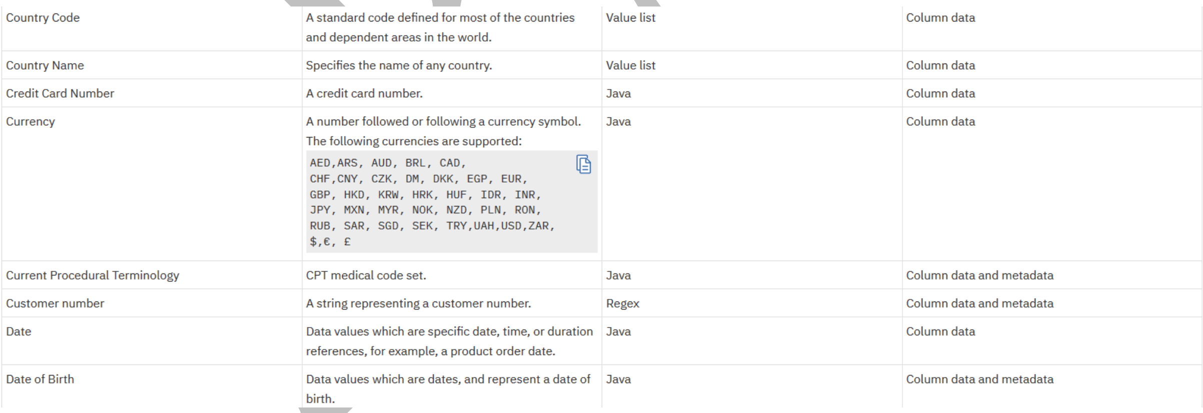
3 Modeling Automation

The modeling of data is a difficult and time-consuming task that typically requires manual work. The data modeller must be familiar with the dataset nature, formats and attributes. Data modeller must also know the statistical properties and constraints over the dataset as well as the importance of features and their relationships and how they affect the applications that are using it. Moreover, the data modeller must know the modeling language of the CSP [3-21] solver and how to translate the data properties into constraint rules of the CSP language. Data analysis (see - Data analysis section above and also Integration with Hecuba and COMPS) of real dataset produces descriptive statistics which can be leveraged to automatically create constraint rules that enable the fabrication of similar synthetic data.

We have implemented a new module in TDF that can ingest descriptive statistics and automatically create a set of corresponding constraint rules and insert them into a TDF project to enable fast creation of new data. This module supports various interfaces for analyses produced by various sources. Among the various sources are the analysis results produced by Hecuba and COMPS. (Further details at – Integration with Hecuba and COMPS).

The required support for the new functionality was added to the web UI and the REST API was extended to include the procedures that ingest the analysis and performs the automated rules creation.

The module is leveraging both metadata and data analysis in order to create the required constraint rules. The meta data includes columns or fields names and their types. In addition to this, in database tables, there are referential integrity constraints, e.g., primary and foreign keys, check constraints, etc. TDF has a template library, which can be extended by the user, with template constraints. The library also contains instances of these templates as concrete rules for common terms. (see **Figure 3**)



Country Code	A standard code defined for most of the countries and dependent areas in the world.	Value list	Column data
Country Name	Specifies the name of any country.	Value list	Column data
Credit Card Number	A credit card number.	Java	Column data
Currency	A number followed or following a currency symbol. The following currencies are supported: AED,ARS, AUD, BRL, CAD, CHF,CNY, CZK, DM, DKK, EGP, EUR, GBP, HKD, KRW, HRK, HUF, IDR, INR, JPY, MXN, MYR, NOK, NZD, PLN, RON, RUB, SAR, SGD, SEK, TRY,UAH,USD,ZAR, $,€, £	Java	Column data
Current Procedural Terminology	CPT medical code set.	Java	Column data and metadata
Customer number	A string representing a customer number.	Regex	Column data and metadata
Date	Data values which are specific date, time, or duration references, for example, a product order date.	Java	Column data
Date of Birth	Data values which are dates, and represent a date of birth.	Java	Column data and metadata

Figure 3. Templates ready for common terms

The data analysis results contain descriptive statistics. Now the module supports single column/field results. The module contains all the required support to parse the JSON file, and to create the internal data analysis representation. Constraint rules can be generated for ALL columns or only for a subset selected by the user. Generated constraint rules are grouped together in a rule set, which is created per table in database projects. For each column, the module is looking at its analysis results and is creating the appropriate constraint. For example, if the column is numeric, and contains lower and upper bounds, the 'max' and 'min' constraints will be created. In case a distribution was found, or even a histogram made of bins, the 'randomWeightedValue' constraint will be created.

The module supports various types of results for numeric, dates and text columns. Constraint rules generated by the automation module, are created with various priorities lower than 'mandatory' by default. This means that the CSP solver can provide a solution that doesn't satisfy all constraints, however, provides some solution that is satisfying as many constraints as possible. This is done mainly to avoid potential errors due to conflicting constraints.

4 Fabricated datasets for the use cases

TDF was used to generate datasets for the following uses cases

- Analysis of relationships through IP address – CAIXA.
- Production process of aluminium die-casting – CRF.
- Accurate location prediction with high traffic and visibility and & Optimization of placement of telecommunication equipment – TID.

Synthetic data was fabricated on an I-BiDaaS dedicated VM into PostgreSQL DB, SQLite and csv files. TDF projects were defined, and the data was successfully generated, however changes in the fabrication approach were made (see section Data fabrication via simulation) due to inaccurate results in the TID use case. The data evaluation by the use case providers is described below.

4.1 Data fabrication via simulation

The 'TID mobility data' dataset is used in the following two use cases

- Accurate location prediction with high traffic and visibility
- Optimization of placement of telecommunication equipment

The dataset defined for this use case in D2.1 posed a new challenge to TDF as the characterization included cyclic dependant distributions, i.e.

Distribution of events over users:

Min.	1st Qu.	Median	3rd Qu.	Max.
1.00	15.00	100.00	350.00	10000.00

Distribution of sectorIDs over users:

Min.	1st Qu.	Median	3rd Qu.	Max.
1.00	4.00	14.00	35.00	200.00

Distribution of events over sectors:

Min.	1st Qu.	Median	3rd Qu.	Max.
1.00	80.00	3500.00	15000.00	1e7

The data requirements are properties learned from real data which are collected from 10s of millions of users over 10s of thousands of sectors. The number of events of a user per day can range between 0 (if mobile phone is shut down) to 10s of thousands. This data is formed by so-called cell network events which are picked up by the antennas that are closer to the mobile phone (for further details see D2.1 and section Accurate location prediction with high traffic and visibility of this deliverable).

Generating such data using TDF constraint rules was challenging and the data that was initially fabricated did not satisfy some of the requirements.

To overcome this, we chose to use a simulation engine that essentially simulates the real way data gets collected. We used SimPy [27], which is a process-based discrete-event simulation framework based on standard Python. Processes in SimPy are defined by Python generator functions and are used in our case to model active customers. SimPy also provides various types of shared resources to model limited capacity congestion points. We implemented the TID mobility data dataset simulation that uses the SimPy framework property that enables simulation to be performed "as fast as possible". This property enables a very efficient methodology to generate extremely large volumes of data as the processes don't interact with each other or with shared resources.

The implementation requires the distributions in quantiles, and a desired number of users. It then spawns the desired number of concurrent users and starts the simulation of creating events.

4.2 Analysis of relationships through IP address

The fabricated data was created using TDF according to a set of rules defined by CAIXA. The rules were refined several times in order to create realistic data for all different fields considering the format of the real data. It is not possible to distinguish a data sample from a field in the synthetic dataset and a sample from the same field in the real dataset. The rules also specified that only a specific percentage of the entries should have a relationship, and the analysis of the created dataset accomplishes that rule. It is hard to specify in a rule the concrete time connectivity patterns that the real data follows and was not included in the specification of the synthetic dataset. However, that parameter was not critical for the relationship analysis done in the use case and the synthetic dataset was completely valid for assessing that there exist the same percentage of relationships as in the real dataset.

Further analysis with a set of algorithms (K-means, DBSCAN) was done using the synthetic dataset and a tokenized version of the real dataset[1].

K-means. The results obtained using K-means clustering on real tokenized data show higher silhouette scores than with the synthetic data, which may suggest that the clusters are in a better agreement with the data than in the synthetic case. Using the synthetic data, only 1 big cluster is obtained, with clusters of size 2 dominating. On the other hand, using the tokenized data, there are 2 big clusters containing most of the points, while the single point clusters dominate. This might offer an explanation for the higher silhouette scores, as single point clusters tend to inflate the metric.

Additionally, a large number of single-point clusters on the tokenized real data might point to users that may have specific transaction patterns and might offer an interesting direction for future research and analysis.

DBSCAN. The results obtained using DBSCAN on real tokenized data show higher silhouette scores for all, except for the final two values of the parameter eps[2]. Coupled with the drastically decreasing number of clusters found for the final two values of eps (from 681 to 6 clusters found), it can be hypothesized that, giving a too loose maximal distance between two points to be considered as being in the same cluster causes almost all of the points to be clustered in a small number of clusters.

Comparing the results on the real tokenized data with the results on synthetic data using DBSCAN, it can be observed that, with both the real and synthetic data, the majority of the clusters found are 2-point clusters. However, with real data, two big clusters containing the majority of the points in the dataset are found, while with the synthetic data, one big cluster was found. Also, increasing the eps parameter on the synthetic data increases the number of clusters, as well as the silhouette score. On the other hand, the opposite effect can be observed on the

[1] By tokenized, we mean an encryption of all the fields of the dataset done inside the premises of CAIXA and sharing the encrypted data. The analysis performed in the use cases allows to extract some conclusions and find relationships with the encrypted data. That enables to do the analysis with encrypted data and decrypt the results once again in CAIXA premises, with the algorithm and keys that will be custodied by CAIXA and will only be available inside CAIXA.

[2] Maximum distance between two samples for them to be considered as in the same neighbourhood.

tokenized real data, where increasing the eps parameter leads to a decrease in both the number of clusters found and in the silhouette score.

4.3 Accurate location prediction with high traffic and visibility & Optimization of placement of telecommunication equipment

To facilitate the early exploration and development phases of the applications in cases where real data was not readily available, TID resorted to sharing the data characteristics and format of the relevant datasets in order to synthesize data using IBM’s Data Fabrication Tool. By realistic synthetic data, we refer to the fabricated data that “mimics” a real dataset, based on metadata descriptions and other data specification rules.

This helped in identifying, to some extent, the practical issues, requirements and idiosyncrasies of each use case, and take the appropriate course of action to address the respective challenges. Moreover, it allowed the relevant partners to think critically about what is possible to achieve within each use case and acknowledge existing limitations, by taking a first look at the fabricated data.

More specifically, TID shared with the consortium anonymized and aggregated data that allowed the data synthesis for the ‘TID_Mobility_data’. The current size of this dataset can scale to over 4TB per day (for a European country). This corresponds to 10s of millions of customers, and some 10s of thousands of sectors in that country. The number of events of a user per day can range between 0 (if mobile phone is shut down) to 10s of thousands. These data are formed by so-called cell network events which are picked up by the antennas that are closer to the mobile phone, thus providing an approximate location of the device. Every transaction of a mobile phone generates one of those events. A transaction can be, for instance, placing or receiving a call, sending or receiving an SMS, asking for a specific URL in your mobile phone browser, or sending a text message or a data transaction from/to any mobile phone app. There are also some synchronization events like, for instance, turning your mobile phone on or off, or when switching between location area networks (relatively big geographical areas comprising several cell towers).

This data comes in tabular form. There are three main tables: one related to the events, and two complementing those, related to the users that generate those events and the antennas that deliver the service and collect those events. Among these, we keep only the information, which is strictly related to mobile phone users' mobility, but log much more. Each event contains the start/end timestamp, the start/end sector ID, the country, and other event data (such as the device type, etc.), the antenna latitude/longitude, the azimuth and beam width, other antenna data (such as the frequency at which the antenna is emitting, or the height at which is located), the user ID, and other customer data (such as contract information from the user).

After the first round of the data fabrication was completed by IBM, TID proceeded with the validation and verification of the fabricated data quality. First, TID extracted from the corresponding PostgreSQL DB, of the I-BiDaaS platform, the relevant data. Following that, TID computed a series of summary statistics, histograms, and density distributions on the fabricated data, and compared them against those reported originally in D2.1.

In the majority of cases, and for most of the variables, we computed the summary statistics, histograms and density distributions. More specifically, we used the R platform to compute descriptive statistics (min, max, mean, median, SD) and IQ distances, which we then matched to those reported in D2.1. Whenever possible, and when density distributions were available, TID computed additional metrics such as the symmetric Kullback-Leibler [24], [25], [26] divergence and the Earth Mover’s Distance [22] [23], which are both measures of distance between two probability distributions. More specifically, the distributions are sets of weighted features that capture the distributions and the EMD is defined as the minimum amount of work needed to change one sequence into another. The notion of work is based on a unit of ground distance. Similarly to the KL divergence, we computed the EMD between the two sets of distributions. For those cases that the distributions deviated from the original ones, they were flagged and are being further investigated.

4.4 Production process of aluminum casting

Waiting to implement the algorithms to anonymize the data, it was necessary to create synthetic data using the IBM's Data Fabrication Tool.

This allowed the other project partners to start working on the data format and the nature of the same.

To allow the realization as close as possible to the real data, real data were analyzed by the CRF, the most significant information was extracted from them, such as the average, the minimum, the maximum, etc. This allowed, with the help of IBM, to set rules to synthesize a first dataset. More specifically, CRF provided at the consortium a file containing the most significant process parameters for the production of 1 million engine crankcase. These parameters reflect the trend of the actual production, with various and heterogeneous information.

The rules for the production of synthetic data were set by CRF with the help of IBM. The rules were specific to each individual parameter and they were independent of the other data of the same crankcase, and they were independent of the parameters of the previous crankcase. The output file of the IBM’s Data Fabrication Tool is a formatted text file, convertible into Excel that contains 1 million rows, which correspond to the single engine crankcase, and 17 columns containing the most important process parameters.

After the hackathon in Melfi, in which anonymized data was delivered to the partners, the correlation between the real parameters and the synthesized parameters was further refined.

For the validation of the synthetic data we proceeded in 2 ways, the first empirical and the second analytical. The empirical technique consisted of delivering these data to the expert production technicians, they did not notice any difference with the actual production data, as there was no distinguishing factor for them. The second analytical technique was carried out by the CRF. The technique is K-Means.

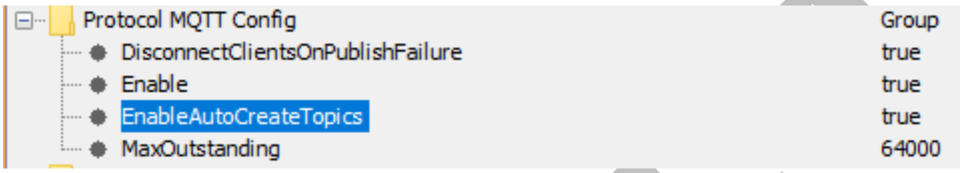
This algorithm allows you to subdivide a set of objects into K groups based on their attributes. Each K group (also called cluster) is characterized by a "representative" element - the centroid - which will be used as a yardstick with the elements present in the input set in order to determine whether a given data belongs to one of the identified clusters.
The technique was tested on a mixed dataset, real data with synthesized data, the algorithm was not to distinguish the reals from the synthesized ones and created mixed subgroups, even with K variations.

5 Integration with I-BiDaaS components

5.1 Integration with the universal messaging bus

TDF was enhanced to support sending the fabricated data in addition to a file or a database to a message broker that supports the MQTT protocol. Universal Messaging supports MQTT version 3.x and IBM's TDF sends the data in a compatible format. This is exactly why we decided to use the ISO standard MQTT for machine-to-machine communication so that the integration of the various components can be done easily. Various programming examples and open source client libraries are available so that the integration with Universal Messaging can be done fast.

In Universal Messaging, the following MQTT relevant parameters can be configured:



```
Protocol MQTT Config                      Group
    DisconnectClientsOnPublishFailure     true
    Enable                                true
    EnableAutoCreateTopics                true
    MaxOutstanding                        64000
```

The most relevant for us was to enable MQTT support in Universal Messaging and allow for automatic creation of topics.

5.2 Integration with Hecuba and COMPS

One of the goals of the I-BiDaaS project is to simplify the generation of synthetic data that can be used for testing and developing purposes. The best way to do so is to use a batch application to analyse the real data and to produce descriptive statistics. This information can then be used to create rules that allow generating synthetic data automatically.

For instance, the batch layer can calculate the max, min and standard deviation of a field, and provide such information to the TDF, so that it can generate internal rules that follow the detected statistical distribution.

Therefore, in the current workflow, the real data is analysed by a new Hecuba module called hecuba.Stats, where the various descriptive statistics metrics have been implemented to run in a distributed environment.

The current implementation supports:

- for Dates, Integers, Floats, Decimals, Longs, and Bigints:
  - count
  - mean
  - stdev
  - variance
  - max
  - min
  - count of empty values.
- for Booleans:
  - count
  - total true
  - total false
  - count empty values.
- for text fields:
  - count

  - max length
  - min length
  - total empty values.
  - For each word in the field:
    - count
    - distinct count

Once the batch application terminates, the same code uses the computed information to generate a JSON file, which follows a pre-determined schema.

```
{
   "columnName":"fk_cod_operacion",
   "columnAnalysisResults":{
      "nbRows":799999,
      "nbOfNullValues":0,
      "nbOfEmptyValues":0,
      "maxDistinctFrequency":36939,
      "minStringValue":"00000",
      "maxStringValue":"validaUsuarioWS",
      "nbOfUniqueValues":282,
      "nbDistinctValues":1570,
      "lengthDistribution":[ {
            "count":560,
            "distinctCount":2,
            "value":23
         },{
            "count":307296,
            "distinctCount":980,
            "value":5
       }],
      "wordsDistribution":[{
            "count":534,
            "distinctCount":0,
            "value":"ARQDECInformeReexecucio"
         },{
            "count":2652,
            "distinctCount":0,
            "value":"01471"
      }]
   }
}
```

Figure 4. An example of JSON file structure for a text field

As shown in Figure 4, the JSON file contains a set of statistics for each column of the target dataset. The file can be uploaded to TDF using its REST API first to set up the project, and then to start the automatic generation of synthetic data.

6 Concluding remarks

This document describes the approach of the I-BiDaaS data fabrication platform tool and how data analysis is helping the modeling automation process. The status of the fabricated data sets that were defined in I-BiDaaS D2.1 for those use case scenarios defined in WP1 is provided along with the quality evaluation methods performed by the use case providers.

Finally, we provide a description of the interactions TDF has with the universal messaging bus, Hecuba and COMPS components.

Work that is still in progress is the generation of a complete and valid dataset for the TID use cases.

Additional enhancements to the TDF tool are support for multi-attribute constraint rule generation and support for additional types of distributions and correlations.

7 References

[1] “Create high-quality test data while minimizing the risks of using sensitive production data.” *IBM InfoSphere Optim Test Data Fabrication*, IBM, 2019, https://www.ibm.com/us-en/marketplace/infosphere-optim-test-data-fabrication.

[2] “Test Data Fabrication.” *Security and Data Fabrication*, IBM Research, 2011, https://www.research.ibm.com/haifa/dept/vst/eqt_tdf.shtml.

[3] “Constraint Satisfaction.” IBM Haifa Research, IBM, 2002, https://www.research.ibm.com/haifa/dept/vst/csp.shtml.

[4] Y. Richter, Y. Naveh, D. L. Gresh, and D. P. Connors (2007), "Optimatch: Applying Constraint Programming to Workforce Management of Highly-skilled Employees", International Journal of Services Operations and Informatics (IJSOI), Vol 3, No. 3/4, pp. 258 - 270.

[5] Y. Naveh, Y. Richter, Y. Altshuler, D. Gresh, and D. Connors (2007), "Workforce Optimization: Identification and Assignment of Professional Workers Using Constraint Programming", IBM J. R&D.

[6] Y. Naveh, M. Rimon, I. Jaeger, Y. Katz, M. Vinov, E. Marcus, and G. Shurek (2006), "Constraint-Based Random Stimuli Generation for Hardware Verification", AI magazine Vol 28 Number 3.

[7] E. Bin, R. Emek, G. Shurek, and A. Ziv (2002). "Using a constraint satisfaction formulation and solution techniques for random test program generation", IBM Systems Journal, 2002.

[8] O. Boni, F. Fournier, N. Mashkif, Y. Naveh, A. Sela, U. Shani, Z. Lando, A. Modai (2012) "Applying Constraint Programming to Incorporate Engineering Methodologies into the Design Process of Complex Systems" Proceedings of the Twenty-Fourth Conference on Innovative Applications of Artificial Intelligence, Toronto, Ontario, Canada. AAAI 2012.

[9] Y. Naveh (2010). "The Big Deal, Applying Constraint Satisfaction Technologies Where it Makes the Difference". Proceedings of the Thirteenth International Conference on Theory and Applications of Satisfiability Testing (SAT'10).

[10] B. Dubrov, H. Eran, A. Freund, E. F. Mark, S. Ramji, and T. A. Schell, (2009). "Pin Assignment Using Stochastic Local Search Constraint Programming" in Proceedings of the 15th International Conference on Priniciples and Practice of Constraint Programming (CP'09), Edited by Ian P. Gent, pp 35-49.

[11] Y. Richter, Y. Naveh, D. L. Gresh, and D. P. Connors (2007), "Optimatch: Applying Constraint Programming to Workforce Management of Highly-skilled Employees", IEEE/INFORMS International Conference on Service Operations and Logistics, and Informatics (SOLI), Philadelphia, pp. 173-178.

[12] S. Sabato and Y. Naveh (2007), "Preprocessing Expression-based Constraint Satisfaction Problems for Stochastic Local Search", Proceedings of The Fourth International

Conference on Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems (CP-AI-OR).

[13] Y. Naveh, M. Rimon, I. Jaeger, Y. Katz, M. Vinov, E. Marcus, and G. Shurek (2006), "Constraint-Based Random Stimuli Generation for Hardware Verification", IAAI 2006.

[14] Y. Richter, A. Freund, and Y. Naveh (2006), "Generalizing AllDifferent: The SomeDifferent constraint", Proceedings of the 12 International Conference on Principles and Practice of Constraint Programming - CP 2006, Lecture Notes in Computer Science, Volume 4204, pages 468-483.

[15] Y. Naveh and R. Emek (2006). "Random stimuli generation for functional hardware verification as a CP application - a demo", IAAI 2006.

[16] Y. Naveh (2005). "Stochastic solver for constraint satisfaction problems with learning of high-level characteristics of the problem topography" CP 2005

[17] F. Geller and M. Veksler (2005), "Assumption-based pruning in conditional CSP", in van Beek, P., ed., CP, "Principles and Practice of Constraint Programming - CP 2005" of Lecture Notes in Computer Science (3709), 241-255 Springer.

[18] R. Dechter, K. Kask, E. Bin, and R. Emek (2002). "Generating random solutions for constraint satisfaction problems", AAAI 2002.

[19] D. Lewin, L. Fournier, M. Levinger, E. Roytman, G. Shurek (1995). "Constraint Satisfaction for Test Program Generation", Internat. Phoenix Conf. on Computers and Communications, March 1995.

[20] Hoffman, E. J., et al. "Constructions for the Solution of the m Queens Problem." Mathematics Magazine, vol. 42, no. 2, 1969, pp. 66–72. JSTOR, JSTOR, www.jstor.org/stable/2689192.

[21] Watkins, John J. (2004). Across the Board: The Mathematics of Chess Problems. Princeton: Princeton University Press. ISBN 0-691-11503-6.

[22] Elizaveta Levina; Peter Bickel (2001). "The EarthMover's Distance is the Mallows Distance: Some Insights from Statistics". Proceedings of ICCV 2001. Vancouver, Canada: 251–256.

[23] C. L. Mallows (1972). "A note on asymptotic joint normality". Annals of Mathematical Statistics. 43 (2): 508–515. doi:10.1214/aoms/1177692631.

[24] Kullback, S.; Leibler, R.A. (1951). "On information and sufficiency". Annals of Mathematical Statistics. 22 (1): 79–86. doi:10.1214/aoms/1177729694. MR 0039968.

[25] Kullback, S. (1959), Information Theory and Statistics, John Wiley & Sons. Republished by Dover Publications in 1968; reprinted in 1978: ISBN 0-8446-5625-9.

[26] Kullback, S. (1987). "Letter to the Editor: The Kullback–Leibler distance". The American Statistician. 41 (4): 340–341. doi:10.1080/00031305.1987.10475510. JSTOR 2684769.

[27] "SimPy - Discrete event simulation for Python", https://simpy.readthedocs.io/en/latest/index.html