Ref. Ares(2016)7198498 - 30/12/2016

HORIZONS 2020 PROGRAMME

Research and Innovation Action – FIRE Initiative

Call Identifier:	H2020–ICT–2014–1
Project Number:	643943
Project Acronym:	FIESTA-IoT
Project Title:	Federated Interoperable Semantic IoT/cloud Testbeds and Applications

Concept and Development for IoT Data Analytics and IoT Stream and Service Management V1

Document Id:	FIESTA-IoT-D32-161130-Draft
File Name:	FIESTA-IoT-D32-161130-Draft.pdf
Document reference:	Deliverable 3.2
Version:	Draft
Editor:	Alireza Ahrabian / Tarek Elsaleh / Francois Carrez
Organisation:	UNIS
Date:	30 / 11 / 2016
Document type:	Deliverable
Dissemination level:	PU, CO

Copyright © 2016 FIESTA-IoT Consortium: National University of Ireland Galway – NUIG-Insight / Coordinator (Ireland), University of Southampton IT Innovation – ITINNOV (United Kingdom), Institut National de Recherche en Informatique & Automatique – INRIA (France), University of Surrey – UNIS (United Kingdom), Unparallel Innovation, Lda – UNPARALLEL (Portugal), Easy Global Market – EGM (France), NEC Europe Ltd. – NEC (United Kingdom), University of Cantabria – UNICAN (Spain), Association Plateforme Telecom – Com4innov (France), Athens Information Technology – AIT (Greece), Sociedad para el desarrollo de Cantabria – SODERCAN (Spain), Ayuntamiento de Santander – SDR (Spain), Fraunhofer Institute for Open Communications Systems – FOKUS (Germany), Korea Electronics Technology Institute KETI (Korea). The European Commission within HORIZON 2020 Program funds the FIESTA-IoT project.

PROPRIETARY RIGHTS STATEMENT

This document contains information, which is proprietary to the FIESTA-IoT Consortium.
This document contains information, which is proprietary to the FIESTA-IoT Consortium.
Neither this document nor the information contained herein shall be used, duplicated or communicated by any means to any third party, in whole or in parts, except with prior written consent of the consortium.

DOCUMENT HISTORY

Rev.	Author(s)	Organisation(s)	Date	Comments
V01	Martin Serrano	NUIG-DEI	YYYY/MM/DD	Initial Draft Proposal
	Amelie Gyrard	NUIG-DERI	2015/05/27	Background and State of the art
	Amelie Gyrard	NUIG-DERI	2015/07/14	LSM upgraded specification ideas
	Amelie Gyrard, Luis Sanchez, Maria Bermudez, Martin Serrano, Philippe Cousin	NUIG-DERI	2015/07/21	Table comparing existing ontologies that we want to exploit in FIESTA-IoT
	Amelie Gyrard	NUIG-DERI	2015/07/24	State of the art IoT ontologies has been moved to D3.1 + aligning iot lite and m3 has been moved as well
	Amelie Gyrard	NUIG-DERI	2015/08/18	Karma data integration tool
	Amelie Gyrard	NUIG-DERI	2015/08/21	Reasoning Architecture API picture
	Amelie Gyrard	NUIG-DERI	2015/09/08	Section Linked Stream Processing, Calbimonte et al., Le-Phuoc et al.
	Amelie Gyrard	NUIG-DERI	2015/09/10 2015/09/17 2015/10/02	Describe Section FIESTA Reasoning Architecture API, section Linked Stream Processing and LD4Sensors, Le-Phuoc et al., Calbimonte et al.,
	Amelie Gyrard	NUIG-DERI	2015/10/06 2015/11/30	Section Challenges to address & limitations of existing work Linked Edit Rules (LER), BASIL
	Amelie Gyrard	NUIG-DERI	2016/04/13 2016/04/16	Reasoning architecture compliant with WP2 picture Description picture Experimenters interaction with the reasoning engine Light reasoning architecture for fiesta-iot Update reasoning architecture API picture
	Amelie Gyrard	NUIG-DERI	2016/04/19 2016/04/27 2016/05/20	Update following WP3 telco to take into feedback Update reasoning architecture pictures and descriptions Section S-LOR demo Update table of content
V02	Amelie Gyrard	NUIG-DERI	2016/05/20	Archived reasoning interaction with FIESTA components (NUIG UNICAN UNIS discussion Francois David)
	Amelie Gyrard	NUIG-DERI	2016/05/20	Updates S-LOR following discussions (UNICAN UNIS NUIG Jorge Ali Amelie Tarek) (algorithm, dataset of

Copyright © 2016 FIESTA-IoT Consortium

			2016/06/20	interoperable rules) Section to differentiate 2 kind of experimenters (reuse existing rules, or editing rules according to their needs)
V02	Amelie Gyrard	NUIG-DERI	2016/06/24	Updates following the reasoning meeting (UNIS/NUIG Tarek, Ali, Francois) SLOR limitations Sequence diagram + description
	Alireza Ahrabian	UNIS	2016/10/06	Updates the contents and contributes to the FIESTA-IoT Analytics chapter
V03	Alireza Ahrabian	UNIS	2016/10/11	Removed and merged sections. Also updated the data analytics section.
	Amelie Gyrard	NUIG-DERI	2016/10/13	Updates various sections
	Alireza Ahrabian/Francois Carrez	UNIS	2016/10/18	Updated references and relevant sections. Namely, description of the FIESTA-IoT reasoner sequence diagram. Also Architecture of relevant components included.
V04	Tarek Elsaleh	UNIS	2016/10/26	Updated Task 3.5 section. Updating other sections.
V05	Alireza Ahrabian	UNIS	2016/11/02	Document almost ready for internal review. Requires, small contributions from NUIG, INRIA, FOKUS.
	Rachit Agarwal	Inria	2016/11/02	Graylog
	João Bosco Jares Amelie Gyrard	NUIG-DERI	2016/11/03	Rule Registration GUI FIESTA-IoT reasoner updates
V06	Alireza Ahrabian	UNIS	2016/11/08	Document prepared for internal review. Technical Reviews.
V07	Alexander Willner	FOKUS	2016/11/12	Fed4Fire related work update
V08	Alireza Ahrabian	UNIS	2016/11/14	Draft Improvement after technical review. Significant modification of the document, namely sections 2 and 4. Prepared document for second Quality Review.
V09	Alireza Ahrabian	UNIS	2016/11/16	Amended document after quality review.
V10	Martin Serrano	NUIG	2016/11/20	Circulated for Approval
Draft	Martin Serrano	NUIG	2016/11/30	EC Submitted

Copyright © 2016 FIESTA-IoT Consortium

TABLE OF CONTENTS

1 INTRODUCTION 6

1.1 EXECUTIVE SUMMARY 6
1.2 AUDIENCE 6
1.3 TERMINOLOGY AND DEFINITION 7

2 RELATED WORK 8

2.1 LINKING DATA 8
WE FIRST PRESENT A SET OF TOOLS DEDICATED FOR THE LINKING OF IOT DATA FROM DIFFERING DOMAINS. 8
2.1.1 LD4Sensors 8
2.1.2 Linked Sensor Middleware (LSM) and CQELS 8
2.2 REASONING ON DATA 9
2.2.1 IntellegO 9
2.2.2 Large Knowledge Collider 9
2.3 DATA ANALYTICS TOOLS 9
2.3.1 Knowledge Acquisition Toolkit (KAT) 9
2.3.2 Weka 3.0 10
2.3.3 ThingSpeak 10
2.4 MONITORING IOT DATA 10
2.4.1 Graylog 10
2.4.2 Nagios 11
2.4.3 Fed4FIRE monitor 11
2.5 LIMITATIONS OF EXISTING WORK 12

3 THE SEMANTIC DATA WORKFLOW TO ANALYSE IOT DATA 13

4 FIESTA-IOT REASONER 14

4.1 RULE BASED REASONING 14
4.2 FIESTA-IOT REASONER-WEB SERVICE 14

5 FIESTA-IOT ANALYTICS 16

5.1 DATA ANALYSIS 16
5.1.1 Data Pre-Processing Techniques 17
5.1.2 Machine Learning Techniques 18
5.2 FIESTA-IOT ANALYTICS - DATA ANALYSIS WEB SERVICE 21
5.3 USE CASE – TRAFFIC DATA ANNOTATION 24

6 FIESTA-IOT MONITORING 26

6.1 REACHABILITY 26
6.2 ANNOTATIONS 26
6.3 DATA AND DATASETS 27
6.4 CONTROLLING THROUGHPUT 27

7 INTERACTION WITH FIESTA-IOT PLATFORM 27

8 CONCLUSIONS AND FUTURE WORK 29

9 REFERENCES 30

Copyright © 2016 FIESTA-IoT Consortium

LIST OF FIGURES

FIGURE 1. THE SEMANTIC IOT DATA WORKFLOW WITHIN FIESTA 13
FIGURE 2. EXPERIMENTERS PRODUCE KNOWLEDGE WITHIN THE FIESTA-IOT PLATFORM 16
FIGURE 3. FIESTA-IOT ANALYTICS: DIGITAL FILTERING EXAMPLE. 17
FIGURE 4. FIESTA-IOT ANALYTICS: OUTLIER REMOVAL EXAMPLE 18
FIGURE 5. LINEAR REGRESSION MODEL (RED LINE) FIT TO A DATA SET (BLUE DOTS). 19
FIGURE 6. A NON-LINEAR RELATIONSHIP BETWEEN THE INPUT-OUTPUT VARIABLES. 20
FIGURE 7. SCATTER PLOT OF TWO SENSOR OBSERVATIONS. 21
FIGURE 8. SCATTER PLOT OF TWO SENSOR OBSERVATIONS. 21
FIGURE 9. FIESTA ANALYTICS (A) CSV FORMATS AND (B) THE HTTP REQUEST FORMAT. 23
FIGURE 10. FIESTA-IOT ANALYTICS WEB SERVICE USING TRAFFICANALYSER EXAMPLE. 23
FIGURE 11. FIESTA ANALYTICS EXAMPLE (UPPER PANEL) THE AVERAGE VEHICLE SPEED AND (LOWER PANEL) THE VEHICLE COUNT TRAFFIC DATA. 25
FIGURE 12. FIESTA ANALYTICS EXAMPLE (UPPER PANEL) FIGURE SHOWING TRAFFIC LEVEL STATUS (LOWER PANEL) SHOWS THE TRAFFIC AVERAGE SPEED STATUS. 25
FIGURE 13. FIESTA ARCHITECTURE INCLUDING THE INTERACTIONS OF THE RELEVANT COMPONENTS WITH THE REASONING, ANALYTICS AND MONITORING TOOLS. 28

LIST OF TABLES

TABLE 1 TERMINOLOGY AND DEFINITIONS TABLE 7
TABLE 2 ADVANTAGES AND DISADVANTAGES OF ANALYTICS APPROACHES. 12
TABLE 3 AN ILLUSTRATION OF THE DATA ANALYSIS CATEGORY ALONG WITH THE CORRESPONDING EXAMPLES OF METHODS INCLUDED IN THE FIESTA-IOT ANALYTICS WEB SERVICE. 22

Copyright © 2016 FIESTA-IoT Consortium

TERMS AND ACRONYMS

IoT	Internet of Things
Jena	Framework to build semantic web applications
Jena TDB	Database to store semantic information called triple.
LSM	Linked Sensor Middleware
CQELS	Continuous Query Evaluation over Linked Streams
KAT	Knowledge Acquisition Toolkit
LD4Sensors	A tool to link sensor data
M3	Machine to Machine Measurement framework
GUI	Graphical User Interface
API	Application Programming Interface
KAT	Knowledge Acquisition Toolkit
LER	Linked Edit Rules
SDR	Semantic Data Repository
CSV	Comma Separated Variable
PCA	Principal Component Analysis
TPS	Testbed Provider Service
K-NN	K- Nearest Neighbours

Copyright © 2016 FIESTA-IoT Consortium

1 INTRODUCTION

1.1 Executive Summary

The FIESTA-IoT project seeks to create the relevant infrastructure for capturing and storing data from domains ranging from smart devices to smart cities. This data enables a wide range of data consumers to exploit such data sets for applications ranging from scientific experimentation to enhancing commercial activity for businesses. While the need for developing infrastructure capable of capturing and storing data sets from a number of differing domains is apparent, the natural question that arises is how to obtain value from such data sets? That is, what are the set of tools and techniques required by the experimenter, in order to transform raw unprocessed IoT data into valuable actionable knowledge? To this end, FIESTA-IoT seeks to provide a range of tools and techniques for the analysis and monitoring of data sets. In particular, this deliverable addresses tasks 3.4 and 3.5, where in particular task 3.4 addresses the need to develop both stream and historical data analytics tools for both reasoning and data analysis. While task 3.5 is concerned with the development of monitoring tools so as to enhance the performance of FIESTA platform being provided to the experimenter.

This document considers the development of tools for the implementation of both stream and historical data analytics for both reasoning and data analysis. In particular we propose to use two functional components that separately process stream and historical data. The first component is the FIESTA-IoT *Reasoning* component that aims to analyse both historical and stream data. This is achieved by linking and applying rules that are either: experimenter generated or rules that already exist in a semantic rule repository. While we also propose a data analysis tool as a web service for experimenters that we refer to as the FIESTA-IoT *Analytics* functional component. The proposed component enables the experimenter to apply sequentially, a set of data analysis techniques. These techniques range from pre-processing algorithms, both supervised and unsupervised machine learning methods and other tools and techniques that have more specific functionality.

Finally, this document provides an overview of the specifications for the tools required for task 3.5. In particular this document considers the need for implementing performance monitoring tools and data throughput control mechanisms. Such systems would enable the continuous monitoring and improvement of the services provided by the FIESTA platform.

1.2 Audience

This deliverable addresses following audiences:

- **Researchers and engineers within the FIESTA-IoT consortium** will take into account various requirements in order to interact with the proposed data analysis tools. Furthermore, this document will provide a set of requirements for monitoring the quality of IoT data generated by Testbeds.
- **Experiment owners who wish to join FIESTA-IoT** will understand the set of data analysis and reasoning tools currently being offered. This document

Copyright © 2016 FIESTA-IoT Consortium

provides a description and use case example in order to highlight the value of such tools to experimenters seeking to use the FIESTA-IoT platform.

- **The data analytics research community** will be interested in integrating their tools into the proposed web services. That is, the data analytics community would be enabled to provide the latest data analytics tools for experimenters to utilise.
- **Members of other Internet of Things (IoT) communities and projects (such as projects of the IERC cluster)** can take this document as an initial reference or inspiration to design and implement their own Testbed, considering the needs for the proposed analytics and monitoring tools.
- **Open call** participants will be able to understand better the technical details needed for them to join the FIESTA-IoT consortium.

1.3 Terminology and Definition

This sub-section intends to clarify the terminology used during this project. This initial step intends to clarify all of the important terms used, in order to minimize misunderstandings when referring to specific parts involved in the generation of data and the FIESTA-IoT concepts. The following definitions (listed in the Table below) were set regarding the domain area of FIESTA-IoT, and so are aligned with terminologies used in the Future Internet Research and Experimentation (FIRE) community and in reference to IoT-related projects (such as IoT-A).

Table 1 Terminology and Definitions table

Term	Definition
Device	Technical physical component (hardware) with communication capabilities to other Information Technology (IT) systems. A device can be attached to, or embedded inside a physical entity, or monitor a physical entity in its vicinity [1]. The device could be: 1. **Sensor**: A sensor is a special device that perceives certain characteristics of the real world and transfers them into a digital representation [2]. 2. **Actuator**: An actuator is a mechanical device for moving or controlling a mechanism or system. It takes energy, usually transported by air, electric current, or liquid, and converts that into some kind of motion [2].
Discovery	Discovery is a service to find unknown resources/entities/services based on a rough specification of the desired result. It may be utilized by a human or another service. Credentials for authorization are considered when executing the discovery [1].
Domain	Refers to an application area where the meaning of data corresponds to the same semantic context. For instance, pressure in Water Management Domain may refer to water pressure on pipes while in Air Quality Domain it refers to atmospheric pressure.
Information	Content of communication; data and metadata describing data. The material basis is raw data, which is processed into relevant information, including source information (e.g., analogue and state information) and derived information (e.g., statistical and historical information) [3].

Copyright © 2016 FIESTA-IoT Consortium

Measurement	The important data for the experimenter. It represents the minimum piece of information sent by a specific resource, which the experimenter needs in order to fulfil the objective of the experiment.
Metadata	The metadata is the additional information associated with the measurement, facilitating its understanding.
Testbed	A Testbed is an environment that allows experimentation and testing for research and development products. A Testbed provides a rigorous, transparent and replicable environment for experimentation and testing [4].
Experiment	Experiment is a test under controlled conditions that is made to demonstrate a known truth, examine the validity of a hypothesis, or determine the efficacy of something previously untried [5].
Semantic Interoperability	Semantic interoperability is the ability of computer systems to exchange data with unambiguous, shared meaning. Semantic interoperability is a requirement to enable machine computable logic, inference, knowledge discovery, and data federation between information systems.
Service	Services (Technology) are services designed to facilitate the use of technology by end users. These services provide specialized technology-oriented solutions by combining the processes/functions of software, hardware, networks, telecommunications and electronics.

2 RELATED WORK

In this section we present the related works corresponding to the reasoning, data analytics and monitoring tools.

2.1 Linking Data

We first present a set of tools dedicated for the linking of IoT data from differing domains.

2.1.1 LD4Sensors

The LD4Sensors/inContext-Sensing tool has been designed within the SPITFIRE EU project. This tool enriches sensor data with the Linked Data by using the Pachube API, the SPITFIRE ontology and the Silk tool to align datasets such as DBPedia, WordNet, Musicbrainz, DBLP, flickrwrappr and Geonames [6]. LD4Sensors provides JSON Web services, API and GUI to automate the annotation and linking process of sensor data. The semantic annotation is done with the Jena library and semantic data are stored using Jena TDB. LD4Sensors integrates a SPARQL endpoint to ease access to semantic sensor data. The semantic dataset and SPARQL endpoint are referenced on the dataset catalogue called DataHub. LD4Sensors provides linking but not reasoning and do not deal with real-time. The authors do not interpret the value of the produced sensor data by using domain-specific knowledge expertise.

2.1.2 Linked Sensor Middleware (LSM) and CQELS

Le-Phuoc et al. design the Linked Sensor Middleware (LSM) approach addressing at the same time querying, linking and real-time challenges. LSM is based on the

Copyright © 2016 FIESTA-IoT Consortium

Continuous Query Evaluation over Linked Streams (CQELS) and combine data the Linked Data [7] [8]. Le-Phuoc et al. developed the SensorMasher and LSM platforms to facilitate publishing 'Linked Stream Data' and making it available to other applications [9]. They develop a user friendly interface to manage environmental semantic sensor networks.

2.2 Reasoning on data

Logic reasoning or inference mechanisms on raw data are necessary to deduce high-level contextual information from low-level sensor data according to Mokhtar et al. [10]. This approach is frequently used in research fields such as context awareness and pervasive computing. From our point of view, we should unify the different approaches since they address different needs.

2.2.1 IntellegO

IntellegO is a semantic perception approach to interpret and reason on sensor data [11]. IntellegO uses an abductive logic framework and Parsimonious Covering Theory (PCT) to interpret data based on an “ontology of perception“. The development and reuse of the background knowledge (i.e., domain knowledge designed by domain experts for instance in healthcare) required for interpreting data is a difficult task and is not considered in this work.

2.2.2 Large Knowledge Collider

Large Knowledge Collider (LarKC) is a scalable distributed platform for web scale reasoning [12]. One of the use cases of LARKC is related to smart cities, more precisely parking lot assistance and predicts traffic congestion problems. They addressed the challenge to take only few milliseconds per query even by querying billions of triples that are continuously being updated.

2.3 Data Analytics Tools

An overview of the data analytics tools that have been developed both in academia and industry is now provided. It should be noted that, the tools stated in 2.3 require the user to either download the tool (as is the case with KAT and Weka 3.0) or require the user to understand a domain specific language such as MATLAB. Therefore it is imperative to provide a data analytics tool as a web service that enables a wide range of users to exploit the capabilities of such a system.

2.3.1 Knowledge Acquisition Toolkit (KAT)

The initial implementation of the Knowledge Acquisition Toolkit (KAT) [13] developed as a Python application (which the user has to download), where the tool provides a set of methods for labelling data sets after initial pre-processing and data abstraction. In particular the tool provides an algorithm workflow that is implemented sequentially: that is a pre-processing method, along with dimensionality reduction, (example: principal component analysis), data abstraction (examples include: clustering) and finally the labelling/annotation of the resulting abstracted data is carried out (an example is assigning for temperature data the labels, hot and cold.

Copyright © 2016 FIESTA-IoT Consortium

2.3.2 Weka 3.0

The work in [14] created a data analysis and machine learning environment implemented in Java, referred to as Weka 3.0. Where this tool is downloaded on the client's machine, where the functionalities include, pre-processing and a range of machine learning algorithms (that is, regression, clustering and supervised learning to name but a few). A drawback for both the KAT and Weka 3.0 tools is the requirement for the data consumer to download such tools, which limits the class of users due to the requirements of installing additional software over existing technologies (such as MATLAB and R). In this work we seek to provide a web service based data analysis platform that caters to a range of users from both novice to more knowledgeable experimenters.

2.3.3 ThingSpeak

ThingSpeak developed by Mathworks [15], provides a system that both collects and stores real-time data obtained from IoT sensors. After collecting and storing IoT data, the user/experimenter analyses/visualizes the data set using the MATLAB computing tool. ThingSpeak also incorporates tools for alerting and carrying out device actuation the user/experimenter as well as providing a scheduler for carrying out relevant tasks.

2.4 Monitoring IoT Data

An overview of existing monitoring tools with similar capabilities for the requirements specified in task 3.5 are now provided.

2.4.1 Graylog

Graylog[1] is a tool to have all the logs generated by different components in a central repository. It also allows system administrators to perform queries on the stored collected logs. Graylog offers web client where system administrators can monitor the performance of the system based on the collected logs. Graylog is built on an idea to discover and resolve issues faster. In order to do so it offers wide range of functionalities such as search on large-scale data, dashboard for quick visualizations of metrics, triggers and alerts and collector that enables easy configuration of the technology used to ship logs to Graylog. Beyond the above-mentioned functionalities, Graylog offers secured access to the logs as logs can hold critical information and REST APIs to access stored information. Graylog requires MongoDB[2], ElasticSearch[3] and Java8+. Its collector can be run on various platforms. Graylog is an open source tool. Its architecture follows support for cluster deployment[4].

With respect to FIESTA-IoT platform that has multiple components running on various technologies (Wildfly, OpenAM and MySQL), Graylog can provide a way to monitor the performance of FIESTA-IoT platform and look for error occurring if any.

[1] https://www.graylog.org

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

[3] https://www.elastic.co/products/elasticsearch

[4] http://docs.graylog.org/en/latest/pages/architecture.html

Copyright © 2016 FIESTA-IoT Consortium

2.4.2 Nagios

According to its legacy webpage[5], Nagios boasts of being the "Industry Standard in IT Infrastructure Monitoring". With a free and open source system core[6] (written in C and released under the GNU Generic Public License, Version 2), it offers a complete framework for monitoring (and alerting if things go wrong) network services (e.g. SNMP, SSH, HTTP, etc.), host resources (e.g. disk usage, system logs, etc.), and even hardware devices, like sensors. Amongst their commercial solutions, Nagios have split their services into three main products or categories:

- Nagios XI. Platform's main product. Its focus it to accomplish monitoring of critical components (e.g. operating systems, applications, network protocols, network infrastructure, etc.)
- Nagios Log Server. Focused on harvesting all the log monitoring and management, it concentrates in a very same place all this information, thus simplifying enormously the process of searching data.
- Nagios Network Analyzer. As its name hints, this latter service carries out a complete data analysis to help system admins have a clear control on the health of the network, by e.g. resolving connection outages, abnormal behaviours or security threats.

Compared to Graylog, Nagios offers a wider range of features, since it is not bounded to centralize the logging of all the components, but also provides another set of features that would allow FIESTA-IoT admins to monitor the whole platform, spanning from the system logging to the network analysis and yielding to an alarm triggering system in case any of the components are malfunctioning (e.g. a Testbed is down, an intruder is trying to enter the get access, etc.).

2.4.3 Fed4FIRE monitor

To realize a federation-wide facility monitoring service, at both the Testbed infrastructure and at the federation levels, within Fed4FIRE all Testbeds implemented support for a facility monitoring system, which is used to steer a First Level Support (FLS) dashboard[7]. The monitoring comprises several functionalities that are in charge of realizing specific measurements. In total, six services are monitored:

1. ICMP ping to the SFA Aggregate Manager (AM) server or some other Testbed server: this checks connectivity over the internet to the Testbed. If this fails, Testbed can likely not be used.
2. AM API GetVersion call: tests the AM component (no credential is needed)
3. AM API Listresources to know the number of free resources. If this is 0, then new experiments cannot be created.
4. Red/Green/Amber internal status of what the Testbed provider monitors himself. This is custom per Testbed, and is based on the Testbed's facility monitoring data.
5. Aggregated status: this aggregates all tests, averages (to avoid false alarms!) and sends emails to the FLS ticket system whenever there is a new alarm or when an

[5] https://www.nagios.org/

[6] https://github.com/NagiosEnterprises/nagioscore

[7] https://flsmonitor.fed4fire.eu.

Copyright © 2016 FIESTA-IoT Consortium

alarm is solved. This creates automatically in the OTRS system, and notifies as such the Testbed expert, which can comment on the ticket.
6. Login status: result of the last SSH login test per Testbed.

2.5 Limitations of existing work

In this section we highlight the limitations of existing technologies (an overview is provided in Table 2), where in particular we seek to propose a set of tools that is both compliant with tasks 3.4 and 3.5. The limitations of existing methods are as follows:

1. Existing reasoning methods do not provide a means for generating and storing rules in a semantic repository such that experimenters can both query and access more effectively.
2. Existing data analysis methods require the user to either download software onto the experimenter's machine, or requires knowledge of specific programming languages.
3. The requirements for a monitoring system of IoT data streams has not been defined and widely accepted.

The main challenges to address are:

1. The development of both reasoning and data analysis tools as a web service, thus enabling a wider range of users to benefit from such added value tools.
2. Thereby providing the means for providing the most effective reasoning and data analysis algorithms for a wide range of users.
3. Provide relevant documentation and tutorials for novice experimenters.
4. Provide the requirements for monitoring IoT data streams, in particular to highlight the relevant capabilities for FIESTA-IoT.

Table 2 Advantages and disadvantages of analytics approaches.

Methods	Pros	Cons
Logic/rule-based reasoning	• Simple rules • Adapted to simple sensors • Easy for beginners (learning & implementation) • Easier to combine rules	• Not adapted to complicated sensors • Heterogeneous rule languages and editors
Data Analysis/Machine Learning	• More elaborate results • Adapted to complicated sensors	• Complicated for non-experts • Complicated for a 'sharing and reusing' approach
Monitoring Tools	• Can provide updates of sensor availability • Would provide a means of improving FIESTA-IoT data collection process	• Specification for such systems for monitoring data quality not well defined

Copyright © 2016 FIESTA-IoT Consortium

3 THE SEMANTIC DATA WORKFLOW TO ANALYSE IOT DATA

The FIESTA platform seeks to design the following workflow, that is, data collected by Testbed providers is first stored using a semantic description of the data. Thus enabling more effective querying of the data, such that both reasoning and data analytics can then be carried out by the experimenter (as depicted in Figure 1). By providing reasoning and data analytics capabilities we seek to add value to the data generated by Testbed providers, such that meaningful and actionable information can be extracted. The proposed analytics tools provided by FIESTA seek to both infer information from the sensor generated data by using reasoning methods, while also providing the necessary data analysis tools to carry out both inference and/or prediction on the data.

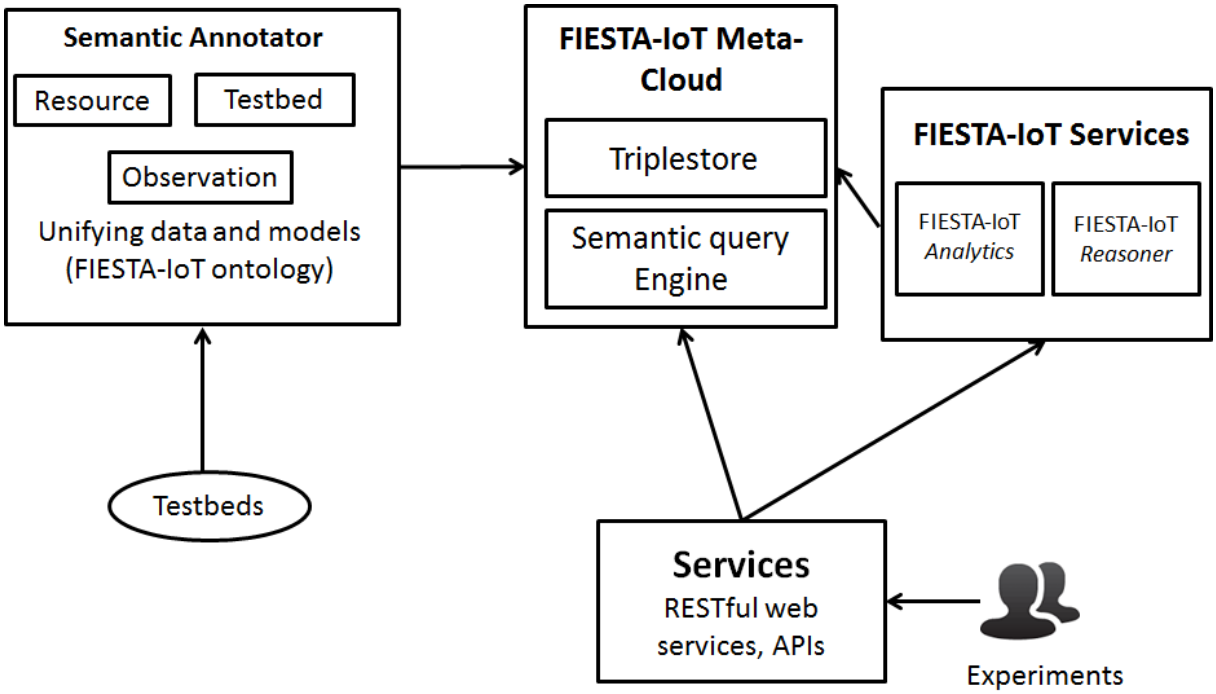


Figure 1. The semantic IoT data workflow within FIESTA

The semantic data workflow comprises the following components:

1. **Semantic Annotator** components enable semantic annotation of IoT data and models generated by heterogeneous Testbeds. IoT data can be raw data (e.g., provided by SmartSantander) or already semantically annotated (e.g., Keti Testbed provides data already semantically annotated with the OneM2M ontology). The M3-lite taxonomy explained in Deliverable D3.1.1 enables unifying IoT data from different Testbeds, application domains and delete any ambiguities regarding a same kind of measurement (e.g., body temperature and room temperature). The Semantic annotators can include wrappers to support different formats of IoT data, in priority, those supported by FIESTA-IoT Testbeds. The semantic annotator enables annotating resource descriptions and observations generated by resources.

Copyright © 2016 FIESTA-IoT Consortium

2. **Semantic query engine** enables the querying of data sets in a Testbed agnostic manner. Furthermore, the querying of processed data can also be carried out, owing to the semantic re-annotation and storage of processed Testbed data.
3. **FIESTA-IoT Reasoning and Data Analytics** components enable both the enriching data with additional information along with the tools required for the extraction of information. The reasoning tool deployed in FIESTA is based on the Jena inference engine, while the data analytics is carried out using a python implemented web service implementation and modification of the KAT tool.
4. **Services** provides access to the experiment to the different components provided by FIESTA-IoT.

4 FIESTA-IOT REASONER

In this chapter we provide an overview of the proposed FIESTA-IoT *Reasoner* component. The component is a rule based inference engine, where the rules are generated and stored in a semantic rule repository (that is a data store for a set of semantically labelled rules that enables interoperability).

4.1 Rule Based Reasoning

Rule based reasoning seeks to infer information based on a set of rules that may vary in complexity. That is, one may have a simple set of if-then-else rules; where a simple example consists of identifying a human interpretable meaning to temperature data, i.e. if the weather is “hot” or “cold”, given the raw temperature data obtained from a sensor. Moreover, more complex set of rules seek to utilize multiple data streams (from potentially multiple domains) so as to infer more complex information. For example, identifying humidity requires multiple data sources such as wind speed, air pressure, precipitation levels etc., where a set of rules are then defined so as to infer if the weather given a location is humid or not.

4.2 FIESTA-IoT *Reasoner*-web service

In this work we seek to develop the FIESTA-IoT *Reasoner* such that the relevant users have the capability of generating and storing rules on the FIESTA-IoT platform. Before introducing the proposed component, we first distinguish two kind of FIESTA-IoT clients:

1. **The client is a Knowledge Producer**: as explained in the architecture document [16] KPs aim at leveraging raw data and inferring new knowledge out of it. If the KP decides to use the FIESTA-IoT *Reasoner* (it could also decide indeed to use other enablers like analytics and ML), it has first to create a correct and complete set of rules and to store it in the Semantic Rule Repository. The second thing it has to deal with is to determine the data set it wants to apply the set of rules on. When both data set (characterised by a SPARQL sentence) and rules (RuleSetID) are known, the reasoner can be invoked as depicted in the sequence diagram below. In KP case, new inferred data can be either directly stored in the FIESTA meta-cloud data endpoint or alternatively returned to the KP for examination, and then uploaded to the data endpoint by the KP itself.

Copyright © 2016 FIESTA-IoT Consortium

2. **The client is an Experimenter** (actually the Experiment Execution Engine): as explained in [16] Experimenters aim at accessing data, carrying out experiments using different techniques (in our case the FIESTA-IoT *Reasoner*) and exploiting the resulting data locally, meaning the experimenter is not expected to store any of the data inferred by the reasoner into the FIESTA-IoT meta-cloud data endpoint, in order to avoid the data storage to be polluted with an uncontrolled amount of data. It could however decide to publish the data to the message bus so that it could be reused by other experiments. The preliminary phase therefore consists of the following steps that can be achieved in any order:
    1. Either browsing the existing set of rules from the Semantic rule repository and selecting one that fits the needs or creating a set of rules and passing it during invocation. However this set of rules would not be stored within the Semantic Rule repository unless it passes elementary tests concerning correctness and completeness, so that it can be reasonably trusted by the FIESTA-IoT community.
    2. Browsing data, selecting data of interest for the experiment and writing a SPARQL request that can be used to load the data set.

Moreover, the rules are implemented as Jena rules since we used the Jena framework to build semantic web applications. Moreover, Jena provides an inference engine to easily execute the Jena rules and deduce additional information. Figure 2 depicts the underlying components of the FIESTA-IoT *Reasoner* below comprises several components:

- **FIESTA-IoT Built-in reasoner** which runs the Jena inference engine.
- **Semantic Rule repository**, a database storing Jena-compliant IF THEN ELSE rules. This repository can be updated by new rules thanks to the rule editor. The rules should be compliant with the M3-lite taxonomy and FIESTA-IoT ontology.
- **Semantic Data repository** stores datasets semantically annotated and compliant with the FIESTA-IoT ontology.
- M3-lite taxonomy is the dictionary used to unify sensor data in the SDR.
- **Query engine** is implemented as a SPARQL query engine, more precisely the Jena ARQ query engine to be fully compatible with other components. The query engine executes SPARQL queries compliant with FIESTA-IoT ontology and M3-lite taxonomy,

  M3 interoperable domain knowledge is a database of ontologies and datasets relevant to build smart experiments. The rules available within the **Semantic Rule Repository are "IF THEN" rules. The THEN part enables to interlink sensor data with external knowledge bases to deduce additional information.**

Copyright © 2016 FIESTA-IoT Consortium

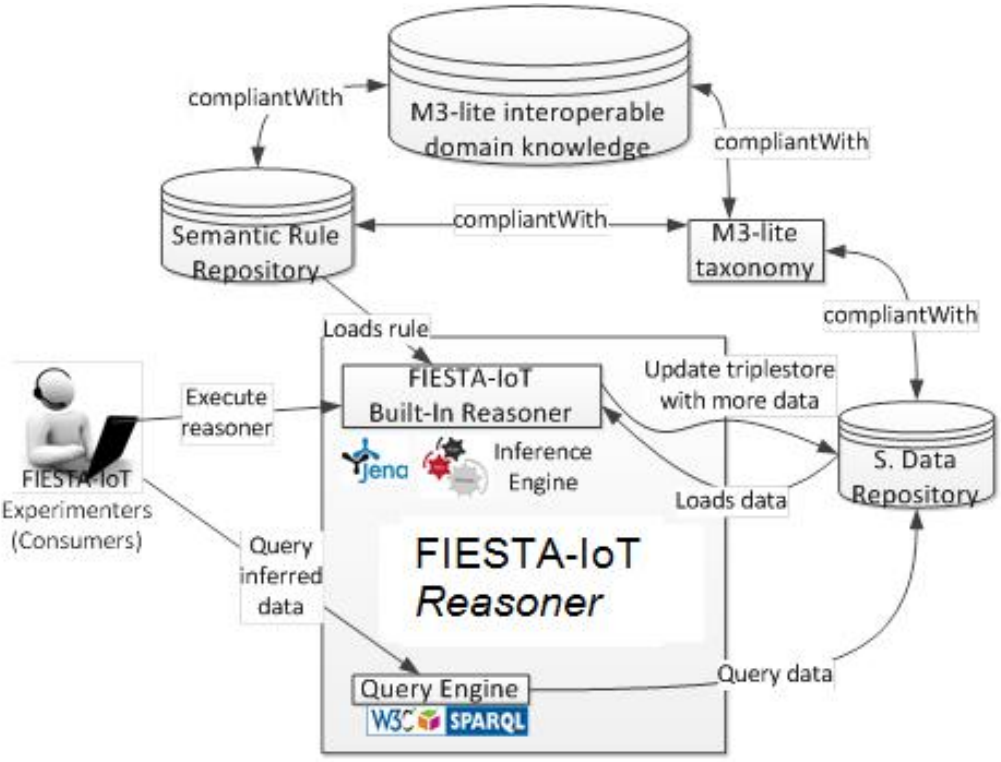


Figure 2. Experimenters produce knowledge within the FIESTA-IoT platform

5 FIESTA-IOT ANALYTICS

This section describes the FIESTA-IoT *Analytics* web service that is derived from the extension of the KAT software.

5.1 Data Analysis

It is imperative to provide data analysis tools so as to enable the experimenter to extract meaningful information from the data obtained from the FIESTA data repository. Examples of data analysis tools that provide an added value to the experimenter may range from, simple filtering algorithms (so to remove unwanted noise components), to more sophisticated supervised machine learning algorithms such as deep learning (so as to identify patterns and relations in the data sets being analysed).

We will now provide a detailed explanation of examples of data analysis techniques that we have initially identified as adding value to the experimenter. Generally data analysis can be split into two categories, the pre-processing stage followed by a learning stage (such that either inference, that is understanding the relationship between input and output variables, or prediction can be carried out). The pre-processing stage is used for the cleaning and removal of unwanted signal components (generally referred to as noise); where the following tools such as digital filtering, outlier removal and missing data processing are generally applied at this stage. While learning stage can be split be split into two sub-categories: namely supervised and unsupervised learning algorithms.

Copyright © 2016 FIESTA-IoT Consortium

5.1.1 Data Pre-Processing Techniques

The following pre-processing methods were considered initially to be included in the FIESTA-IoT Analytics tool, namely:

1. Digital Filtering – A finite impulse response (FIR) filter, for lowpass, bandpass and highpass filtering. That is, the removal of low frequencies (lowpass filtering), between a range of frequencies (bandpass filtering) and the removal of high frequency components (highpass filtering). This method is important for the removal of unwanted signal components that corrupt the desired signal. It should be noted that the user would need to define the type of filtering required (that is lowpass, bandpass etc.) as well as providing the frequency ranges they require to remove. Please see Figure 3, for an illustrative example. Figure 3 (a) Illustrates the desired signal (left) and unwanted noise signal (right). (b) Shows the addition of the desired and unwanted signal shown in the upper panel. (c) Shows the application of lowpass FIR filter on the noisy signal shown in the middle panel.
2. Outlier Removal – Many machine-learning algorithms are sensitive to outliers. That is the performance of the methods degrades as the number of outlier's increases. Accordingly, we provide an outlier tool that is based on winsorization [17]. The user defines only one parameter that is the percentage of from the highest and lowest value in the data set is clipped (an example is shown in Figure 4). At Figure 4 at (Upper panel) Original signal with outliers. (Lower panel) The output of the original signal with outliers after applying winsorization

It should be noted that, the set of pre-processing method will not be limited to the above methods. For the initial implementation of the FIESTA-IoT Analytics it was found that the above pre-processing methods were required for the initial cleaning and conditioning of data for employing machine learning methods.

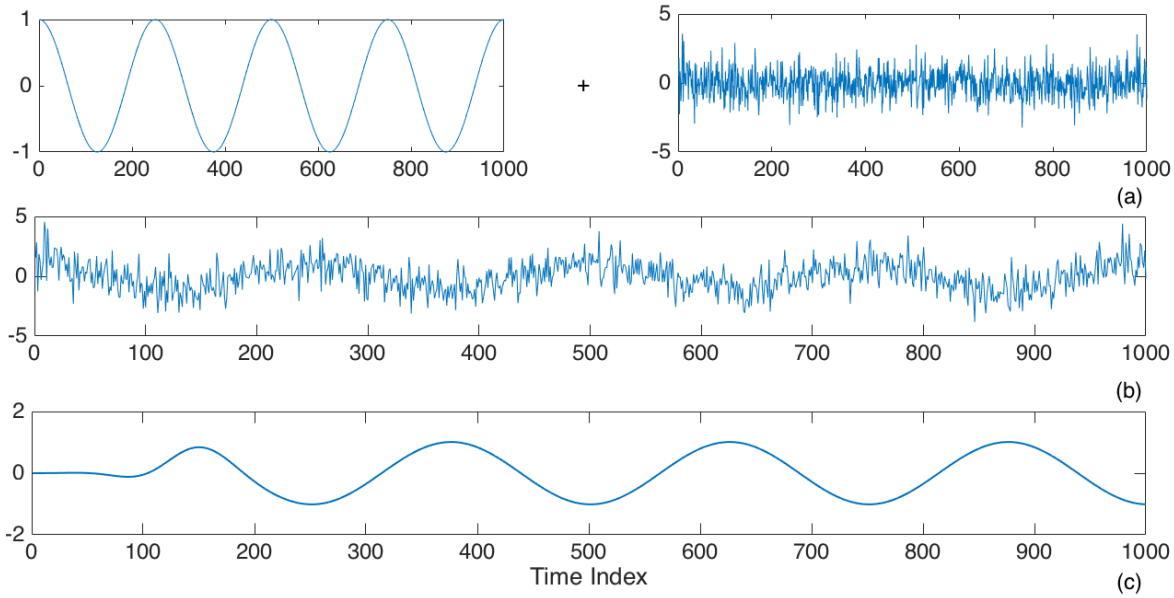


Figure 3. FIESTA-IoT Analytics: Digital Filtering Example.

Copyright © 2016 FIESTA-IoT Consortium

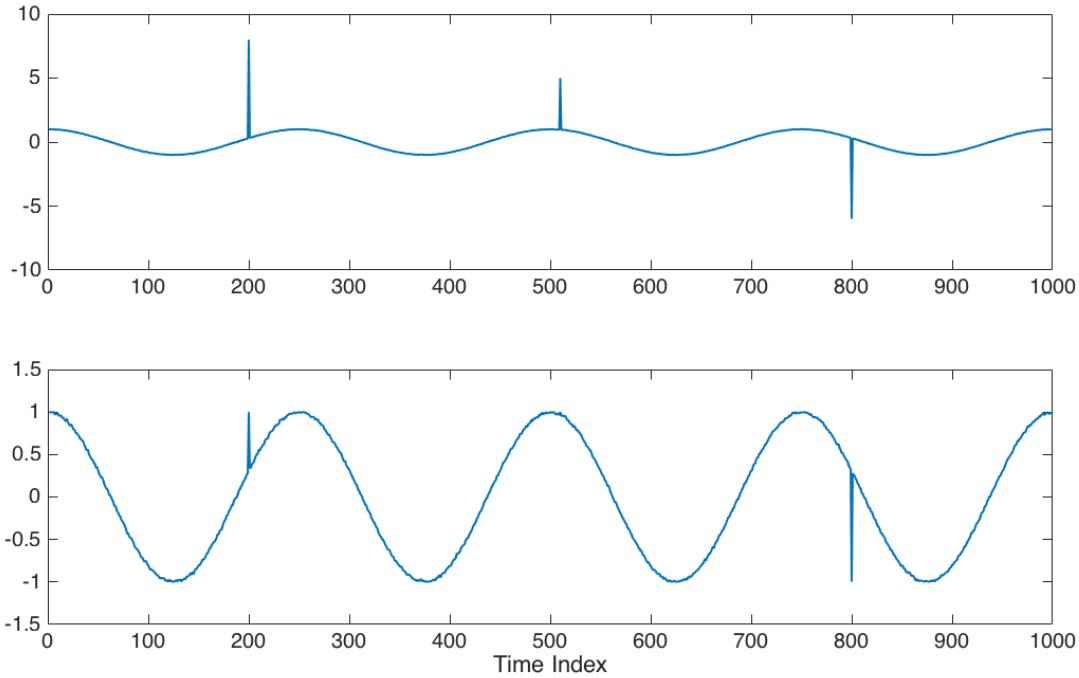


Figure 4. FIESTA-IoT Analytics: Outlier Removal Example

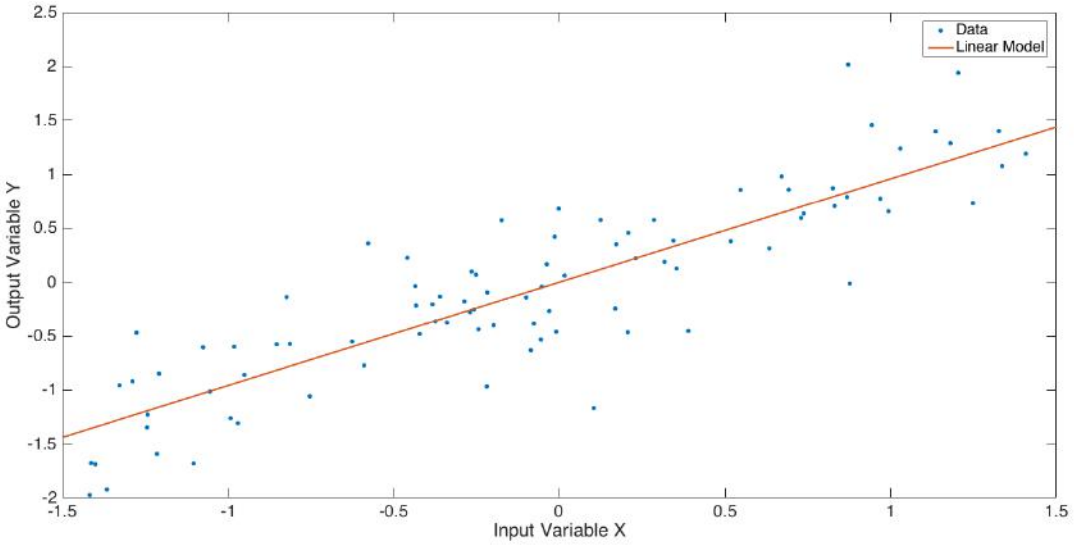
5.1.2 Machine Learning Techniques

As mentioned at the start of this chapter, machine learning techniques are utilized for two general problems: *inference* and *prediction*. Furthermore, machine learning methods can be divided into two more categories, namely supervised and unsupervised learning.

Supervised learning seeks to identify a functional relationship between the data when an input-output relationship is required by the experimenter for the data set being analysed. For example, if one considers the output set of data points Y that may correspond for example to data obtained from a sensor measuring air pollution. While the input variable X may be the number of cars. One can then find a functional (either linear or nonlinear) relationship between the output Y with the input X. Such a problem is generally referred to as supervised learning, where training is first carried out in order to identify the parameters of the functional form specified by the experimenter. Such that either inference or prediction can then be carried out. Examples of supervised learning algorithms that will be initially included in the FIESTA-IoT Analytics platform include the following (James, Witten, Hastie, & Tibshirani, 2015).

At Figure 5 Linear regression model (red line) fit to a data set (blue dots). It should be noted that, the input-output relationship has a linear relationship that is captured effectively using a linear model

Copyright © 2016 FIESTA-IoT Consortium



.

Figure 5. Linear regression model (red line) fit to a data set (blue dots).

1. Linear Regression – This supervised learning method seeks to find a linear functional relationship between the output Y and the set of input variables X_p (where the subscript *p* corresponds to the variable index). An advantage of linear regression is the relatively simple interpretability of model while maintaining reasonable prediction performance. For example consider the following example: that is we wish to generate a linear model between the input and output, $\mathrm{Y} = a\mathrm{X}$, where a is the linear parameter that relates the input X to the output Y. As a result, the following inference (interpretation) can be made, if the input variables changes by, $\Delta\mathrm{X}$ then the output will change by $a\Delta\mathrm{X}$. A simple illustration of a linear regression model is shown in Figure 5.
2. K-Nearest Neighbors (K-NN) regression – While linear regression may provide an interpretable relationship between the input and output variables. The performance of the method for predicting the output given the input data points may degrade. This may arise owing to a non-linear relationship with respect to the model parameters (it should be noted that, transformations of the input variables themselves can be carried out in order to carry out linear regression). Given the input data ($\mathrm{X}_{train,}\mathrm{Y}_{train}$), we seek to estimate the output $\widehat{\mathrm{Y}}_{test}$, given the test data X_{test} (which is a subset of the $\mathrm{X}_{train,}$), by averaging the K output training samples using the corresponding K nearest (using the Euclidean distance) input training data X_{train} points to X_{test} . An illustrative example of K-NN is provided in Figure 6.

At Figure 6 A non-linear relationship between the input-output variables. Estimation using linear regression (red line) does not effectively capture the relationship between the input and output. However, it should be noted that the K-NN (yellow line) is able to better fit the data

Copyright © 2016 FIESTA-IoT Consortium

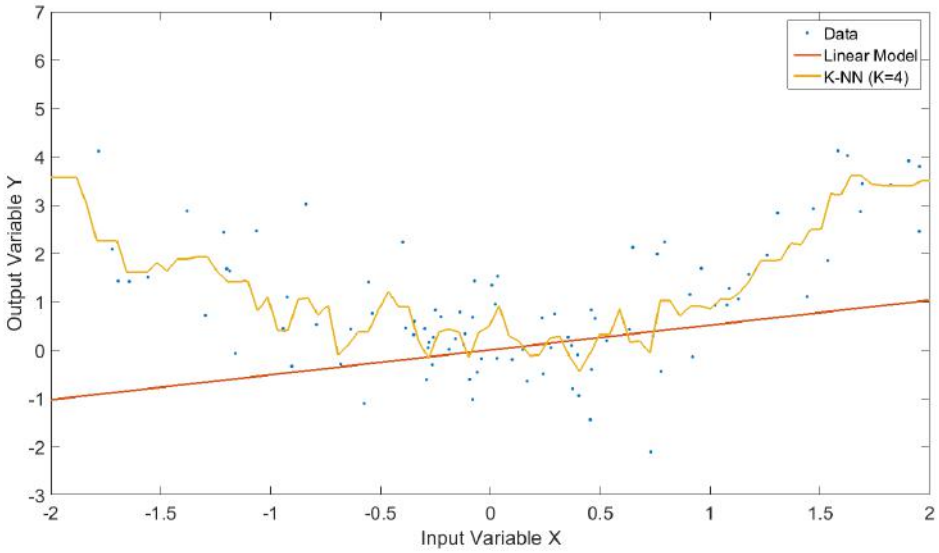


Figure 6. A non-linear relationship between the input-output variables.

Unsupervised learning seeks to identify structures and patterns in the data set, where an explicit input-output relationship is not known (or required). In such problems where only the input data is only available with no explicit output, the challenge is to identify groups or cluster the data in order to understand the relationship between the variables (this is often carried out in exploratory data analysis). To this end, we have also initially included the following unsupervised learning algorithms [18]:

1. K-Means Clustering - This algorithms seeks to identify clusters or subgroups of the data points being analysed. Clustering is often performed as part of exploratory analysis of data sets, where the experimenter seeks to identify group structure within the data. An example of clustering analysis given two sensors is shown in Figure 7, where each point corresponds to the observation time index. The output of the two sensors can therefore be assigned to one of the two clusters shown in Figure 7, thus providing unsupervised classification of data. K-means clustering seeks to identify a set of K non-overlapping clusters, where each data point is assigned to one of the K clusters. The algorithm achieves this by iteratively estimating the clusters such that the total distance between each point within the K clusters is minimized.
2. Principal Component Analysis (PCA) – Given a large number of sensors, it is often not practical or possible (for very large number of sensors) to visualize the structure between the data. Accordingly methods such as PCA seek to identify a lower dimensional representation of the data that captures the most significant variability of the data. For example, consider Figure 8 where there are two sensors such that the data has a large variation along one direction (this is illustrated by the red arrow). By projecting the original data of the two sensors along this direction of highest variation (this is referred to the as the principal component scores), we can obtain a lower dimensional representation of the data thus enabling more effective analysis of the original data set.

Figure 7 Scatter plot of two sensor observations. The data exhibits clustering (first cluster shown by crosses and second cluster shown by squares) which can be extracted using clustering algorithms.

Copyright © 2016 FIESTA-IoT Consortium

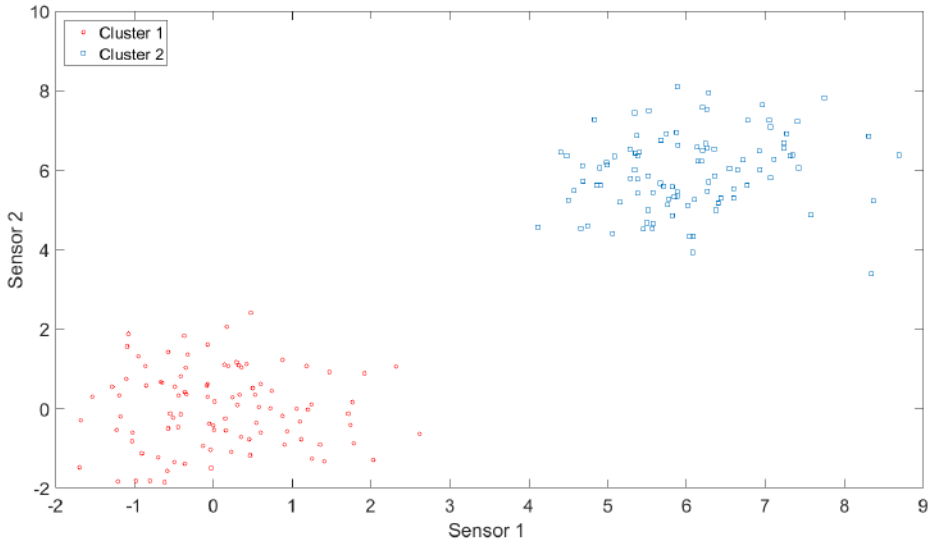


Figure 7. Scatter plot of two sensor observations.

At Figure 8, scatter plot of two sensor observations. The data exhibits clustering (first cluster shown by crosses and second cluster shown by squares) which can be extracted using clustering algorithms

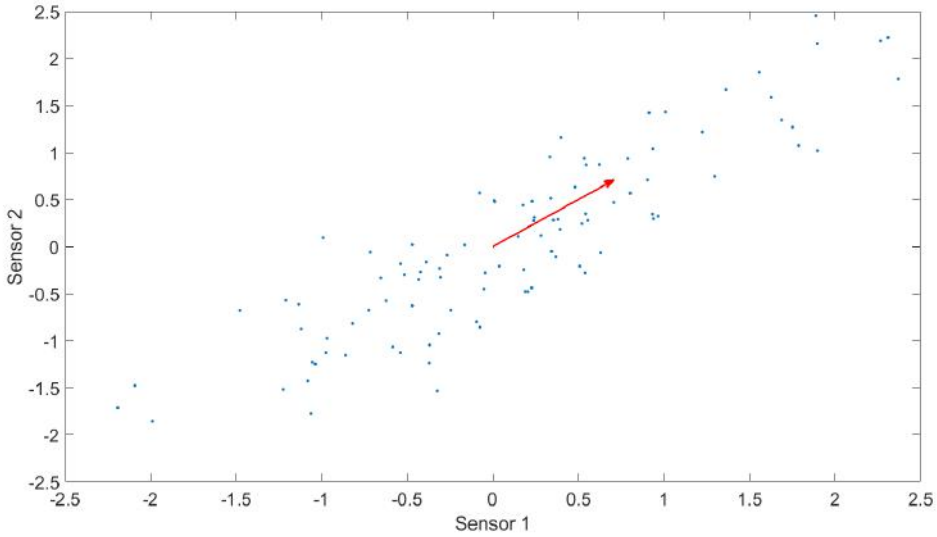


Figure 8. Scatter plot of two sensor observations.

5.2 FIESTA-IoT Analytics - Data Analysis Web Service

Existing data analysis tools offer such systems as software that requires the user/data consumer to download the product. However, such systems may require the user to learn a programming language (such as Java and Python) for the use of such systems, while also requiring software updates so as to access the latest data analysis techniques.

As a result, it is imperative to provide open access data analysis tools for consumers as a web service, which provide the following benefits: namely for novice/beginner data consumer, the tools that would enable them to analyse and obtain useful

Copyright © 2016 FIESTA-IoT Consortium

information. While for the more advanced/experienced user the FIESTA-IoT *Analytics* component seeks to provide the most effective data analysis tools for a given data set (we generally refer to a data set as information being obtained from IoT sensors).

For example, such a tool would provide relevant documentation for the beginner data consumer. With examples of data processing work flows, while for the more experienced user the most up-to-date tools developed in academic institutions would be provided. Thus a wide range of users can evaluate the tools and thus identify the most useful data analysis tool for a given data set.

To this end, this work presents a web service (known as the FIESTA-IoT *Analytics* web service) for data consumers that provides access to range of tools and techniques for processing time series data. The proposed FIESTA-IoT *Analytics* service is a Python implemented system, where a data experimenter/consumer would send an HTTP request of the relevant algorithms and corresponding parameters they seek to have executed on a data set.

Pre-Processing	**1.** Outlier Removal **2.** Digital Filtering
Supervised Learning	**1.** Linear Regression **2.** K-Nearest Neighbor
Unsupervised Learning	**1.** Principal Component Analysis **2.** K-means clustering
Other	**1.** Correlation Coefficient **2.** Spectral Estimation

Table 3 An illustration of the data analysis category along with the corresponding examples of methods included in the FIESTA-IoT Analytics web service.

In particular the data analysis tools are grouped into the following categories, namely: 1) pre-processing methods, 2) supervised learning, 3) unsupervised learning algorithms and finally 4) other techniques and methods (examples of techniques belonging to this class are spectral and dependence estimation methods) [19] [20] [18]. Table 3 presents examples of the relevant algorithms for each group, where users will find relevant information for each method via online documentation. Furthermore, the proposed data analysis tool as a web service would enable more rapid updates of the list of algorithms (as algorithms proposed by new research groups will also be included). We now provide an overview of the data format and REST POST request in order to utilize the FIESTA-IoT *Analytics* service:

1. Create a comma separated value (CSV) file with the following format[8]. Where the sensor columns alternate be- tween data and time-stamps (that is the following column header: sensor 1 data, time-stamp sensor 1, sensor 2 data, time-stamp sensor 2 data, etc.), an example of which is shown in Figure 9 (a). Furthermore, the CSV file needs to be available as web link, such that the FIESTA-IoT Analytics service can discover the data.

[8] In future iterations of the proposed tool, we need to consider how and where the SPARQL query needs to be executed in order to retrieve data sets in the CSV format required.

Copyright © 2016 FIESTA-IoT Consortium

2. Send an HTTP request to the FIESTA-IoT *Analytics*, where an example of the exact format for making the request is shown in Figure 9 (b). Where the list of methods and corresponding parameters are included (online documentation would help the user determine the number of parameters required for each method). Furthermore, an HTTP link to the data source also needs to be provided. At Figure 9 (a) CSV format for both the sensor observations and time stamps. (b) The HTTP request format, for the relevant data analysis tools.

Sensor 1	Time-Stamp 1	Sensor 2	Time-Stamp 2	...
1.43	2014-02-13T11:30:00	0.02	2014-02-13T11:40:00	
1.21	2014-02-13T11:35:00	-0.4	2014-02-13T11:50:00	

(a)

```
{
 "Method": ["method 1","method 2","method 3"],
 "Parameters":["parameter 1","parameter 2","parameter 3"],
 "DataPointer":["http://link to CSV data"]
}
```

(b)

Figure 9. FIESTA Analytics (a) CSV formats and (b) The HTTP request format.

At Figure 10, an example of the input to the FIESTA-IoT Analytics web service using the TrafficAnalyser function. There are 5 input parameters. The first parameters corresponds to the variable $\boldsymbol{l}$, the second to the distribution (in this case “P” corresponds to Poisson) and the last three parameters correspond to the labels a,b,c.

```
{
 "Methods": ["TrafficAnalyser"],
 "Parameters":["3,P,0,1,2"],
 "DataPointer":["http://link to CSV data"]
}
```

Figure 10. FIESTA-IoT Analytics web service using TrafficAnalyser example.

It should be noted that, the CSV format was selected so as to readily identify columns pertaining to the time-stamp of the corresponding output observations, such that re-sampling of the observations to common sampling frequency is achieved (we align the sampling frequency to the sensor with the lowest sampling frequency). Furthermore, the proposed FIESTA-IoT *Analytics* service requires the data consumer to select the correct sequence of methods. For example, one cannot execute the Fourier transform on a dataset followed by the execution of a pre-processing algorithm (an example being bandpass filtering). To this end, we provide an error message when such incorrect sequences of data analysis methods are selected and accordingly refer the user to the documentation. Finally, the output of the processed time series data is returned to the user as a CSV file.

Copyright © 2016 FIESTA-IoT Consortium

5.3 Use Case – Traffic Data Annotation

In this particular use case example, we provide a data analysis tool (referred to as: TrafficAnalyser) that obtains the relevant statistics such that labelling/annotation of both the average vehicle speed and vehicle count data[9] can be assigned, thereby providing human interpretable information from traffic sensor data [21]. That is, we seek to identify a confidence interval for the average of the measurements $x_d(t)$, where t corresponds to the time in hours and minutes over one day, and d corresponds to the set of dates. Accordingly, we define the confidence interval around $x_d(t)$ as follows

$$\mu(t) - l\sigma(t) \leq x_d(t) \leq \mu(t) + l\sigma(t)$$

where $\mu(t)$ is the sample mean

$$\mu(t) = \frac{1}{D}\sum_{d} x_d(t)$$

and $\sigma(t)$ is the sample standard deviation, such that depending on the underlying statistical distribution assumed for the data (that is for vehicle count we assume a Poisson distributed data set, while for the average vehicle speed we assume a Normal distribution),

$$\sigma^2(t) = \begin{cases} \frac{1}{D}\sum_{d} x_d(t), & \text{if } x_d(t) \text{ is Poisson Dist.} \\ \frac{1}{D}\sum_{d} (x_d(t) - \mu(t))^2, & \text{if } x_d(t) \text{ is Normal Dist.} \end{cases}$$

where D corresponds to the number of dates and the user defined parameter l corresponds to the significance level of the Normal test. Furthermore, the user then defines three input labels $\{a, b, c\}$, such that annotation, $x_d^{sem}(t)$ of the data set $x_d(t)$ can then be carried out, that is

$$x_d^{sem}(t) = \begin{cases} a, & \text{if } x_d(t) - \mu(t) \leq -l\sigma(t) \\ b, & \text{if } |x_d(t) - \mu(t)| < l\sigma(t) \\ \text{c}, & \text{if } x_d(t) - \mu(t) \geq l\sigma(t) \end{cases}$$

As an example for the vehicle count data, the following labels can be selected: $\{a = \text{"BelowAverage"}, b = \text{"Normal"}, c = \text{"}AboveAverage\text{"}\}$**, thereby providing a human interpretable meaning to the numerical vehicle count data. Finally, of the input parameters and methods selected for the traffic count data set is in Figure 10. Finally, Figure 11 illustrates the numerical representation for both the average vehicle speed (upper panel) and vehicle count (lower panel). While** Figure 12 presents the corresponding output of the TrafficAnalyser method stated above, where it can be observed that such a representation can be interpreted more effectively, than the original data set.

[9] Traffic data obtained from: http://iot.ee.surrey.ac.uk:8080/datasets/traffic/traffic_feb_june/index.html

Copyright © 2016 FIESTA-IoT Consortium

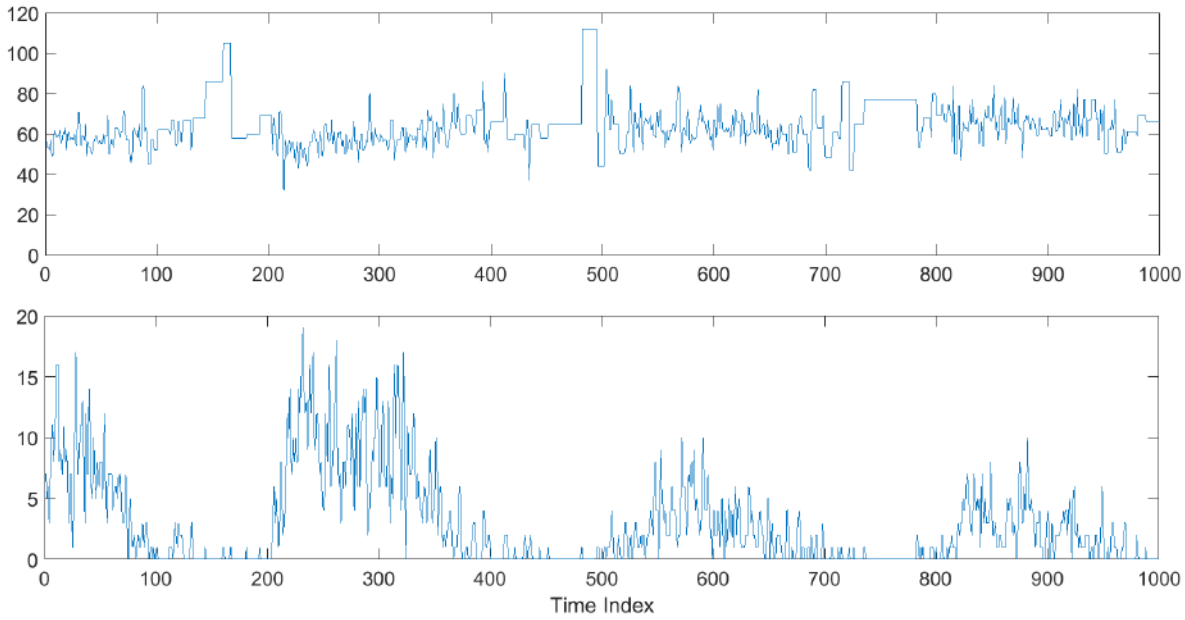


Figure 11. FIESTA Analytics Example (Upper panel) The average vehicle speed and (lower panel) the vehicle count traffic data.

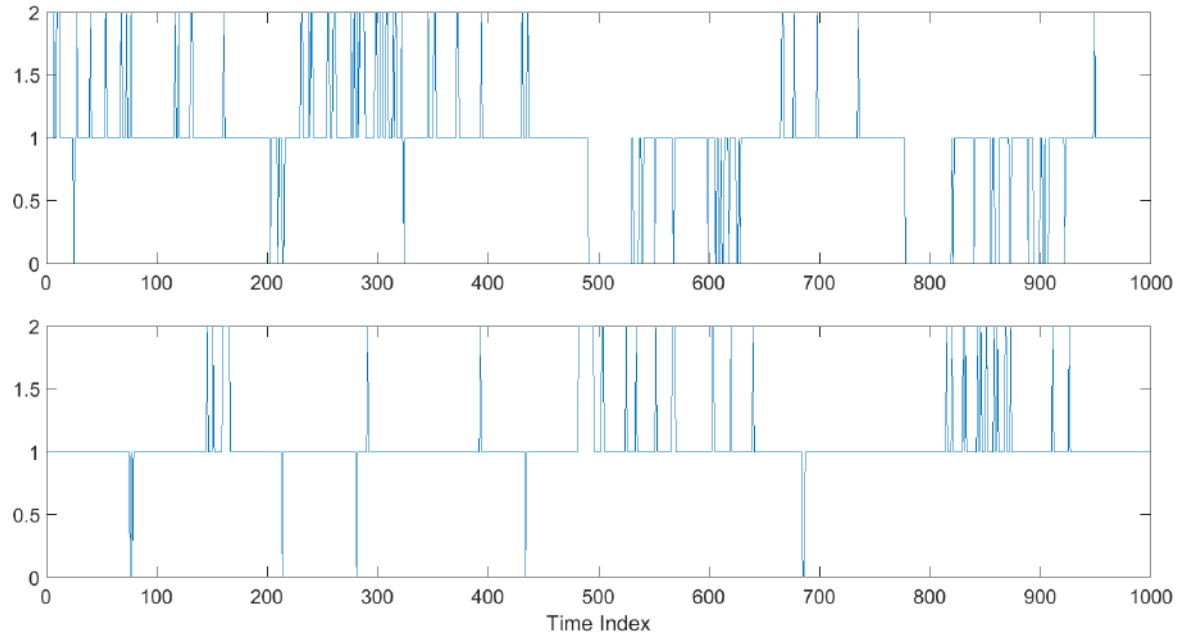


Figure 12. FIESTA Analytics Example (Upper panel) Figure showing traffic level status (lower panel) shows the traffic average speed status.

Copyright © 2016 FIESTA-IoT Consortium

6 FIESTA-IOT MONITORING

The task objective describes the management of IoT Streams at the Resource level. But it is expected that Testbed providers by default would not handover the management of their Resources to an external system entity, which would involve functions such as changing the frequency of sensor sampling for a Resource, or schedules of unavailability for energy conservation. Also, giving control to another entity would constitute the reservation of a Resource(s), which can affect the experiments of other FIESTA-IoT users, in terms of availability. Hence management of IoT data streams from the perspective of the FIESTA-IoT platform should be handled at the southbound interface between the Testbed data endpoint and the FIESTA-IoT Data Message bus. When it comes to monitoring, this should involve:

1. The verification of reachability of the exposed IoT Service/Data endpoints declared by a Testbed.
2. The quality of the data in terms of:
    1. values within correct range
    2. annotations with correct property values.

When a Testbed registers its Resources with the FIESTA-IoT Registry, it should indicate a level of service that it can provide. These can be translated into parameters such as:

1. Number of Resources
2. Uptime/availability of Resources
3. Maximum frequency / data collection rate / throughput
4. Maximum Freshness of data
5. Jitter
6. Availability (% of time)
7. data collected (local storage)
8. Availability

6.1 Reachability

The FIESTA-IoT monitoring component can check that the Testbed and/or their Resources can be reached by the endpoint URLs that have been declared upon registration. This can involve basic PING tests using ICMP (if supported by Testbeds) or HTTP GET requests. This will only check if the endpoint is able to respond.

The next step can be to test the Testbed Provider Service (TPS) interface exposed by the Testbeds. This involve polling for data by retrieving the last reading, or retrieving a set of readings between a specified time interval. If the Testbed supports a push mechanism, then the monitoring component can subscribe to it through the message bus.

6.2 Annotations

Once IoT registry endpoints are checked for reachability, the body of the response should be checked. This would involve the validation of the format of the body declared in the response header, and the structure of the body.

Copyright © 2016 FIESTA-IoT Consortium

An auditing feature can make use of the annotation validation service to check if the data has been annotated correctly. This includes checking quantity kinds and units are consistent.

6.3 Data and Datasets

The data itself should also be checked. This would involve checking if the value is not null. The value can then be checked if it is within a valid range based on the domain of interest that the Resource is a part of.

6.4 Controlling Throughput

For the FIESTA-IoT *Analytics* tool, any change in sampling frequency during time is not a desirable property (as re-sampling of data streams needs to be carried out using a constant sampling frequency). However, potential solutions for providing a means for controlling throughput, utilise the DMS. As the DMS can regulate the throughput of data towards the Metacloud data repository, by a factor of the minimum sampling frequency declared by a Testbed for a Resource. This can be based on:

1. the global flow of data to the FIESTA-IoT platform
2. usage by experimenters
3. the rate of change in data values

If the minimum sampling rate is not enough for an experimenter, a report can be generated to inform a Testbed provider to address the issue. This notification can be done via email.

7 INTERACTION WITH FIESTA-IOT PLATFORM

Figure 13 illustrates the primary interaction of the FIESTA components with both FIESTA-IoT *Reasoner* and *Analytics* tools (as the FIESTA-IoT *Monitoring* tool is at the conceptual stage the interaction with the architecture has been omitted). The details of the interactions for the separate components are as follows:

1. FIESTA-IoT *Reasoner* – The experimenter first discovers and selects the data sets of interest from the IoT registry. Next the experimenter has the option of either creating a new set of rules that are tailored to the specific application of interest; or a set of existing rules are selected from the semantic rule repository. The inference engine then executes the aforementioned rules on the selected data set, where the processed data set is potentially re-annotated and stored semantically on the SDR (labelled as S. Data Repository in Figure 13).
2. FIESTA-IoT *Analytics* – The experimenter first discovers and selects the data set for analysis, where the data analysis methods to be executed along with the appropriate parameters are also defined. Once defined, the FIESTA-IoT Analytics component first retrieves the data set from the SDR, where the data analysis methods are then sequentially executed on the data set. The processed data set is then pushed back to the experimenter.
3. FIESTA-IoT Monitoring tool – The monitoring tool will interact with the message bus, where data being pushed into the SDR will be monitored in real-time. For a given Testbed sensor with substandard data quality, feedback will be provided to the Testbed provider, such that appropriate correcting actions can be taken.

Copyright © 2016 FIESTA-IoT Consortium

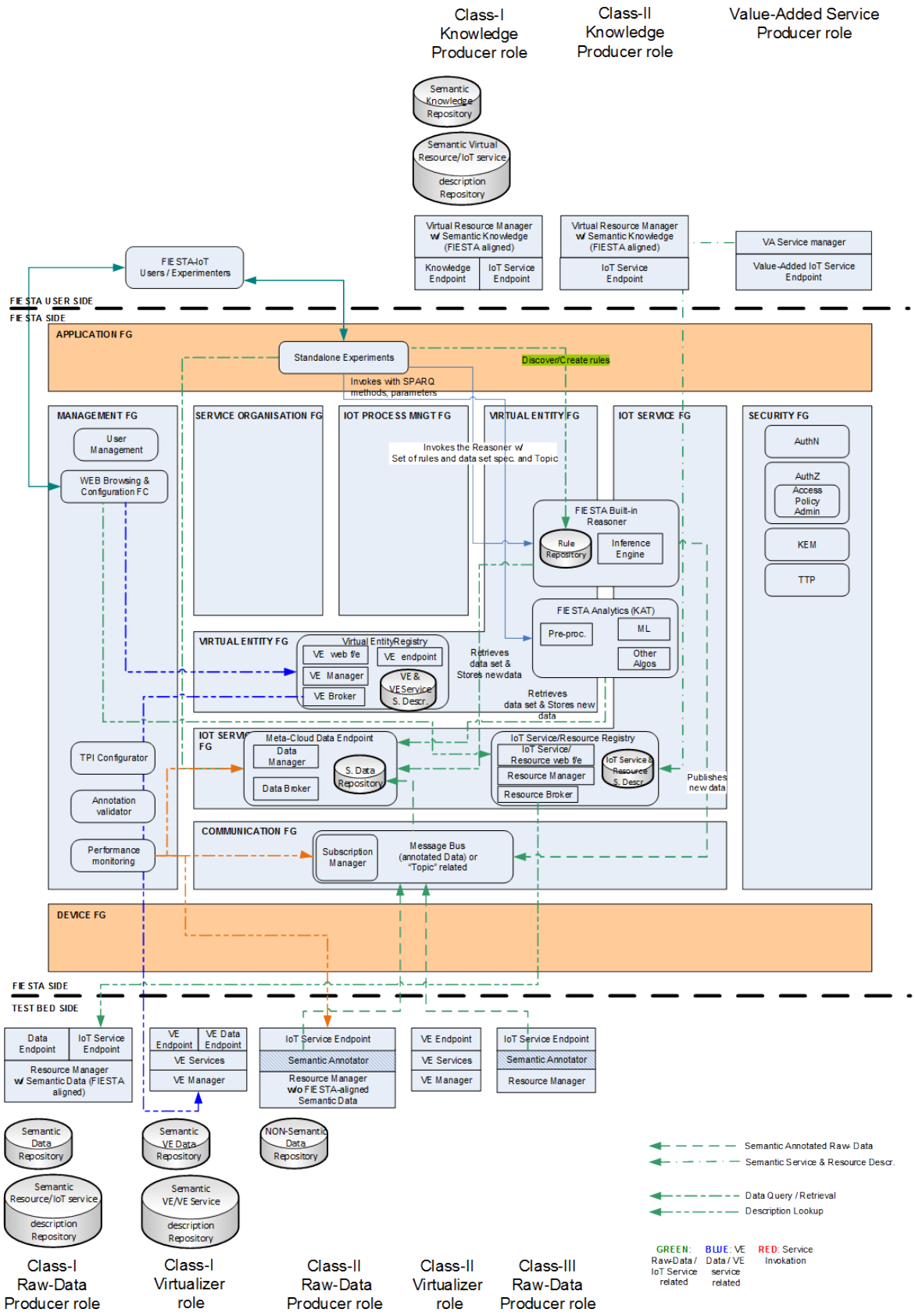


Figure 13. FIESTA architecture including the interactions of the relevant components with the reasoning, analytics and monitoring tools.

8 CONCLUSIONS AND FUTURE WORK

In this document we have presented both the conceptual and development approaches for two data analytics components referred to as: FIESTA-IoT *Reasoner* and FIESTA-IoT *Analytics*, along with a conceptual overview of the data monitoring mechanisms.

The reasoning component enables the user to select from either a set of rules that are stored on the semantic rule repository or for the development of user generated rules, for processing data sets. The processed data is then re-annotated and stored in the SDR. Furthermore, a comprehensive description of the framework required for developing the components such as the semantic rule repository and inference engine was provided. Finally this document also provides a description of the interaction between FIESTA-IoT *Reasoner* and the relevant components from FIESTA-IoT.

While the FIESTA-IoT Analytics component enables the experimenter to select a set of pre-processing and machine learning algorithms so as to allow the experimenter to carry out both inference and prediction. This document provides an overview of the machine learning and pre-processing methods included in the component, along with a description of how the FIESTA-IoT *Analytics* component interacts with FIESTA-IoT. Furthermore, this document provided a use case example that illustrated the FIESTA-IoT *Analytics* component annotating traffic data. This report also discussed the need for tools for both monitoring data streams, while providing examples of potential tools that can be utilized for achieving the stated goals.

Future work with respect to the FIESTA-IoT *Reasoner* will seek to potentially replace Jena Rules within SLOR by SPARQL CONSTRUCT rules to encourage interoperability since SPARQL is a W3C recommendation. Furthermore, the implementation of rules will not depend on the Jena Semantic Web framework SPARQL CONSTRUCT rules and a SPARQL query engine enables the inference of new information in the same way that it has been done with Jena rules and the Jena inference engine. SPARQL CONSTRUCT rules would be compliant with the FIESTA-IoT ontology. We also plan, to integrate CQELS, a SPARQL query engine adapted to real-time/stream data since it has been designed by one of the partners from the FIESTA-IoT consortium. CQELS addresses real-time challenges but does not provide high level abstraction of IoT data that we addressed in this deliverable.

Future work regarding the FIESTA-IoT *Analytics*, will seek to integrate the proposed component with the rest of the FIESTA-IoT architecture. We will also explore the expansion of the number of techniques that are available to the experimenter. Furthermore, we will seek to investigate the role of knowledge producers in developing and exposing data analysis techniques to experimenters.

Copyright © 2016 FIESTA-IoT Consortium

9 REFERENCES

[1] IoT-A, “Deliverable D1.5: Final architectural reference model for the IoT,” 2013.

[2] IoT-A, “Deliverable D1.2: Initial architectural reference model for IoT,” 2011.

[3] IEEE, “Guide for monitoring, information exchange, and control of distributed resources interconnected with electric power systems,” 2007. [Online]. Available: http://goo.gl/gEQlqd. [Accessed 31 March 2016].

[4] A. Gavras, “Experimentally driven research white paper,” 2010.

[5] A. H. Soukhanov, K. Ellis and M. Severynse, The american heritage dictionary of the english language, Boston: Houghton Mifflin, 1992.

[6] M. Leggieri, A. Passant and M. Hauswirth, “Incontext-sensing: Lod augmented sensor data?,” in *In Proceedings of the 10th International Semantic Web Conference (ISWC 2011)*, 2011.

[7] D. Le Phuoc, “A native and adaptive approach for linked stream data processing,” PhD thesis, 2013.

[8] D. Le-Phuoc, M. Dao-Tran, J. Xavier Parreira and M. Hauswirth, “A native and adaptive approach for unified processing of linked streams and linked data,” in *In The Semantic Web–ISWC 2011*, 2011.

[9] D. Le-phuoc and M. Hauswirth, “Linked open data in sensor data mashups,” 2009.

[10] S. Ben Mokhtar, D. Fournier, N. Georgantas and V. Issarny, “Context-aware service composition in pervasive computing environments,” in *In Rapid Integration of Software Engineering Techniques*, 2006.

[11] C. Henson, A. Sheth and K. Thirunarayan, “Semantic perception: Converting sensory observations to abstractions,” *IEEE Internet Computing,* vol. 16, no. 2, pp. 26-34, 2012.

[12] D. Fensel, F. van Harmelen, B. Andersson, P. Brennan, H. Cunningham, E. Della Valle, F. Fischer, Z. Huang, A. Kiryakov, T.-I. Lee and et.al, “Towards larkc: a platform for web-scale reasoning,” in *IEEE International Conference in Semantic Computing*, 2008.

[13] F. Ganz, D. Puschmann, P. Barnaghi and F. Carrez, “A practical evaluation of information processing and abstraction techniques for the internet of things,” *IEEE Internet of Things Journal,* vol. 2, no. 4, pp. 340-354, 2015.

[14] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann and I. H. Witten, “The weka data mining software: An update,” *SIGKDD Explorations,* vol. 11, no. 2, pp. 10-18, 2009.

[15] Mathworks, “ThingSpeak,” [Online]. Available: https://thingspeak.com/.

[16] FIESTA-IoT, “Delivarable 2.4: FIESTA-IoT meta cloud architecture,” 2015.

Copyright © 2016 FIESTA-IoT Consortium

[17] W. J. Dixon, “Simplified estimation from censored normal samples,” *The Annals of Mathematical Statistics,* vol. 31, no. 2, pp. 385-391, 1960.

[18] G. James, D. Witten, T. Hastie and R. Tibshirani, An introduction to statistical learning, Verlag, NY: Springer, 2015.

[19] S. Garcia, J. Luengo and F. Herrera, Data preprocessing, Verlag, NY: Springer, 2014.

[20] P. Stoica and R. L. Moses , Spectral analysis of signals, Upper Saddle River, NJ: Prentice Hall, 2005.

[21] P. Anantharam, P. Barnaghi, K. Thirunarayan and A. Sheth, “Extracting city traffic events from social streams,” *ACM Transactions on Intelligent Systems and Technology,* vol. 6, no. 4, p. 1086–1108, 2015.

Copyright © 2016 FIESTA-IoT Consortium