# D2.1 Foundations of Semantic Data Models and Tools, IoT and Big Data Integration in Multi-Cloud Environments

<table>
<thead>
<tr>
<th>Deliverable Id :</th>
<th>D2.1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Deliverable Name :</td>
<td><strong>Foundations of Semantic Data Models and Tools, IoT and Big Data Integration in Multi-Cloud Environments</strong></td>
</tr>
<tr>
<td>Status :</td>
<td>Final</td>
</tr>
<tr>
<td>Dissemination Level :</td>
<td>PU</td>
</tr>
<tr>
<td>Due date of deliverable:</td>
<td>M12</td>
</tr>
<tr>
<td>Actual submission date :</td>
<td>M13</td>
</tr>
<tr>
<td>Work Package :</td>
<td>WP2</td>
</tr>
<tr>
<td>Organization name of lead contractor for this deliverable :</td>
<td>WINGS</td>
</tr>
<tr>
<td>Author(s):</td>
<td>Vera Stavroulaki, Dimitris Kelaidonis, Kotas Petsas, Antonis Moustakos, Panagiotis Vlacheas, Panagiotis Demestichas, Kazuo Hashimoto</td>
</tr>
<tr>
<td>Partner(s) contributing :</td>
<td>ALL</td>
</tr>
</tbody>
</table>

**Abstract:** This document provides an analysis of available Semantic Data Models, Data Standards and Semantic Tools for Data Handling and Querying, with special focus on those domains with major impact in the envisaged use cases/demonstration scenarios. Moreover, the document provides an overview of requirements and challenges for necessary data model components as well as for iKaaS Data Processing and Knowledge Acquisition over multi-cloud infrastructures. In addition the document presents selected data model components as well as first concepts for Data Processing and Knowledge Acquisition mechanisms, including formal definition of knowledge types and knowledge acquisition processes in the scope of the iKaaS use cases.
Contents

1 Introduction................................................................................................................. 4
  1.1 The scope of the iKaaS project................................................................. 4
  1.2 iKaaS data model overview................................................................. 5
  1.3 Overview of iKaaS Data processing and Knowledge Acquisition .... 7
  1.4 iKaaS application domains and scenarios........................................... 9
2 Recent technology trends and iKaaS advances and innovation .. 11
  2.1 Semantic data models and data standards.............................. 11
    2.1.1 State-of-the-art ............................................................................ 11
    2.1.2 iKaaS advances and innovation ................................................. 23
  2.2 Semantic tools for data handling and querying .................. 24
    2.2.1 State-of-the-art ............................................................................ 24
    2.2.2 iKaaS advances and innovation ................................................. 27
  2.3 Data processing and Knowledge Acquisition over multi-cloud environments ................................................. 29
    2.3.1 Big Data in the Context of IoT .................................................... 29
    2.3.2 Knowledge as a Service framework ............................................. 33
    2.3.3 Tools exploited for data processing and knowledge acquisition implementations ................................................. 36
3 Requirements for iKaaS data model and iKaaS data processing and knowledge acquisition ................................................. 41
  3.1 Requirements for data model components ...................... 42
    3.1.1 City data model requirements ...................................................... 42
    3.1.2 Health data model requirements ............................................... 43
    3.1.3 Access Rights data model requirements .................................. 44
    3.1.4 Virtual Entity data model requirements .................................. 45
    3.1.5 Service data model requirements .............................................. 48
    3.1.6 Knowledge data model requirements ....................................... 50
    3.1.7 User data model requirements .................................................... 52
    3.1.8 Use of data model components by iKaaS architecture components ................................................. 53
3.2 Requirements for iKaaS Data processing and Knowledge Acquisition ................................................................. 54

4 iKaaS Data Model components ................................................................. 58
  4.1 City data model ........................................................................... 61
  4.2 Health data model ....................................................................... 63
  4.3 Access Rights data model ............................................................. 63
  4.4 Virtual Entity data model .............................................................. 66
  4.5 Service data model ....................................................................... 70
  4.6 Knowledge data model ................................................................. 75
  4.7 User data model ........................................................................... 79

5 Data processing and Knowledge Acquisition over multi-cloud environments ........................................................................................................ 83
  5.1 Knowledge in iKaaS applications/use cases ................................... 83
    5.1.1 Knowledge definition ............................................................ 83
    5.1.2 Form of knowledge .............................................................. 83
  5.2 Platform knowledge .................................................................... 89
  5.3 Generic Data processing and Knowledge Acquisition mechanisms ........................................................................................................ 90
    5.3.1 Data Processing for Anomaly (Event) Detection from Environmental Data ............................................................... 90
    5.3.2 Complex Event Processing .................................................... 93
    5.3.3 Tools for Data collection and Annotation from Mobile Devices ................................................................. 95

6 Conclusions ......................................................................................... 99

7 Appendix: General requirements for IoT Big Data processing and Analytics ........................................................................................................... 100
  List of figures .................................................................................. 103
  List of Tables ................................................................................... 104
  List of Abbreviations ..................................................................... 105
  List of References .......................................................................... 107
1 Introduction

This document provides an analysis of available Semantic Data Models, Data Standards and Semantic Tools for Data Handling and Querying, with special focus on those domains that have a major impact in the envisaged use cases/demonstration scenarios (Section 2). Moreover, the document provides an overview of requirements and challenges for the data model components (Section 3.1) as well as for iKaaS Data Processing and Knowledge Acquisition approaches over multi-cloud infrastructures (Section 3.2 and Appendix in Section 7). Following the requirements elicitation, the document presents the first design of the developed data model components (Section 4) as well as the first concepts for Data Processing and Knowledge Acquisition mechanisms, including formal definition of knowledge types and knowledge acquisition processes in the scope of the iKaaS use cases (Section 5). In this section, the scope of iKaaS and the concept of iKaaS data model and Data Processing and Knowledge Acquisition shall be briefly described, before being described in depth in later sections.

1.1 The scope of the iKaaS project

iKaaS is a project to develop a platform to integrate essential technologies, such as big data analysis, privacy management, IoT (Internet of Things) in multi-cloud environments.

Applications considered are context aware services and environment monitoring services (for more details one can refer to the iKaaS D5.1 deliverable [1]). The context aware services are applied in the healthcare applications and include the healthcare support service at Tago-nishi in Sendai and the ambient assisted living, both of which differentiate themselves from conventional healthcare services by in-depth monitoring of users’ context. Environmental monitoring services investigated in the iKaaS project are the community management service in Tago-nishi and pollen sensing service in Madrid, while some aspects are also addressed in the ambient assisted living use case (such as home environment monitoring, e.g. temperature, luminosity, etc. as well as conditions in a city e.g. weather, pollution, traffic).
Therefore, the application data processed in the iKaaS platform are health status data of individuals, water/electricity/gas usage at each household, environmental sensing data (temperature, humidity, pollen density, pollution, weather, etc.) and geospatial data of a town.

Figure 1 shows the conceptual description of application, data and essential technology in the iKaaS project.

**1.2 iKaaS data model overview**

In this section, the iKaaS data model shall be described in brief. The iKaaS data model is defined for the ease of data integration among services. The conceptual positioning of the iKaaS data model is depicted in Figure 2.

The iKaaS data model consists of a set of component data models and dependency relations among them. The data model components
correspond to particular services and concepts included to different application and services domains. Indicatively, the environment sensing systems (CESP (Community Environment Sensing Platform), Madrid system), healthcare support systems (healthcare support system in Tago-nishi, ambient assisted living), city model in town management system, privacy, etc.

Each component data model has its own data class definition and a set of dependency relations to interact with other data models. Dependency is a reference relation or linkage relation, which is represented with necessary protocols or standards. For instance, the geospatial data model defines geometry and attributes of geographic features and also defines a linking mechanism to provide location information to other data models. This avoids having every individual model duplicating location information, thereby causing inconsistencies in geographic information modelling. Linking mechanisms will reference a class in the geospatial data model that provides entity of location information of linked features in form of URIs to concept location description model instances, such as CityGML. Figure 3 shows the relationship between the various identified data model components. In section 4, each data model component is described in terms of its data class definition and the links to other data models.

Further to the above, a detailed description of the particular advances and innovation of the proposed data models and the tools for semantic data handling is presented in the sections 2.1 and 2.2 respectively. Indicatively, iKaaS aims to provide a universal data model that will be able to exploit and combine existing data models (e.g. CityGML) for the provision of an integrated version for the modelling of the Cloud-IoT platforms. Moreover, the provision of a hybrid technological solution for the semantic data management is proposed as the solution for the manipulation of heterogeneous data, associated with the different conceptual entities, such as services, VEs, etc. Additionally, a detailed correlation among the different data models and the project objectives is provided in section 3.1.
Figure 3: Dependence relation in the iKaaS data model

1.3 Overview of iKaaS Data processing and Knowledge Acquisition

Data processing and Knowledge Acquisition in iKaaS includes functionality for data analytics, learning, knowledge inference and reasoning to support autonomous behaviour and self-adaptation of applications. These methods aim to realise a) enhanced awareness with respect to external situation(s), operational context and human/social aspects, b) learning capabilities for building knowledge and experience related to situation and past application behaviour adaptation, so as to enable faster processing of data, more efficient and reliable behaviour adaptation and control, etc. and c) reasoning capabilities to support the optimal autonomous application/system behaviour adaptation, taking into account current context, knowledge and policies.

This will allow the acquisition of knowledge and experience from real world IoT data, provided by individual smart objects or also by other stakeholders/data sources, enabling the notion of Knowledge as a Service (KaaS), which can be exploited by various situation/context aware applications/stakeholders. Enabling KaaS will not only contribute to the realisation of enhanced situation aware applications, but will also foster new business models where various stakeholders
may provide knowledge to other service providers, which can in turn exploit this knowledge to develop new Services. Different types of knowledge may be derived and exploited in various use cases. In the following some indicative examples of knowledge and associated learning mechanisms are provided (it should be noted that these are far from exhaustive). As a first example, home automation in the scope of Assisted Living is assumed. In this context, knowledge refers to user patterns in terms of configuring the home environment (e.g. temperature, lighting, etc.) In this sense data processing and knowledge acquisition mechanisms deal with learning user patterns to forecast user desires with respect to home and appliances configuration and proactively take actions/offer recommendations.

For example, the user sets the alarm to 6:00 am. Since the user usually gets up at 7:00 am, the heating, lighting, etc. are operated accordingly (e.g. heating switches on at 6:45 am). Through learning mechanisms, the system can derive that the heating should be switched on at 5:45 am - even though the user has not said/done something to explicitly request this. In addition, the system can learn from user behaviour/patterns to improve its performance. For example if the user keeps adjusting the temperature (turning the heating/air-conditioning up/down, switching the lights on earlier than that in the system configuration) these adjustments are recorded along with time, weather, date information to gradually (autonomously) derive knowledge on what the user prefers most. As an additional example, we assume the case of remote health monitoring. Vital signs of an elderly/disabled person are monitored via a wearable device (e.g. a smart watch). Activities and patterns of movements (e.g. walking, sleeping patterns, eating patterns, physical and social activity, etc.) can also be monitored via motion detection sensors, accelerometers, etc. These patterns can be assessed for inference of the physical condition of an individual. In this case, data processing and knowledge acquisition mechanisms deal with learning patterns in user physical status and behaviour to identify pattern irregularities (any abnormality in usual patterns). In this context, knowledge refers to the user’s health/physical status. If something unusual/potentially problematic is observed, a notification is issued to a designated doctor/health care professional. Notifications/alarms can be raised in case something is not yet abnormal but the recorded values show a trend towards a potential problematic situation (e.g. increasing blood pressure which has still not reached a certain threshold may still be worrying).
The data model component for representing knowledge in a particular domain, e.g. “Smart Home”, “Smart City”, etc. is described in detail in section 4. A formal definition (mathematical formulation) of knowledge in the scope of the various iKaaS use cases is provided in section 5. With respect to the iKaaS innovation, as it is described in detail in the section 2.3 an indicative proposal refers to the combination of existing data processing solutions with innovative Machine Learning algorithms (based on Time-series and Bayesian Networks) so as to provide high accuracy and reliability on prediction capabilities. Additionally, the knowledge derivation as an explicitly modelled data in a knowledge database, will allow the provision of knowledge to any authorized third party entity, which in turn can ideally use this data so as to improve its functionalities and the performance of these as provided to end-users and applications.

1.4 iKaaS application domains and scenarios

This section provides an overview of the iKaaS applications and corresponding scenarios.

The iKaaS project has defined 3 application scenarios and aims to design mechanisms for the acquisition and sharing of data and knowledge in the scope of these applications by utilizing the iKaaS data model. Table 1 provides an overview of the iKaaS applications.

<table>
<thead>
<tr>
<th>Service</th>
<th>Application Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Environmental health service (Madrid use case)</strong></td>
<td>Environmental health service will offer real time information about the levels of pollution and pollen and their distribution in Madrid City.</td>
</tr>
<tr>
<td><strong>Town and health management service (Tago-nishi use case)</strong></td>
<td>Town management organization provides town asset management service and energy management service. It also provides health support service for the residents.</td>
</tr>
<tr>
<td><strong>Ambient Assisted Living</strong></td>
<td>The AAL use case cover different service domains, including smart home and smart city. AAL will</td>
</tr>
</tbody>
</table>
provide services for the real-time monitoring of the environmental condition both indoor and outdoor and health status of end-user. It will support the provision of notifications to the end-user with respect to their health status, suggestions for navigation in the smart city, etc.
2 Recent technology trends and iKaaS advances and innovation

This section provides an overview of relevant state of the art for Semantic data models and data standards (sub-section 2.1), Semantic tools for data handling and querying (sub-section 2.2) and Data processing and Knowledge Acquisition over multi-cloud environments (sub-section 2.3).

2.1 Semantic data models and data standards

2.1.1 State-of-the-art

The Semantic Web is a Web 3.0 web technology and constitutes a way for linking data between systems or entities, allowing the enrichment and the self-description of data interrelations that are available across the globe on the web. Essentially, the semantic web tries to convert the human understandable data to machine understandable data. More specifically, in the current web there are data that is represented in web documents/files such as HTML, XML, JSON, etc., which are readable by humans and machines. The humans have the ability to understand the meaning of the provided data but the machines meet difficulties to extract meaning from the current data forms. This situation leads to the need for data models for the management of distributed data in the Web. There are numerous technologies and standards that have been designed and deployed to realise the semantic web. In the rest of this section, an overview of the relevant Semantic Web technologies / technological standards for semantic modelling is provided, as well as a detailed description of the existing semantic data models standards.

The W3C, that leads the Semantic Web technologies and concepts standardisation, is helping to build a technology stack to support a “Web of data,” [2]. The ultimate goal of the Web of data is to facilitate large-scale data integration and processing and to develop systems that can support trusted interactions over the network. Several technologies have been introduced for the implementation of Semantic Web that make possible the Web of data by building vocabularies, creating data in standard data exchange formats and making the web capable of storing data. There are many well-known standardized technologies which realize the linked data process: RDF (Resource Definition Framework) [3]; OWL (Web Ontology Language) [4]; SKOS (Simple Knowledge Organization System) [5]. The Linked Data [6][7] concept is coming as the basic...
reference to web of data, namely as an addition to the extension of the current web of documents, by adding in the same time, the idea of supplying structured data to the web, which means that data must be linked so that “the web of data” can be explored by people or machines thus making the web more useful. In particular, Linked Data consists of connecting data which is in the web and is not related to one another. In order to link and structure these data, RDF [3] and HTTP (Hypertext Transfer Protocol) [8] are used. Hence the new linked-data-web is not cutting off from the traditional document-based web, so document-based web has to be considered and, therefore it is necessary to have some knowledge about it.

The current web infrastructure (i.e. the “web of documents”) is constituted by some Standard Web Technologies, such as: URLs and URIs [9] that are structured as a set of characters used to identify resources on the internet or to identify hypertext documents called HTML [10], in which the creation of the web content is delegated; and the HTTP [8] as a protocol to transfer hypertext. So the point is that the “web of data” uses these URIs as resources identification, the HTTP for retrieving resources of its descriptions; and RDF (the chosen standard web technology for linked data, which is a XML-based metadata data model) for describing resources. The above features provide the ability for reliable data processing linking data, as well as make them manageable (perform queries/updates) by using corresponding semantic standards for the data management, such as SPARQL [62].

The Linked Data [11]concept, introduced by Tim-Berners Lee in 2006, suggested the following principles to publish linked-data; i) URIs for the identification of things, ii) HTTP URIs so that these things can be referred to and looked up (“dereferenced”) by people and user agents, iii) the provision of useful information about the thing when its URI is dereferenced, using standard formats such as RDF/XML, and iv) the involvement of links to other, related URIs in the exposed data to improve discovery of other related information on the web.

Focusing on the data modelling standards and the existing data models, the Open Geospatial Consortium (OGC) [12] provides a set of different standards for the data modelling that have already started being integrated with the W3C Semantic Web technologies standards aiming to improve interoperability and integration of spatial data on the Semantic Web [13].
Table 2: Conceptual layers data modelling standards

<table>
<thead>
<tr>
<th>Level</th>
<th>Conceptual Layer</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Standards that describe data models for information</td>
<td>Provides abstract conceptual schemas for describing the application schemas.</td>
</tr>
<tr>
<td>2</td>
<td>Standards for information management</td>
<td>Description of data sets and services</td>
</tr>
<tr>
<td>3</td>
<td>Standards for information services</td>
<td>Support the specification of information services.</td>
</tr>
<tr>
<td>4</td>
<td>Standards for encoding of information</td>
<td>Support the interchange of information between systems.</td>
</tr>
<tr>
<td>5</td>
<td>Application schemas</td>
<td>Provide a description of the semantic structure of the dataset</td>
</tr>
</tbody>
</table>

Using the layers illustrated in Table 2, Table 3 is an overview of existing OGC and Semantic Web standards, complemented by the description of their main features, the implementation technologies that support their reference implementations, as well as an indicative matching of the modelling category in which the corresponding standard belongs and/or could be applied. All referenced standards/models are described in more detail in the following.

Table 3: Overview of existing OGC and other Semantic Web standards

<table>
<thead>
<tr>
<th>Standards</th>
<th>Layer</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Catalog Service for the Web (CSW)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Geographically Encoded Objects for RSS feeds (GeoRSS)</strong></td>
<td></td>
</tr>
<tr>
<td><strong>Geographic information -- Ontology --: part 1</strong></td>
<td>X</td>
</tr>
<tr>
<td><strong>Geographic information -- Ontology --: part 2</strong></td>
<td>X</td>
</tr>
<tr>
<td><strong>Geographic Query Language for RDF Data (GeoSPARQL)</strong></td>
<td>X</td>
</tr>
<tr>
<td><strong>Geography Markup Language (GML)</strong></td>
<td>X</td>
</tr>
<tr>
<td><strong>GeoJSON</strong></td>
<td>X</td>
</tr>
<tr>
<td>Place Identifier</td>
<td>X</td>
</tr>
<tr>
<td>-------------------</td>
<td>---</td>
</tr>
<tr>
<td>CityGML</td>
<td>X</td>
</tr>
<tr>
<td>Observations and Measurements (O&amp;M)</td>
<td>X</td>
</tr>
<tr>
<td>SensorML</td>
<td>X</td>
</tr>
<tr>
<td>Sensor Observation Service (SOS)</td>
<td>X</td>
</tr>
<tr>
<td>TransducerML (TML)</td>
<td>X</td>
</tr>
<tr>
<td>WaterML</td>
<td>X</td>
</tr>
<tr>
<td>IEEE11073PHD</td>
<td>X</td>
</tr>
<tr>
<td>Continua Alliance Guidelines</td>
<td>X</td>
</tr>
<tr>
<td>HL7</td>
<td>X</td>
</tr>
<tr>
<td>Web Processing Service</td>
<td>X</td>
</tr>
<tr>
<td>Web Registry Service</td>
<td>X</td>
</tr>
<tr>
<td>iCore Service Model</td>
<td>X</td>
</tr>
<tr>
<td>iCore Real World Knowledge Model</td>
<td>X</td>
</tr>
<tr>
<td>iCore System Knowledge Model</td>
<td>X</td>
</tr>
<tr>
<td>IoT-A Entity Model</td>
<td>X</td>
</tr>
<tr>
<td>IoT-A Resource Model</td>
<td>X</td>
</tr>
<tr>
<td>IoT-A Service Model</td>
<td>X</td>
</tr>
<tr>
<td>IPSO Application Framework</td>
<td>X</td>
</tr>
<tr>
<td>NGSI 9/10 specification</td>
<td>X</td>
</tr>
</tbody>
</table>

**Geographic information/knowledge:** Geospatial information can be classified into “Topography” which represents objects in the real world visually, and “Theme” which has context to connect itself to geographical location.

“Height” is a topography. “Slope” can be extracted from analysis of the height as new knowledge. This slope can be used for pedestrian support or driving support. In healthcare support service, they may introduce the following knowledge in order to recommend a walking route for an elderly person who is lacking exercise but also suffering from high blood pressure:

- The walking route should not contain any slope steeper than 5 degree.
“Location of Traffic Accident” can be an example of theme. “Hazardous Location of Traffic Accident” can be generated as new knowledge by trend analysis on the location of traffic accident. Hazardous location of traffic accident is the useful information as a countermeasure for traffic accident or pedestrian safety support.

Ontology of Geospatial information can be classified as either class level or instance level ontology. Class level ontology describes relations among geospatial classes, such as “Road” to “Street” or “Building” or “House”. On the other hand, instance level ontology describes relations among instances when the same place is referred to different expressions such as “Address”, “Name commonly used”, “Postal code”.

Ontology can be used as a dictionary of morphological analysis for searching geospatial data and metadata. “Namazu [14]” is an open source full-text retrieval search system that is widely used as a search engine for geospatial data clearing house. The maintenance and update of a dictionary, however, is an ongoing challenge, since a dictionary is heavily dependent on human knowledge. Morphological analysis is a tool for ontology maintenance we may use, and Namazu is introduced as one of useful tools mainly used in Japan.

**Ontology of Geographic information** [15][16]: Standards for ontology in the field of geographic information have been considered since 2002 in the Technical Committee 211 (TC211) of the International Standard Organization (ISO). There is a widespread agreement amongst experts that geospatial information should or can be seamlessly combined with other cloud data for various services with ontology information. Currently, there are two documents that cover subjects of conceptual framework and implementation rules on Web Ontology Language (OWL) that have been drafted. Implementation of the iKaaS data model can include aspects related to those drafts in combination with CityGML and other existing standards. [16] is a technical specification of the framework to treat geospatial knowledge. It is regarded as a guideline to design iKaaS local DB since it contains a geospatial knowledge. [17] shows a set of rules to map geospatial data to owl data. It illustrates how to integrate geospatial knowledge with other type of knowledge.
Although we have not yet discussed the treatment of knowledge of this type shown in [17], we should consider this issue sooner or later.

**Place Identifier**[17][18]: The Place Identifier (PI) architecture which was proposed by the Japanese body of TC211 also meets the iKaaS data model requirements and use cases. TC211 has been working on PI and published a standard for conceptual architecture and a committee draft (CD) for a linking mechanism. PI is a type of a gazetteer which provides location linking capabilities together with contextual information. Place is identified using either coordinate identifiers, geographic identifiers, or virtual world identifiers such as URI. The reference model defines a mechanism to match multiple Place Identifiers to the same place. In addition, a data structure and set of service interfaces are also defined in this reference model. Such a flexible reference mechanism is necessary to consider for the treatment of geospatial knowledge in iKaaS.

**Geographically Encoded Objects for RSS feeds (GeoRSS)** [19]: As RSS and Atom become more prevalent as a way to publish and share information, it becomes increasingly important that location is described in an interoperable manner so that applications can request, aggregate, share and map geographically tagged feeds. GeoRSS was designed as a lightweight, community driven way to extend existing feeds with geographic information. GeoRSS supports the modelling and the management of XML data and can be applied for the modelling of geographical locations.

**Geographic Query Language for RDF Data (GeoSPARQL)** [20]: it supports representing and querying geospatial data on the Semantic Web. GeoSPARQL defines a vocabulary for representing geospatial data in RDF, and it defines an extension to the SPARQL query language for processing geospatial data. In addition, GeoSPARQL is designed to accommodate systems based on qualitative spatial reasoning and systems based on quantitative spatial computations. GeoSPARQL supports OWL and RDF for the modelling of data, SPARQL for the query of data and can be applied on the domain for the locations modelling.

**Geography Markup Language (GML)** [21]: is an XML grammar for expressing geographical features. GML serves as a modelling
language for geographic systems as well as an open interchange format for geographic transactions on the Internet. As with most XML based grammars, there are two parts to the grammar – the schema that describes the document and the instance document that contains the actual data. A GML document is described using a GML Schema.

GML is a specification of XML based encoding for geospatial information. It is accepted as a standard format for geospatial information exchange among different computer systems. iKaaS should consider this technology for geospatial knowledge transfer from one site to the other.

CityGML [22]: is an open data model and XML-based format for the storage and exchange of virtual 3D city models. It is an application schema for the Geography Markup Language version 3.1.1 (GML3), the extendible international standard for spatial data exchange issued by the Open Geospatial Consortium (OGC) and the ISO TC211. The aim of the development of CityGML is to reach a common definition of the basic entities, attributes, and relations of a 3D city model. This is especially important with respect to the cost-effective sustainable maintenance of 3D city models, allowing the reuse of the same data in different application fields. CityGML can be used for the modelling of location, places, buildings, areas, etc. and support XML for the data representation. CityGML defines a set of data structure for 3D representation of city space and XMLSchema based on GML. The City Data Model we are developing is based on CityGML, however, we are trying to extend it by adding new classes and features for iKaaS town management use case.

In the following an overview of further existing OGC and Semantic Web standards is provided.

Catalog Service for the Web (CSW) [23]: supports the ability to publish and search collections of descriptive information (metadata) for data, services, and related information objects. Meta-data in catalogues represent resource characteristics that can be queried and presented for evaluation and further processing by both humans and software. Catalogue services are required to support the discovery and binding to registered information resources within an information community. CSW supports the modelling and management of XML
data and can be applied for the modelling of data, resources and services in the web and defines common interfaces to discover, browse, and query meta-data about data, services, and other potential resources. CSW defines the standard interface among retrieval systems regarding geospatial information. It is important in that it enables system traversal retrieval.

**Observations and Measurements (O&M)** [24]: specifies an XML implementation for the OGC and ISO Observations and Measurements (O&M) conceptual model, as well as constitutes an essential dependency for the OGC Sensor Observation Service (SOS) Interface Standard. More specifically, this standard defines XML schemas for observations, and for features involved in sampling when making observations. These provide document models for the exchange of information describing observation acts and their results, both within and between different scientific and technical communities. It can be applied for the modelling of the sensor data by focusing on the sensor observations and data measurements modelling, supporting XML and XMLS.

**SensorML** [25]: provides standard models and an XML encoding for describing any process, including the process of measurements by sensors and instructions for deriving higher-level information from observations. Processes described in SensorML are discoverable and executable. All processes define their inputs, outputs, parameters, and method, as well as provide relevant metadata. SensorML models detectors and sensors as processes that convert real phenomena to data. It is XML-based and supports data modelling to allow the sensor discovery, as well as to support operations such as sensor geolocation, processing of sensor observations, sensor programming mechanism and subscription to sensor alerts.

**Sensor Observation Service (SOS)** [26]: standard is applicable to use cases in which sensor data need to be managed in an interoperable way. This standard defines a Web service interface which allows querying observations, sensor meta-data, as well as representations of observed features. Further, this standard defines means to register new sensors and to remove existing ones. Also, it defines operations to insert new sensor observations. This standard defines this functionality in a binding independent way; two bindings
are specified in this document: a Key Value Pair binding and a SOAP (Simple Object Access Protocol) binding. SOS can be applied for the data modelling and management in the context of the Sensor Web.

**TransducerML (TML)** [27]: standard developed to describe any transducer (sensor or transmitter) in terms of a common model, including characterizing not only the data but XML formed metadata describing the system producing that data. TML enables the interoperability of heterogeneous sensor systems by providing a self-describing data exchange protocol based on XML. This enables the fusion of multi-sensor, multi-source data into a common operating picture. Further to that, TML maintains relative and absolute time sequencing of data from various (or all as required) sensors within a system. TML enables the analysts to compare temporal and spatially similar collected data and or compare disparate temporal or spatially collected data thus providing multiple domain coupling. It can be applied in different context from the data and resource modelling.

**WaterML** [28]: is a standard information model for the representation of water observations data, with the intent of allowing the exchange of such data sets across information systems. Through the use of existing OGC standards, it aims at being an interoperable exchange format that may be re-used to address a range of exchange requirements, some of which are described later in this document.

**IEEE11073PHD** [29]: is a standard for exchange of personal health device (sensor) data between a medical device agent (e.g. blood pressure or glucose monitor) and a manager (e.g. mobile phone or computer). The standard includes a domain information model. Objects are converted into binary data using Medical Device Encoding Rules (MDER). IEEE11073PHD is the data standard referenced by the Bluetooth Health Device Profile (HDP) [30], Zigbee Health Care Profile, USB Personal Healthcare Devices Definition [31], NFC Personal Health Device Communication Technical Specification [32], and the Continua Health Alliance. The nomenclature can also be used within other cases, such as a HL7 message of a Continua WAN interface.

**Continua Alliance Guidelines** [33]: The communication chain defined by the industry group Continua Health Alliance spans from personal health devices (e.g. blood pressure monitors) to professional
health information exchange, using previously developed standards such as IEEEl1073PHD and HL7, harmonized with the professionally used IHE profiles. Data are transmitted over WAN using a SOAP (IHE PCD-01) or REST hData (with HL2.6 payload) interface. Adherence to the guidelines is mandatory in Danish [34] and Norwegian [35] government programs.

**HL7 [36]**: HL7 (Health Level 7) is a series of standards mainly representing patient data in a health record context. HL7v2, based on delimited text and without a set data model, is still used and recommended. HL7v3 messages did not catch on, however the clinical document architecture (CDA) represented as XML has caught on. The upcoming HL7 FHIR [37] will leverage these previous standards as well as web standards, and support RESTful architectures.

**Web Processing Service (WPS) [38]**: Interface Standard provides rules for standardizing inputs and outputs (requests and responses) for geospatial processing services, such as polygon overlay. The standard also defines how a client can request the execution of a process, and how the output from the process is handled. It defines an interface that facilitates the publishing of geospatial processes and clients’ discovery of and binding to those processes. The data required by the WPS can be delivered across a network or they can be available at the server. The WPS standard could be applied on data modelling in terms of inputs/outputs managements.

**Web Registry Service (WRS) [39]**: is a software component that supports the run-time discovery and evaluation of resources such as services, datasets, and application schemes. WRS could be applied for the modelling of data and the data models management of resources, services, etc.

**iCore Service Model** [40]: constitutes one of the main specifications in the iCore platform [41] that support the decision of how services are represented in an IoT environment, both internally and in exposure at the iCore system boundaries, e.g. services representing specific applications towards external actors, or use of external services. In particular, the iCore service model is a service model considering service mash-ups of real-time data processing nodes,
which uses virtual abstract representations of things/object (Virtual Objects - VOs), as well as compositions of them (Composite VOs - CVOs), as underlying execution building blocks, and can include as such various event processing techniques/solutions.

**iCore Real World Knowledge (RWK) Model** [42]: this model is introduced in the iCore platform so as to satisfy the requirements for modelling of concepts beyond the physical objects / things. More specifically, it includes abstract Real World (RW) objects like particular events or phenomena occurring in the real world playing an important role in everyday life. The RWK Model has been designed so as to be exploitable to support of inferring new RWK concepts that will be specified and stored in a machine-interpretable format. For this reason the introduction of Semantic Web technologies such as the OWL and RDF was required to support RWK Model compatibility.

**iCore System Knowledge Model** [42]: this models handles to the running services (if not derived in another way, e.g. by query), by indicating particular system features such as 'System is up and running', 'Service up-time is X seconds', etc. This information is represented by this model and can be exploited in various ways, such as the System performance and/or availability time monitoring. The data are structured by using RDF, while an OWL ontology provides the annotation of the particular concepts that are included in the model.

**IoT-A Entity Model** [43]: an entity can have certain aspects that need to be taken into account. For example, when one needs to know about the location of an entity or the features of interest that data is available for. The OWL-DL representation has been used to define the entity model.

**IoT-A Resource Model** [43]: a resource is the core software component that represents an entity in the digital world. The resource concept has datatype properties that specify its name (hasName), an ID (hasResourceID) and a timezone defined in an external ontology (hasTimeZone). A resource also has a functional location property (hasResourceLocation) that links to the Location concept. This location could be the location of the device the resource
runs on. The functional restriction denotes that a resource can only have a link to one location instance.

**IoT-A Service Model** [43]: resources are accessed by services which provide functionality to gather information about entities they are associated with or manipulate physical properties of their associated entities. The OWL-S specification has been designed as upper ontology for the Semantic Web Services. According to this specification, Semantic Web resources provide services which are described by their service profile, service model, and service grounding. Assuming potential IoT users are interested in information about the real world entities, they will search with terminology concerning entities of several domains.

**GEOJSON Model** [44]: GeoJSON is a format for encoding a variety of geographic data structures. A GeoJSON object may represent a geometry, a feature, or a collection of features. GeoJSON supports the following geometry types: Point, LineString, Polygon, MultiPoint, MultiLineString, MultiPolygon, and GeometryCollection. Features in GeoJSON contain a geometry object and additional properties, and a feature collection represents a list of features. A complete GeoJSON data structure is always an object (in JSON terms). In GeoJSON, an object consists of a collection of name/value pairs -- also called members. For each member, the name is always a string. Member values are either a string, number, object, array or one of the literals: true, false, and null.

**IPSO Application Framework** [45]: provides a not yet standardised way to define Internet of Things (IoT) applications, by using IETF standards and web technologies such as HTTP, REST, XML, JSON, and COAP. It enables users to describe resources, interfaces, data, and interaction models. Companies such as Google, Intel, Cisco, Bosch, Ericsson, Nokia, Oracle, and others have already become members of the IPSO alliance taking advantage of the interoperability IPv6 has to offer on constrained devices.

**NGSI 9/10 specification** [46]: The Open Mobile Alliance (OMA) has defined the NGSI-9 and NGSI-10 interfaces through a detailed specification in order to manage context information based on context entities. The core functionalities of these interfaces are to provide
context information (update operations), to consume context information (query while also subscribe/publish operations) and finally to discover context entities based on query operations (register and discover operations). The NGSI-9 interface is focused on the register and discovery operations for context entities. Meanwhile, the NGSI-10 interface’s main operations are to create/read/update/delete context information. The context information model is defined by a context entity. In the context entity the information is organized as context elements that include a set of context attributes and associated metadata. An implementation of these interfaces has been performed by FIWARE [47], where RESTful communication is established with a broker that implements the NGSI-9 and NGSI-10 interfaces. In the particular implementation of NGSI-9 and NGSI-10 interfaces, the context information is represented through XML mark-up language or JSON data-interchange format.

2.1.2 iKaaS advances and innovation

The iKaaS project semantic modelling activities aim to design and develop an integrated semantic data model. The semantic data model will integrate different conceptual entities, information of which can be exploited in the context of the knowledge generation processes.

In the current related work, as well as in the research activities as they have been included in the literature, so far, the feature of an integrated semantic data model is missing. There are a lot of notable research efforts that have produced critical results in the field of the semantic data models for different domains, such as Smart Cities, Internet of Things, e-Health, etc. These models are mostly presented in isolation from each other that leads to the requirement for strong effort positioning (OPEX increase) for the integration for the provision of a unified result in the context of more complex and challenging areas/application domains.

The semantic data modelling in iKaaS comprises of a set of different data models for diverse concepts, such as security, data sources, IoT entities, etc., towards the development of a unified semantic data model for the Cloud IoT environments. These data models may constitute either already existing models deployed outside of ikaas project, or models the designed in the context of iKaaS project, such as the Service Model. The new designed models add new features/properties on the description of particular concept, such as
the services. A high-level overview of the integration of the semantic data models is presented in Figure 3, which actually presents the dependencies among the iKaaS proposed data models. Through the representation of the iKaaS data models dependencies it is obvious that all the proposed models, in the different contexts, relate with each other and create a unified structure of semantic data that can be exploited for the knowledge building in different sectors.

Consequently, the innovation introduced by iKaaS in the context of the Semantic data modelling and standards, relates to the provision of a semantic data model composed by different conceptual semantic data models, existing of new designed in the context of iKaaS. The iKaaS data model can be ideally used by the knowledge building mechanisms for the generation of solid, high-accuracy and reliable knowledge.

### 2.2 Semantic tools for data handling and querying

#### 2.2.1 State-of-the-art

As already introduced in the previous section, the Semantic Web includes a set of different technological standards and/or semantic tools that support data handling, querying and modification (update/delete). Specifically, Semantic Web technologies enable people to create data stores on the Web, build vocabularies, and write rules for handling data. Linked data are empowered by technologies such as RDF, SPARQL, OWL, and SKOS [48]. Taking into account the W3C approach [48], the Semantic Web technologies and tools can be classified in the following categories: a) Linked Data, b) Vocabularies /Ontologies, c) Semantic Data Reasoning and Inference, and d) Semantic Data management that includes data stores, query and modification of data (update/delete). In the rest of this section there is the description of each category in a way of a general overview of what each category includes, which technological aspects are covered, as well as which requirements are satisfied. The outcome of the current section will be used as the basis for section 2.3.2 that refers to the requirements for mechanisms and Semantic tools for data handling and querying.

**Linked Data** [49]-[51]: As already introduced, the Semantic Web is a Web of Data, where data may correspond to dates and titles and part numbers and chemical properties and any other data one might conceive of. The collection of Semantic Web technologies (RDF, OWL,
SPARQL, etc.) provides an environment where applications can query these data, draw inferences using vocabularies, etc. However, to make the Web of Data a reality, it is important to have the huge amount of data on the Web available in a standard format, reachable and manageable by Semantic Web tools. Furthermore, not only does the Semantic Web need access to data, but relationships among data should be made available, too, to create a Web of Data. This collection of interrelated datasets on the Web can also be referred to as Linked Data. Aiming to achieve the Linked Data creation, different technologies should be available for a common format (RDF), to make either conversion or on-the-fly access to existing databases (relational, XML, HTML, etc.). It is also important to be able to setup query endpoints to access that data more conveniently. W3C provides a palette of technologies, such as RDF, RDFa, OWL, RDFS, SPARQL, etc., to get access to the data.

**Vocabularies / Ontologies** [52]-[56]: On the Semantic Web, vocabularies define the concepts and relationships used to describe and represent an area of concern. Vocabularies are used to classify the terms that can be used in a particular application, characterize possible relationships, and define possible constraints on using those terms. In practice, vocabularies can be very complex (with several thousands of terms) or very simple. There is no clear division between what is referred to as “vocabularies” and “ontologies”. The trend is to use the word “ontology” for more complex, and possibly quite formal collection of terms, whereas “vocabulary” is used when such strict formalism is not necessarily used or only in a very loose sense. Vocabularies are the basic building blocks for inference techniques on the Semantic Web. In particular, the vocabularies and/or ontologies can be used in order to help data integration when, for example, ambiguities may exist on the terms used in the different data sets, or when a bit of extra knowledge may lead to the discovery of new relationships. Consider, for example, the application of ontologies in the field of health care. Medical professionals use them to represent knowledge about symptoms, diseases, and treatments. Pharmaceutical companies use them to represent information about drugs, dosages, and allergies. Combining this knowledge from the medical and pharmaceutical communities with patient data enables a whole range of intelligent applications such as decision support tools that search for possible treatments; systems that monitor drug efficacy and possible side effects; and tools that support epidemiological research.
**Semantic Data Reasoning and Inference** [57]-[60]: The inference constitutes the result of the reasoning process on Semantic Data. On the Semantic Web, the source of such extra information can be defined via vocabularies or rule sets. Both of these approaches draw upon knowledge representation techniques. In general, ontologies concentrate on classification methods, putting an emphasis on defining 'classes', 'subclasses', on how individual resources can be associated to such classes, and characterizing the relationships among classes and their instances. Rules, on the other hand, concentrate on defining a general mechanism on discovering and generating new relationships based on existing ones, much like logic programs, like Prolog [60], do. In the family of Semantic Web related W3C Recommendations RDFS, OWL, or SKOS are the tools of choice to define ontologies, whereas RIF [59] has been developed to cover rule based approaches. Focusing on the main aim of the inference application into the Semantic Web, inference constitutes one of the tools of choice to improve the quality of data integration on the Web, by discovering new relationships, automatically analysing the content of the data, or managing knowledge on the Web in general. Inference based techniques are also important in discovering possible inconsistencies in the (integrated) data.

**Semantic Data Management (Store / Query / Modify)** [61]-[70]: RDF is a standard model for data interchange on the Web. RDF has features that facilitate data merging even if the underlying schemas differ, and it specifically supports the evolution of schemas over time without requiring all the data consumers to be changed. RDF extends the linking structure of the Web to use URIs to name the relationship between things as well as the two ends of the link (this is usually referred to as a “triple”) [61]. Using this simple model, it allows structured and semi-structured data to be mixed, exposed, and shared across different applications. This linking structure forms a directed, labelled graph, where the edges represent the named link between two resources, represented by the graph nodes. This graph view is the easiest possible mental model for RDF and is often used in easy-to-understand visual explanations. The above specification defines the syntax and semantics of the SPARQL query language [62] for RDF. SPARQL can be used to express queries across diverse data sources, whether the data are stored natively as RDF or viewed as RDF via middle-ware. SPARQL contains capabilities for querying required and optional graph patterns along with their conjunctions and disjunctions. SPARQL also supports extensible value testing and
constraining queries by source RDF graph. The results of SPARQL queries can be results sets or RDF graphs. The RDF data can be stored on systems either as RDF Files on the Native storage system or as RDF Statements into RDF Stores [61][63]. Nowadays, there is a wide variety of RDF Stores / Triple Stores that offer storage and query capabilities on RDF data. These RDF Stores differ with each other in various aspects such as the total amount of actual storage space that take of the system storage so as to store the data, the total load and query/modification time, etc. Some of the most popular RDF Store systems are: a) Allegrograph [64], OpenLink Virtuozo [65], RDFox [66], openRDF Sesame (RDF4j) [67], Jena Fuseki [68], Bigdata(R) [69], etc. Each one of the above technologies, as well as additional similar technologies for the RDF data storing, constitutes ideal solutions that could be applied in the context of the iKaaS platform. Taking into account the requirements in the context of framework/platform architecture, combined with the corresponding features of each RDF Store technology, there is particular interest on the combination of three major technologies in the context of the Semantic Data management: a) Allegrograph Storage system [64], Apache Jena Ontology API [70] and c) Sesame Java framework [67] for RDF data storage, processing and handling.

The above approaches arise from the vision to build a powerful Semantic Data management system with high performance and reliability that focuses not only on one particular technology but on the combination of different powerful Semantic Tools for data handling, querying and modification. Specifically, Sesame is a powerful Java framework that includes creating, parsing, storing, inferencing and querying over RDF data and offers an easy-to-use API that can be connected to all leading RDF storage solutions [67]. On the other hand Allegrograph offers an integrated system-based RDF data storage solution that can be exploited so as to store very-large scales of semantic data, that in turn, can be manipulated in various ways (querying, modification, reasoning, etc.) by exploiting provided capabilities by Sesame Framework API, as well as by exploiting particular data management solutions by Apache Jena API such as the Ontology API that can contribute to the Query RDF data parsing, RDF Data Serialization, etc.

2.2.2 iKaaS advances and innovation

As it is clearly identified, the iKaaS project aims to the development of an integrated multi-Cloud architecture for the distributed
knowledge building in contemporary IoT ecosystems. Through the introduction of the universal semantic data model (section 2.1) it is possible to achieve semantic data integration and exploitation for the design of advanced knowledge models, as well as for the development of well-defined structured knowledge data.

The added value of the iKaaS project side with respect to the tools for the semantic data management, including handling, query, etc., lies in the vision of the integration of the existing technologies for the provisioning of high-performance hybrid technological solutions. The provision of such solutions will lead to the design and the deployment of integrated semantic data management tools that will work as enablers for the distributed Cloud-based semantic data management. Such tools may include storage solutions (e.g. OpenRDF, Allegrograph), query/update endpoint (e.g. SPARQL/SPARUL).

The integration of the diverse semantic tools (e.g. RDF DBs, SPARQL endpoints, Inference engines, etc.) will also allow the management of the data for the knowledge generation through the exploitation of semantic technologies for the knowledge data integration and the knowledge representation (e.g. RDF models and Ontologies for the iKaaS universal data model). Moreover, the innovation of the iKaaS project relates to the exploitation and the application of the semantic data management tools for the contemporary Cloud-based IoT environments, by integrating various different domains, such as the Smart City, Smart Home, etc. Specifically, through the exploitation of semantic data modeling technologies, combined with semantic data tools for the storage, querying on and modification of data, iKaaS will build an integrated environment of federated distributed semantic data stores.

The semantic data federation will allow the dynamic, on-demand creation and deployment of cloud-based IoT services that will be described by semantic data structured by specific data models, defined in the context of iKaaS. The realization of the data federation is lied on the semantic database federation in terms of the support of federated queries on the distributed service data. Consequently, iKaaS offer the Global Service Catalogue as the semantic storage federation for the distributed Local Service Catalogues, and through this way it is achieved the federation of the distributed data that describe the services. The services will enable a new era for the Cloud IoT, allowing the provision of services based on particular contextual
requirements, avoiding the deployment of services with low added value for the different application domains.

2.3 Data processing and Knowledge Acquisition over multi-cloud environments

2.3.1 Big Data in the Context of IoT

In this section the overall context of big data relevant to IoT is described. The term big data has emerged because of challenges on overall data processing, data sharing, storage, search engine, virtualization, etc., that the traditional technologies were not sufficient for. Many of these challenges are due to IoT. These Big data challenges are defined by 4Vs, these are Volume-, Velocity, Variety, Veracity. The volume means large data size in 100s of terabytes. The velocity means the real-time and/or streaming nature of data. The variety means the heterogeneous data (structure and unstructured, diverse data models & query language, and diverse data sources). The veracity means data uncertainty due to data inconsistency, incompleteness, ambiguities, latency, model approximations, etc.

![Big Data Properties](image)

In earlier days the big data concept was mostly applied in financial analysis, where the data are coming from every transaction. However due to IoT, in which thousands of devices and system are connected, there is real-time, semi-real time diverse set of data coming from those devices. Therefore all 4Vs of the big data challenges are related to IoT. However the velocity is the main IoT Big data challenge because of real-time and streams of data coming from diverse IoT devices and sensors. In the context of IoT-Big data, therefore often Real-Time Big Data terminology is replaced by the IoT Big Data. Data coming from the IoT devices have to be processed in real-time to
arrive for reliable and intelligent decisions. These real-time stream/time-serious IoT data should be processed and be computed following decentralized approaches. In these the processing is not only for data at the centralised cloud-level rather some kind of pre-processing is performed at the edge/local devices and gateway level.

For example, healthcare wearables (like ECG) produce up to 1000 events per second which is a challenge for real-time processing considering miniaturized devices and the number of such devices. The pre-processing should be done at the devices and gateway level, in order to accelerate the process as well as secure the data provision and sharing. Next is the volume, for example, GE gathers 50 million pieces of data from 10 million sensors each day. A wearable sensor produces about 55 million data points per day.

**State of the art**

The IoT-big data user application is very much different than the typical IoT application. IoT application means collection of devices, system and remotely monitoring and actuation of the sensors/actuator, devices, etc. However the IoT-big data application is mostly data driven, data insight from IoT devices. The applications of IoT Big Data analytics can be classified into five main categories (a) Predictive analytics (b) Prescriptive analytics, (c) Descriptive analytics, (d) Monitoring and (e) Control and optimization. All these require a deep understanding of the domains, situation and the requirements of applications by users. The data are coming not only from the devices but rather users and application context, their requirements and preference. These data analytics can propose advanced applications, systems, users, etc., knowledge to offer value-added data-driven services. Gaining insights and knowledge in real time and actionable insights can lead to performance optimization. All the above five applications are inter-related and require multiple tools like machine learning, reasoning, optimization, etc.
**Figure 5: IoT–Big Data Applications**

Predictive analytics are used in many applications where users require services that can foresee the situation and act on it. For iKaaS, early prediction of health of elderly people, prediction of environment pollen level in smart city, etc., is offered as services. Prescriptive analytics can provide many possible actionable decisions and also can provide the trade-off between them. Descriptive analytics offers the insights into the situation and helps in deep understanding. In iKaaS, for example understanding the current health condition and daily activity level of elderly people, understanding of current environment situation, pollen, CO2 level in a smart city. In healthcare service understanding the cause of diseases and diagnosis through analytics as well as finding whether there is any emergency at the moment. Monitoring, control and optimization are legacy applications, but with big data analytics they can be improved immensely. Thus, analytics can indeed offer multiple services such as observing behaviour of things, gaining important insights and processing in real time for immediate actions.

This vision boils down to solving multiple challenges: to store all the events and devices (velocity & volume), a highly scalable and distributed database, to run queries over the stored events; continuous stream processing and query (velocity & volume) to perform analytics (data mining and machine learning, distributed algorithms) over the data to gain insights.

For example, real-time fall detection and potential reaction for aging population. Real-time detection and action represent multiple challenges.

**iKaaS advances and innovation**

In this case, iKaaS project aims to combine existing technologies, so as to provide innovative, accurate and high performance software solutions. By exploiting Apache’s Hadoop with MapReduce it is
possible to achieve classification of the data, collected from IoT devices, user preferences and/or user profiles as well. In general, MapReduce is composed of the procedures of Mapping, which gives the capability of filtering and sorting the input data, and Reducing, which performs a summary operation. The filtering is an important operation, especially for the data being collected from the IoT devices, because there are possibilities of fault measurements, such as values being out of scope. These advantages of big data analysis would help the Machine Learning processes designed in the context of iKaaS platform, such as the Bayesian Networks technique that is envisioned to be applied in the AAL use case.

**Figure 6: Data Processing, Machine Learning and Knowledge Building flow diagram**

An indicative example of how iKaaS could combine Hadoop data processing capabilities with Machine Learning is presented in a high-level flow diagram in Figure 6. In particular, the input data being collected from many sensors in a specific region in Smart City for the prediction of environmental pollution level, could be grouped with MapReduce and crosschecked, in order to remove the faulty measurements and the out of scope values. Then the results of this process could be included in large-scale datasets that are used for training the Machine Learning algorithms. In this way, more accurate predictions will be achieved, since the training data will be filtered and classified; facilitating in this way the evolution of the Machine Learning algorithm in terms of the training result. In case of a
prediction of high environmental pollution in the center of a Smart City, by running again MapReduce on saved user profiles, it could classify elderly people with respiratory problems in a group and inform the specific users about the trade-off between choosing to reach their destination through the center, which is the optimal route, or by a healthier route, which protects them from the deterioration of their health. The other users will be informed to go through the optimal route. As a result, the accuracy of the prediction increases and also, the decision making is faster.

Consequently, the advantage that iKaaS brings is associated with the exploitation of data from the different data sources so as to build well-defined, grouped and classified dataset for the Machine Learning algorithms. This essentially relates to the evolution of the Machine Learning techniques in terms of their performance and accuracy by exploiting advanced legacy techniques and tools for the data processing.

2.3.2 Knowledge as a Service framework

State of the art

Data Analytics: This layer includes high-level analytical applications similar to R or Tableau delivered over a cloud computing platform which can be used to analyze the underlying data. Users can access these technologies in this layer through a web interface where they can create queries and define reports that will be based on the underlying data in the storage layer.

Data Management: In this layer, higher level applications such as Amazon Relational Database Service (RDS) and DynamoDB are implemented to provide distributed data management and processing services. Technologies contained in this layer provide database management services over a cloud platform. For example, if a user needs to have an Oracle database over the cloud, then using Amazon RDS such a service can be instantiated. As data-management services are managed by the service provider, such as taking periodic backups of data, this eases deployments and reduces resources requirements for their upkeep.

Computation Layer: This layer is composed of technologies that provide computing services over a web platform. For example, using Amazon Elastic MapReduce (EMR), users can write programs to manipulate data and store the results in a cloud platform. This layer
includes the processing framework as well as APIs and other programs to help the programs utilize it.

**Data Storage Layer:** This data storage layer is typically distributed in a portable file system, which can be scalable on demand. This can be a typical HDFS file system composed of a name node and a cluster of data nodes, which can be scaled up according to demand. This layer has high availability built into the system. The data storage layer will have a presentation component where users can directly interact with the HDFS file system to upload data for analytics.

**Cloud Infrastructure:** In this layer cloud platforms such as open stack or VMware ESX server provide the virtual cloud environment that forms the basis of the KaaS stack. This layer can also perform usage monitoring and billing for the services rendered. Although this layer is a part of KaaS, it is not accessible directly through external applications. For example, this layer does not have a presentation option and hence cannot be accessed in the same way the AWS layer would be.

**Data Infrastructure:** This layer is composed of the actual data center hardware and the physical nodes of the system. Data centers are typically composed of thousands of servers connected to each other by a high-speed network line enabling transfer of data. The data centers also have routers, firewalls, and backup systems to insure protection against data loss. This data center provides computing and storage power to multiple consumers of the KaaS service.

---

**Figure 7:** Knowledge as a service framework [72]
iKaaS advances and innovation

The large amount of data that could be delivered by distributed data sources in the multi-Cloud iKaaS ecosystem (e.g. environmental sensors, body area network, user preferences, etc), require innovative solutions so as to be efficiently handled and analysed. The analysis will take place in the data analytics layer, which can create queries that will be the input to the Machine Learning algorithms, such as Bayesian Network algorithms. The final output is the knowledge, which is offered for the creation of innovative services.

The process followed for the derivation of knowledge is:

- collecting data from the IoT devices using VEs, and cache/store them into files that will be used as the first rough input for the creation of large-scale training datasets for the Machine Learning algorithms training,
- data analysis and filtering with Big Data analytic algorithms (e.g. map reduced based algorithms),
- exploitation of the training datasets and analysing of the filtered data with machine learning algorithms,
- Knowledge derivation.

When the procedure of knowledge generation is completed, it is stored in a Knowledge DB and is available at any time for various uses, such as decision making.

An indicative example on how the extracted knowledge can be used efficiently is presented below. Assume a Time-Series Forecasting algorithm that predicts the future vital signs for many people in a City. By gathering this information for each individual in a DB and analysing it in its entirety, the result would provide statistics for the general population of the City. From those statistics, knowledge is generated about the health status of the society for a specific period in a year. As a result, an innovative service could be created, that predicts the health status of the general population and warns doctors and/or hospitals to be prepared for a possible crisis in the City.

In conclusion, the aim described above is to exploit the generated knowledge and provide innovative services. This could be achieved by exploiting existing solutions and design principles on big data analysis so as to ‘sift’ the rough data so as to be input to large-scale and continuously growing dataset for the training of the Machine Learning algorithms that will build and derive the knowledge. In relation to this part the next subsection present the state of the art of the tools and
that could be used so as to realize the knowledge as a service process.

2.3.3 Tools exploited for data processing and knowledge acquisition implementations

State of the art

Concerning the state of the art for tools that may be exploited for data processing and knowledge acquisition most current implementations utilize Hadoop or a variation of it while others are built on its filesystem implementation (HDFS with HBase) or other specific Hadoop components.

The Apache Hadoop [73] project develops open source software for distributed computing:

- **Hadoop Common**: The common utilities that support the other Hadoop modules.
- **Hadoop Distributed File System (HDFS)**: A distributed file system that provides high-throughput access to application data.
- **Hadoop YARN**: A framework for job scheduling and cluster resource management.
- **Hadoop MapReduce**: A YARN-based system for parallel processing of large data sets.

These modules developed for the Hadoop project are widely used as a whole or individually in other open/closed source projects.

Apache Spark [74] is an open-source cluster computing framework. Spark's in-memory primitives provide performance up to 100 times faster than Hadoop's two-stage disk-based MapReduce paradigm. By allowing user programs to load data into a cluster's memory and query it repeatedly, Spark is well-suited to machine learning algorithms.

Apache Mahout [75] is an environment for quickly creating scalable machine learning applications. To achieve this, Mahout uses the Hadoop MapReduce algorithms without, however, restricting contributions to only using this paradigm. Contributions that run on a single node or on a non-Hadoop cluster can also be used.

Apache Pig [76] is a platform for analysing large data sets that consists of a high-level language for expressing data analysis programs, coupled with infrastructure for evaluating these programs.
Pig programs are highly parallelizable in order to handle big data sets with ease. Pig's infrastructure layer consists of a compiler that produces sequences of Hadoop Map-Reduce programs while its language layer consists of a textual language called Pig Latin.

**Cloudera** [77] a distributor of Hadoop software, provides a data-processing framework using Hadoop. Their aim is to provide an open source framework with enterprise grade performance. Inside the framework, apache spark is utilized for in-memory data processing while Impala provides a massively-parallel SQL engine. Cloudera’s framework is also accompanied by tools that aid data management (Cloudera navigator) and data searches (Cloudera Search).

The **Hortonworks Data Platform** [78] is built completely as open source software and is based on the Apache Hadoop project. Specifically, it uses the YARN framework, the HDFS filesystem as well as Apache Hive to provide access to the data via SQL queries.

**Infobright** [79] is a focused analytical database provider with a column-store DBMS (based on MySQL) aimed at fast analysis of up to 50 terabytes per server. Relative to big data 50 TB is a small data size but Infobright uses a high compression method that provides a 20:1 to 40:1 ratio and a data-skipping technology making it well suited to machine data such as click streams, mobile data, log files, and sensor data. The database is designed for symmetric multiprocessor servers instead of massively parallel processing. While this database is based on MySQL it can be integrated with Hadoop so that it is used as a store for less used data (e.g. historical values) while the SQL engine provides fast access to recent/most used data.

**Kognitio** [80] is an analytical accelerator for Hadoop. An in-memory analytical platform, it is a massively parallel relational database for analytics that extends a Hadoop environment to enable Big Data queries to run extremely quickly, returning billions of records in seconds from standard SQL queries.

**Meteor** [81] Meteor is a JavaScript app platform, offering a complete full-stack framework for delivering web and mobile apps entirely in JavaScript. Meteor radically simplifies the development process for reactive app development. Users expect more of their apps than ever before. Meteor can deliver on those expectations quickly and effectively while providing a delightful, integrated developer experience.
**MapR** [82] is a complete distribution for Apache Hadoop that packages more than a dozen projects from the Hadoop ecosystem to provide a broad set of big data capabilities. MapR in essence provides the same capabilities as a Hadoop distribution with extra features packed in (i.e. the HDFS filesystem in the MapR distribution has been swapped with a Network File System (NFS) alternative to allow for greater scalability and ease of access) and also fixes regularly any issues that exist within the Hadoop project.

**SAP Hana** [83] is an in-memory analytics platform. When data are truly big or unstructured, Hana supports various Hadoop distributions, with Hana accessing data through Hive. Additionally, when data needs to be archived Hana uses SAP IQ that offers a compressed, columnar DBMS adapted for use with MapReduce processing. Hana also has a built-in predictive analytics library, R language support, spatial processing, natural language processing, and text analytics libraries.

**Weka** [84] is a workbench that contains a collection of visualization tools and algorithms for data analysis and predictive modelling, together with graphical user interfaces for easy access to this functionality. The implementation has been done in Java and Weka can also be utilised in other applications as a library of machine learning applications while the usage of a single training file format (Attribute Relation File Format – ARFF) makes switching algorithms easy. Weka supports several standard data mining tasks. More specifically, data pre-processing, clustering, classification, regression, visualization, and feature selection. All of Weka’s techniques are predicated on the assumption that the data are available as a single flat file or relation, where each data point is described by a fixed number of attributes. Weka also provides access to SQL databases using Java Database Connectivity and can process the result returned by a database query.

**Apache Tez** is an extensible framework for building high performance batch and interactive data processing applications, coordinated by YARN in Apache Hadoop. Tez improves the MapReduce paradigm by dramatically improving its speed, while maintaining MapReduce’s ability to scale to petabytes of data. Important Hadoop ecosystem projects like Apache Hive and Apache Pig use Apache Tez, as do a growing number of third party data access applications developed for the broader Hadoop ecosystem”[85]
Apache Storm is a free and open source distributed real-time computation system. Storm makes it easy to reliably process unbounded streams of data, doing for real-time processing what Hadoop did for batch processing. Storm has many use cases: real-time analytics, online machine learning, continuous computation, distributed RPC, ETL, and more [86].

Apache Flink [87][88], like Apache Hadoop and Apache Spark, is a community-driven open source framework for distributed Big Data Analytics. The Apache Flink engine exploits data streaming and in-memory processing and iteration operators to improve performance. Apache Flink is an open source platform for scalable batch and stream data processing. Flink’s core is a streaming dataflow engine that provides data distribution, communication, and fault tolerance for distributed computations over data streams. Flink runs on top of HDFS and YARN, focusing on ease of programming.

iKaaS advances and innovation

The iKaaS project aims to combine some of the state-of-the-art tools for data processing and knowledge acquisition, so as to provide knowledge from the IoT ecosystems in a distributed manner. The integration of these tools could lead to efficient data management, in the terms of analysis and machine learning processes.

The data storage layer that is described in the previous section, can be implemented by using the Hadoop Distributed File System (HDFS) which offers scalability on demand. The dataset that can be in various different heterogeneous formats, compatible with the specific Machine Learning algorithm that is used for a particular situation can be stored in the HDFS. Essentially, the HDFS could constitute an ideal file system for heterogeneous data format that are used for the training of algorithms that are responsible for the knowledge building. With the means of scalability, it is possible to scale up the Hadoop cluster, by adding data nodes, and upload large volumes of data to HDFS for analysis.

To achieve the distributed knowledge acquisition, it is wise to use Hadoop based tools with, either built-in or external, machine learning mechanisms. The reason is that these tools distribute the data to the Hadoop cluster, which is scalable and the number of data-nodes depend on the amount of data collected by the IoT environments. The distribution makes it easier to process and analyse the large volumes
of data. The result of analysis is the extraction of valuable information, which is the input to the machine learning algorithms. As a result, the integration of the Hadoop based tools with the machine learning mechanisms will produce an effective system for the knowledge generation. The innovation of the iKaaS project relates to the creation of such integrated systems in every local cloud, which exist in the different domains (e.g. Smart Home, Smart City), for the analysis of the collected data, from the contemporary cloud-based IoT ecosystems and the generation of distributed knowledge.

The knowledge acquisition from the systems described above, will allow the creation of innovative cloud-based services. In case these services have requirements for more input data in some domains, in order to provide better results, the system is flexible and its size can differ from local cloud to local cloud, without any software changes to the data analytics and machine learning algorithms.

Consequently, the innovation introduced by iKaaS in the context of combining data processing and knowledge acquisition tools, relates to the provision of an integrated system with flexible scalability and high accuracy, which can be effective in a multi-cloud based IoT environment.
3 Requirements for iKaaS data model and iKaaS data processing and knowledge acquisition

This section describes the requirements identified for the data models and the knowledge acquisition processes for the multi-Cloud IoT environments as they are investigated in the context of iKaaS project. The identified requirements in each category are matched with project objectives. Table 4 presents the identified objectives, in terms of their number and title, as a reminder for the reader’s facilitation. Further to that, sections 3.1 and 3.2 present the requirements for data model components and for the data processing and data acquisition respectively.

Table 4: iKaaS Objectives

<table>
<thead>
<tr>
<th>Objective</th>
<th>Title</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obj_1</td>
<td>Define a <strong>universal data model</strong> based on the Semantic Web for data collected from IoT and stored in cloud platforms</td>
</tr>
<tr>
<td>Obj_2</td>
<td>Define and develop the various concepts in <strong>trust-management-by-design</strong> for the data</td>
</tr>
<tr>
<td>Obj_3</td>
<td>Develop and implement a <strong>decentralized heterogeneous secure multi-cloud</strong> environment spanning across borders</td>
</tr>
<tr>
<td>Obj_4</td>
<td><strong>Knowledge-as-a-Service</strong> platform</td>
</tr>
<tr>
<td>Obj_5</td>
<td>Develop more than one <strong>application for validation</strong></td>
</tr>
<tr>
<td>Obj_6</td>
<td><strong>Validate</strong> the analytics methodology and determine the utility of the distributed knowledge-base through <strong>experiments</strong> in model <strong>smart cities</strong> and <strong>smart homes</strong></td>
</tr>
</tbody>
</table>
3.1 Requirements for data model components

Seven different data model components have been identified covering seven key aspects of the iKaaS architecture and Platform, namely Virtual Entities (for the representation of IoT infrastructure elements, further analysed in section 4.4 and D4.1 [89]), Services, Knowledge, User, Access Rights, City and Health. It should be noted that this set of data model components will be enhanced as work progresses towards the specification, implementation and validation of the Universal Data model. For example one of the next steps is to address the addition of resource description aspects to further increase the cloudification of the platform and enable taking service composition/migration etc. decisions, considering the practical capabilities and limitations of the underlying infrastructure. This section describes the set of different requirements that are associated with each of the iKaaS data model components specified so far.

3.1.1 City data model requirements

<table>
<thead>
<tr>
<th>Table 5: City data model requirements</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Description</th>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>C-1</td>
<td>Non-Functional</td>
<td>City data model should comply with an international standard, known as CityGML, for the sake of interoperability.</td>
<td>Obj_1</td>
</tr>
<tr>
<td>C-2</td>
<td>Functional</td>
<td>The data specification of CityGML should be extended to form a selected set of mandatory elements used in the project for sake of the satisfaction of city management under the rules of international standard.</td>
<td>Obj_1</td>
</tr>
<tr>
<td>C-3</td>
<td>Functional</td>
<td>Every element of town should be defined as an individual object.</td>
<td>Obj_1, Obj_5, Obj_6</td>
</tr>
<tr>
<td>C-4</td>
<td>Functional</td>
<td>Data should be able to update object by object for the sake of data</td>
<td>Obj_1</td>
</tr>
</tbody>
</table>
From the viewpoint of town management, it is necessary to define the life line infrastructure as managed objects. This among other implies the including underground objects such as gas pipes, water pipes and drainage system.

iKaaS project requests to equip a variety of sensors in town, each of which should be defined as an individual object with its location using the City data model as reference in the VE Model description.

City data model must provide all necessary geographic information so that other iKaaS data models can use it to identify the location on the earth.

### 3.1.2 Health data model requirements

**Table 6:** Health data model requirements

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Description</th>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>H-1</td>
<td>Non-Functional</td>
<td>Health data must be harmonizable with IEEE11073 and HL7CDA2 as described by Continua Health Alliance.</td>
<td>Obj_1</td>
</tr>
<tr>
<td>H-2</td>
<td>Functional</td>
<td>Health data should be classified based on data source and measurement method.</td>
<td>Obj_1</td>
</tr>
</tbody>
</table>
### 3.1.3 Access Rights data model requirements

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Description</th>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>A-1</td>
<td>Functional</td>
<td>A privacy certificate (defined in D3.1) must be required so that a security gateway (defined in D3.1) interprets the national regulation that governs data transfer for the country where the application exists.</td>
<td>Obj_1, Obj_2, Obj_3</td>
</tr>
<tr>
<td>A-2</td>
<td>Functional</td>
<td>A privacy certificate must be required so that a security gateway confirms the validity of an application.</td>
<td>Obj_1, Obj_2, Obj_3, Obj_5</td>
</tr>
<tr>
<td>A-3</td>
<td>Functional</td>
<td>A token must be required so that a security gateway confirms that an application is permitted to access data on a local cloud under the rules that govern data transfer both for the country where the local cloud sets up and the country where the application exists.</td>
<td>Obj_1, Obj_2, Obj_3</td>
</tr>
<tr>
<td>A-4</td>
<td>Non-Functional</td>
<td>The access rights model should support the association of the access rights with particular user roles.</td>
<td>Obj_1, Obj_2</td>
</tr>
</tbody>
</table>
A-5  Non-Functional  The access rights model should be associated with particular users that are authorized to use data from services and VEs directly.  Obj_1, Obj_2

A-6  Functional  The access rights model should include in the privacy certificate information on the country that supports the application.  Obj_1, Obj_2, Obj_3, Obj_5

A-7  Functional  The access rights model should include in the privacy certificate information about the authorized countries that have access to data.  Obj_1, Obj_2, Obj_3, Obj_5

A-8  Functional  The access rights model privacy certificate authorized countries must be correlated with particular data Ids lists.  Obj_1, Obj_3, Obj_5

3.1.4 Virtual Entity data model requirements

Table 8: Virtual Entity requirements

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Description</th>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>VE-1</td>
<td>Functional</td>
<td>A VE must provide a digital abstraction of a device.</td>
<td>Obj_1, Obj_3, Obj_4, Obj_5</td>
</tr>
<tr>
<td>VE-2</td>
<td>Functional</td>
<td>A VE must provide a digital abstraction of an object observed.</td>
<td>Obj_1, Obj_3, Obj_4, Obj_5, Obj_5</td>
</tr>
<tr>
<td>VE-3</td>
<td>Non-functional</td>
<td>Digital abstraction contains a unique identification of a VE.</td>
<td>Obj_1, Obj_4, Obj_5</td>
</tr>
<tr>
<td>VE-4</td>
<td>Non-functional</td>
<td>A VE must contain list of functionalities provided by</td>
<td>Obj_1, Obj_4, Obj_5</td>
</tr>
<tr>
<td>VE</td>
<td>Description</td>
<td>Object References</td>
<td></td>
</tr>
<tr>
<td>------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------</td>
<td></td>
</tr>
<tr>
<td>VE-5</td>
<td>A Digital abstraction includes context information (e.g. location).</td>
<td>Obj_1, Obj_4, Obj_5</td>
<td></td>
</tr>
<tr>
<td>VE-6</td>
<td>All literal information at the VE must be linked to information describing the meaning of it (meta data).</td>
<td>Obj_1, Obj_4, Obj_5</td>
<td></td>
</tr>
<tr>
<td>VE-7</td>
<td>Digital abstraction includes address information.</td>
<td>Obj_1, Obj_3, Obj_5</td>
<td></td>
</tr>
<tr>
<td>VE-8</td>
<td>The VE Digital abstraction includes description information that allow the discovery of them by using particular search criteria.</td>
<td>Obj_1, Obj_3, Obj_4, Obj_5</td>
<td></td>
</tr>
<tr>
<td>VE-9</td>
<td>The VE search description information is used for finding addresses of suitable VEs.</td>
<td>Obj_1, Obj_4, Obj_5</td>
<td></td>
</tr>
<tr>
<td>VE-10</td>
<td>A VE may add further status information about objects like where it is fixed (bedroom, garden ...), battery status, energy supply, maintenance duration, error profile ...), distinguish between unique entry and periodic entry sampling and sampling period (stored in the local data base).</td>
<td>Obj_1, Obj_3, Obj_4, Obj_5</td>
<td></td>
</tr>
<tr>
<td>VE-11</td>
<td>A VE should be able to</td>
<td>Obj_1, Obj_4,</td>
<td></td>
</tr>
<tr>
<td>VE-12</td>
<td>Functional</td>
<td>A VE linked to a mobile device should be able to acquire readings from all sensors of the mobile device to maximize usability of the mobile device’s capabilities in iKaaS services.</td>
<td><strong>Obj_1, Obj_4, Obj_5</strong></td>
</tr>
<tr>
<td>---------</td>
<td>--------------</td>
<td>---------------------------------------------------------------------------------------------------------------------------------</td>
<td>--------------------------</td>
</tr>
<tr>
<td>VE-13</td>
<td>Functional</td>
<td>A VE may support annotation of data collected from the associated device according to the iKaaS data model.</td>
<td><strong>Obj_1, Obj_4, Obj_5</strong></td>
</tr>
<tr>
<td>VE-14</td>
<td>Functional</td>
<td>A VE should provide information that can be beneficial for cloud optimization purposes, since VEs are the “end-points” of a Local Cloud.</td>
<td><strong>Obj_1, Obj_4, Obj_5</strong></td>
</tr>
<tr>
<td>VE-15</td>
<td>Functional</td>
<td>A VE should include information that will allow the dynamic composition and provision of complex services, supported by intelligent iKaaS platform mechanisms.</td>
<td><strong>Obj_1, Obj_3, Obj_5, Obj_6</strong></td>
</tr>
<tr>
<td>VE-16</td>
<td>Functional</td>
<td>A VE must include appropriated information with respect to its functional capabilities.</td>
<td><strong>Obj_1, Obj_3</strong></td>
</tr>
<tr>
<td>VE-17</td>
<td>Functional</td>
<td>A VE should provide appropriate information with respect to its</td>
<td><strong>Obj_1, Obj_3, Obj_5, Obj_6</strong></td>
</tr>
<tr>
<td>ID</td>
<td>Type</td>
<td>Description</td>
<td>Objective</td>
</tr>
<tr>
<td>-----</td>
<td>--------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-----------------</td>
</tr>
<tr>
<td>S-1</td>
<td>Functional</td>
<td>A Service must provide the abstract description of service as they are considered in the iKaaS platform context.</td>
<td>Obj_1, Obj_5</td>
</tr>
<tr>
<td>S-2</td>
<td>Functional</td>
<td>A Service must be characterized as the graph model node / class, which includes a set of particular features and constitutes the super class for the “Simple Service” and the “Complex Service” subclasses.</td>
<td>Obj_1</td>
</tr>
<tr>
<td>S-3</td>
<td>Functional</td>
<td>A “Simple Service” must support the representation of the class which inherits all features from super-class Service and it has as additional features the one-to-many (1...n) Virtual Entity Functions</td>
<td>Obj_1</td>
</tr>
<tr>
<td>S-4</td>
<td>Functional</td>
<td>A “Complex Service” must support the representation of the class which inherits all features from super-class Service and it has as additional features it composition of “Simple Services” complemented with a particular Service Logic.</td>
<td>Obj_1</td>
</tr>
<tr>
<td>S-5</td>
<td>Functional</td>
<td>A Service may support</td>
<td>Obj_1</td>
</tr>
<tr>
<td>S-6</td>
<td>Functional</td>
<td>A Service should provide information that can be beneficial for cloud optimization purposes, such as the Local cloud switch.</td>
<td>Obj_1, Obj_3, Obj_5</td>
</tr>
<tr>
<td>-----</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>S-7</td>
<td>Non-Functional</td>
<td>A Service must be complemented by one or more domains.</td>
<td>Obj_3, Obj_5</td>
</tr>
<tr>
<td>S-8</td>
<td>Non-Functional</td>
<td>A Service must be complemented by a physical or a logical location.</td>
<td>Obj_3, Obj_5</td>
</tr>
<tr>
<td>S-9</td>
<td>Non-Functional</td>
<td>A Service may be complemented by one or more particular parameters.</td>
<td>Obj_1, Obj_3, Obj_5</td>
</tr>
<tr>
<td>S-10</td>
<td>Non-Functional</td>
<td>A service should be associated with at least one user that is characterized as its consumer.</td>
<td>Obj_1, Obj_3, Obj_5, Obj_6</td>
</tr>
<tr>
<td>S-11</td>
<td>Non-Functional</td>
<td>All literal information at the Service level must be linked to information describing the meaning of it (metadata).</td>
<td>Obj_1</td>
</tr>
<tr>
<td>S-12</td>
<td>Functional</td>
<td>A “Complex Service” must execute particular service logic that describes specific actions that is triggered by particular conditions.</td>
<td>Obj_1, Obj_3</td>
</tr>
<tr>
<td>S-13</td>
<td>Functional</td>
<td>A Service should provide appropriate information that will facilitate the operation of the iKaaS platform mechanisms.</td>
<td>Obj_1, Obj_3</td>
</tr>
<tr>
<td>S-14</td>
<td>Functional</td>
<td>A Service must be complemented by appropriate information that refer to the domain that supports and/or it is able to</td>
<td>Obj_1, Obj_3</td>
</tr>
</tbody>
</table>
be deployed.

| S-15 | Non-Functional | A Service should support the provision of appropriate information that will allow its dynamic deployment and management in the context of multiple cloud environments. | Obj_1, Obj_3, Obj_5, Obj_6 |

### 3.1.6 Knowledge data model requirements

Table 10: Knowledge data model requirements

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Description</th>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>K-1</td>
<td>Functional</td>
<td>The knowledge data model must represent the knowledge data that will be generated in the context of the iKaaS platform.</td>
<td>Obj_1, Obj_4</td>
</tr>
<tr>
<td>K-2</td>
<td>Functional</td>
<td>The knowledge data model must support the different and heterogeneous datasets and algorithms that will be used for the generation of knowledge data.</td>
<td>Obj_3, Obj_4</td>
</tr>
<tr>
<td>K-3</td>
<td>Functional</td>
<td>The knowledge must be associated with at least one domain</td>
<td>Obj_1, Obj_4</td>
</tr>
<tr>
<td>K-3</td>
<td>Functional</td>
<td>The knowledge should be complemented by particular data that refer to the Services that are included in the context of the knowledge building.</td>
<td>Obj_1, Obj_4</td>
</tr>
<tr>
<td>K-5</td>
<td>Non-Functional</td>
<td>The knowledge should be complemented by particular user specific data that may refer either to the VE owners or Service consumers.</td>
<td>Obj_1, Obj_2, Obj_4</td>
</tr>
<tr>
<td>K-5</td>
<td>Functional</td>
<td>The knowledge must be associated with a particular dataset that represents the core part of the data that used for the knowledge generation.</td>
<td>Obj_1, Obj_4</td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
<td>-------------------------------------------------------------------------------------------------</td>
<td>--------------</td>
</tr>
<tr>
<td>K-7</td>
<td>Functional</td>
<td>The knowledge data model must support the abstraction of the concepts that can be part of the knowledge so as to achieve the creation of abstract and adaptive data structures based on the model.</td>
<td>Obj_1, Obj_3, Obj_4</td>
</tr>
<tr>
<td>K-8</td>
<td>Non-Functional</td>
<td>The Knowledge should be complemented by one or more domains.</td>
<td>Obj_3, Obj_4</td>
</tr>
<tr>
<td>K-9</td>
<td>Non-Functional</td>
<td>The Knowledge should be complemented by one or more conceptual objects.</td>
<td>Obj_4</td>
</tr>
<tr>
<td>K-10</td>
<td>Non-Functional</td>
<td>The Knowledge may be complemented by one or more particular conceptual object parameters.</td>
<td>Obj_4</td>
</tr>
<tr>
<td>K-11</td>
<td>Non-Functional</td>
<td>The knowledge should include appropriate information that will be able to be re-used into the context of multiple cloud environments as well as will be able to be exploited for different purposes, such as the combination of prototype complex service capabilities.</td>
<td>Obj_3, Obj_4, Obj_5, Obj_6</td>
</tr>
<tr>
<td>K-12</td>
<td>Non-functional</td>
<td>The knowledge data model should be flexible and extensible in order to allow the inclusion of parameters that were not originally considered.</td>
<td>Obj_1, Obj_4</td>
</tr>
</tbody>
</table>
### 3.1.7 User data model requirements

**Table 11: User data model requirements**

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Description</th>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>U-1</td>
<td>Functional</td>
<td>The user model must present the abstract representation of a user in the context of the iKaaS system.</td>
<td><strong>Obj_1</strong></td>
</tr>
<tr>
<td>U-2</td>
<td>Functional</td>
<td>The user must be associated with particular preferences, policies / access and/or usage rights with respect to the VEs and the Services.</td>
<td><strong>Obj_1, Obj_2, Obj_3</strong></td>
</tr>
<tr>
<td>U-3</td>
<td>Functional</td>
<td>The user should be classified by using particular user roles that will be associated with particular access rights by exploiting the iKaaS Access Rights models instantiations.</td>
<td><strong>Obj_1, Obj_2, Obj_3</strong></td>
</tr>
<tr>
<td>U-4</td>
<td>Functional</td>
<td>Users should be allowed to set access/usage rights for the resources they own either they are the VEs or some service.</td>
<td><strong>Obj_1, Obj_2, Obj_3</strong></td>
</tr>
<tr>
<td>U-5</td>
<td>Functional</td>
<td>The system should enable the representation, storing and retrieval of user preferences, requirements and constraints in user profiles.</td>
<td><strong>Obj_1, Obj_2, Obj_3, Obj_4</strong></td>
</tr>
<tr>
<td>U-6</td>
<td>Functional</td>
<td>The system should estimate the values of user profile inferred parameters, based on the set of observable parameters.</td>
<td><strong>Obj_3, Obj_4</strong></td>
</tr>
</tbody>
</table>
3.1.8 Use of data model components by iKaaS architecture components

This section describes requirements of referencing mechanisms between individual data models. Applications have to be compliant with the rules defined except cases where no dependency exists.

Table 12 shows major requirements on each individual data model. For a description of the iKaaS components one can refer to the iKaaS D4.1 deliverable [89].

Table 12: Overview of use of component data models in iKaaS architecture

<table>
<thead>
<tr>
<th>Component data Model</th>
<th>iKaaS component using the data model</th>
</tr>
</thead>
<tbody>
<tr>
<td>City (Geospatial) data model</td>
<td>Stored in “Local Cloud DB” in RDF format, used by Local and Global data processing. For the exchange of geospatial information, iKaaS extension of CityGML is used.</td>
</tr>
<tr>
<td>Health data model</td>
<td>Local Cloud DB, Global Knowledge DB, Local and Global data processing.</td>
</tr>
<tr>
<td>Access rights data model</td>
<td>Security gateway. Internal data model is not accessible. Only a set of APIs are open for other iKaaS components.</td>
</tr>
<tr>
<td>Virtual entity data model</td>
<td>Global Service Catalogue, Local Service Catalogue, Global Service Manager, Local</td>
</tr>
</tbody>
</table>
Service data model
Global Service Catalogue, Local Service Catalogue, Global Service Manager, Local Service Manager, Local and Global Data Processing.

Knowledge data model
Global Knowledge DB, Local DB, Global and Local Data Processing.

User data model
Global Knowledge DB, Local DB, Global and Local Data Processing.

### 3.2 Requirements for iKaaS Data processing and Knowledge Acquisition

**Table 13:** Requirements for Data processing and Knowledge acquisition

<table>
<thead>
<tr>
<th>ID</th>
<th>Type</th>
<th>Description</th>
<th>Objective</th>
</tr>
</thead>
<tbody>
<tr>
<td>DP-1</td>
<td>Functional</td>
<td>Data processing and Knowledge Acquisition mechanisms should enable estimation of potential future problems and prediction of future events.</td>
<td>Obj_4</td>
</tr>
<tr>
<td>DP-2</td>
<td>Functional</td>
<td>Data processing and Knowledge Acquisition mechanisms should be able to retrieve measurements produced by Virtual Entities.</td>
<td>Obj_4</td>
</tr>
<tr>
<td>DP-3</td>
<td>Functional</td>
<td>Data processing and Knowledge Acquisition mechanisms will derive knowledge from various data sources</td>
<td>Obj_4</td>
</tr>
<tr>
<td>DP-4</td>
<td>Functional</td>
<td>Data processing and Knowledge Acquisition mechanisms will be responsible for updating the knowledge information</td>
<td>Obj_4</td>
</tr>
<tr>
<td><strong>DP-5</strong></td>
<td>Functional</td>
<td>in the local DB/Global Knowledge DB.</td>
<td><strong>Obj_4, Obj_5</strong></td>
</tr>
<tr>
<td>---</td>
<td>---</td>
<td>---</td>
<td>---</td>
</tr>
<tr>
<td><strong>DP-6</strong></td>
<td>Functional</td>
<td>Data processing and Knowledge Acquisition mechanisms will provide endpoints accessible by applications to retrieve data (for complex services)</td>
<td><strong>Obj_3, Obj_4, Obj_5</strong></td>
</tr>
<tr>
<td><strong>DP-7</strong></td>
<td>Functional</td>
<td>Data processing and Knowledge Acquisition mechanisms should be able to detect anomalies on the fly, coming from various data sources.</td>
<td><strong>Obj_3, Obj_4, Obj_5</strong></td>
</tr>
<tr>
<td><strong>DP-8</strong></td>
<td>Functional</td>
<td>Knowledge should be built/acquired on efficiency, on existing VEs and Services and their availability (platform/system knowledge).</td>
<td><strong>Obj_3, Obj_4, Obj_5</strong></td>
</tr>
<tr>
<td><strong>DP-9</strong></td>
<td>Functional</td>
<td>Data processing and knowledge acquisition mechanisms should allow the building of knowledge related to decisions regarding the platform/system configuration and to exploit this knowledge in order to decrease the time required to reach a decision. This includes functionality to identify whether a problem/situation currently addressed is similar to an older one for which a solution is already available.</td>
<td><strong>Obj_3, Obj_4, Obj_5</strong></td>
</tr>
<tr>
<td>DP Code</td>
<td>Type</td>
<td>Description</td>
<td>Objects</td>
</tr>
<tr>
<td>---------</td>
<td>------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>DP-9</td>
<td>Functional</td>
<td>Data processing and Knowledge Acquisition mechanisms shall be able to aggregate and infer relevant context/situation parameters (e.g. time, location, environmental conditions, preference of users) to describe the current situation and anticipate changes to it.</td>
<td>Obj_1, Obj_3, Obj_4</td>
</tr>
<tr>
<td>DP-10</td>
<td>Functional</td>
<td>Data processing and Knowledge Acquisition mechanisms must be able to “carry-on” with minimal features rather than “hang” if devices/VEs are disconnected.</td>
<td>Obj_1, Obj_4</td>
</tr>
<tr>
<td>DP-11</td>
<td>Non-Functional</td>
<td>Data processing and Knowledge Acquisition mechanisms shall respect the ownership of devices, VEs and data propagated by devices and VEs.</td>
<td>Obj_1, Obj_2, Obj_3, Obj_4</td>
</tr>
<tr>
<td>DP-12</td>
<td>Non-Functional</td>
<td>Data processing mechanisms must be capable of real-time, low-latency data processing.</td>
<td>Obj_4, Obj_5, Obj_6</td>
</tr>
<tr>
<td>DP-13</td>
<td>Non-Functional</td>
<td>Data processing and Knowledge Acquisition mechanisms must enable the notion of Knowledge as a Service (KaaS), which can be exploited by various situation/context aware applications/stakeholders, taking into account privacy and security restrictions (i.e. not all data and derived</td>
<td>Obj_1, Obj_4</td>
</tr>
<tr>
<td>DP-14</td>
<td>Non-Functional</td>
<td>It should be possible to exploit Data processing and Knowledge Acquisition mechanisms across different administrative and business domains.</td>
<td>Obj_3, Obj_5, Obj_6</td>
</tr>
<tr>
<td>-------</td>
<td>----------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>---------------------</td>
</tr>
<tr>
<td>DP-15</td>
<td>Non-Functional</td>
<td>Data processing and Knowledge Acquisition mechanisms may evaluate the quality information of the data received and determine whether data quality requirements are satisfied, so as to further process data for knowledge building and storage.</td>
<td>Obj_4</td>
</tr>
</tbody>
</table>
4 iKaaS Data Model components

In this section, the structure of individual data model components shall be described. Focusing on the investigation results arisen in the corresponding section 2, there is a set of different models that have been introduced either as Semantic Web standards, or as validated research work results in the context of various projects. Such models, can be used either as part of the abstract iKaaS model, or features of them can be included into generic semantic data containers of the iKaaS data model. Following this approach, iKaaS aims to achieve the development of a fully aligned, with the Semantic Web Linked Data principles [49]-[51], integrated data model that will satisfy multiple requirements of different data models. In the context of the above, as already introduced, seven (7) different Semantic Data Model components have been developed by covering seven(7) key aspects of the iKaaS architecture and Platform, namely Virtual Entities, Services, Knowledge, User, Access Rights, City and Health. These are presented and described in detail in the following sub-sections.

The table below presents the different identified semantic data models complemented with a short description of the innovation/differentiation that is introduced in the context of the iKaaS project.

Table 14: Semantic data model innovation

<table>
<thead>
<tr>
<th>Model</th>
<th>Innovation</th>
</tr>
</thead>
<tbody>
<tr>
<td>City Model</td>
<td>The City data model aims to introduce an innovative integrated data model for the description of the smart city concepts focusing on the contextual information, the description of location, places, buildings, etc, based on the CityGML standard. The main innovation introduced by the iKaaS City model, is the extension of the City Object data model description concepts for the application on smart cities supported by distributed multi-cloud IoT environments.</td>
</tr>
<tr>
<td>Model</td>
<td>Description</td>
</tr>
<tr>
<td>-----------</td>
<td>-------------</td>
</tr>
<tr>
<td>Health Model</td>
<td>No health data model advances are made within the iKaaS project. The presented standards and guidelines represent the current state-of-the-art, and future standards (eg. HL7 FHIR) will likely be harmonized with the IEEE11073PHD data model. There is no need for an additional health data model for iKaaS use cases. However, combining this health data model with environmental modelling can be considered an iKaaS advance.</td>
</tr>
<tr>
<td>Service Model</td>
<td>The service data model introduces an integrated semantic data model for the service management in the Cloud-IoT based IoT environments. In particular the iKaaS service in comparison to other service models that exist so far, introduces the unique feature of the multiple-data types composition for the service provision, migration and the service-related knowledge building in contemporary Cloud-IoT ecosystems. Specifically, it allows for the inclusion of data of a wide nature from the application level to the device level, considering all of them as factors for the service creation, management, as well as for the knowledge building that may refer to the service manipulation considering various data types (e.g. user preferences, service logic, device communication capabilities).</td>
</tr>
<tr>
<td>VE Model</td>
<td>While various existing models and modelling languages, such as the SensorML focus on the data models for the description of the IoT sensing environments, the VE Model introduces the innovative feature of the IoT device abstract semantic model so as to be exploited by innovative autonomous mechanisms that support the decision making on dynamic service creation. Essentially the VE Model is introduced here as a model that can make reference to other existing ontologies for the device aspects (e.g. the SSN ontology), introducing in the same time features that are important for the dynamic service provisioning in the multi-Cloud environments (e.g. the VE Function Features that</td>
</tr>
</tbody>
</table>
refer to some type of indicators that describe the function capabilities). Moreover, this model is associated with the feature of its re-usability in the context of diverse and different description concepts, including both sensing and actuation IoT environments, having the capability to respond to the contemporary requirements of multi-cloud IoT environments. Essentially, the VE model can reuse the existing ontologies for the description of heterogeneous devices that have been described in the context of totally different IoT domains, so as to create a structured model that cannot be limited by the device type, since it can exploit existing ontologies or in case of lack of ontology/description modules it can be applied so as to describe a new entity from scratch without having the limitation of whether it is a sensor or an actuator.

<table>
<thead>
<tr>
<th>Access Rights Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>This model aims to address a variety of data and/or existing ontologies that are associated with the security aspects of the Cloud-based IoT systems. Through the exploitation of the access rights model it will be possible to develop unified security policies and rules, which will allow control of the access on data, the mechanism management, etc. One of the innovative key features of the models is associated with its capability to be associated directly and/or indirectly with different representation concepts, such as the user, the VE, the services.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>User Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>The user data model aims to introduce the innovative feature of the user data exploitation in two critical mechanisms: a) cloud service management and b) cloud-based IoT environments knowledge building. In particular, the iKaaS user model extends existing data models for the user data modelling such as the iCore [41] user data model. The basic difference that makes the model innovative refers to the correlation of user preferences with knowledge generation processes for the intelligent service management, as well as</td>
</tr>
</tbody>
</table>
the exploitation of the use profile data and preferences for the management of services in terms of their migration across multi-cloud environments.

| Knowledge Model | This model constitutes the first integrated approach as it has been described in the literature so far, that tries to integrate abstract conceptual features for the design of a generic knowledge model that will represent the knowledge in a unified structure. The real innovation of this approach lies in the fact of the advanced knowledge building and machine learning mechanisms development that will enable the realization of the Knowledge-as-a-Service platform. Based on the knowledge model, iKaaS will be able to represent in a more structured way the knowledge data. |

### 4.1 City data model

The City data model defines knowledge related to geospatial features, geometry of features for UI of graphical representation for applications, and mechanism of geospatial referencing for other data models. Overall instances or part of model concepts can be used so as to describe data related information for the VE Model that includes the location feature.

The structure of the data model is based on the well-defined XML based geospatial data model called CityGML published by OGC: Open Geospatial Consortium. The data model has been modified for application with necessary objects, and defined extended CityGML specification which is called ADE: Application Domain Extensions. An overview of the city data model structure is shown in Figure 8.
A Mechanism for geospatial referencing is based on existing ISO standard spatial referencing by coordinates, spatial referencing by geographic identifier, and other concrete standards. City data model provides geospatial attributes to referencing data models through the mechanisms. Other data model shall define linking attributes to root class of the city data model.
City data model as a part of iKaaS data model shall be converted from CityGML to RDF to be enhanced with additional semantic information.

Geometry of the city data model can be used for representation of 3dimensional geospatial objects for applications such as a virtual reality application package called Unity.

4.2 Health data model

The Health data model defines the outputs of various sensors related to the user’s health and wellbeing. In the healthcare domain typically only a certain section of a larger model is implemented on a case-by-case basis.

Health data should be modelled in accordance with the 11073 PHD DIM [29] (Figure 9). Additional associations between the medical sensor device (MDS) and user; and measured metric and user; should be defined, with user information containing the information defined in Patient Identification Segment of the PCD-01 transaction HL7 message[90].

![Figure 9: Relationships of the medical device system and measurement systems in 11073 PHD, with an added user/patient. The part relevant to iKaaS is outlined with red.](image)

4.3 Access Rights data model

Figure 10 depicts the Access Rights data model that can be used in the context of iKaaS platform so as to model data with respect to the access rights. There are two core concepts that are associated with
the access rights root node, a) the “Privacy Certificate” and the “User Role”.

Figure 10: Access Right data model

Each local cloud has the security gateway at the connection point with the global cloud. The queries from applications and the data on the local cloud are all exchanged through the security gateway. Before requesting the local cloud for data, the application is required to be issued a privacy certificate by the privacy certificate authority (CA) of the same country, and then must obtain a token from the security gateway.

Table 15 includes the description of the corresponding parameters that are listed into the Privacy Certificate, as well as the description of the iKaaS user association with the access rights model, through the user role.

Table 15: Access Rights description

<table>
<thead>
<tr>
<th>Access Rights Model concepts</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Privacy Certificate</td>
<td>The privacy certificate makes it possible for the security gateway to interpret the rules in the country where the application exists. The privacy CA is built for each country and creates the privacy certificate on the basis of the</td>
</tr>
</tbody>
</table>
national regulations that prevail in that country and information concerning the application. The privacy certificate has associations with the CA Country, with one or more LC Countries as well as with a particular application. In addition, it includes the expiration date property that refers to the expiry date of the privacy certificate and the signature that is generated with the private key of the privacy CA. The public key is distributed to security gateways.

### CA Country
The name of the country where the privacy CA is established. It also refers to the name of the country where the application exists.

### LC Countries
The names of the countries that the application is permitted to access.

### LC Data IDs
The identifiers indicate the types of data that the application is permitted to access. The values are nested in each value of LC Countries.

### Application
This concept includes all appropriate properties that refer to the application, by including: a) Application IP that refers to the IP address of the application, b) Application ID that define the type of service that the application provides and c) the Application PK that is the public key of the application.

### User Role
This entity defines the role of the user in the system. In particular, the user can have different roles into the iKaaS system that are associated with particular access rights. Indicatively, some roles for the users that could be considered could be “owner”, “consumer”, “administrator”, etc.

Further to the above information that are associated with the Access Rights model, the role of the **Issuance of Token function** should be highlighted, which actually exploits the above information through an appropriate API. In particular, this function is called when the application obtains the token to access local cloud DBs. The application is then required to specify the data IDs that it wants to have access to and to send the privacy certificate. When issuing a token, the security gateway refers to the privacy certificate in order
to comply with the rules of the country where the application exists. A *token* is transmitted to the application along with its *expiry date* after being encrypted with the value of the parameter *Application PK* listed on the privacy certificate. The application is required to decrypt the token with its own private key. When the application transmits the query to the local cloud DBs through the security gateway, the token is treated as a common key of a message authentication code (MAC).

### 4.4 Virtual Entity data model

A Virtual Entity (VE) is the virtual abstract representation of a Real World Object (RWO) with Information and Communication Capabilities (ICT), a device such as *sensor, actuator, Smartphone, etc.*, that in turn is associated to one or more non-ICT Objects, such as *buildings, places, persons, etc.*

As already mentioned a VE represents an **ICT Object** and is owned by one VE owner. The VE may have one or more VE Parameters that refer on specific information regarding the VE. Furthermore, a VE represents the Functionality that is offered by ICT object. In particular, the VE is associated with **VE Functions** that, in turn, have specific Input and Output parameters, whilst are described in terms of **VE Function Features**, such as **Utilities** (*add positive meaning on the function*) and **Costs** (*add negative meaning on the function*) as well as it has **Access Rights** and **Billing Costs**. Moreover, the virtual abstract representation includes information for the further description of ICT and non-ICT Objects. Such information is classified in **Parameters** and **Location** parameters of objects. An ICT Parameter can include information about the specifications of ICT object and other necessary data regarding the ICT. The parameters that are associated with an ICT object describe essentially the specific features that characterize the ICT Object. For instance, in case where we have as an ICT Object, a sensor, a potential ICT Parameter could be the range or the accuracy of sensor. On the other hand, the objects, which belong in the real world, have a physical location that is described in terms of geographical coordinates through the Geo Location parameters. At this point it should be highlighted that an ICT and non-ICT can have the same or different Geo Location. A typical example for this situation arises when we have as ICT Object, a *camera* that observes a *building* that is some meter far away of it. In
this case the [ICT_Object = Camera] hasICTLocation “X” and the [non-ICT_Object = <citygml:Building>] hasNonICTLocation “Y”.

The information that is described above can be clearly readable and understandable by humans (human readable and understandable data) [91] but it is quite difficult to be readable and understandable by machines (machine readable and understandable data) [91]. In order to allow the machines to be able to understand the meaning of data and consequently to infer conclusions on them, semantic enrichment of data should take place. In addition through the semantic enrichment of data, the semantic interoperability between heterogeneous entities in the iCore system is enabled. A possible and an efficient way to achieve this, is through the use of ontologies that belong to semantic web technologies [92][93].

**Figure 11**: iKaaS Virtual Entity Model

The Figure 11 depicts the iKaaS VE Model that aims to be used as a cross-platform virtualization model for the semantic description of the VEs that can be included in different services.

The above model may refer to a particular instance of measurement data that are associated with the output parameter of a particular VE Function that perform sensing measurements, such as a pollution sensor, a temperature sensor, etc. The instance of the measured data
can be based on the graph representation that is depicted in Figure 12. The "hasMeasurement" property that is included into the "Output parameter" concept could be associated (in case of a sensor) with a particular measurement data instance. Through the above association, the output parameter is associated with measured data, that are complemented with particular annotations such as the measurement unit, measurement accuracy, etc. Further to that, the measured data will be correlated with a particular location that is actually specified through the "Location" concept that is included in the VE Model. The Location can be either statically defined during the registration of the VE (into the context of a simple service) in case we refer to a static sensor that is installed in some building for instance, or it can be continuously updated in case we refer to a mobile entity such as the VE ‘Buses’, or the wearable devices that are continuously changing their location and taking at the same time data measurements from various environmental sensors, such as pollen and pollution sensors. The classification between a static VE and a mobile VE, in terms of its location changing, can be highlighted by the "isMobile" that is included as data property into the Virtual Entity concept in the VE model.

The structured measured data can be stored into Local Cloud DBs or it can be directly streamed from the VE to the third-party entities that consume them, such as the data processing component.

![Figure 12: VE produced data measurements structure](image-url)
"measurement": {
  "hasID": "@id",
  "hasName": "TemperatureMeasurement",
  "hasMeasurementProperty": {
    "hasURI": "@id/properties/{property_id}",
    "measurementProperty": [
      {
        "type": "measurementData",
        "value": 25
      },
      {
        "type": "measurementDataType",
        "value": "Integer"
      },
      {
        "type": "measurementUnit",
        "value": "Celsius"
      },
      {
        "type": "timestamp",
        "value": "2015-10-13T10:30:00"
      },
      {
        "type": "location",
        "value": "@{ve_url}/location"
      }
    ]
  }
}

**Figure 13:** VE produced data measurements structure - JSON Indicative example

![Ontology Visualization](image.png)

**Figure 14:** iKaaS VE Model Ontology Visualization
Taking into account the overview of the VE Model concepts, given above, as well as by following the structured information as it is depicted in the high level design of the VE Model, an OWL ontology arises that is depicted in Figure 14, as an ontology visualization graph using Protege tool [94]. This OWL ontology is defined into RDF/XML format and it will be uploaded in an online space so as to be publicly accessible through its own unique Persistent Uniform Resource Locator (PURL) [95].

### 4.5 Service data model

Figure 15 depicts the high level design of the iKaaS Service Model, based on the modelling requirements that have been identified in the corresponding section. Specific application and evaluation of the model has been performed in the context of the Ambient Assisted Living use case. Furthermore, the model has been designed in a way that allows its application in different concepts and use cases. Consequently, the iKaaS Service Model constitutes an abstract Semantic Model that can cover the modelling requirements for heterogeneous types of services in the context of the iKaaS platform.

![Figure 15: iKaaS Service Model](image)

As can be observed, the model is comprised by one super-class named ‘Service’, complemented with its first level concepts and properties, as well as by two sub-classes the ‘Simple Service’ and the ‘Complex Service’ that inherit the features of the super-class ‘Service’ and they are complemented with some additional specific features.
The tables below provide the description of the three core classes of the model, complemented by the description of their corresponding features. In particular, Figure 16 presents the description of the super class service concepts / nodes (aka meta-data containers), the description of the Simple Service nodes and the description of the Complex Service nodes.

**Table 16:** Service description

<table>
<thead>
<tr>
<th>Service Concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Service</td>
<td>The root concept of the model that refers to the service entity that is identified by a unique URI, has been created in a specific time that can be described by using an existing Time ontology (such as the one in [96]), and it belongs to a specific type as it is identified in the iKaaS platform, namely Simple or Complex Service. The Service node has three different associations that links this node to the other three nodes (described in the following). The ‘Domain’ concept presents the domain, on which the Service has been built so as to support specific requirements, as well as to represent particular domain features. Each domain can be described in terms of its type (e.g.: Smart Home, Smart City, etc.), a name and a textual description in natural language.</td>
</tr>
<tr>
<td>Domain</td>
<td>Instance of User Model</td>
</tr>
</tbody>
</table>
This concept includes meta-data with respect to the location on which the particular service was requested (e.g. a service for a neighbourhood somewhere in as Smart City, or the Location of a Smart Home, etc.).

This concept works as the meta-data container for the description of extra features either for the service or for the Domain, so as to allow the introduction of additional properties and features that are not included as first level properties into the service model as well as in order to allow the introduction of external, existing ontologies features. Each parameter can be described by a particular name in textual format as well as can include many feature sets as key-pair values for the type – value properties that are included into the concept.

This concept describes all appropriate data that refer to the user that consumes the service. As meta-data, container can include additional external features from existing ontologies so as to allow the extension of specific features for the user class. The latest is allowed by introducing the key-pair sets type-value for the introduction of external references to existing ontologies.

**Figure 17:** Simple and Complex service concepts / components analysis
**Table 17: Simple Service description**

<table>
<thead>
<tr>
<th>Simple Service Concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple Service</td>
<td>This concept essentially constitutes the instantiation of the root node for the simple service that is complemented by a unique identifier, expressed as URI. Each simple service is associated with one or more Virtual Entity functions that actually support the service functionality.</td>
</tr>
</tbody>
</table>

| Virtual Entity Function | This concept includes a set of different URIs that link to different VE Functions that compose the Simple Service. Each URI leads indirectly, through the reverse association of ‘offersFunction’ named ‘isOfferedBy’ to a particular instance of a VE that is represented by the VE Model. |

**Table 18: Complex Service description**

<table>
<thead>
<tr>
<th>Complex Service Concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Complex Service</td>
<td>The ‘Complex Service’ concept essentially constitutes the instantiation of the root node for the complex service that is complemented by a unique identifier, expressed as URI. Each complex service is associated with more than one Simple Services that actually compose its features to provide more complex capabilities.</td>
</tr>
</tbody>
</table>

| Service Logic           | The ‘Service Logic’ concept refers to the workflow of the complex service. The service logic is associated with complex service by a specific property that is referred as ‘executesServiceLogic’. The service logic is combined by different compound conditions and compound actions. |

| Compound Conditions     | This concept describes the sets of the conditions that are included into the service logic and are associated one-to-one with a particular action. |

<p>| Compound Actions        | The ‘Compound Actions’ concept includes the sets of the different Actions that can be included into the service logic and can be triggered by one and only one particular condition. |</p>
<table>
<thead>
<tr>
<th>Concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Condition</td>
<td>The ‘Condition’ concept is uniquely identified by a URI and includes the key-pair feature for the type-value association of particular properties that can be used for the description of the condition. In general terms, the types of the condition could refer to a particular class (e.g. VE_Function), to a particular logical expression and to a particular condition filter, such as a threshold. Indicatively, through the combination of the condition type-value properties could arise an expression for the condition as the following: ‘&lt;&lt;VE_Function&gt;&gt; : Sensor_Measurement &gt; Threshold’</td>
</tr>
<tr>
<td>Action</td>
<td>The ‘Action’ concept is uniquely identified by a URI and includes the key-pair feature for the type-value association of particular properties that can be used for the description of the action. In general terms, the types of the action could refer to a particular class (e.g. VE_Function), to a particular class property and to a particular action filter, such as a state action. Indicatively, through the combination of the action type-value properties could arise an expression for the action as the following: ‘&lt;&lt;VE_Function&gt;&gt; : State = OFF’</td>
</tr>
</tbody>
</table>

Through the definition of the above concepts, with their included properties, an OWL ontology is derived that is depicted in Figure 18, as an ontology visualization graph using Protege tool [94]. This OWL ontology is defined in RDF/XML format and it will be uploaded in an online space so as to be publicly accessible through its own unique Persistent Uniform Resource Locator (PURL) [95].
4.6 Knowledge data model

This section presents and describes the iKaaS Knowledge data model, and gives an indicative example of the instantiation of the Knowledge data model. In particular, Figure 19 presents the high-level overview of the graph-based representation of the Knowledge data model that can be used for the development of the corresponding Ontology for the description of its concepts and their relations, as well as it can be used as the base for the deployment of RDF Graph representations for the Knowledge representation.

Figure 19: iKaaS Knowledge data model
The structured data as RDF Graphs that actually correspond to knowledge representations, can be stored in RDF Stores. In the iKaaS platform, based on the first implementation approach, the RDF Store that will store the RDF Knowledge representation data, will be the Global Knowledge DB and the Local Cloud DB. Table 19 presents the description of the corresponding parts that are involved in the Knowledge data model.

Table 19: Knowledge data model description

<table>
<thead>
<tr>
<th>Complex Service Concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Knowledge Root node</td>
<td>The ‘Knowledge Root Node’ presents the root node of the iKaaS Knowledge data model that points to a domain specific Knowledge Model, such as the Smart Home or the Smart City Knowledge data model</td>
</tr>
<tr>
<td>Domain</td>
<td>The ‘Domain’ node presents the domain to which the Knowledge data model refers and describes its concepts. Each domain can be described in terms of its type (e.g.: Smart Home, Smart City, etc.), a name and a textual description in natural language. Each domain may include one or more parameters, the ‘Domain Parameters’, which may describe domain specific concepts required for specific situations, such Date &amp; Time, Indoor Location, Status of a process and/or situation, etc.</td>
</tr>
<tr>
<td>Knowledge Conceptual Object</td>
<td>The ‘Knowledge Conceptual Object’ concept can be used so as to present the potential Knowledge concepts that can be involved in a specific domain. This concept can be considered as an abstract meta-data container that can include any data-type/concept of external / third-party ontologies and/or data model. In a general approach, it could be considered as the node that can describe the components, which are included in a specific domain (e.g. Smart Home: [Sensors &amp; Actuators, Persons, Places, etc.], Smart City: [Places, Mobile Devices, QR Codes, etc.])</td>
</tr>
<tr>
<td>Concept</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Object Feature</td>
<td>The 'Object Feature' concept works as assistant node to the 'Knowledge Conceptual Object' concept and it is considered as part of the Knowledge data model, in order to support the capability for further description of a Knowledge concept, where it is required. Indicatively, the object feature could include information about the available devices that are represented by particular VEs that are indirectly associated with a simple service through the association between the simple service and the VE Functions of the VEs that belong in the corresponding domain that the knowledge data is referred.</td>
</tr>
<tr>
<td>Dataset</td>
<td>This concept refers to the dataset that has been used as part of data for the knowledge generation. The dataset can refer to various different datasets instances, such as data measurements, actuator workflows, application data (e.g. weather data, traffic data, etc.).</td>
</tr>
<tr>
<td>Instance of Service model</td>
<td>This concept refers to the instantiation of a particular service as it is considered in iKaaS and instantiated based on the service model. Actually, it can describe either a domain specific or a generic (multi-applied/re-usable) service, which is developed under a specific concept so as to serve specific purposes. For instance, in a Smart Home, a domain specific service refers to the Smart Health Monitoring, which actually can be re-used outside of the initial concept. The service is described in terms of a URI as its unique identifier, a name and textual description in natural language, while it includes a link, as a URI, to the Simple/Complex Service that has been used for its instantiation.</td>
</tr>
<tr>
<td>Instance of user model</td>
<td>This concept refers to the instantiation of a user model that actually may represent the user that uses the corresponding service that is part of the domain, which belong to the knowledge data as an instantiation of the knowledge data model.</td>
</tr>
</tbody>
</table>

Through the definition of the above concepts, with their included properties, an OWL ontology is derived that is depicted in Figure 20, as an ontology visualization graph using Protege tool [94]. This OWL
ontology is defined in RDF/XML format and it will be uploaded in an online space so as to be publicly accessible through its own unique Persistent Uniform Resource Locator (PURL) [95].

**Figure 20:** iKaaS Knowledge data model Ontology Visualization

Further to the above, Figure 21 presents an indicative instantiation of the iKaaS Knowledge data model as it is applied in the context of a Smart Home Local Cloud Proof of Concept prototype (Ambient Assisted Living Use Case). For visualization purposes the Knowledge data model instance has been designed as a UML diagram [97] that except for the main concepts that are included in the model, includes, as enumerations, various indicative entities and/or raw data-type that can be part of each knowledge entity, such as the Knowledge concept, which in this case can be a device (e.g. sensors, actuator) or a person, etc. Consequently, taking into account the Knowledge data model instantiation sample it can be even better observed that the iKaaS Knowledge data model allows the easy integration of external / third-party models or model entities with zero adaptation. Namely, it is not required to create a new ontology entity or vocabulary concept that would represent a Knowledge concept of the type device, person, etc., but the only requirement corresponds to the need of a simple reference of required concept into an external ontology, such as the SSN ontology [98] for sensor devices.
Section 5.1 provides a formal definition of knowledge and presents how this can be formulated in indicative examples in the scope of the iKaaS use cases. As presented in section 5.1, knowledge types can roughly fall under the following categories: observations, history and knowledge, the latter being derived with the application of a certain method (e.g. time series forecasting, Bayesian statistics, etc.). It should be noted that the observation, history and knowledge data can be mapped to the Dataset node of the Knowledge data model, whereas the equations applied to derive the knowledge can be mapped to the Knowledge conceptual object. The knowledge data model as well as the formulation of knowledge and corresponding data processing and knowledge acquisition mechanisms are work in progress and will be further enhanced as the project progresses.

**4.7 User data model**

The User Model is a general abstract representation of users that includes a set of different concepts/entities. These concepts/entities are defined in a way that gives the ability for an abstract representation of user data. These are defined so as to cover the need for modelling of 'static information' as well as 'dynamic and not specific information' that can be related with the user and can refer to
different concepts. Thus, the User Model has to provide an abstraction of user data and it works as a high-level abstraction model.

The users in general can have different sets of data that represent their properties. In particular, static information that is denoted as User Characteristics can present some standard characteristics of the user such as name, surname, etc. Concepts from Friend-of-a-Friend (FOAF) ontology could be very useful for the description of User Characteristics. FOAF provides a full updated vocabulary that allows the representation of machine-readable data about people, SW agents, Social groups, etc., in the Web. Moreover, it is assumed that the user may have one or more different User Profiles that in turn are linked with different User Preferences and User Policies. In addition, as is described above, a user can have various and diverse User Roles such as the owner of a virtual entity, the consumer of a service, etc.

Depending on the User Role, the associations between the entities, in the User Model, can be differentiated because each role has different properties and associations. A user in role of an ‘Observed Entity’ cannot have an association with Service entity as the consumer of a Service. On the other hand, the User Policies and User Preferences that are defined by a ‘Service Request’ can be totally different from those that are defined by an ‘Observed User’. For instance a ‘Service Requester’ can provide a set of preferences and policies for the customization of a requested application, whilst an ‘Observed User’ can define preferences and policies that will be used to customize its observation of Situation Observers. However, it should be noted that there may be cases in which a user has both roles at the same time.

Figure 22 depicts the representation of the User Model diagram, comprising entities associations and their meta-data. In addition, Table 20 presents some relevant information with respect to the main entities that can be included in the User Model.
**Figure 22:** User Model Entities Diagram

**Table 20:** User model concepts description

<table>
<thead>
<tr>
<th>User Model concept</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>The root node of the User Model that refers to the User that can be either a Human or a Software Agent.</td>
</tr>
<tr>
<td>User Role</td>
<td>This entity defines the role of the user in the system. In particular, the user can have different roles into the iKaaS system that are associated with particular access rights instances that are based on the access rights models.</td>
</tr>
<tr>
<td>User Characteristic</td>
<td>An entity that comprises static information regarding the User. Such types of information can refer to the name, surname, age, gender, etc., which constitute standard and stable characteristics for the user independently of his/her profile. (e.g.: [Name = ‘Sample_Name’, Surname = ‘Sample_Surname’, Age=‘30’, Gender = ‘Female’, ....] &amp; ([Profile_ID = ‘ID_x’, Profile_Type = ‘Smart Office’) or [Profile_ID = ‘ID_y’, Profile_Type = ‘Smart Home’]).</td>
</tr>
<tr>
<td>User Profile</td>
<td>This entity defines different instantiations of user profiles that can be associated with a specific user. Each distinct user profile is unique for a user.</td>
</tr>
</tbody>
</table>
and is linked with that user through a unique identifier.

<table>
<thead>
<tr>
<th>User Preference</th>
<th>This entity is used so as to allow the expression of user preferences that relate with user activity in the iCore system. In particular, the user preferences in case of:</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>- a ‘Service Requester’ can define specific properties for the customization of Services (e.g.: ‘Send me notifications about the traffic status in the Area_X’ or ‘GUI Language = English’)</td>
</tr>
<tr>
<td></td>
<td>- an ‘Observed User’ can express specific properties for the customization of the observation, (e.g.: ‘change my profile if a set of properties are satisfied’).</td>
</tr>
</tbody>
</table>

Alternative implementations of the user model can be introduced by various existing ontologies, by making reference to their concepts and properties as different key-value pairs complemented with type-value data, where the type refers to a specific property type and the value to the corresponding value of the selected property type.
5 Data processing and Knowledge Acquisition over multi-cloud environments

5.1 Knowledge in iKaaS applications/use cases

5.1.1 Knowledge definition

It is assumed that knowledge consists of rules and facts. There should be at least two views for knowledge definition, depending on the treatment of “fact”, as depicted in Table 21.

Table 21: Views for knowledge definition

<table>
<thead>
<tr>
<th>View</th>
<th>Implementation of fact</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fact is a data</td>
<td>Facts are represented as a part of knowledge in cloud DBs.</td>
</tr>
<tr>
<td>Fact is not a data</td>
<td>Facts are implemented as a sequence of procedures which would work on “data” stored in cloud DBs.</td>
</tr>
</tbody>
</table>

5.1.2 Form of knowledge

Pollen density map in Madrid use case

Knowledge types

There might be two types of knowledge:

1. Observation
   - \( \{<m_i, t_i, x_i, y_i, z_i> \mid i = 1, 2, 3, \ldots \} \)
     
   Where
   - \( m_i \) is a measurement of pollen density at time, \( t_i \), and location, \( <x_i, y_i, z_i> \).
   - \( \{ <W_i, t_i, x_i, y_i, z_i> \mid i = 1, 2, 3, \ldots \} \)
     
   Where \( W_i = <S_i, D_i, R_i> \),
   - \( S_i, D_i, R_i \) are, respectively, a measurement of wind strength, wind direction and an amount of rainfall at time, \( t_i \), and location, \( <x_i, y_i, z_i> \).

2. Knowledge
   - The pollen density is derived with the use of a diffusion equation with drift term.
As mentioned in sub-section 4.6 Observations and derived Knowledge data can be mapped to the Dataset node of the Knowledge data model.

From a limited number of observations (1), pollen density is calculated at every place in Madrid. The pollen density map is created overlapping the calculation result on top of Madrid city map.

Let $K_m$ be a set of knowledge in Madrid system.

- (1) observation is a part of $K_m$
- (2) knowledge is a part of $K_m$

The calculation result is an assumption derived from (1) observation & (2) knowledge (diffusion equation). The derived assumptions are also assumed to be a part of $K_m$.

**Knowledge format**

(1) Observation will be a set of key-value pairs.
(2) Knowledge will be the output of the application of a set of equations.

The pollen map is obtained by overlaying the assumption (the calculated pollen density in Madrid City) onto the map of Madrid City (Geospatial knowledge).

(3) Assumptions are a set of key-value pairs.
(4) Map of Madrid City is a geospatial knowledge (RDF)

In a broader sense, the pollen map will be regarded as a part of knowledge in $K_m$. In addition:

- An assumption, derived from knowledge $K$, is also a part of $K$
- Any combination (mashup) of knowledge, the pieces of which are a part of knowledge $K$, is also a part of knowledge $K$.

**Knowledge acquisition**

The pollen density is derived with the use of the diffusion equation. The diffusion equation may not predict a correct pollen density, since the calculated wind flow might be different from the actual flow due to the viscosity or disturbance created geospatial objects. It would be possible to define a knowledge acquisition problem such that
**Input**: Pollen density observation, wind strength, wind direction, amount of rainfall.

**Output**: Pollen density of all the areas of Madrid.

**Process**: Pollen density will be calculated by diffusion equation, but the wind flow path is not straight as calculated due to the geospatial objects and the rainfall will cause a decrease of pollen density. It would be possible to reduce the measurement errors by modifying the additional terms of diffusion equation. This will become a knowledge acquisition process.

Given that bus terminals as well as buses are equipped with several pollen sensors it is possible to obtain a dataset for this learning problem.

**Town management in Tago-nishi**

**Knowledge types**

In the town management service at Tago-nishi, the prediction of rain/snow fall is under consideration. There might be three types of knowledge:

1. **Observation**
   - \{<r_i, t_i, x_i, y_i, z_i>| i = 1, 2, 3, ...\}
   - Where \( r_i \) is a measurement of rainfall at time, \( t_i \), and location, \( <x_i, y_i, z_i> \).

2. **History**
   - \{<R_j, t_j, x_j, y_j, z_j>| j = 1, 2, 3, ...\}
   - Where \( R_j \) is a historical measurement data of rainfall at time, \( t_j \), and location, \( <x_j, y_j, z_j> \).

3. **Knowledge**
   - Output from application of equations for time series prediction and/or the application of unsupervised/deep learning
   - Given the observation, \( T \), which shows how much rain has fallen by time \( t_1 \), predict how much more rain will fall by time \( t_2 (t_1 < t_2) \).

As mentioned in sub-section 4.6 Observations, History and derived Knowledge data can be mapped to the Dataset node of the Knowledge data model.
Knowledge format

(1) Observation is a time series.
(2) History is a time series.
(3) Knowledge is a set of equations.

The input is Observation (1) and the output is a set of rainfall time series.

(4) Assumption is a time series.

Knowledge acquisition

The equation and/or the algorithm may predict a future rainfall referring to the historical rainfall data. It would be possible to find a set of rainfall time series under the similar weather context in the past. The uncertainty would come from that the weather context is not exactly the same. It would be possible to define a knowledge acquisition problem such that:

**Input:** Rainfall time series, history of rainfall

**Output:** Amount of rainfall expected in future time-frame and in particular area.

**Process:** Rainfall will be calculated by extrapolating a similar time series of rainfall. Given T be a time series obtained by observation, \{< T_i, \alpha_i > | i = 1, 2, 3, ...\} is a set of similar time series where \( \alpha_i = \text{similarity}(T, T_i) \). In order to compare rainfall environment, \( \text{similarity}(T, T_i) \) may consist of similarity of time series and context.

\[
\text{similarity}(T_X, T_Y) = \text{similarity}_{\text{timeseries}}(T_X, T_Y) + \text{similarity}_{\text{context}}(T_X, T_Y)
\]

Given time series of rainfall \( T_X \) and \( T_Y \), \( \text{similarity}_{\text{timeseries}}(T_X, T_Y) \) is calculated without any uncertainty. \( \text{similarity}_{\text{context}}(T_X, T_Y) \), however, may contain some uncertainty since context has many parameters but the number of observable parameters are limited.

It would be possible to reduce the prediction errors by modifying the terms of context similarity term. This will become a knowledge acquisition process.

Assisted Living use case - Home automation

**Knowledge types**

Various types of knowledge are envisaged in the scope of this use case. As an indicative example we consider here knowledge in user
patterns to derive user desires with respect to home and appliances configuration and proactively take actions/offer recommendations. For example, if the user keeps adjusting the temperature (turning the heating/air-conditioning up/down, switching the lights on earlier than in the system configuration) these adjustments are recorded along with time, weather, date information to gradually (autonomously) derive knowledge on what the user prefers most. For the sake of simplicity, in the following we focus only on the temperature settings.

(1) Observation
  \[ \{<ts_i, t_i, x_i> | i = 1, 2, 3, \ldots\} \]
  Where
  \( ts_i \) is a temperature setting at time, \( t_i \), and location within the home, \( x_i \).

(2) Knowledge
Knowledge in this case is the estimation of future temperature settings that the user would like to make by exploiting Bayesian statistics. Knowledge is derived by calculating the probability that the temperature setting will have a specific value for a certain user, given a particular location within the home and time zone.

As mentioned in sub-section 4.6 Observations, and derived Knowledge data can be mapped to the Dataset node of the Knowledge data model.

**Knowledge format**

(1) Observation will be a set of key-value pairs.
(2) Knowledge will be a set of probabilities.

**Knowledge acquisition**

**Input:** Temperature setting observation, location, time.

**Output:** Probability of the level of user settings for a specific device (in this case heating), given a certain location and time.

**Process:** Concepts from Bayesian statistics are applied in order to estimate the probability of the level of user settings for a specific device (in this case heating), given a certain location and time. More specifically, the aim is to utilize a method according to which instantaneous estimations are updated by taking into account existing information on the user. In other words, the aim is to calculate an adapted probability, based on the following formula:
\[ P_{\text{adapted}, n} = w_{\text{hist}} * P_{\text{adapted}, n-1} + w_{\text{instant}} * (1 - |P_{\text{adapted}, n-1} - P_{\text{instant}, n}|) * P_{\text{instant}, n} \]

Where:

- \(|x|\): represents the absolute value of \(x\),
- \(n\): is the current moment,
- \(P_{\text{adapted}, n}\): denotes the adapted probability estimation at moment \(n\),
- \(P_{\text{adapted}, n-1}\): denotes the adapted probability’s previous value,
- \(P_{\text{instant}, n}\): stands for the current instantaneous estimation and
- \(w_{\text{hist}}\) and \(w_{\text{instant}}\): reflect the weights attributed to the historical estimation and the current instantaneous estimation, respectively. Their value is in the interval \((0, 1)\) and their sum equals 1 \((w_{\text{hist}} + w_{\text{instant}} = 1)\).[99]

Beyond Bayesian methods other sample solution approaches may rely on reinforcement learning.

**Assisted Living use case - Remote Health monitoring**

**Knowledge type**

In this example, the focus is on monitoring vital signs of an elderly/disabled person via a wearable device (e.g. a smartwatch). Activities and patterns of movements (e.g. walking, sleeping patterns, eating patterns, physical and social activity, etc.) can also be monitored via motion detection sensors, accelerometers, etc. These patterns can be assessed for inference of the physical condition of an individual. Knowledge in this case refers to the user physical status and behaviours to identify pattern irregularities (any abnormality in usual patterns). If something unusual/potentially problematic is observed a notification is issued to a designated doctor/health care professional. Notifications/alarms can be raised in case something is not yet abnormal but the recorded values show a trend towards a potential problematic situation (e.g. increasing blood pressure which has still not reached a certain threshold may still be worrying). For the sake of simplicity, in the following we focus only on the monitoring of vital signs.

\[(1) \quad \text{Observation} \]

\[- \{<v_{ij}, t_j, x_j, y_j, z_j> \mid i, j = 1, 2, 3, \ldots \} \]
Where
\( v_{s,i,j} \) is a measurement for vital sign \( i \) at time, \( t_j \), and location, \( <x_j, y_j, z_j> \).

(2) History
\[ \{ <V_{i,j}, t_j, x_j, y_j, z_j> | i,j = 1, 2, 3, \ldots \} \]
Where \( V_i \) is a historical measurement data for vital sign \( i \) at time, \( t_j \), and location, \( <x_j, y_j, z_j> \).

(3) Knowledge
The forecasted, most likely value of a vital sign such as blood pressure for a certain user, given a certain time of the day, week, month and location (if relevant). This can be derived through the use of time series forecasting potentially also in conjunction with other techniques (unsupervised learning).

As mentioned in sub-section 4.6 Observations, History and derived Knowledge data can be mapped to the Dataset node of the Knowledge data model.

**Knowledge format**

(1) Observations as a time series.
(2) History as a time series.
(3) Knowledge as forecasted time series based on Observations and History.

**Knowledge acquisition**

**Input**: Vital signs measurements time series, history of vital signs measurements

**Output**: Value of vital signs expected in the future

**Process**: Time series forecasting are applied in order to derive a future time series for vital signs measurements.

**5.2 Platform knowledge**

Section 5.1 provided some examples of knowledge acquisition mainly from the viewpoint of application development.

In addition, in a similar manner, there may be knowledge that is applicable to the iKaaS platform. This may be totally independent of
application knowledge. One example would be the choice of cloud resources. The iKaaS platform will choose cloud resource depending on the status of the cloud resources. The decision of resource choice will be context dependent. It may be possible to define this as a learning process.

One example of a technique that can be applied for enabling learning related to the efficiency of decisions is k-Nearest Neighbour(s) (k-NN), which allows to identify similar contextual situations[100][101]. This way current context (at the point when a decision needs to be made such as on the choice of cloud resources) can be compared to contextual situations encountered in the past (reference contexts). Given the current context and a set of reference contexts the closest contextual situation to the current one can be estimated according to a “distance” calculation between context parameters/features (such as time, location, etc.). This distance is calculated among corresponding pairs of context parameters and then an overall distance for each reference context from the current context is calculated as the sum of the individual distances. Based on these calculations and some overall “distance” threshold a reference context that is closest to the current context can be derived (if it exists). This then allows deriving a previously good decision associated with a reference context to be applied directly without the need of running an optimisation process from scratch.

This approach can also be applied for instance for deciding on the potential re-use of existing Complex Services when there is a new request for a Complex Service, allowing to determine the suitability of previously created Complex Services. (Relevant context parameters/features in this case include functions comprised in Complex, time at which a Complex Service was created/requested, location at which a Complex Service was created/requested, etc.) Such aspects will also be further investigated in the context of T3.1.

### 5.3 Generic Data processing and Knowledge Acquisition mechanisms

#### 5.3.1 Data Processing for Anomaly (Event) Detection from Environmental Data

**State of the Art**

Research into smart city is usually focused on the definition and architecture level, it still remains though on the high-level view of the
smart city. However, the general definition of a smart city is uncertain, and its structure is quite different, because the requirements of each city is different; moreover, what could be the establishment of a smart city depends on local circumstances. Therefore, it is worth doing the work of specific scenarios or applications, looking for ways to help create smart cities technology.

Development and deployment of sensors as well as improved low-power communications technology increase the amount of sensing data collected from the sensors. Most of these sensor data represent the state of the environment in corresponding locations within the city. This provides a better understanding of the status and conditions of the locations. Smart City aims at using such remote sensing data to build “city intelligence”. Citizens and authorities thus can better understand their city, furthermore, can make appropriate decisions or correct problematic situation.

In order to obtain intelligence and derive knowledge using only sensing data is not enough; It is required to put sensing data into context by taking into account external information (such as location, time, and events that may affect the sensing data). However, the question is how to merge sensing data and external information together. Applying external information to all sensing data may be a waste of time and cost, because much of the sensing data only represent normal behaviours, which are not of interest in most cases. Therefore, it is better to identify the particular interested sensing data and then apply the relevant external information to them.

Anomalies in sensing data always represent special issues that are interesting in most cases. They can be clarified and understood by correlating external information [102][103][104]. By understanding the meaning of anomalies, their sources and effects can be determined and relevant knowledge can be built, where necessary corrective measures can be triggered.

The approach consists of three main steps: anomaly detection, external information extraction, and correlation and verification of anomalies and external information.

Anomaly detection aims to provide a means to identify abnormal data points or patterns that indicates interested events. As it is assumed the events are unknown beforehand, supervised approaches that limited in specific known events may not feasible. Thus unsupervised
anomaly detection techniques (distance-based, Kolmogorov Complexity-based, and Functional Data Analysis) are focused on.

Preliminary results show that the detected anomaly may be partly associated with particular source and events. However, there are still many challenges in anomaly detection, including the adaptation of the parameters, setting of threshold values, acquisition of the information of source of event, correlation between anomalies and the source of the event. It is recommended to use Kolmogorov complexity-based approach and functional data analysis with H-modal depth, because they have better distinguishing capabilities of the output of abnormal and normal patterns.

Lack of information about events is a problem that cannot be solved by anomaly detection techniques. Thus it is required to seek other approaches. People publish information about events they are involved in on social networks. Hence, it is feasible to obtain event information from social networks. In Twitter, millions of tweets are published every day. Among the tweets, many real world events are talked about. These tweets may reflect events that are highly related to anomalies in sensing data. This offers the potential to detect anomaly-related events information from tweets.

Once measurements of events and correlations between events and sensing anomalies are verified, they can be stored as knowledge to be used later in a twofold manner for iKaaS services. First, when a certain event is predicted/planned to take place then its effect on the environmental conditions in a certain area can be deduced by consulting the stored knowledge and alerts can be sent out in advance to the possibly affected people. Second, by monitoring the environmental conditions and consulting the stored knowledge, possible events that led to the specific measurements can be identified and actions for them be taken. As an example, an increase of pollution levels at a junction can be correlated to a severe traffic jam, which can then be attributed to factors such as a road accidents or malfunction of traffic lights. In both cases, appropriate actions to cope with the root causes of the event can be triggered.

**iKaaS advances and innovation**

Current event detection approaches on tweets can detect specific events [105][106] or general open events[107][108]. Since anomaly-related events cannot be limited in one particular event, specific
events detection is not feasible. General open events detection extract events that span a time period. Detected events contains both news and real world events without distinctions, thus it is hard to filter out events that are directly related to anomalies in sensing data. Hence, anomaly-related events detection on tweets requires further research which may depend on a combination of supervised and unsupervised approaches. It also relies on features extracted via natural language processing approaches.

Moreover, few approaches provide impact measurements of detected events. In order to correlate events with sensing data, it is particularly important to provide a quantitative estimation of event impact on real world. The measurements of events will be estimated according to event type, number of tweets, etc. These measurements can be aggregated based on event type and should reflect the impact on the real world. Then they can be used to correlate with anomalies in sensing data.

5.3.2 Complex Event Processing

State of the art

Smart Cities concept is linked to big amounts of data, data from sensors monitoring the environment across the city, data from the transport infrastructures (public and private), data from health systems, etc. all this data has a heterogeneous nature. Moreover, new sensors are being deployed on the public transport system vehicles introducing new variables to take into account when evaluating their measurements, as the same sensor could have different behaviour due to external factors like temperature or humidity that will change depending on the place and time.

In IoT vast amount of information is collected from many types of sensors and stored for further processing and derivation of knowledge. It is of paramount importance to ensure the quality of the data both because invalid data can lead to unnecessary wastage of storage resources as well as can reduce the quality/validity of the generated knowledge.

The discarding of data is based on patterns which can be as easy as compare values with valid thresholds or more complex patterns like compare data steaming from many sensors. In order to perform this filtering task Complex Event Processing (CEP) technique is a useful
tool as it provides the capability of looking for patterns over bit amounts of data on the fly. Also CEP engines provide the capability of changing or adding new patterns easily.

The introduction of Big Data brings quick access to big volume of heterogeneous data that can be used to improve the filtering process by means of providing, for instance, more accuracy factors or thresholds.

Complex Event technique was firstly considered as a complementary tool of Big Data: “CEP is the solution to many of the issues raised by big data, especially where that is instrumented data arising from sensors, logs (web, security or otherwise), smart meters, RFID, GPS or similar.”[116]. But trends in Big Data and CEP technologies are being developed to build systems where both feed the other to improve their results and provide solution to known and new problems in many different fields like healthcare [118]. This combination of technologies is being explored in the FP7 COSMOS project (www.iot-cosmos.eu), where it is being developed in the field of traffic monitoring [119].

**iKaaS advances and innovation**

One of the challenges present in all environmental sensing networks is accuracy and precision of sensor between different sensors platforms and networks. And this issue can become bigger when sensors are under quick changes of environmental conditions (temperature, humidity) when they are moving.

Although the accuracy and precision of data steaming from sensors can be analysed off-line, one of the keys of IoT is real-time data, for instance in Madrid Use case one of the objectives is to build real-time maps of pollen and pollution particles distribution that brings up the need of a tool able to fix deviations on measurements on the fly.

And the situation can be more complex if sensors from different manufacturers co-exist in the same network, as the behaviour of sensor and correction factors are not the same.

Leveraging the knowledge acquired in other projects like COSMOS and providing a light Complex event processing engine able to run in small devices and collaborating with Data Bases and Analytics tools,
we will provide a distributed data processing mechanism able to self-adapt to the environment.

This self-adapting data processing mechanism will be applied to the filtering and calibration of sensor networks autonomously. In scenarios like Madrid use case the measures taken by new deployed sensors will be analysed against measurements from legacy sensor networks to derive patterns on deviations and fix them on the fly.

### 5.3.3 Tools for Data collection and Annotation from Mobile Devices

#### State of the Art

In the iKaaS project, mobile devices carried by the users are expected to play a key role with respect to the collection of data needed for the iKaaS provided services in a Smart City context (the interested reader can also refer to [1] for a more elaborate view on the already identified use cases). In addition, the data collected by mobile devices can play a key role not only for the provisioning of services but also for the multi-cloud platform optimization itself.

Towards that end in the context of the iKaaS project, we envision the development of mobile phone applications able to provide the following data and the corresponding annotation of them.

- iKaaS user profile data
- Data from the sensors of mobile devices

The iKaaS user profile data can expose useful information about the user/owner of a device that assist building his/her social profile. For example, the age of a user can be part of the user profile data so that in the case of environmental-induced alarms (i.e. pollution levels above a tolerable threshold) then alarms can be sent to those more prone to be affected due to their age. In a similar manner, in case of the identification of road faults or other structural failures in a city, alarms can be sent only to people above a certain age who are expected to be affected. In a similar manner one can expect that factors such as employment sector and education level can be part of the user profile data. The former helping to identify citizens in already “environmental-heavy” conditions (e.g. working in road works) that are more affected by increase in pollution levels, while the latter can help define the nature/granularity of alarms that can be sent to citizens depending on the language and complexity they can
understand. This list of user profile data can be expanded to include factors such as **gender**, **home town**, etc. if it is foreseen that such information can further assist in customizing the iKaaS provided services.

With respect to data coming from mobile devices, we foresee their use in a twofold way.

First, as data useful for the provisioning of environmental/health oriented services. Such data can include Location, Accelerometer, Magnetometer, Gyroscope and Light readings. Location can assist in defining which iKaaS users are affected (or expected to be affected) for example by pollution levels at a certain place. Readings related to the physical position and movement can be helpful for identifying falls and other dangerous situations that users may find themselves in and assist in triggering suitable alarms for providing assistance to these users. In a similar manner, the level of light can assist in automating the process of switching on the lights in the room where a person with disabilities/difficulties in movement resides.

Data from mobile devices, can also assist in the multi-cloud optimization itself rather than only in iKaaS service optimization. More details on this will be given in the corresponding WP3 deliverable, we do give though here an overview of the requirements in terms of data from mobile devices and how we foresee those being used for.

Data from mobile devices can help identifying the suitability of devices for use in the provisioning of services. This device suitability identification can be seen as “fixing” the end-points of the service provisioning chains, allowing as such (once the end-points are fixed) to then “fix” the location/placement of service provisioning functions in a way that optimizes both the service and the overall cloud performance. For example, if a device is identified as suitable, which is attached at a certain local cloud, then it would make sense for the other service functionalities needed to be instantiated at that local cloud so as to be close to the data source. Factors that be considered to determine the suitability of devices can be related to location/mobility pattern, battery levels, availability of sensors, data quality from sensors, user away and reaction times (e.g. to make sure the user carries the device with them and is able to see an alert on time and react to it).
iKaaS innovation

The specific tool, while not demonstrating any “algorithmic” innovation, possesses certain desired features. Being designed from scratch enhances its security since using a third party library always comes with the possibility of data phishing. Additionally, having direct access to the source code, leads to a lightweight mobile application without any redundant piece of code. Moreover, our approach offers the “opt-out” functionality for users of it during data gathering, a functionality that is not implemented in similar tools.

In a similar manner we believe that user social relationships can assist in defining suitable end-points and local clouds for the instantiation of service functions as well in service optimization. For example if two people are deemed as being in a “trusted relationship”, then in the case of a fall or accident of the first person, an alert can be sent only to the other “trusted” person to go and assist. Once “trusted” people are identified in a service provisioning chain, then their devices and local cloud resources (if owned) can then be leveraged upon for service provisioning. E.g. if a person goes to a trusted person’s home, then these home “local cloud” resources can be utilized and have the notion of the “follow me as I move” service.

One of the fundamental elements of sociability is social interaction. Humans are social beings that interact with each other through verbal or non-verbal communication. Real-world social interactions occur when people are in proximity, maintain mutual-facing directions and exchange verbal signals. Furthermore, they are considered to be a significant component of social signals [109] that people exchange in daily life. Pentland argues in [110] that location, proximity and signalling behaviour of the user are key properties of human networks that affect the propagation of information. Thus, providing an opportunistic and unobtrusive method to quantify users’ social interactions would constitute an important step in human behaviour understanding for several fields including social sciences.

Social interaction detection is comprised by two major and important components: a) interpersonal distance estimation and b) relative orientation computation. Proxemics has preoccupied the field of psychology due to importance of interpersonal distance among people in order to characterise their relation. In [111], Hall mapped interpersonal space among people into four interaction zones based
on their relationship: a) Public, b) Social c) Personal d) Intimate. To detect these interaction zones, an accurate interpersonal distance estimation technique is required. In parallel, Mehrabian [112] showed that on-going social interactions are enabled when users are in shorter distances and tend to face each other directly. Following, Kendon [113] introduced F-formations referring to the spatial formations that people perform when they socially interact. Hence, to understand if people interact in real world there is a need to estimate both their interpersonal distance and their relative orientation. Therefore an appropriate tool (mobile phone application) needs to be developed that will provide these types of information, which then will be fed into a data processing functionality (see corresponding WP3 deliverable [114]) that based on these will calculate and keep updating a social score value, based on which the level of “trust” between pairs of people can be derived.

ikaas advances and innovation

We design, implement and evaluate a novel opportunistic approach for detecting social interactions by using only off-the-shelf mobile phones which does not require user involvement in the inference process. Initially we develop a novel method for detecting users’ interpersonal distance in a fine-grained manner based on only 6 Bluetooth RSSI samples. We estimate users’ relative orientation by improving a state-of-the-art technique [115] for facing direction detection, independent to device on-body wearing position and enhance these through collaborative sensing for faster, privacy preserving and real-time inference. Additionally, the system requires neither any external hardware nor any firmware modifications because it leverages Bluetooth’s native capability for ad-hoc discoverability and communication, making it suitable for pervasive deployment by simply downloading an app. Each device performs inference online in order to eliminate any privacy issues occurring when transmitting data to third parties. Devices calculate interpersonal distance and relative orientation with respect to each nearby user. Finally, we perform classification on the occurrence of a social interaction, depending on the selected target class (proximity or interaction zone).
6 Conclusions

This document provided an analysis of available Semantic Data Models, Data Standards and Semantic Tools for Data Handling and Querying, with special focus on those domains with major impact in the envisaged use cases/demonstration scenarios (Section 2). Moreover, the document provided an overview of requirements and challenges for necessary data model components (Section 3.1) as well as for iKaaS Data Processing and Knowledge Acquisition over multi-cloud infrastructures (Section 3.2 and Appendix in Section 7). The document also presented selected data model components covering seven key aspects of the iKaaS architecture and Platform, namely Virtual Entities, Services, Knowledge, User, Access Rights, City and Health (Section 4). Some first concepts for Data Processing and Knowledge Acquisition mechanisms were also discussed, including formal definition of knowledge types and knowledge acquisition processes in the scope of the iKaaS use cases (Section 5).

Description with respect to iKaaS advances and innovation in the different field elaborated in this document, has been provided so as to position in a clear way the iKaaS original work versus existing technologies. Moreover, the data models requirements are aligned with the project objectives. The presented data model components, as well as the formulation of knowledge and corresponding data processing and knowledge acquisition mechanisms are works in progress and will be further enhanced as the project progresses. The next steps are the further improvement and enhancement of the data model components and the data processing and knowledge acquisition mechanisms towards the specification, implementation and validation of the Universal Data model and mechanisms for Knowledge acquisition and service provision. For example one of the next steps for the enhancement of data model components is to address the addition of resource description aspects to further increase the cloudification of the platform in support of service composition/migration etc. decisions, taking into account the practical capabilities and limitations of the underlying infrastructure. Future work on data processing and knowledge acquisition mechanisms will focus on additional State-of-the-Art analysis and more detailed specification of knowledge and corresponding mechanisms.
7 Appendix: General requirements for IoT Big Data processing and Analytics

There are a few major requirements for IoT-big data processing and analytic platform that it offers the dynamically management capabilities of IoT data/objects but it also provides connectivity to the diverse heterogeneous objects, considering the interoperability issues. The interoperable and federated devices and system can share the data and knowledge across multiple systems in the environments. Next is deriving useful information/data insight and knowledge from this connection and large volume of IoT data. The platform needs to offer ubiquitous accessibility and connectivity in facilitation of maximum accessibility as well as connectivity of the diverse heterogeneous objects/services and various volumes of users including mobility. Due to the mobility the functionalities should be adaptable and data should be migrated and transferable from devices to devices/system to system. Dynamic management/orchestration of users, billions of devices as well as massive amount of data produced by those connected devices, maximum resource utilization, and enabling and sharing of IoT resources (objects, applications, platforms) are all necessary. Personalized, secure, and privacy by design services based on preferences of users and requirements including real-world context are the important requirements. Some of the requirements are dealt here briefly.

i) Intelligent and Dynamic

The platform should include intelligent and autonomic features in order to dynamically manage the platform functions, components and applications. The intelligence should be decentralised and distributed from edge/data sources to applications. The platform should also be capable of making a proactive decision, dynamic deployment, and intelligent decision to understand the context of the operating and deployment environment, users and application requirements and preference, etc. The intelligence should not only be on the application but also on the lower level system function like objects which can make decision on the best adaptable connection of the devices. They have to be dynamic enough in order for realization of resources sharing and service as well as computing migration techniques. Dynamic metering may be necessary when IoT devices are shared. Considering performance targets/constraints, offloading from
clients/hosts to cloud is necessary but the performance should be guaranteed.

**ii) Distributed and decentralized**
The platform should include the distributed information processing and computing capabilities, distributed storage, distributed intelligence, and distributed data management capabilities. The platform functionalities and capabilities are distributed and decentralised across smart devices, gateway/server and multiple cloud environments. Most of the state-of-the-art platforms are centralized in which cloud-based platform used to collect the data and process/store are centralized. However as IoT is distributed and decentralized, more distributed processing and storage of the massive data as well as cloud functionalities is a must. Decentralized (and infrastructure-less) clouds will be the order of the day through processing capabilities and positioning data closer to users.

**iii) Scalability and elasticity**
The platform has to be scalable to address the connectivity from small to large number of the devices, manage the different scale of the data and services, as well as users. It is because more and more devices are connected to the platform and more and more users and using the services and applications. One particular devices and system should be scale to serve multiple users and similarly multiple devices are provisioned for one user. Therefore data management, storage and processing should be scaled between local cloud/gateway to cloud to apps.

The iKaaS processing and computing technologies requirements are horizontally as well as vertically scalable to handle large datasets. Horizontal scaling means adding multiple autonomous computers that work together to increase the overall capacity of the system. In iKaaS, horizontal scaling will be done between local cloud/gateway and cloud system. Vertical scaling means adding more processors or virtual machines or memory, thereby increasing capability of the system. Most of the traditional systems have to scale up vertically so that they can process and analyse larger data sets. Vertical scaling relies mainly on technological innovations to achieve higher performance. In iKaaS both horizontal and vertical scaling are the basic requirements.
iv) **Real-Time and streaming processing**
Real-time data processing and service provisioning of “Big data”, is necessary. Un-structured and semi-structured data coming from distributed sources should be processed to provide real-time/near real-time services.

v) **Heterogeneous (unified view)**
Interoperability between cloud/IoT services and infrastructure, and federation between cloud, Big data and IoT devices has to be taken into account in order to realize full potential of IoT big data and cloud. These three technologies are defined for specific purposes, therefore the heterogeneous nature. The unified and holistic integration views of these three technologies are basic iKaaS requirements. Open software components, standard data structure and modelling and abstraction of heterogeneous IoT devices and the data is necessary. Therefore the software components and functionalities should be defined considering the standard APIs. Standard APIs to deal with heterogeneity need to be evolved. Data always raise heterogeneity problems: many data formats, many metadata schema descriptions, mix of various levels of complexity, etc., are the cases in point. The target here is to deliver a data model and the specification of required mechanisms for exploiting both structured and unstructured data, for moving from raw data to linked data, enabling the adoption of a common understanding, the recognition of similar data, and unambiguous description of relevant information for multimodal and cross-domain smart space applications.

vi) **Security and privacy**
Security and privacy by design is needed including different privacy and security features like data integrity, localization, confidentiality, SLA, security and privacy-preserving data management modules. Holistic approaches are required to address privacy & security issues across value chains including privacy by design aspects, software algorithms and new data management models.
List of figures

Figure 1: Conceptual description of application, data and essential technology in the iKaaS project .......................................................... 5
Figure 2: Conceptual description of the iKaaS data model ................. 5
Figure 3: Dependence relation in the iKaaS data model ................... 7
Figure 4: Big Data Properties[71] .................................................. 29
Figure 5: IoT-Big Data Applications ............................................. 31
Figure 6: Data processing, Machine Learning and Knowledge building flow diagram .......................................................... 32
Figure 7: Knowledge as a service framework [72] .......................... 34
Figure 8: Overview of the city data model structure ....................... 62
Figure 9: Relationships of the medical device system and measurement systems in 11073 PHD, with an added user/patient. The part relevant to iKaaS is outlined with red. ....................... 63
Figure 10: Access Right data model ............................................. 64
Figure 11: iKaaS Virtual Entity Model ......................................... 67
Figure 12: VE produced data measurements structure .................... 68
Figure 13: VE produced data measurements structure - JSON Indicative example .......................................................... 69
Figure 14: iKaaS VE Model Ontology Visualization ....................... 69
Figure 15: iKaaS Service Model .................................................. 70
Figure 16: Service first level concepts / components analysis ...... 71
Figure 17: Simple and Complex service concepts / components analysis .......................................................... 72
Figure 18: iKaaS Service Model Ontology Visualization .............. 75
Figure 19: iKaaS Knowledge data model ..................................... 75
Figure 20: iKaaS Knowledge data model Ontology Visualization .... 78
Figure 21: iKaaS Knowledge data model instantiation .................. 79
Figure 22: User Model Entities Diagram ....................................... 81
List of Tables
Table 1: Overview of iKaaS applications ...........................................9
Table 2: Conceptual layers data modelling standards .......................13
Table 3: Overview of existing OGC and other Semantic Web standards .................................................................13
Table 4: iKaaS Objectives ..................................................................41
Table 5: City data model requirements .................................................42
Table 6: Health data model requirements .............................................43
Table 7: Access Rights data model requirements ...................................44
Table 8: Virtual Entity requirements ...................................................45
Table 9: Service model requirements ..................................................48
Table 10: Knowledge data model requirements ....................................50
Table 11: User data model requirements ..............................................52
Table 12: Overview of use of component data models in iKaaS architecture ........................................................................53
Table 13: Requirements for Data processing and Knowledge acquisition ........................................................................54
Table 14: Semantic data model innovation ..........................................58
Table 15: Access Rights description .....................................................64
Table 16: Service description ..............................................................71
Table 17: Simple Service description ...................................................73
Table 18: Complex Service description .................................................73
Table 19: Knowledge data model description .......................................76
Table 20: User model concepts description .........................................81
Table 21: Views for knowledge definition ............................................83
<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>CEP</td>
<td>Complex Event Processor/Processing</td>
</tr>
<tr>
<td>CESP</td>
<td>Community Environment Sensing Platform</td>
</tr>
<tr>
<td>CSW</td>
<td>Catalogue Service for the Web</td>
</tr>
<tr>
<td>CSW</td>
<td>Catalog Service for the Web</td>
</tr>
<tr>
<td>CVO</td>
<td>Composite Virtual Object</td>
</tr>
<tr>
<td>DB</td>
<td>Database</td>
</tr>
<tr>
<td>ECG</td>
<td>Electrocardiogram</td>
</tr>
<tr>
<td>GML</td>
<td>Geography Markup Language</td>
</tr>
<tr>
<td>HDFS</td>
<td>Hadoop Distributed File System</td>
</tr>
<tr>
<td>HL7</td>
<td>Health Level 7</td>
</tr>
<tr>
<td>HTML</td>
<td>HyperText Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hyper Text Transfer Protocol</td>
</tr>
<tr>
<td>ICT</td>
<td>Information and Communication Technology</td>
</tr>
<tr>
<td>IoT</td>
<td>Internet of Things</td>
</tr>
<tr>
<td>ISO</td>
<td>International Standards Organization</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>ML</td>
<td>Markup Language</td>
</tr>
<tr>
<td>OGC</td>
<td>Open Geospatial Consortium</td>
</tr>
<tr>
<td>OWL</td>
<td>Web Ontology Language</td>
</tr>
<tr>
<td>PURL</td>
<td>Persistent Uniform Resource Locator</td>
</tr>
<tr>
<td>RDF</td>
<td>Resource Description Framework</td>
</tr>
<tr>
<td>REST</td>
<td>Representational State Transfer</td>
</tr>
<tr>
<td>RIF</td>
<td>Rule Interchange Format</td>
</tr>
<tr>
<td>RSS</td>
<td>Rich Site Summary</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>RWK</td>
<td>Real World Knowledge</td>
</tr>
<tr>
<td>SKOS</td>
<td>Simple Knowledge Organization System</td>
</tr>
<tr>
<td>SOAP</td>
<td>Simple Object Access Protocol</td>
</tr>
<tr>
<td>SOS</td>
<td>Sensor Observation Service</td>
</tr>
<tr>
<td>SPARQL</td>
<td>SPARQL Protocol and RDF Query Language</td>
</tr>
<tr>
<td>SQL</td>
<td>Structured Query Language</td>
</tr>
<tr>
<td>SSN</td>
<td>Semantic Sensor Network</td>
</tr>
<tr>
<td>TC211</td>
<td>Technical Committee 211</td>
</tr>
<tr>
<td>TML</td>
<td>Transducer ML</td>
</tr>
<tr>
<td>UML</td>
<td>Unified Modelling Language</td>
</tr>
<tr>
<td>URI</td>
<td>Uniform Resource Identifier</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>VE</td>
<td>Virtual Entity</td>
</tr>
<tr>
<td>VO</td>
<td>Virtual Object</td>
</tr>
<tr>
<td>W3C</td>
<td>World Wide Web Consortium</td>
</tr>
<tr>
<td>WPS</td>
<td>Web Processing Service</td>
</tr>
<tr>
<td>WRS</td>
<td>Web Registry Service</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
</tbody>
</table>
List of References

[17]. ISO 19155:2012, Geographic information -- Place Identifier (PI) architecture,
[18]. ISO/CD 19155-2, Geographic information -- Place Identifier (PI) architecture -- Part 2: Place Identifier (PI) linking,
[31]. USB Implementers Forum. USB Device Class Definition for Personal Healthcare Devices http://www.usb.org/developers/docs/devclass_docs/

[33]. Interoperability design guidelines for personal health systems. Continua Health Alliance. Orderable online http://www.continuaalliance.org/


[35]. https://www.regjeringen.no/en/aktuelt/standardisering-av-velferdsteknologi-med-continua/id2356200/

[36]. http://www.hl7.org/

[37]. http://www.hl7.org/fhir/


[44]. GeoJSON, Full specification can be find at http://geojson.org/geojson-spec.html


[108]. A. Ritter, O. Etzioni and S. Clark “Open domain event extraction from twitter”, In Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 1104-1112), August 2012.


[116]. Philip Howard. “CEP and big data.”, Bloor 2011
Partners:

Coordinator’s contact:

Prof Klaus Moessner
University of Surrey
k.moessner@surrey.ac.uk

This project has received funding from the European Union’s Seventh Framework Programme for research, technological development and demonstration under grant agreement no 643262.