Journal of Biomedical Informatics 94 (2019) 103179

Contents lists available at ScienceDirect

Journal of Biomedical Informatics

journal homepage: www.elsevier.com/locate/yjbin

Supporting integrated care with a flexible data management framework built upon Linked Data, HL7 FHIR and ontologies

Vassilis Kilintzis*, Ioanna Chouvarda, Nikolaos Beredimas, Pantelis Natsiavas, Nicos Maglaveras

Lab of Computing, Medical Informatics and Biomedical Imaging Technologies, Medical School of Aristotle University of Thessaloniki, Thessaloniki, Greece

ARTICLE INFO

Keywords:
Linked Data
Ontology
Telemonitoring
HL7 FHIR
OWL
Web services
Integrated care

ABSTRACT

In this paper we present the methodology and decisions behind an implementation of a telehealth data management framework, aiming to support integrated care services for chronic and multimorbid patients. The framework leverages an OWL ontology, built upon HL7 FHIR resources, to provide storage and representation of semantically enriched EHR data following Linked Data principles. This is presented along with the realization of the persistent storage solution and communication web services that allow the management of EHR data, ensuring the validity and integrity of the exchanged patient data as self-describing ontology instances. The framework concentrates on flexibility and reusability, which is addressed by regarding the aforementioned ontology as a single point of change. This solution has been implemented in the scope of the EU project WELCOME for managing data in a telemonitoring system for patients with COPD and co-morbidities and was also successfully deployed for the INLIFE EU project with minimal effort. The results of the two applications suggest it can be adopted and properly adapted in a series of integrated care scenarios with minimal effort.

1. Introduction

Integrated care refers to a model of patient-centered care, with organized links among multiple levels of care management, coordinated services, and collaboration among professionals across care delivery even of separate organizations. It focuses on the continuum of healthcare delivery around patients and populations, on prevention and management of patients with chronic diseases and/or multiple morbidities [1]. Approaches and levels of integrated care implementation can be context-specific, depending on policies and legislations, as well as on the public–private partnerships. The objective is to ensure high quality and efficient care where and when it is needed, towards better health outcomes while controlling costs. The WELCOME project aims at the development of a technical solution that will leverage integrated care of chronic obstructive pulmonary disease (COPD) and comorbidities via wearable technologies, user applications and cloud computing [2].

Central data management is at the heart of an integrated care solution that entails multiple sources of data and information along with multiple access points, a strong temporal aspect, as well as different computational workflows. This employs cloud data storage, management and security, an ability to support storing and retrieving multiple long-term streaming sensor data, other biosensor data and user reports along with numerous calculated features, metadata and provenance information as well as decision support system outputs.

The benefits of Integrated medical data management in a standardized and semantically enhanced manner is highlighted in various papers [3–5]. Although such an approach could be supported by the current state-of-the-art standards and technologies, it is only partially followed by modern systems. The review by [6] on using Ontologies and Semantic Integration Methodologies to Support Integrated Chronic Disease Management in Primary and Ambulatory Care reveals that there is big potential but still little evidence in this area. Similarly in [7] the use of semantics for data quality in integrated chronic disease management is shown to be promising but still very immature.

A modern framework for medical data management should meet several challenges to be embraced not only by developers but also by knowledge management experts and thus take part in the modern integrated care landscape. It should adhere to current state-of-the-art standards and best practices for the definition of the data model, the web services and the format of the exchanged messages. Such a framework should be easy to maintain and update, while at the same time, it must be able to manage seamlessly a disparity of types of data. Moreover, it must provide the required flexibility to meet the needs not only of standard EHR systems but also of PHRs, Telehealth systems or even clinical trial support systems with minimum modification, since

* Corresponding author at: Lab of Computing, Medical Informatics and Biomedical Imaging Technologies, Medical School of Aristotle University of Thessaloniki, Aristotle University of Thessaloniki, Univ. Campus, Thessaloniki, 54636, Greece. Tel.: +306947077831.
E-mail address: billyk@med.auth.gr (V. Kilintzis).

https://doi.org/10.1016/j.jbi.2019.103179
Received 25 October 2018; Received in revised form 4 April 2019; Accepted 15 April 2019
Available online 24 April 2019
1532-0464/

all these are perceived as pillars in the modern integrated care landscape. The framework must leverage the advantages of cloud computing, to provide unrestricted storage with native integrity checking, along with the standardized data exchange mechanism for the communication with external applications of multiple stakeholders. It must address the needs of managing medical data with robustness and sufficient performance, while at the same time, acting as the base for knowledge discovery and development of learning health systems [8], by relying on the application of Linked Data principles as described by Tim Berners-Lee [9].

We introduce a *flexible scaffolding*[1] *framework* for medical data management that is based on a semantic model, exposes Restful web services and has the following virtues:

(A) It allows an easy manipulation of the managed concepts by altering the semantic model, both on new deployments (*reusability,* the framework can be used in different neighboring domains with minor modifications), and also on an already deployed framework (*flexibility*, easy adaptation to post-deployment requirements concerning addition of concepts to the data model).
(B) It has a single point of maintenance (*maintainable* and *sustainable*). Possible modifications do not require changes on multiple layers, additionally, during such modifications, the framework does not require the procedure of re-compilation/re-deployment.
(C) It relies on modern health informatics standards and follows the Linked Data principles, thus enables usage of the stored information in ways not perceived at the time of the current system development.

The format of the paper is as follows. First, we present the rationale behind this work that defines the design goals, then we present the current state-of-the-art, out-of-the box solutions in comparison with the proposed framework along with the modern technologies that are relevant to the three parts of our framework (data model, persistent storage, data management web service). In the methods section we present the characteristics of the implemented framework. Finally, in the application scenarios section we present the results of the deployment of the framework into two paradigm domains.

2. Objectives

In this section we present the objectives that acted as a roadmap for the development of the framework. These objectives reflect the requirements for the design and implementation of a framework that supports data management of integrated care and it is built upon Linked Data, HL7 FHIR and Ontologies. Apart from the requirements derived from best practices for the data management of integrated care (in general and also in the context of our WELCOME scenario), we present also additional objectives in order to fulfill our goals towards the development of a framework that is beyond state-of-the-art in terms of flexibility, reusability and semantic interoperability.

The design goals of the presented framework were to:

a. ***Formally define/represent*** the data model entities relevant to the telehealth and integrated care domains.

Typically, the definition of the relevant entities of a domain is the first step of any system that manages data. A data model is the definition of how the data corresponding to the relevant entities of a domain should be organized and how those entities and thus the corresponding data relate to each other. As suggested by the application scenario of WELCOME this effort was aiming at the domain of management of patients with COPD, congestive heart failure (CHF), diabetes and depression. In this domain, we must take into account that, beyond common medical record data on the diseases in focus, there are several entities referring to portable/wearable devices' measurements and also to secondary calculated parameters, some deriving from the processing of biosignals recorded via tele-health devices. An example is the inclusion of arrhythmia characterization, based on the analysis of home-based ECG, in the data model. Additionally, since a concrete and complete data model is almost impossible to be defined one off, we must be able to make future changes and additions, minimizing the risk to the rest of the model and the overlaying framework layers.

b. ***Persistently store*** medical record and telehealth data entities related to the domain of focus.

Data must be stored in a way that can be later retrieved by applications other than the one creating them. This is a necessity in integrated care as multiple users, and multiple entry points play a role in the chronic and comorbid patient management. This process must be performed fast and with consistency, especially when bound in a decision support procedure. Also, the selected persistent storage solution should not bind to an architecture (i.e. vendor/product) or specific file system to ensure viability of the system in the long run, and capability for data repurposing.

c. Provide a web service to robustly ***manage*** and ***exchange*** the data of the domain.

The framework must provide a web service that enables the exchange and modification of data remotely through a standard way. The endpoints of the web service should follow common practice guidelines that are familiar to developers of the potential clients. The exchanged messages' format must be such that it does not require either complex or entity specific transformations before data storing. This requirement for high simplicity of the web services is a complementary prerequisite towards a robust data management and exchange framework.

d. ***Define, formally represent*** and ***enforce*** the rules that, once applied, ensure that the stored information is ***valid and meaningful***.

To ensure the accuracy and reliability of the persistently stored data the framework should provide a way to formally represent integrity constraints (upon their definition by the domain experts) and also a robust way of enforcing these constraints. Domain, Entity and Referential integrity [10] should be available not only as definition but also as functional constraints. This is important due to the impact of data quality in the health care domain, and especially as data are produced in telehealth and in various unattended environments. In a project like WELCOME, the definition of data integrity rules requires the involvement of both knowledge management experts and domain experts (i.e. pulmonologist, cardiologist).

e. Adhere as much as possible to **interoperability standards**.

Healthcare data management is a domain with various proposed solutions and knowledge that accumulated through years of research. The use of standards is a prerequisite almost in all medical systems. The HL7 set of standards is arguably the most acknowledged standardization effort in the field of health informatics. HL7 FHIR[2] is a modern and leading open standard for interoperability, and thus ideal to act as source of inspiration and repository of best practices regarding the

[1] Scaffolding: The compiler uses this specification to generate code that the application can use to create, read, update and delete database entries, effectively treating the template as a "scaffold" on which to build a more powerful application. (https://en.wikipedia.org/wiki/Scaffold_(programming)).

[2] https://www.hl7.org/fhir/.

domain.

f. Provide semantic interoperability of data through ***Linked Data*** principles.

For chronic care and comorbidity management, as in the case of the WELCOME project, complicated information flows are automatically employed upon data arrival, to process data and generate features. These are combined as inputs in decision support for the patient and the coordinated healthcare professional team [11]. Therefore, data have to be integrated and actionable for routine use of such systems. Moreover, data produced in coordinated care and tele-health applications, require harmonization prior of being used for other purposes, i.e. for research. To facilitate such functionalities in the future, a system can inherently follow the principles of Linked Data and thus minimize the need for manual mapping of concepts and complex data transformations. Those principles dictate the use of Uniform Resource Identifiers (URIs) to uniquely identify resources and the definition of exchanged data in a way that they include not only actual values but also the relationships among them.

g. The web service supporting the communication should be auto-deployed, agnostic of the managed concepts and thus promote the framework's ***flexibility and reusability***.

To enhance the susceptibility of the framework to changes, the web service should not require any source code change upon common modification tasks of the data model. Specifically, addition, removal or update of the model's concepts must reflect to web service endpoints without re-compilation of the web-service. The latter must apply for both new and existing deployments. This is important since the need to incorporate additional entities and semantics originating from requirements emerging after deployment is not a rare case. Transferring the modification cost only to the semantic data model layer reduces the required total effort resulting in a maintainable and sustainable overall framework. Designing the web service to be agnostic of the concepts represented by the exchanged data enables the redeployment of the framework in neighboring domains and thus render it reusable in other applications.

h. Support the framework through ***free and open source tools***.

In order to provide a solution that can be adopted by the community without restrictions the framework must rely on technologies that are free to use by anyone. This can protect the framework from vendor lock-in practices and provide the basis for a system that is reusable and sustainable.

The design goals presented were aiming to a scaffolding framework for the management of telehealth data in a robust manner, retaining flexibility to accept modifications after deployment and the capability to be reused in different neighboring domains. The support for flexibility and reusability must be coupled with limited required effort and without the need of changes in all layers of the system. Finally, the target framework must be inherently interoperable through its coherence with accepted standards in a basic level (e.g. data types) and not through transformation operations that may produce loss of fidelity.

3. Related work

In this section, we provide an overview of similar data collection and integration approaches applied in the context of recent or ongoing research projects, focusing on Integrated Care scenarios. The presented overview focuses on the respective projects' use cases, data modeling and management approach, application of widely accepted interoperability standards, and use of semantic data annotation capabilities. Also, we present a review of the state-of-the-art technical solutions and frameworks that are targeted at flexible management of data.

UniversAAL is an open source platform providing an interoperability framework upon which individual applications could communicate to exchange data in specific use cases [12], focused on Ambient Assisted Living (AAL) scenarios, including health data monitoring.[3] UniversAAL was initially an EU-funded research project and its data model was based on the so-called Consolidated Reference Model providing the application specific semantics organized as an underlying conceptual model. In its current version, the UniversAAL project provides an open source platform based on a Service Oriented Architecture which engages ontologies in the form of RDF as the main semantic interoperability layer among the various services.[4]

In the context of the inCASA European project, a telemonitoring system has been developed focusing on elderly chronic patients [13] and monitoring both activity and physiological data, providing data analysis and alerts. The inCASA telemonitoring framework employs IEEE 11073 medical device standards over ZigBee[5] for communication from the sensor to the local gateway and IHE PCD-01 HL7 profile to communicate data to the respective service through General Packet Radio Service (GPRS). Transmitted data include peripheral oxygen saturation (SpO2), blood pressure, activity data, etc., and they are transmitted as HL7 (version 2) messages exchanged through a WSDL interface defined by IHE.[6] The inCASA technical implementation engaged Semantic Web technologies (e.g. SPARQL and SWRL) and reasoning upon semantically annotated data to produce alerts and notify the respective consumer applications.

Respectively, the Dem@Care project developed a multi-sensor activity telemonitoring solution to facilitate the timely diagnosis, assessment, maintenance and promotion of self-independence of dementia patients[7]. The Dem@Care framework also leverages a Service Oriented Architecture approach, as client applications communicate with the Dem@Care services using standard web service protocols (WSDL/SOAP) using a specific XSD schema to semantically identify communicated entities [14]. Furthermore, the Dem@Care exploits Semantic Web technologies (e.g. ontologies) to semantically annotate data and also exploit automatic reasoning capabilities combined with rules in form of SPARQL queries to detect specific types of physical activities.

C3 Cloud[8] is an ongoing EU-funded project, aiming to develop an ICT infrastructure to enable continuous coordination of patient-centred care activities by a multidisciplinary care team and patients/informal care givers. The main concept of the project is the design of a personal care plan, based on available clinical guidelines. Information exchange is designed upon HL7 FHIR and REST web services to provide interoperability with proprietary EHR systems used on pilot sites, based on HL7 Care Plan Domain Analysis Model (DAM). onFHIR platform is used as the project's storage layer, providing secure FHIR compatible storage upon MongoDB, a widely used NoSQL database.[9, 10, 11] There is also provision for an "Integrated Terminology Server" as a central semantic infrastructure to facilitate semantic interoperability and also provide reasoning capabilities.[12, 13]

The OpSIT project was a Germany based research project which focused on the formal definition and synthesis of the various

[3] http://www.universaal.info/page/about/.
[4] http://www.universaal.info/page/service-providers/.
[5] https://www.zigbee.org/.
[6] IHE International https://www.ihe.net/.
[7] http://www.demcare.eu/.
[8] http://c3-cloud.eu/c3/project-summary.
[9] http://c3-cloud.eu/sites/default/files/ICIC2018-LaleciErturkmen-PaperID443.pdf.
[10] https://onfhir.io/.
[11] http://www.hl7.org/documentcenter/public/wg/java/SRDConFHIR.pptx.
[12] http://c3-cloud.eu/c3/project-summary.
[13] http://c3-cloud.eu/sites/default/files/pdf/InformaticsForHealthPosterFinal.pdf.

healthcare processes using Business Process Modeling Notation (BPMN), using telehealth via smart devices (e.g. a smart watch) as its main use case [15]. The typical architecture of transferring data through local gateways is employed and the data storage layer is based on a relational database (MySQL). The need for semantic annotation of collected data is clearly identified and an ontology-oriented approach is outlined in [16] to enhance interoperability of data exchange among various IT systems and also to enhance data processing capabilities through AI.

Finally, the MobiGuide[14] project's main objective was to create a scalable, secure, ubiquitously accessible, and user-friendly mobile solution for designing, deploying, and maintaining Patient Guidance Systems based on clinical guidelines and personal health records. The MobiGuide project's approach to EMR data management and exchange uses HL7 vMR frames for back-end definition and connection to hospital EMRs (through manual mapping) and converts HL7 vMRs to newly defined openEHR archetypes in order to be used by the Clinical Decision Support system [17]. This approach effectively reconciliates the differences between the two widely known families of standards (i.e. openEHR and HL7) maintaining semantic abilities offered by the ontology section of openEHR archetypes and the interoperability traits offered by HL7 vMRs.

Summarizing, the need for both syntactic and semantic interoperability is clearly identified in the respective projects. Service Oriented Architectures above REST or SOAP paradigms were typically engaged to tackle syntactic interoperability. Regarding semantic interoperability, HL7 standards (currently evolved to HL7 FHIR) along with openEHR archetypes were the most prominent message exchange paradigm. Semantic Web technologies have also been widely employed as they provide both semantic interoperability and automatic reasoning capabilities. Regarding the data storage layer, both relational databases and also RDF triple stores have been employed. Nevertheless, most of the presented approaches do not follow standards-based data management on all levels of data processing (i.e. data exchange, storage, querying, reasoning), on the other hand for approaches that present high adherence to standards (i.e. MobiGuide), semantics, instead of being a core element not only of the model but also of the exchanged and stored data are concentrated in one layer of their approach as an addition to support specific functionality. Also, the presented approaches and decisions taken by the aforementioned efforts adequately support the specific program's needs for data modeling and exchange through defined communication APIs,[15] on the other hand the aim for reusability in different deployments is out of scope for most of those efforts. In efforts with support for extensibility via addition of new concepts (i.e. MobiGuide) the required modification involves changes on multiple layers of the proposed system.

Regarding existing technical solutions that aim at providing flexible data management systems, OData[16] is an approach of standardizing the way normal create, retrieve, update, delete (CRUD) operations are provided through REST web services. OData is standardized through OASIS[17] and can be considered the closest thing to a defacto standard in prescribing a RESTful communication's API. It prescribes the way that such web services facilitate CRUD and querying operations in a way independent of the actual backend implementation. In the .NET ecosystem, typically the data would be stored to an SQL server backend and an Entity Framework[18] ORM model would be built on top of it in order to provide an object-oriented view of data. Subsequently the EF model would be used from the ASP.NET Web API's services in order to expose the CRUD operations on the client's side. Respectively, in the Java ecosystem, tools such as Jersey,[19] Spring MVC[20] would be used to provide the REST service layer, and an ORM like Hibernate[21] would be used to provide an object-oriented modeling of data, as data would be typically stored in a relational DBMS.

The above implementation approaches present OData compatibility, are tested by a wide community and provide multilevel data validation (from REST input validation to SQL data integrity). On the other hand, they lack a single point of maintenance and there is no semantic interoperability support. In case a change occurs in the data model, the whole software stack needs to be recompiled. Taking the .NET stack as an example, if a table is added in the database, the EF model has to be rebuilt and the respective Web API service endpoint has to be manually added. While the mainstream scaffolding implementations support many different formats, surprisingly they have not yet adopted the Linked Data principles and the respective technologies such as RDF.

3.1. Terminologies and standards

A semantic data model describes the meaning of its instances but also allows the interpretation of the meaning directly from the instances. This enhancement that a semantic model offers to the domain model definition is important and, along with the expressivity provided by semantic tools, makes ontologies a modern way of describing a domain data model. There are several documented attempts to use ontologies as data model to support EHR data [18], personalized care of chronic diseases [19] and also Decision Support Systems [20–22].

The Web Ontology Language (OWL) is a family of knowledge representation languages for authoring ontologies. Ontologies are a formal way to describe taxonomies and classification networks, essentially defining the structure of knowledge for various domains [23]. OWL DL is a sublanguage of OWL and it was defined to support those users who want the maximum expressiveness[22] without losing computational completeness (all entailments are guaranteed to be computed) and decidability (all computations will finish in finite time) of reasoning systems.

One of the most acknowledged standard in health informatics is HL7. HL7 is an organization dedicated to providing a comprehensive framework and related standards for the exchange, integration, sharing, and retrieval of electronic health information that supports clinical practice and the management, delivery and evaluation of health services. HL7 FHIR (Fast Healthcare Interoperability Resources) is a new HL7 standard for exchanging electronic health records. It builds upon previous HL7 data format standards, but also leverages more modern technologic concepts and approaches, aiming to be more developer-friendly. FHIR solutions are built from a set of modular components called "Resources". These resources can easily be assembled into working systems that solve real world clinical and administrative problems.[23]

It is important to note that FHIR is an open[24] standard and at the time of the framework implementation phase published as Draft Standard for Trial Use (DSTU).

Health ontologies and terminologies are widely used in health care information systems. One of the most easy to use and freely accessible repository of health related ontologies is BioPortal [24] of the National

[14] http://www.mobiguide.eu/.
[15] Application programming interface (API).
[16] http://www.odata.org/.
[17] https://www.oasis-open.org/org.
[18] https://msdn.microsoft.com/en-us/data/ef.aspx.
[19] https://jersey.java.net/.
[20] http://docs.spring.io/spring/docs/current/spring-framework-reference/html/mvc.html.
[21] http://hibernate.org/.
[22] OWL DL includes all OWL language constructs with restrictions such as type separation (a class cannot also be an individual or property, a property cannot also be an individual or class).
[23] https://www.hl7.org/fhir/summary.html.
[24] FHIR is licensed under Creative Commons "No Rights Reserved", and states that "You can create derivative specifications or implementation-related products and services". https://www.hl7.org/fhir/license.html.

Center for Biomedical Ontology. It incorporates search and representation mechanisms for several health ontologies and terminologies such as SNOMED Clinical Terms,[25] International Classification of Diseases (ICD),[26] Logical Observation Identifier Names and Codes (LOINC),[27] and World Health Organization's Anatomical Therapeutic Chemical (ATC)[28] classification system among others. One of the functionalities of BioPortal is that it provides Persistent Uniform Resource Locators (PURLs) for all concepts defined in the ontologies it hosts. PURLs are Web addresses or Uniform Resource Locators (URLs) that act as permanent identifiers in the face of a dynamic and changing Web infrastructure.

3.2. Persistent storage and data integrity

Regardless of their origin, scope, or size, all the relevant data can be classified in three broad categories: Structured Information, BLOBs (Binary Large OBjects), and the metadata concerning the BLOBs belonging to the first one.

The issue of storing BLOBS is a trivial one, technology-wise. A variety of solutions exist, ranging from the storage of flat files on the server's file system, accessed through a basic custom-built web service, to more sophisticated approaches of using an existing, commercial, file storage cloud solution such as Amazon S3, Microsoft Azure BLOB Storage or Google Cloud Storage.

What is of concern, however, is the selection of a storage engine for the handling of structured information data. The main options to consider when selecting a storage engine are the following:

- Relational SQL databases
- NoSQL databases

Relational databases are by far the most widely used solution to save data in enterprise systems, however they also have several disadvantages. It is not easy to maintain an evolving data model in a relational database. Furthermore, relational databases have no "out-of-the-box" support for semantic web technologies, which means that we would have to implement semantic web technologies on a layer out of storage.

NoSQL is a term encompassing various families of database technologies, that differ from traditional relational databases in the mechanism used to store and retrieve data. These families include key-value stores, key-document stores, column-family stores and graph stores [25]. NoSQL databases tend to be cluster-friendly, schema-less and usually lack model definition and integrity checking mechanisms. Furthermore, they implement custom, non-standardized ways to access data leading to vendor lock-in. There exists, however, the specific case of RDF Triple Stores. Those are graph databases that store triples of information, a triple being a data entity consisting of subject-predicate-object [26]. Triples can be imported and exported using queries, RDF and other formats.

Triple stores offer a variety of advantages. There is a standardized, vendor independent way of accessing data, SPARQL [27] for querying, and RDF serializations for exchanging data. Triple stores can be queried over HTTP, making it easy to fit them inside a service-oriented architecture. They allow easier maintenance of an evolving data model, while, at the same time, natively embedding semantic interoperability. RDF triple stores adhere to the various W3C standards related to RDF (RDF, SPARQL, GraphStore Protocol). This means they offer a very flexible solution, as one can simply swap one storage engine for another, with minimum effort and without any significant changes to existing codebase or overall architecture.

OpenLink Virtuoso Universal Server is open-source, provides commercial support if required, and can be deployed in a cluster setup. Regarding the performance, studies [28] have presented that Virtuoso, deployed as a backbone storage engine in an integrated Web Services approach based on RDF data, can perform as fast as traditional relational databases. Virtuoso is also found to outperform other SPARQL-enabled and semantic repositories either in real-word [29,30] or generated datasets [31].

It is important to note, that one main characteristic of semantic repositories is that they operate under the Open World Assumption [32] where the absence of a statement does not, in principle, mean that the statement is false. In essence, RDF Triple Stores do not provide a concrete, standard way to enforce constraints in the traditional, relational approach [33,34]. This is partly, due to the way those storage engines evolved. In the past, they were primarily used as views of data that were already stored in a relational database, therefore, it was the database's engine responsibility to ensure they were valid and of good quality. With the introduction of SPARQL 1.1 UPDATE language, it is now possible to use RDF Triple Stores as a storage engine. This has led to the emergence of a variety of solutions to enforce constraint checking and make them operate more like relational databases in that regard. Examples of these kinds of approaches include OWL Restrictions, Resource Shapes (ReSh),[29] Shape Expressions (ShEx),[30] Description Set Profiles (DSPs),[31] Stardog ICV,[32] Pellet ICV,[33] SPARQL and SPIN (SPARQL Inferencing Notation) [35]. SPIN is a W3C Member Submission, offers an open source Java API and is distributed by a commercial entity (TopQuadrant Inc.) that actively supports it. In SPIN, standard SPARQL is used to formulate constraints, and the SPIN API checks RDF data against those constraints.

4. Methods

The presented framework consists of three interconnected parts: (a) The underlying data model implemented as an OWL-DL ontology, (b) the integrity mechanisms along with the persistent storage solution, and (c) the RESTful web service dynamic interface that exposes the data management functionality.

In this section we present the main methods and software tools used in the design and development of our scaffolding framework.

4.1. Definition and construction of the data model

The data model was based on (a) semantic technologies, (b) HL7-FHIR, (c) Medical Terminologies. After putting these technologies in the scope of our work, the steps for combining them in the data model are described.

The methodology that we followed [36] in order to define and implement the semantic data model proposes a series of steps in order to construct an ontology. The main steps of this methodology are:

A. Explicitly determine and demarcate the subject-matter or domain of the ontology.
B. Determine what are the concepts and relations amongst concepts in this subject-matter and concretize this information in the form of a representational artifact.
C. Ensure logical, philosophical and scientific coherence, coherence and compatibility with other relevant ontologies and human intelligibility.

[25] https://www.snomed.org/snomed-ct.
[26] http://www.who.int/classifications/icd/en/.
[27] https://loinc.org/.
[28] https://www.whocc.no/atc_ddd_index/.
[29] https://www.w3.org/Submission/shapes/.
[30] https://shex.io/.
[31] http://dublincore.org/documents/dc-dsp/.
[32] https://www.stardog.com/blog/data-quality-with-icv/.
[33] https://www.w3.org/2001/sw/wiki/ICV.

D. Formalize and implement the representational artifact in a computer language in some specific computing context.

Regarding the implementation of the above methodology each step has been addressed as follows:

A. We have decided that the ontology must represent the entities, required to store medical record data and telehealth data related to COPD, CHF, diabetes and depression. The selected entities were represented as ontology classes.
B. The concepts were represented initially as a simple list of domain entities in a simple table. Subsequently the relations between them were defined along with the required restrictions (i.e. restrictions regarding referential integrity, logical limits, data type conformance, etc.) which would ensure that the stored information is valid and meaningful.
C. The logical and scientific coherence was ensured through iterative review of the defined relations by a group of knowledge management experts and three domain experts (pulmonologists and cardiologist).
D. The representation language was OWL-DL.

4.1.1. The semantic model

The resulting ontology is presented in detail in [37]. In order to preserve this paper's coherence, we will repeat the main characteristics.

In this ontology the domain entities are presented as OWL classes while the instances of the defined OWL classes correspond to the actual data managed by the framework.

The semantic model maintains three significant aspects in accordance with HL7 FHIR. The first is the use of FHIR primitive and complex data types as available holders for all types of information handled by our framework. The rationale is that FHIR data types are mature enough, while they are also well defined and constrained to be represented as a distinct ontology. The resulting ontology may be imported by other ontologies that describe their particular health domain and are not in need of the complexity introduced by a FHIR-based EHR messaging system. More importantly, having the same semantics in the data types of the exchanged information between our framework and HL7 FHIR, ensures that, conversion of the managed information to the wire format of HL7 FHIR (e.g. JSON) can done without loss of fidelity. Details on the methodology that we have followed in the definition of HL7 FHIR primitive and complex data types in RDF/OWL are presented in [38].

The second is the definition of all classes as direct or indirect children of three classes, named after HL7 FHIR Resource categories, i.e. *Clinical*, *Administrative* and *Conformance*. This is merely done for better organizing classes in accordance to HL7 FHIR rather than practical reasons, since those classes have no properties attached.

Finally, the semantic data model leverages the semantics of HL7 FHIR Resources by defining one class per FHIR Resource along with its' properties and attributes. This was a significant effort resulting into the definition of 87 OWL classes and 180 OWL object properties. In Fig. 1, the definition of the FHIR Resource *Condition* as an OWL class in our ontology is presented along with the corresponding object properties.

The framework's ontology comprises five ontologies, each one defined for a specific purpose, to form the semantic data model. This improves the maintainability of the ontology, since specific parts can be updated independently, but more importantly, enables the reusability of some parts of the framework's ontology without any modification in other contexts. The five ontologies are:

(a) The "*FHIR primitive and complex data types ontology", d*efining the primitive and complex data types of the FHIR framework, alongside their validation rules [38]. In this ontology, HL7 FHIR primitive and complex data types are defined as OWL classes and the property that points to the actual literal value is the *rdf:value* property.
(b) The *"Storage server specific properties ontology",* defining properties, that can be assigned to specific ontology classes, that can be examined by the Restful API to evaluate referential integrity. For example, the *hasForeignKey* property is used to define that a referenced resource must pre-exist. In the case of the *Condition* class this property is used to define that a particular patient instance must exist in order to enter a condition for him/her. It must be noted that this ontology is not mandatory to be hierarchically second. Nevertheless, since the FHIR data type ontology is considered the cornerstone of the model, it is placed second.
(c) The *"FHIR Resources and properties ontology",* defining the required HL7 FHIR Resources and their corresponding properties in OWL. This ontology also includes, along with the definition of each class, detailed annotation and cardinality restrictions on the properties. For example, as depicted in Fig. 1, the *Condition.clinicalStatus* property must have exactly one value for the exchanged (*Condition*) instance to be valid.
(d) The "*Extension to FHIR Resources ontology",* defining classes partially matching the semantics of HL7 FHIR Resources. This ontology is developed to match domain needs by altering or adding semantics on existing HL7 FHIR Resources. For example in our context, any Observation may be derived both from other Findings or Observations and not only by other Observations as in HL7 FHIR.
(e) The *"Domain specific data entities ontology",* defining the entities required to model the domain of integrated care services for multimorbid patients. This ontology includes definitions of classes that match specific domain needs. Defining domain entities as subclasses of the FHIR Resources' classes specializes the concepts managed by the framework. The *Observation* class and its' defined sub-classes provide an evident example. Instead of using the instances of the *Observation* class (corresponding to the FHIR Resource Observation) for storing actual data (of any observation type), a unique sub-class for each domain entity is defined. In this aspect, the classes named *BodyTemperature and BloodGlucose* managing data regarding body temperature and blood glucose respectively are defined as sub-classes of the *Observation* class. Having a specific sub-class, for body temperature, allows us to provide semantic links for the entity (to SNOMED via BioPortal's PURLs[34]), define SPIN rules to enforce logical values (i.e. 25 °C to 44 °C) and to restrict the accepted data type to be only of type *Quantity*.
(f) The aforementioned five ontologies are hierarchically connected, i.e. the "*Storage server specific properties ontology"(b) imports* "*FHIR primitive and complex data types ontology"(a) and "FHIR resources and properties ontology"(c) then imports (b) etc.* and each layer added, enhances the semantics of the framework's data model.

The ontology that we have developed is available online either for reuse or for review purposes.[35] In Fig. 2 a part of the class hierarchy and the definition of the *BodyTemperature* class is presented.

4.2. Definition and construction of the persistent storage and integrity mechanisms

For our framework OpenLink Virtuoso Universal Server was selected as the persistent storage solution and SPARQL GraphStore Protocol [39], was selected as the internal communication protocol between the data management web service and Virtuoso, using a one graph per resource approach, while SPIN was used to represent the required functional restrictions on the stored data.

We have used OpenLink Virtuoso Universal Server not only for the

[34] Specifically for this example, we have linked our BodyTemperature class to SNOMED CT term "Body temperature (observable entity)" via the PURL http://purl.bioontology.org/ontology/SNOMEDCT/386725007.
[35] http://lomi.med.auth.gr/ontologies/WELCOME_entities.

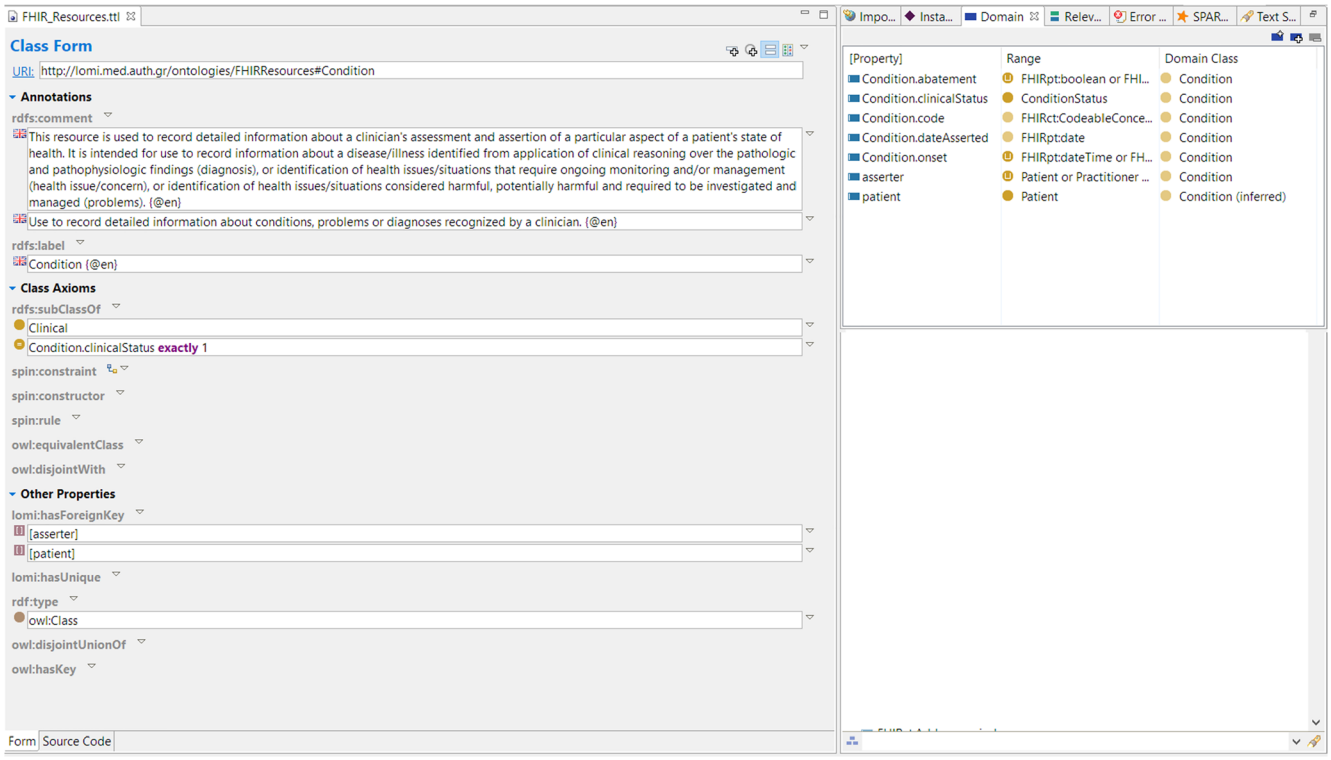

Fig. 1. Condition FHIR Resource representation as an ontology class. In upper right the corresponding properties (i.e. properties with Condition class as Domain) are depicted.

persistent storage of the exchanged data but also to persistently store the ontology. Although Virtuoso was the choice for an RDF Triple Store, it requires little extra effort to change to another one in future setups if new triple store technologies arrive that perform better than the adopted solution. This is due to certain design choices, such as the fact that no vendor-specific SPARQL functions are used, no non-prescribed SPARQL query forms (such as SPARQL DESCRIBE) are used, and communication is strictly over standard SPARQL HTTP protocols (no binary drivers). In addition integration tests for the framework include not only Virtuoso, but also Apache Jena Fuseki[36] to ensure standards compliance and avoid the possibility of future vendor lock-in issues.

After reviewing recent [40,41] and older work [42], we opted for the use of SPIN for enforcing constraints when storing data on an RDF Triple Store as it appeared to combine maturity, accessibility, integration with the rest of our development toolset, and a relatively smooth learning curve.

SPIN rules were defined and are used to validate and enforce value level constraints. For example, one value constraint is that *Observation.value* (the property about the actual measurement value) for *BodyTemperature* cannot be more than 44 °C. This constraint is enforced by the SPIN rule shown in Listing 1. These types of constraints are evaluated every time a specific resource is created or updated.

Unlike value level constraints, enforcing referential integrity and, similarly, unique values' constraints, cannot easily achieved using the same technique. In relational databases, the use of indexes allows for very fast evaluation of these constraints. Using SPIN and RDF graphs, however, one must either load the entire graph in-memory, an expensive operation, or use a triple store that supports SPIN inference, essentially enforcing vendor lock-in. We, therefore, opted to identify these types of constraints by defining and using several custom RDF properties.

An example can be seen in Listing 2[37] above, where part of the definition of the *Device* class is shown. When a device is assigned to a patient the corresponding patient URI must already exist in the triple store, while, at the same time, no two devices can have the same identifier. At runtime, these types of constraints are transformed to appropriate SPARQL queries that are executed directly against the triple store's SPARQL endpoint.

4.3. Definition and construction of the web service interface

The top layer of our scaffolding framework's architecture is the communication with the consumers of the service via an application programing interface (API). We implemented a RESTful API for this purpose following the established best practices for REST [43–46]. REST is widely used, it supports several types of security methods and open source client implementations exist for almost all platforms.

In the presented framework structured information is exchanged in the form of RDF graphs, serialized in Turtle,[38] and stored in an RDF Triple Store. Turtle was selected since it is more compact than RDF/XML and N-triples while it is also significantly more human readable, facilitating development of client applications and debugging in case of errors.

An overview of this software architecture stack is shown in Fig. 3.

The REST API consists of three main services: (a) the Ontology/Schema service, (b) the RDF Resources Service, and (c) the Files service. These services and their endpoints have been designed in accordance to general established best practices for REST, also considering the design

[36] https://jena.apache.org.

[37] Non-standard namespace prefixes presented in Listing 2 are (a) (*footnote continued*) FHIRResources: which refers to "*FHIR Resources and properties ontology" and* (b) lomi: which refers to "*Storage server specific properties ontology"*.

[38] Turtle, the Terse RDF Triple Language, is a concrete syntax for RDF as defined in the RDF Concepts and Abstract Syntax W3C Recommendation. Turtle is an extension of N-Triples carefully taking the most useful and appropriate things added from Notation 3 while keeping it in the RDF model. (https://www.w3.org/TeamSubmission/turtle/).

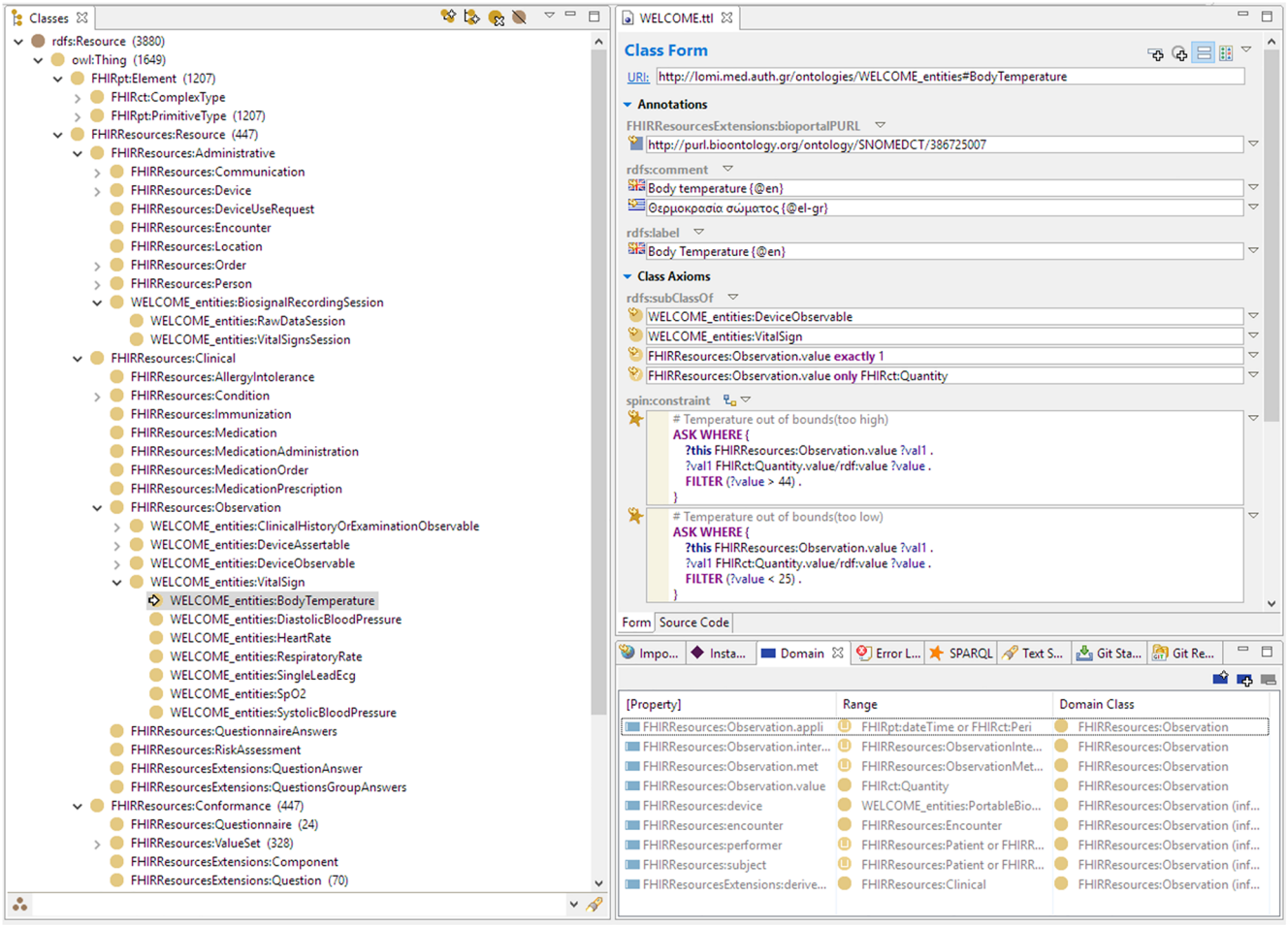

Fig. 2. Ontology hierarchy.

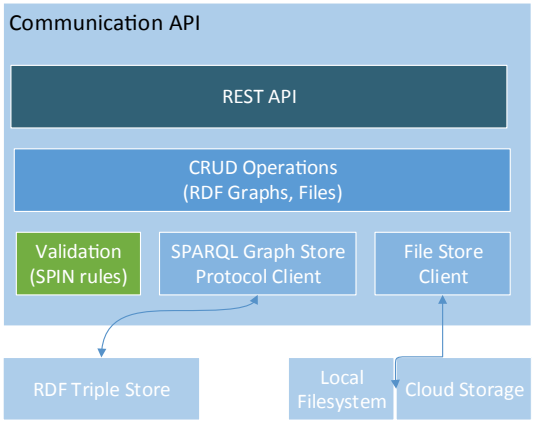

Fig. 3. Communication API software architecture stack.

of REST APIs developed specifically for semantic technologies[39,40] and the HL7 FHIR API.[41]

A. The Ontology/Schema service is responsible for the storage and retrieval of the framework's ontology. Endpoints are provided for administrators to store and update the data model (ontology). Application developers can retrieve the data model in order to design, implement and maintain their applications. As the data model may evolve, client applications can dynamically calculate and display information dependent on the stored model (e.g. they can incorporate addition of new questionnaires or new observation types).

B. The RDF Resources Service handles the storage and retrieval of structured data. Clients can create, retrieve, update, delete (CRUD paradigm) and search for resources. The base documentation for this service is simple enough to fit in Table 1. This end point is designed according to the following two key design decisions:

(1) dynamic: as it accepts resources directly based on the ontology. The addition of a new resource in the ontology (e.g addition of *BodyHeight concept*) creates a valid end point in the communication API (e.g /data/*BodyHeight*), which can be directly employed. It should be highlighted that no specific endpoint needs to be built for each entity in the ontology model. The RDF Resources Service automatically updates the available signatures as soon as the data model's ontology is updated with new entities or relationships. It is important to note that, corresponding endpoints for the creation of new instances are generated only for the terminal nodes of the ontology's class hierarchy. As a result, the framework is able to enforce (for each entity) rules and restrictions defined in the terminal nodes themselves and/or inherited from their super classes.

(2) generic: as it can be used to store information regarding Patient personal details, patient's measurements or communication messages between doctors in a homogeneous manner.

Additionally, the complexity of the underlying ontology is hidden

[39] http://www.w3.org/TR/sparql11-http-rdf-update.
[40] http://www.w3.org/TR/ldp.
[41] https://www.hl7.org/fhir/http.html.

Table 1
RDF Resources Service.

Endpoint	HTTP VERB	ACTION	RESULT
/data/{Resource Type}	POST	CREATE	Store a new resource of the specified {Resource Type}
	e.g. /data/Patient (create a new Patient resource)		
/data/{Resource Type}/{UUID}	GET	READ	Retrieve a specific resource
	e.g. /data/BodyTemperature/f0ad1dcb-7273-4ba7-9b13-9438e7963717 (retrieve specific BodyTemperature resource)		
/data/{Resource Type}/{UUID}	PUT	UPDATE	Update a specific resource
	e.g. /data/VitalSignsSession/f555ec19-841d-490c-8649-bc2a22e103e0 (update specific VitalSignsSession resource)		
/data/{Resource Type}/{UUID}	DELETE	DELETE	Delete a specific resource
	e.g. /data/Patient/159feaee-df29-4ba0-8abb-56123dad7dbb (delete specific Patient resource)		
/data/{Resource Type}	GET	READ	Retrieve a collection of resources of the specified {Resource Type}
	e.g. /data/Patient (retrieve a list of Patient resources)		

from the developer using the service, who only needs a basic understanding of how RDF data are represented. The API consumer may perform search operations through the inclusion of a GET parameter. For example, a request of the form

```
GET api/data/VitalSignsSession?q=Period.end,after,2018-01-01T14:45:00.000Z
```

would return a collection of Vital Sign Sessions that ended after January 1st 2018 14:45 UTC.

The filtering parameter can be multi-valued, allowing string, numerical, temporal or boolean values, and also URI references, as criteria. For example:

```
GET api/data/Patient?q=res,like,Gender_male&q=Person.birthdate,before,1970,asc
```

would return a collection of male Patients, born before 1970, and sorted by their date of birth in ascending order.

C. The Files service handles binary files. Clients can store, retrieve and delete BLOBs such as EDF files for sensor data, or PDF files from external reports.

The communication API features gzip compression[42] for all API endpoints, supports also the ETag[43] mechanism to achieve HTTP caching, thus further reducing bandwidth usage, and concurrency control for the update operations on resources. The CRUD Layer of the software stack depicted in Fig. 3 is the module responsible for the actual CRUD operations exposed by the REST API. API endpoints rely on corresponding method calls of this module to perform their operations. The CRUD Layer utilizes the SPIN API to evaluate SPIN rules (that enforce integrity), a SPARQL/Graph Store Protocol Client to communicate with the Triple Store and with the file store.

5. Results

5.1. Application scenario

In Fig. 4 an example deployment scenario of our framework is presented. In this example an application server is consuming the web services offered by the framework to provide EHR data related functionality to an end user. The deployment sequence of the framework is as follows:

(a) The domain expert along with knowledge management expert modify/enhance the ontology's final semantic level, to include domain specific entities and restrictions.
(b) The Communication API and the Persistent storage server are deployed, configuring deployment settings such as communication addresses. This step does not require the previous step to be completed since the API automatically reshapes the web service endpoints based on the definition of the terminal nodes.

The ease of deployment using free and open source software for the ontology editing, communication API and the persistent storage server favor the adaptation and use of our framework in scenarios not envisioned at the time of development. It must be noted that since the framework components communicate in a standardized manner (e.g. SPARQL 1.1 Graph Store HTTP Protocol) future vendor lock-in issues are mitigated. Sustainability of the framework is also supported by the loose coupling of the framework's components (i.e. the communication API is agnostic of the ontology entities and the vendor of the triple store).

5.2. Adaptation and reuse

The presented framework is ***reusable*** since it can be re-deployed for use in neighboring domains. The scenarios that the framework can be reused to manage data include, among others, personal health record systems, well-being management and telemonitoring research applications in various health domains. To support these domains, in a new deployment, definitions in the *"Domain specific data entities ontology"* layer of the model must be performed to add or remove classes and, if needed properties to match the concepts that correspond to the specific domain and scenario of use. A schematic example of this procedure (i.e. the transition from WELCOME scenario to the INLIFE scenario) is depicted in Fig. 5.

The framework presented was developed in order to support the data management requirements of WELCOME EU funded project for patients with COPD with comorbidities. The fifth layer of the model, for this deployment was named "WELCOME Entities" and included entities that were defined not only to describe the domain but also to satisfy end-user application requirements since all patient related data were managed by the framework. Entities and hierarchies of entities were defined as children of FHIR Resource classes. For example, "Biosignal Recording Session" was defined as child of FHIR Resource Administrative class having WELCOME specific children classes "Raw Data Session" and "Vital Signs Session". The resulting framework was deployed in Amazon Web Services[44] cloud infrastructure and was used to manage the data of the WELCOME's pilot studies. Two pilot studies were performed during WELCOME using the cloud deployment of our framework, one in Greece monitoring 17 patients and one in the UK monitoring 14 patients. Both pilot studies had a six-month duration, monitoring almost one month each patient and included also data from a specialized wearable vest device along with the results of sophisticated feature extraction from bio-signals such as multi-lead ECG (Electrocardiography) and EIT (Electro Impedance Tomography). The above feature extraction procedures resulted in big volume of data to be managed by the API as shown in Table 2. The deployed framework operated robustly without any down time and the pilot data analysis confirmed that there was no data loss or data corruption during the system use. The use of AWS Elastic Beanstalk[45] allowed us to effortlessly and automatically scale the number of deployed API instances when needed without downtime.

The framework has been adapted with minor changes in the ontology to support the Guardian Angel tool of the *INdependent LIving support Functions for the Elderly (INLIFE)* EU project platform. Guardian Angel (GA) is a telemonitoring platform that consists of a mobile

[42] http://tools.ietf.org/html/rfc7230.
[43] http://tools.ietf.org/html/rfc7232.
[44] https://aws.amazon.com/.
[45] https://aws.amazon.com/elasticbeanstalk/.

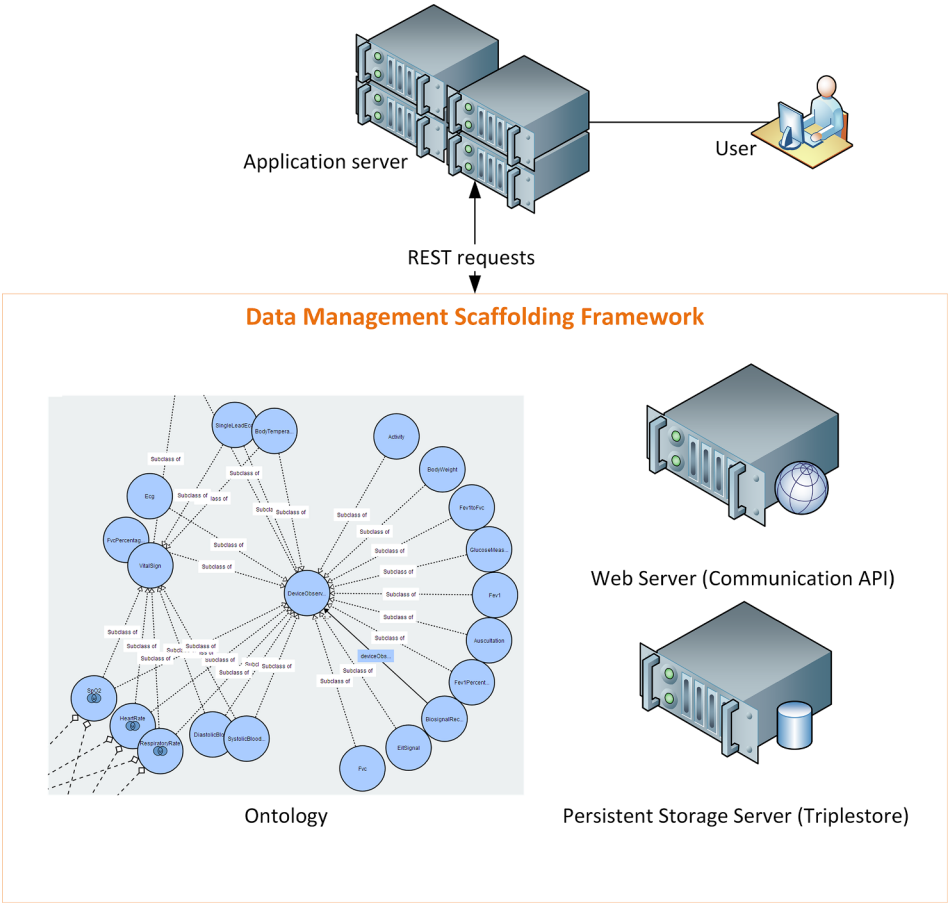

Fig. 4. Flexible Data Management Framework deployment scenario diagram.

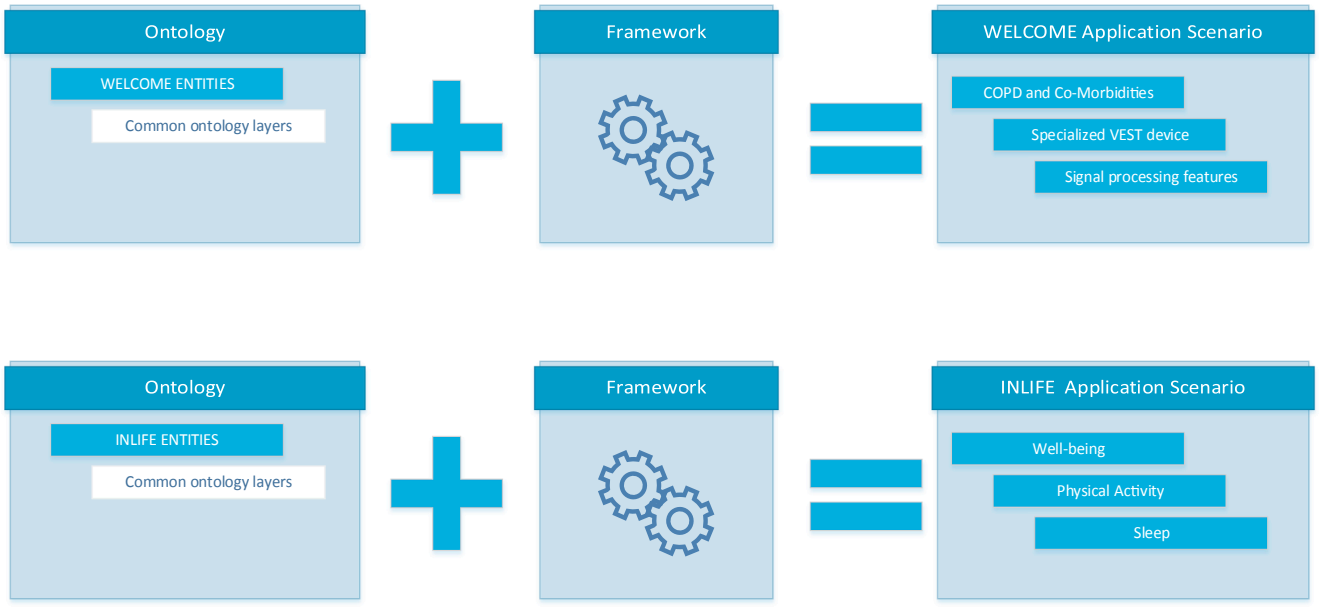

Fig. 5. The use of the scaffolding framework in two scenarios, the "Wearable Sensing and Smart Cloud Computing for Integrated Care to COPD Patients with Comorbidities (WELCOME)" and the "INdependent LIving support Functions for the Elderly (INLIFE)" EU project.

application for monitoring users on the move throughout their daily activities and a web-based review and analysis application for a meaningful presentation of the monitored data by the respective care-givers. The supported sensors for the study included a wrist wearable device and bluetooth portable devices for Peripheral oxygen saturation (SpO2) and Blood pressure measurement. The changes required for our framework to support GA were again limited only in the fifth layer of the ontology ("*Domain specific data entities ontology*") named *"INLIFE Entities"* for this deployment. These changes included removal of classes that were corresponding to specific entities of the WELCOME project,

```
# Temperature out of bounds(too high)
ASK WHERE {
?this (FHIRResources:Observation.value/FHIRct:Quantity.value)/rdf:value ?value .
FILTER (?value > 44) .
  }
```

Listing 1. SPIN constraint for high BodyTemperature.

```
FHIRResources:Device
  rdf:type owl:Class ;
  lomi:hasForeignKey (
      FHIRResources:patient
    ) ;
  lomi:hasUnique (
      FHIRResources:Device.identifier
     ) ;
```

Listing 2. Example definition of a class (i.e. Device).

Table 2
Amount of stored data during the pilot studies of WELCOME and INLIFE Guardian Angel.

	WELCOME	INLIFE GA
Total Number of RDF Triples	2,643,249	40,238,291
Ontology RDF Triples	12,388	12,195
Number of FHIR Resources	116,102	1,879,182
Actual size of Virtuoso DB	274 MB	4.30 GB
File Storage	228 GB 32,363 .edf files, with a total size of 82 GB, signal recordings, 4778 .jpg files with a total size of 257 MB, related to EIT Analysis 1798 .mat (Matlab) files with a total size of 144 GB, related to EIT Analysis 1798 .mp4 video files with a total size of 1.65 GB, EIT Reconstruction 7794 .xml files, with a total size of 100 MB, related to Feature Extraction	2.33 GB (per Patient aggregated data)

(e.g. “Biosignal recording session”) which were no longer applicable in INLIFE. Also specific concepts of the INLIFE domain such as the “Daily Step Count” or “Rem Non-Rem Sleep Ratio” were defined as new classes. The overall effort required for the transition of the model from “WELCOME Entities” to “INLIFE Entities” was in the order of person days.[46] Having defined the ontology, the framework was deployed in local servers to support the pilot study which started at Dec 2016. The framework was managing the synchronization of the pilot data, monitored by the application in almost real-time (at most after 5 min after each measurement was taken), from 92 elderly patients with mild cognitive impairment along with COPD or diabetes or mobility problems for a continuous period of six months. The framework operated with no problems and zero down time throughout both projects. It must be noted that although detailed performance benchmarking was not performed for the complete framework, there was no noticeable difference in response times with that experienced in ordinary SQL based communication APIs.

In Table 2 the amount of the stored data during WELCOME and INLIFE Guardian Angel pilot studies is presented.

Table 2 highlights the use-case differences observed between WELCOME and INLIFE. In the case of WELCOME, the bulk of the data traffic consisted of the exchange of binary data. These were mainly biosignal recordings (.edf files) and files relating to secondary analysis and features. The RDF store’s role was to record every other aspect of the generated data (from the patient or the Healthcare Professional side) alongside the metadata concerning the binary files. This led to a rather small database size when compared to the rest of the storage. On the other hand, in the case of INLIFE the relatively simple format of the recorded data allowed us to directly capture the entirety of the recorded information inside the RDF store. These two, very different, use cases serve as another demonstration of the reusability advantage of our approach.

During the period of the pilot studies, few new requirements were presented. Excluding other parts of the project’s software, these requirements also introduced modifications to the data model and the communication API to enable management of new concepts. These modifications were applied to the framework using the Ontology/Schema service of the communication API to update the semantic model. Upon such a change, the RDF Resources Service is automatically updated providing the required functionality in the already deployed framework.

An example of that procedure was performed during WELCOME project’s pilot. An additional requirement was presented to manage data about *Risk Assessments* that where performed not only by a *Practitioner*, but also assessments originating from the developed Clinical Decision Support System (CDSS). To support this, two modifications were made to the model using the Ontology/Schema service.

[46] Having a concrete input by the domain experts, i.e. having well-defined concepts alongside their respective invariants based on the scenario of use was a prerequisite of this task.

The first was the definition of the *DecisionSupportSystem* class as direct child of the *Device* class to store information regarding the CDSS (e.g. version number). The second was to modify the property performer of *RiskAssessment* to accept also as possible values instances of the new *DecisionSupportSystem* class. These modifications, applied via the Ontology/Schema service, automatically enabled the management by the API of the new concepts and thus fulfilled the requirement.

Although a detailed usability and performance evaluation of the framework was not performed during the aforementioned studies, the results of the deployment in two pilot studies proved the robustness, reusability and flexibility of the presented Data Management Scaffolding Framework. The framework supported successfully the data management operations without data loss or data corruption, with zero downtime and with minor redeployment effort in the transition from WELCOME scenario to INLIFE GA scenario.

6. Discussion

In this paper we present a flexible scaffolding framework for integrated care health data management, focusing on tele-health streaming data. The presented framework is standards-based, semantically enhanced data processing framework and provides the ability to adapt in new use cases or data models using a *single point of change*, without the need for a software rebuild. This advancement can be considered of high significance in Integrated Care scenarios which typically combine multiple healthcare pathways and require adaptation to specific locally applied information workflows or data models. Applying this single-point-of-change principle, only the underlying ontology semantic model would be modified and all the data processing framework components (i.e. communication API endpoints, data storage model, integrity rules) would adjust accordingly without the need of a code rebuild and therefore, without the need for a software engineer.

The framework is based on an expandable OWL/RDF ontology inspired from HL7 FHIR, it supports the exchange of OWL instances serialized in turtle format and implements the appropriate measures to ensure integrity and validity of the exchanged data. Having stored semantically enriched EHR data in the form of RDF graphs also simplifies the reuse of those data in retrospective studies since EHR data are self-explanatory in contrast to proprietary representations in an RDBMS where the explanation of the used model and a transformation to a new representation most probably will require significant manual effort.

The REST API is dynamic, relying solely on the ontology, allowing management of additional entities and properties without any change in the API code, as the respective ontology is the core of the presented scaffolding framework. The framework was deployed with success to support data management in the context of two EU-funded projects focusing on the tele-health domain but referred to different use cases. Despite the different conceptual models of the two pilots, the overall adaptation required minor changes only in the underlying ontology.

Integrated Care scenarios typically refer to large and diverse data collections (e.g. body sensors, EHR data, administrative data, etc.) and therefore they are one of the Big Data paradigm typical use cases. The Big Data paradigm is often presented using the so-called 4Vs, i.e. *volume, velocity, variety and veracity*.[47] The presented framework could facilitate the handling of challenges imposed by data *variety,* i.e. the different forms of data, and *veracity,* i.e. the data uncertainty mostly due to poor data quality. The adoption of Linked Data as the presented framework's main data paradigm enhances the ability to integrate a *variety* of data schemes (i.e. different formats, different semantics, etc.) using widely accepted standards focused on data interlinking. Furthermore, the inherent semantic annotation of data used can significantly improve the respective data *veracity* as it reduces ambiguity in data description in all of the data processing steps.

One of the main issues faced during the development of the project was the immaturity of HL7 FHIR leading to changes in FHIR resources that sometimes needed to be reflected in the ontology. Continuous integration of HL7 FHIR changes was facilitated because of the framework's single point of change paradigm.

Another decision that had to be made was regarding the enforcement of validation rules. The consequence of selecting SPIN to perform constraint checking is that the operation cannot be natively conducted on the persistence layer, but has to be located one layer above it, between the storage and the data management web service. Although, this seems a drawback, it is a standard approach, even in relational databases, to check input data for validity and quality before delegating them to the RDBMS. Also, as it was already mentioned the actual validation rules rely on the ontology so again there is a single point of maintenance.

Overall, this framework succeeds its targets: (a) we prove that it is reusable by showing its easy adaptation from one domain (WELCOME project) to the next (INLIFE project), (b) we maintain a cost-effective solution by employing open source software that is easily maintained and expanded, (c) it is scalable and sustainable, by being minimally platform dependent and by employing a NOSQL storage engine, as well as a self-explanatory communication API, which can be well supported in cloud services. The HL7 FHIR, medical terminologies and linked data layer set the basis for interoperability and versatile use of the data and in contrast to the reviewed existing approaches in the framework HL7 FHIR and RDF/OWL are used in all data processing steps (i.e. data exchange, storage, querying, reasoning). The aforementioned qualities constitute a framework that can robustly support Integrated Care Services.

Among the limitations of this work that would need further attention while continuing this work are: (a) a full Interoperability functionality, that could be offered by a FHIR server/client exchange mechanism, in order to communicate with standard EHR systems in the Health IT, (b) the possible use of additional data exchange formats, such as JSON-LD,[48] to reduce the usability barrier related to the steep learning curve of RDF, (c) the incorporation of mechanisms for handling data streams in a more optimized manner. This would potentially be dealt with by combining the triplestore with columnar databases [47] or stream databases [48], (d) investing on mechanisms for easy and flexible querying of the data, based on the semantics and linked data qualities. It should be noted that due to legal and ethical reasons we do not provide access to the data collected during the frameworks pilot applications. Still, we identify the need to use Linked Data as the main data management paradigm, as data collected in Integrated Care scenarios could have important secondary uses in the context of research and this is one of the presented framework's objectives. Finally, apart from the common security and data safety measures, such as the use of encrypted communication and data redundancy solutions that were applied during the two deployments, the framework has to support specialized security practices that would refer to the application domain, such as, defining and enforcing authorization access rules for specific data resources. The identified extensions would further mature the framework towards a digital single market in health.

Acknowledgements

The authors would like to thank Prof. Sanjay Mehrotra from the IEMS Dept., Northwestern University, Evanston, IL for the critical review of the manuscript. The research leading to these results has been partially funded from the FP-ICT Programme under Grant Agreement no 611223 – WELCOME (http://www.welcome-project.eu/).

[47] https://www.ibmbigdatahub.com/infographic/four-vs-big-data.

[48] https://json-ld.org (JSON for Linking Data).

References

[1] D. Kodner, All together now: a conceptual exploration of integrated care, Healthc. Q. 13 (2009) 6–15, https://doi.org/10.12927/hcq.2009.21091.

[2] I. Chouvarda, N.Y. Philip, P. Natsiavas, V. Kilintzis, D. Sobnath, R. Kayyali, J. Henriques, R.P. Paiva, A. Raptopoulos, O. Chetelat, N. Maglaveras, WELCOME - Innovative integrated care platform using wearable sensing and smart cloud computing for COPD patients with Comorbidities, 2014 36th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBC 2014 (2014) 3180–3183, https://doi.org/10.1109/EMBC.2014.6944298.

[3] C.W. Kelman, A.J. Bass, C.D.J. Holman, Research use of linked health data – A best practice protocol, Aust. N. Z. J. Public Health. 26 (2002) 251–255, https://doi.org/10.1111/j.1467-842X.2002.tb00682.x.

[4] M. Aranguren, J. Fernandez-Breis, M. Dumontier, Special issue on linked data for health care and the life sciences, Semant. Web. 5 (2014) 99–100, https://doi.org/10.3233/SW-130115.

[5] B. Tilahun, T. Kauppinen, Potential of linked open data in health information representation on the semantic web, in: SWAT4LS, 2012: pp. 3–6.

[6] H. Liyanage, S.-T. Liaw, C. Kuziemsky, A.L. Terry, S. Jones, J.K. Soler, S. de Lusignan, The Evidence-base for using ontologies and semantic integration methodologies to support integrated chronic disease management in primary and ambulatory care: realist review. Contribution of the IMIA primary health care informatics WG, Yearb. Med. Inform. 8 (2013) 147–154.

[7] S.T. Liaw, A. Rahimi, P. Ray, J. Taggart, S. Dennis, S. de Lusignan, B. Jalaludin, A.E.T. Yeo, A. Talaei-Khoei, Towards an ontology for data quality in integrated chronic disease management: a realist review of the literature, Int. J. Med. Inform. 82 (2013) 10–24, https://doi.org/10.1016/j.ijmedinf.2012.10.001.

[8] L. Lessard, W. Michalowski, M. Fung-Kee-Fung, L. Jones, A. Grudniewicz, Architectural frameworks: defining the structures for implementing learning health systems, Implement. Sci. 12 (2017) 78, https://doi.org/10.1186/s13012-017-0607-7.

[9] T. Berners-Lee, Linked Data, 52 (2011) 284–292, < https://www.w3.org/DesignIssues/LinkedData.html > (accessed February 18, 2015).

[10] H. Darwen, C.J. Date, The third manifesto, ACM SIGMOD Rec. 24 (1995) 39–49, https://doi.org/10.1145/202660.202667.

[11] I. Chouvarda, V. Kilintzis, N. Beredimas, P. Natsiavas, E. Perantoni, I. Vogiatzis, V. Vaimakakis, N. Maglaveras, Clinical flows and decision support systems for coordinated and integrated care in COPD, 3rd IEEE EMBS Int. Conf. Biomed. Heal. Informatics, BHI 2016, 2016, https://doi.org/10.1109/BHI.2016.7455938.

[12] S. Hanke, C. Mayer, O. Hoeftberger, H. Boos, R. Wichert, M.-R. Tazari, P. Wolf, F. Furfari, universAAL - An open and consolidated AAL platform, in: Ambient Assited Living 4 Dtsch. AALKongress, 2011, doi:10.1007/978-3-642-18167-2_10.

[13] H. Gokalp, J. de Folter, V. Verma, J. Fursse, R. Jones, M. Clarke, Integrated telehealth and telecare for monitoring frail elderly with chronic disease, Telemed. e-Health. (2018), https://doi.org/10.1089/tmj.2017.0322 tmj.2017.0322.

[14] K. Avgerinakis, A. Briassouli, I. Kompatsiaris, Recognition of activities of daily living for smart home environments, in: Intell. Environ. (IE), 2013 9th Int. Conf., 2013, doi:10.1109/IE.2013.37.

[15] J. Schubert, S. Ghulam, L. Prieto-González, Integrated care concept using smart items and cloud infrastructure, Procedia Comput. Sci. 63 (2015) 439–444, https://doi.org/10.1016/J.PROCS.2015.08.365.

[16] L. Prieto González, C. Jaedicke, J. Schubert, V. Stantchev, Fog computing architectures for healthcare, J. Inform., Commun. Ethics Soc. 14 (2016) 334–349, https://doi.org/10.1108/JICES-05-2016-0014.

[17] A. González-Ferrer, M. Peleg, B. Verhees, J.-M. Verlinden, C. Marcos, Data Integration for Clinical Decision Support Based on openEHR Archetypes and HL7 Virtual Medical Record, Springer, Berlin, Heidelberg, 2013, pp. 71–84, , https://doi.org/10.1007/978-3-642-36438-9_5.

[18] C. Martínez-Costa, S. Schulz, Ontology content patterns as bridge for the semantic representation of clinical information, Appl. Clin. Inform. 5 (2014) 660–669, https://doi.org/10.4338/ACI-2014-04-RA-0031.

[19] T. Mallaug, K. Bratbergsengen, Long-term temporal data representation of personal health data, Adv. Databases Inf. Syst. 9th East Eur. Conf. ADBIS 2005, Tallinn, Est. Sept. 12–15, Proc., Springer, Berlin, Heidelberg, 2005, pp. 379–391, , https://doi.org/10.1007/11547686_28.

[20] N. Lasierra, A. Alesanco, S. Guillén, J. García, A three stage ontology-driven solution to provide personalized care to chronic patients at home, J. Biomed. Inform. 46 (2013) 516–529, https://doi.org/10.1016/j.jbi.2013.03.006.

[21] D. Riaño, F. Real, J.A. López-Vallverdú, F. Campana, S. Ercolani, P. Mecocci, R. Annicchiarico, C. Caltagirone, An ontology-based personalization of health-care knowledge to support clinical decisions for chronically ill patients, J. Biomed. Inform. 45 (2012) 429–446, https://doi.org/10.1016/j.jbi.2011.12.008.

[22] S.R. Abidi, S. Hussain, M. Shepherd, Ontology-based modeling of clinical practice guidelines: a clinical decision support system for breast cancer follow-up interventions at primary care settings, Stud. Health Technol. Inform. 129 (2007) 845–849.

[23] H. Knublauch, D. Oberler, P. Tetlow, E. Wallace, A semantic web primer for object-oriented software developers, W3C Ed. Draft (2006) < http://www.w3.org/TR/ > .

[24] P.L. Whetzel, N.F. Noy, N.H. Shah, P.R. Alexander, C. Nyulas, T. Tudorache, M.A. Musen, BioPortal: enhanced functionality via new Web services from the National Center for Biomedical Ontology to access and use ontologies in software applications, Nucleic Acids Res. 39 (2011) W541–W545, https://doi.org/10.1093/nar/gkr469.

[25] P.J. Sadalage, M. Fowler, NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence, Pearson Education, 2013.

[26] F. Gandon, G. Schreiber, RDF 1.1 XML Syntax, W3C Recomm. 25 Febr. 2014 (2016) 1–35, < http://www.w3.org/TR/2014/REC-rdf-syntax-grammar-20140225/ > (accessed October 1, 2014).

[27] C.B. Aranda, O. Corby, S. Das, L. Feigenbaum, P. Gearon, B. Glimm, S. Harris, S. Hawke, I. Herman, N. Humfrey, N. Michaelis, C. Ogbuji, M. Perry, A. Passant, A. Polleres, E. Prud'hommeaux, A. Seaborne, G.T. Williams, SPARQL 1.1 Overview, W3C Recomm (2013), < http://www.w3.org/TR/sparql11-overview/ > (accessed October 1, 2014).

[28] V. Kilintzis, N. Beredimas, I. Chouvarda, Evaluation of the performance of open-source RDBMS and triplestores for storing medical data over a web service, in: 2014 36th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBC 2014, IEEE, 2014, pp. 4499–4502, doi:10.1109/EMBC.2014.6944623.

[29] M. Voigt, A. Mitschick, Yet another triple store benchmark? practical experiences with real-world data, Citeseer. (n.d.).

[30] M. Morsey, J. Lehmann, S. Auer, A.C. Ngonga, Ngomo, usage-centric benchmarking of RDF triple stores, AAAI Conf. Artif. Intell. 26 (2012) 2134–2140.

[31] C. Bizer, T. Heath, T. Berners-Lee, Linked data-the story so far, Int. J. Semant. Web Inf. Syst. 5 (2009) 1–22.

[32] R. Reiter, On closed world data bases, Readings Artif. Intell. Elsevier, 1981, pp. 119–140, , https://doi.org/10.1016/B978-0-934613-03-3.50014-3.

[33] T. Bosch, E. Acar, A. Nolle, K. Eckert, The role of reasoning for RDF validation, Proc. 11th Int. Conf. Semant. Syst. (2015) 33–40.

[34] G. Schenner, S. Bischof, A. Polleres, S. Steyskal, Integrating distributed configurations with RDFS and SPARQL, CEUR Workshop Proc. (2014) 9–15.

[35] W3C, SPIN - Overview and Motivation, (2011), < https://www.w3.org/Submission/spin-overview/ > .

[36] A. Spear, Ontology for the twenty first century: An introduction with recommendations, Inst. Form. Ontol. Med. Inf. Sci. Saarbrücken, Ger, 2006.

[37] V. Kilintzis, N. Beredimas, P. Natsiavas, I. Chouvarda, N. Maglaveras, A fully functional HL7 FHIR based ontology for telehealth data, 17th Int. HL7 Interoperability Conf. IHIC 2017, 2017, pp. 19–26.

[38] N. Beredimas, V. Kilintzis, I. Chouvarda, N. Maglaveras, A reusable ontology for primitive and complex HL7 FHIR data types, Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS (2015) 2547–2550, https://doi.org/10.1109/EMBC.2015.7318911.

[39] C. Ogbuji, SPARQL 1.1 Graph Store HTTP Protocol, W3C (2013), < http://www.w3.org/TR/sparql11-http-rdf-update/ > (accessed October 1, 2014).

[40] T. Bosch, K. Eckert, Towards description set profiles for RDF using SPARQL as intermediate language, Int. Conf. Dublin Core Metadata Appl. (2014) 129–137.

[41] T. Bosch, K. Eckert, Requirements on RDF constraint formulation and validation, Int. Conf. Dublin Core Metadata Appl. (2014) 95–108.

[42] C. Fürber, M. Hepp, Using {SPARQL} and {SPIN} for data quality management on the semantic web, in: 13th Int. Conf. Bus. Inf. Syst., 2010, pp. 35–46.

[43] C. Pautasso, RESTful web services: principles, patterns, emerging technologies, Web Serv. Found. Springer New York, New York, NY, 2014, pp. 31–51, , https://doi.org/10.1007/978-1-4614-7518-7_2.

[44] M. Masse, REST API Design Rulebook: Designing Consistent RESTful Web Service Interfaces, 2011.

[45] T.I. Hunter, CONSUMER-CENTRIC API DESIGN., BLURB, 2015.

[46] B. Mulloy, Web API design, 2013.

[47] D.J. Abadi, P.A. Boncz, S. Harizopoulos, Column-oriented database systems, Proc. VLDB Endow 2 (2009) 1664–1665, https://doi.org/10.14778/1687553.1687625.

[48] D. Carney, U. Çetintemel, M. Cherniack, C. Convey, S. Lee, G. Seidman, M. Stonebraker, N. Tatbul, S. Zdonik, Monitoring streams: a new class of data management applications, in: Proc. 28th Int. Conf. Very Large Data Bases, 2002, pp. 215–226.