FP7-ICT-611140 CARRE

Project co-funded by the European Commission
under the Information and Communication Technologies
(ICT) 7th Framework Programme

D.3.1. Aggregator Module Generic Design

J. Piwinski, R. Kloda, A. Third

July 2014

CARRE Contacts

Project Coordinator:	Eleni Kaldoudi	kaldoudi@med.duth.gr
Project Manager:	Koula Zigeridou	kzygerid@alex.duth.gr

DUTH Democritus University of Thrace	Eleni Kaldoudi	kaldoudi@med.duth.gr
OU The Open University	John Domingue	john.domingue@open.ac.uk
BED Bedfordshire University	Enjie Liu	Enjie.Liu@beds.ac.uk
VULSK Vilnius University Hospital Santariskių Klinikos	Domantas Stundys	Domantas.Stundys@santa.lt
KTU Kaunas University of Technology	Arunas Lukosevicius	arunas.lukosevicius@ktu.lt
PIAP Industrial Research Institute for Automation & Measurements	Roman Szewczyk	rszewczyk@piap.pl

Disclaimer

CARRE is a Specific Targeted Research Project partially funded by the European Union, under FP7-ICT-2013-10, Theme 5.1. “Personalized health, active ageing & independent living”.

Document Control Page

Project

Contract No.:	611140
Acronym:	CARRE
Title:	Personalized Patient Empowerment and Shared Decision Support for Cardiorenal Disease and Comorbidities
Type:	STREP
Start:	1 November 2013
End:	31 October 2016
Programme:	FP7-ICT-2013.5.1
Website:	http://www.carre-project.eu/

Deliverable

Deliverable No.:	D.3.1
Deliverable Title:	Aggregator Module Generic Design
Responsible Partner:	PIAP - Roman Szewczyk
Authors:	J. Piwinski, R. Kloda, A. Third
Input from:	all partners
Peer Reviewers:	E. Kaldoudi (DUTH), G. Gotsis (OU)
Task:	T.3.1. Generic overall design of aggregator module
Task duration:	5 months: 1 March 2014 to 31 July 2014
Work Package:	WP3: Data & Metadata Harvesting
Work Package Leader:	KTU – Arunas Lukosevicius

Due Date:	31 July 2014
Actual Delivery Date:	31 July 2014
Dissemination Level:	PU
Nature:	R
Files and format:	Deliverable report: 1 pdf file
Version:	03.2
Status:	☐ Draft ☒ Consortium reviewed ☒ WP leader accepted ☒ Coordinator accepted ☐ EC accepted

Document Revision History

Version	Date	Modifications	Contributors
v01.0	25 June 2014	new content	Jan Piwinski
v01.1	11 July 2014	remarks	Eleni Kaldoudi
v01.2	16 July 2014	remarks	Vaidotas Marozas
v01.3	24 July 2014	2nd version of deliverable.	Jan Piwinski
v01.4	30 July 2014	OU contributions	Allan Third
v01.5	30 July 2014	remarks	George Gkotsis
v02.1	31 July 2014	Allan Third – OU contributions	Allan Third
v03.0	31 July 2014	revision based on reviews	Jan Piwinski
v03.1	31 July 2014	editing	Arunas Lukosevicius
v03.2	31 July 2014	editing	Eleni Kaldoudi

Table of Contents

List of Figures

List of Tables

N/A

Executive Summary

This document discusses the generic design of all aggregators for all the types of data sources addressed in CARRE system. Each aggregator is going to be implemented as a web service and will be implementing the CARRE RDF scheme, as this is being developed in T.2.4. Also, CARRE aggregators will communicate with the intended data source via interfaces based on generic and/or domain specific standards in order to harvest and process any relevant data and/or metadata. The objective of this report is to demonstrate how the sensors data could be processed to achieve seamless communication between system components and provide data coherence.

About CARRE

CARRE is an EU FP7-ICT funded project with the goal to provide innovative means for the management of comorbidities (multiple co-occurring medical conditions), especially in the case of chronic cardiac and renal disease patients or persons with increased risk of such conditions.

Sources of medical and other knowledge will be semantically linked with sensor outputs to provide clinical information personalised to the individual patient, so as to be able to track the progression and interactions of comorbid conditions. Visual analytics will be employed so that patients and clinicians will be able to visualise, understand and interact with this linked knowledge and also take advantage of personalised empowerment services supported by a dedicated decision support system.

The ultimate goal is to provide the means for patients with comorbidities to take an active role in care processes, including self-care and shared decision-making, and to support medical professionals in understanding and treating comorbidities via an integrative approach.

Terms and Definitions

The following are definitions of terms, abbreviations and acronyms used in this document.

Term	Definition
API	Application Programming Interface
CAM	CARRE Aggregator Module
CCM	Communication Client Module
CEM	Communication Engine Module
CRUD	Create, Read, Update, Delete
DoW	Description of work
DS	Data Source
DTM	Data Translation Module
FM	Filtering Module
MPLS	Multi Protocol Label Switching
PKI	Public Key Infrastructure
QoS	Quality of Service
RDF	Resource Description Framework
REST	Representational State Transfer
TLS	Transport Layer Security
UDDI	Universal Description Discovery and Integration
UML	Unified Modeling Language
USDL	Universal Service Description Language
W3C	World Wide Web Consortium
WSDL	Web Service Description Language

1. Introduction

This report presents the main system components and detailed functional architecture of the aggregator, CARRE schema interface and CARRE sematic repository functionalities and describes web services development, functionality and technical requirements as well as the generic design of aggregator module and its use cases.

The data sources addressed in the CARRE project include three main data categories:

1) **Personal physiological data from wearable, portable, mobile devices** the patient carries in non-clinically controlled environments, i.e. at home, at work, etc.; challenges in this source category include the selection and parameterization of the sensors, robustness and security of the sensor network, the integration of heterogeneous sensor interfaces and the extraction of semantic information and thus the semantic description for sensor signals.

2) **Personal data** for the patient that refers to information **on medical condition** as well as on **lifestyle and other environmental and emotional issues** (location, mood, intention, etc.). Medical data can potentially be retrieved from personal health records (if available) or by hospital/clinical information systems (where applicable) or provided by the patients itself via the CARRE platform. Other personal information on patient lifestyle will be harvested from on-line social presence of the patient (e.g. social media). Challenges in this source category include semantic integration of personal medical data (via healthcare standards, including ontologies and medical information communication standards); semantic text mining of social media (via text annotation and entity recognition techniques); and data security and privacy issues.

3) **Medical data sources**, including scientific literature and related **medical evidence from authoritative on-line databases and educational material** mainly addressed to the patient. For this type of sources mainly metadata will be collected as well as links to the actual data. Challenges here include source discovery, integrating heterogeneous sources (via generic and domain specific standards); and semantic annotation of data wherever this is not already provided by the source API or if provided is inadequate.

The particular data sources identified are described in detail in CARRE Deliverable D.2.3. The project will use a generic common design for the aggregators of all these data sources. Common principles underlining the development of all aggregators are the following: each aggregator (implemented as web service) will implement the CARRE RDF scheme and will communicate with the intended data source (via interfaces based on generic and/or domain specific standards) in order to harvest any relevant data and/or metadata. An overview of our approach (as already described in the DoW) is shown in Figure 1.

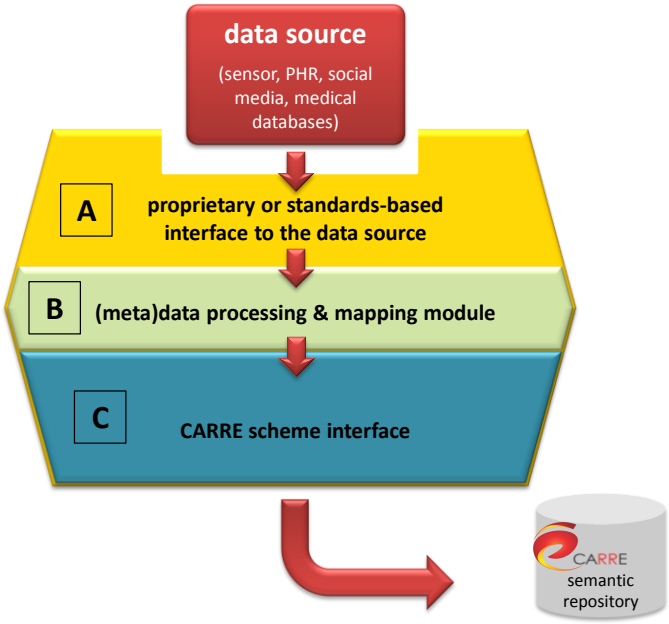


Figure 1. Generic design of a CARRE data source aggregator.

Part A of the aggregator is specific to each individual data source and will either implement a proprietary interface for this source or (if the data source supports it) implement specific communication standards implemented by the data source. Part B of the aggregator will undertake any processing required, and this is specific to each data type (i.e. this part will be the same for different sensors/sources that provide the same data, e.g. mobile sensors from different vendors all measuring the same physiological quantity). Finally, Part C will be implementing a generic interface with CARRE repository via the common CARRE scheme. All aggregator component design and specifications will be the outcome of this task.

Section 2 introduces the generic design of the aggregator module and its use cases, and presents a detailed functional architecture for the aggregator. Section 3 section presents the CARRE schema interface and CARRE semantic repository functionalities. Finally, Section 4 presents some background information on web services technology, which will be used for the implementation of the aggregator.

2. Generic Design of the Aggregator

This chapter presents the generic design of the aggregator. First, we present use cases specific to the aggregator module. Then we present the functional architecture of the aggregator using UML diagrams, describing both the structure of the aggregator and its functionality.

2.1. Aggregator use cases

As a starting point for the generic design of the aggregator, a list of use cases has been identified in order to understand the role of this entity in the CARRE system and to model its behavior.

In the various use cases identified, the following actors are involved:

- The *Operator* performing management and configuration actions on the Aggregator;
- The *Data Source* is interfaced with the Aggregator;
- The *CARRE repository*, with which the Aggregator is interfaced;
- The *Aggregator*, which provides interconnection between the Data source and the CARRE repository.

All the use cases can be grouped into two categories:

- *System management*: includes use cases handled by the operator to configure and perform management activities on the aggregator;
- *Information update*: includes the use case that describes how the aggregator treats new raw data coming from data source.

In the following subsections a detailed description of each elementary use case is provided.

2.1.1. Use Case: Data source information update

ID	AM_UC1
Title	Data source information update
Goal	New data is available at data source
Domain	Data management/monitor
Description	When the data source has new information available, the Aggregator must take this information, decide if it is useful for filtering capability and, in positive case, send it to the CARRE repository. The Aggregator will take care to translate it into the proper CARRE data format before the CARRE repository is reached
Participants	Data Source; Aggregator
Results	The Aggregator collects the new information, filters it and translates it in the

	proper data format and sends it to the CARRE repository
Pre-conditions	The Aggregator has been correctly setup, information thresholds have been configured on the basis of system needs
Post-conditions	-
Incoming Information	Raw data from the data sources
Outgoing information	Information Identifiers, Information Values
Flow of events	The Aggregator receives the raw data from the all sources, filters it and in case translates it into CARRE proper data format. Aggregator provides the new information to the CARRE repository.

2.1.2. Use Case: Policy configuration

ID	AM_UC2
Title	Policy configuration
Goal	New policy to be introduced in the system or modification of existing policies (including deletion of policies)
Domain	Data management/monitor
Description	The Operator of Data Source (DS) configures the Aggregator. Namely, it cans insert/delete/modify the policies to control the filtering and aggregation capabilities of the data source, on the basis of the needs of the system.
Participants	Initiator: Operator Recipient: Aggregator
Results	The Aggregator policies are updated
Pre-conditions	The Aggregator has been correctly setup
Post-conditions	-
Incoming Information	Updated policy
Outgoing information	Acknowledgement of the successful policy update process
Flow of events	Operator accesses the Aggregator configuration manager and makes configuration operations of the new policy or update/delete existing policies. The Aggregator is reconfigured and an acknowledgment of the successful operation is sent back to the operator

2.1.3. Use Case: Metadata structure configuration

ID	AM_UC3
Title	Metadata structure configuration
Goal	New metadata structure to be introduced in the system or selection of an already stored metadata structure
Domain	Data management/monitor
Description	The Operator of DS can configure the Aggregator in order to indicate the metadata structure of the data to be transmitted to the CARRE repository. Namely, it can insert/delete/modify the metadata structure and in addition the Aggregator can store several metadata structures so that the operator has just to select the proper one.
Participants	Initiator: Operator Recipient: Aggregator
Results	The metadata structure is updated
Pre-conditions	The Aggregator has been correctly setup
Post-conditions	-
Incoming Information	New metadata structure or command to select one already in stored in the

	Aggregator
Outgoing information	Acknowledgement of the successful metadata structure update process
Flow of events	Operator accesses the Aggregator configuration manager and makes configuration operations of the new metadata structure or update/delete/select existing ones. The Aggregator is reconfigured and an acknowledgment of the successful operation is sent back to the operator

2.1.4. Use Case: Communication protocol configuration

ID	AM_UC4
Title	Communication protocol configuration
Goal	Selection of a communication protocol to interface the Aggregator with the CI monitoring system or installation of a new communication protocol
Domain	Data management/monitor
Description	The Operator of DS can select the communication protocol to be used to connect the Aggregator and the data source. In particular, the Aggregator will be programmes with several most common communication protocols and the operator can easily select one of them. In addition, the operator can also install a new communication protocol.
Participants	Initiator: Operator Recipient: Aggregator
Results	The communication protocol is configured or installed
Pre-conditions	The Aggregator has been correctly setup
Post-conditions	-
Incoming Information	Command to select an already installed protocol or new communication protocol to be installed
Outgoing information	Acknowledgement of the successful configuration process
Flow of events	Operator accesses the Aggregator configuration manager and makes configuration operations in order to select one of the already installed communication protocol or to install a new communication protocol. The Aggregator is reconfigured and an acknowledgment of the successful operation is sent back to the operator

2.1.5. Use Case: Aggregator testing

ID	AM_UC5
Title	Aggregator testing
Goal	Need to test Aggregator operation
Domain	Data management/monitor
Description	The operator in a CI can test the correct behaviour of the local adaptor.
Participants	Initiator: Operator Recipient: Operator
Results	Test information is collected by the operator
Pre-conditions	The Aggregator has been correctly setup
Post-conditions	-
Incoming Information	Operator request of functional testing of Aggregator functionalities
Outgoing information	Test results
Flow of events	Operator accesses the Aggregator and makes test requests. The Aggregator performs the tests and provides the operator with the result of the tests.

2.2. Aggregator detailed functional architecture

The main functional modules of the generic aggregator are shown in Figure 2 and include the following: Communication Client, Filter, Data translator, Communication engine as well as interfaces: on the side of CARRE repository and data sources (DS).

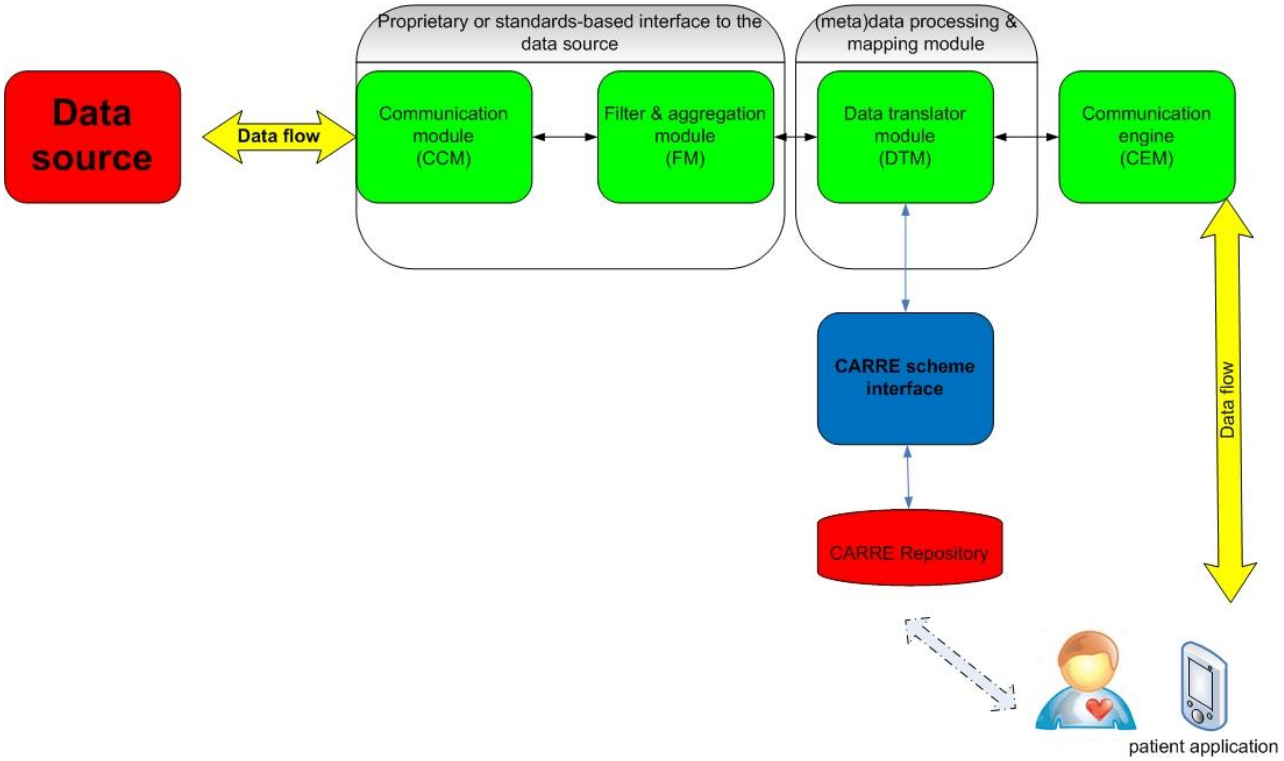


Figure 2. CARRE Aggregator overall schematic block diagram.

In the following subsections, a detailed description of each module is provided.

2.2.1. Communication Client Module

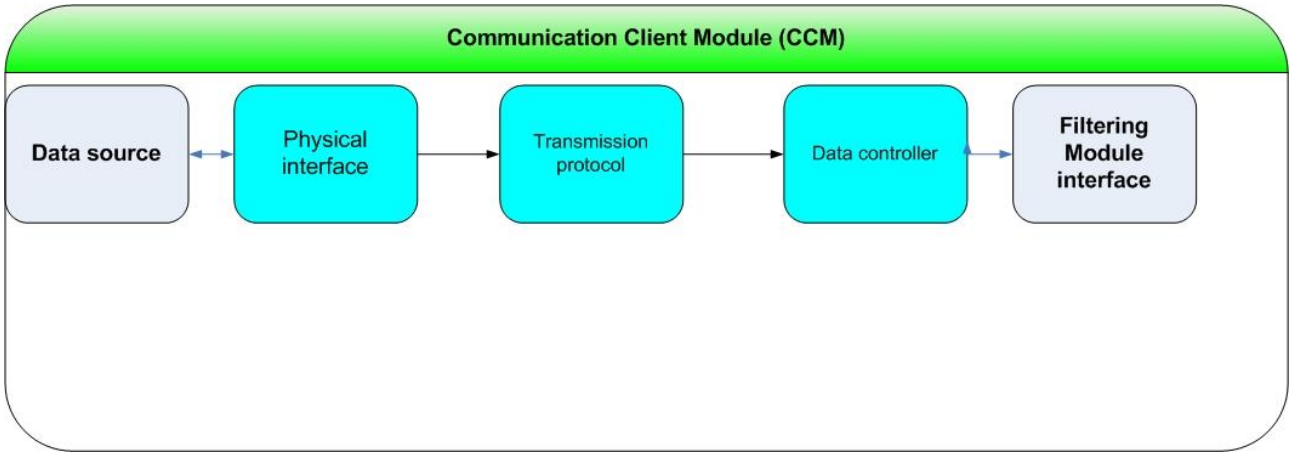


Figure 3. Communication Client Module detailed diagram.

Communication Client Module provides physical connection to DS. Transmission protocol stores the most used communication protocols, in order to allow reconfigurability. Only one selected transmission protocol is

used each time. Data coming from the DS are prepared to be stored in input buffer by data controller. This Client guarantees flexibility and adaptability of Aggregator Module from the point of view of use with different data sources.

From data flow point of view DS interface is based on TCP/IP protocol. Physical interface utilizes IEEE 802.3 standard protocol.

This module enables the configuration of all the functionalities for the administration of the Aggregator, including communication protocol setup (to configure the communication protocol with the DS), permission setup (to configure access security policies), setup filtering and aggregation parameters (to configure the raw data filtering and aggregation policies) and select defined structure output data format (to configure the metadata structure of the data to be given as output to the CARRE repository).

2.2.2. Filtering Module

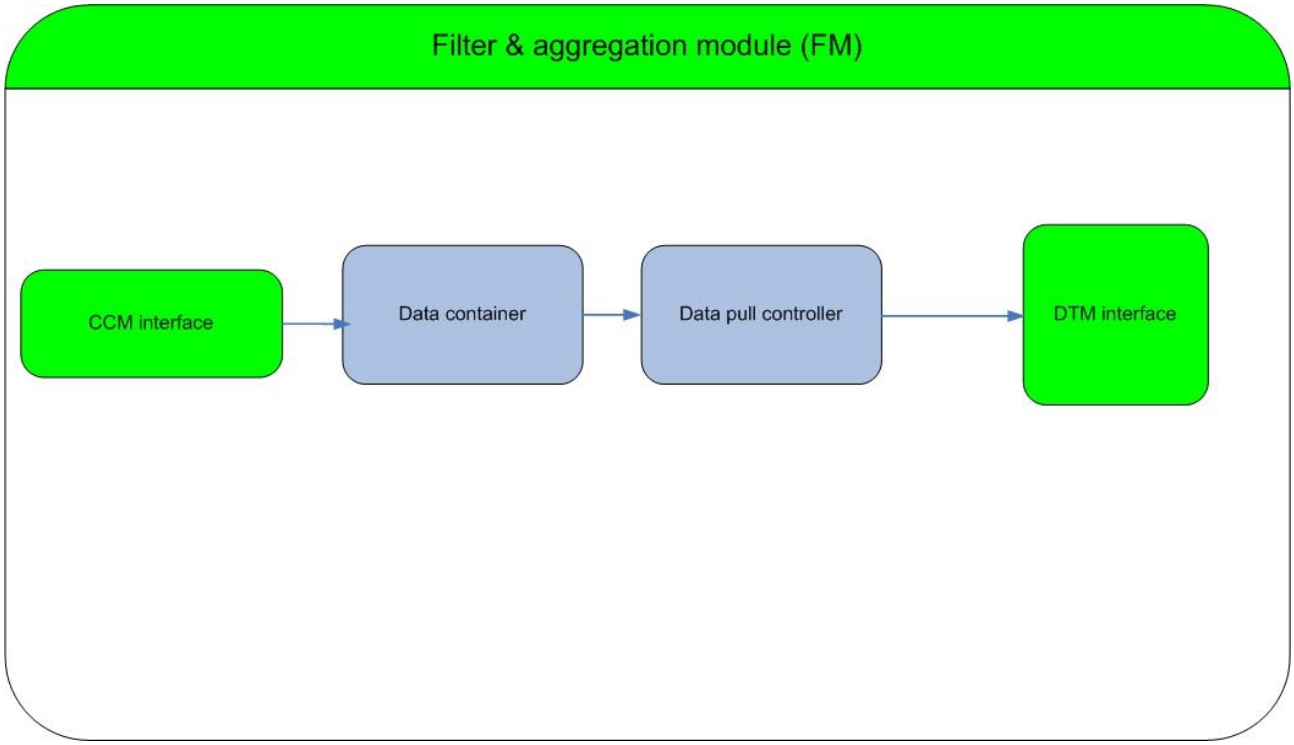


Figure 4. Filtering Module detailed diagram.

The Filtering module is responsible for receiving from buffer all restricted set of information and provide them to Data Translation Module (DTM). This module is under control of Data pull controller containing specialized scripts for data flow control.

This module will control data aggregation on the basis of filtering and aggregation rules implemented in specialized masks. Both aggregation data controller and filtering data controller are synchronized by real time according to the needs of data flow.

Filtering and aggregation mask is generated automatically on the basis of data provided by the operator into the filtering module.

2.2.3. Data Translator Module

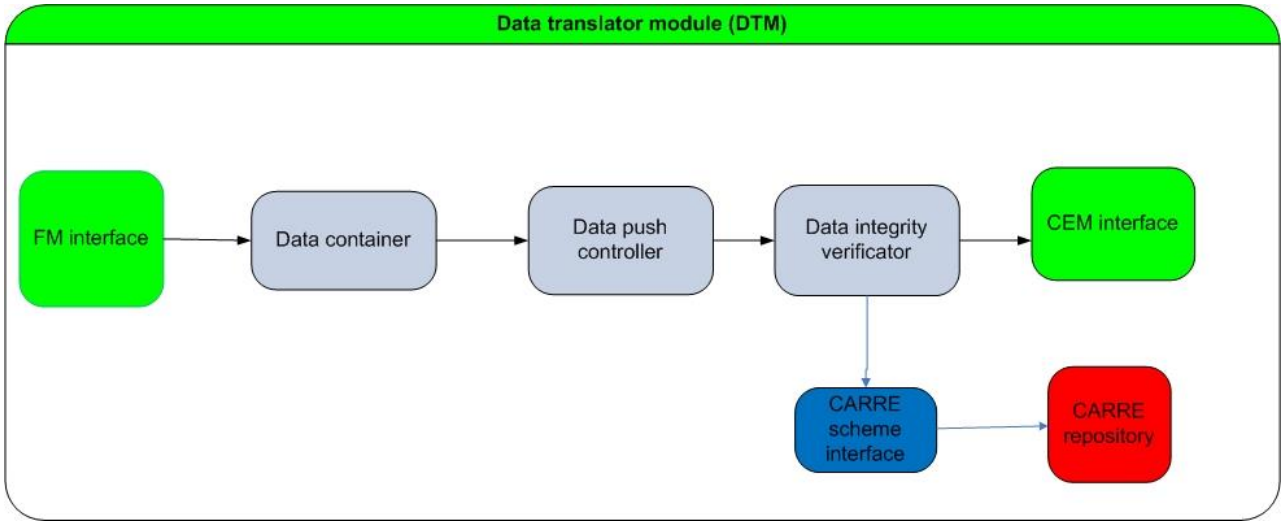


Figure 5. Data Translation Module detailed diagram.

The Data Translator Module receives data from the special buffer and prepares all the information to be stored in CARRE Repository. Moreover data can be published. During operation of the Data Translator Module. data integrity is simultaneously verified.

2.2.4. CARRE scheme interface

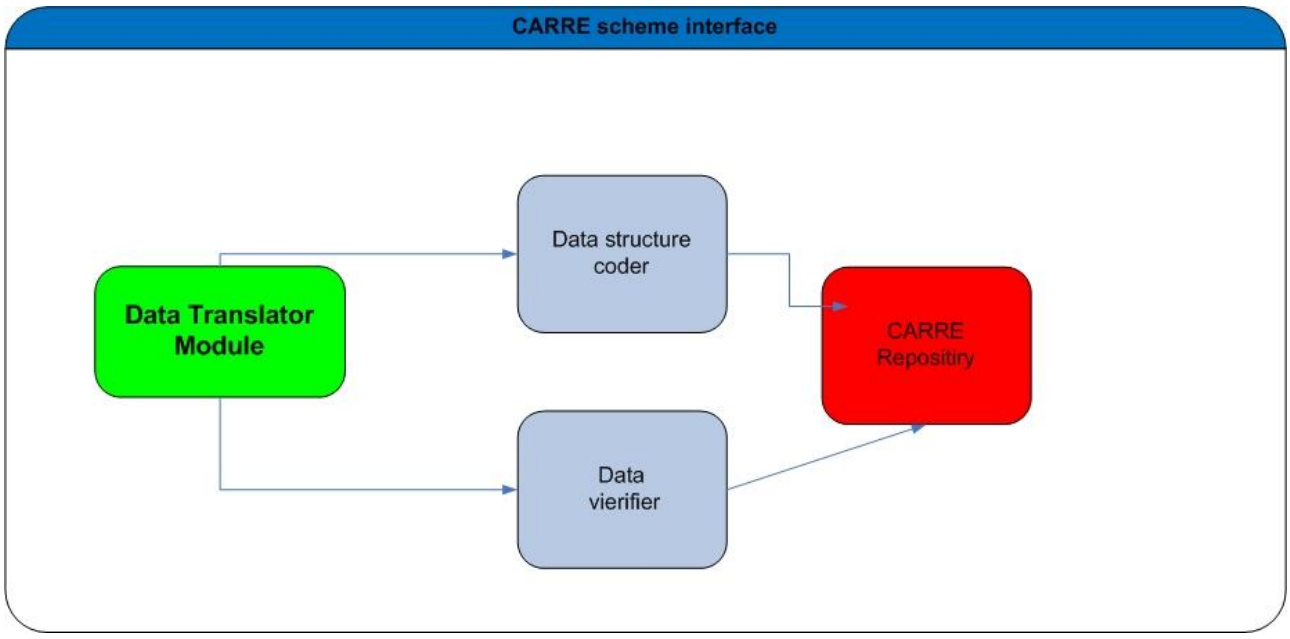


Figure 6. CARRE scheme interface.

This interface implements the mechanisms that correct the format output from Aggregator to CARRE Repository. It also provides rules for the verification of data integrity. Moreover the data structure coder is supervised by the Data verifier to avoid data corruption.

2.2.5. Communication Engine Module

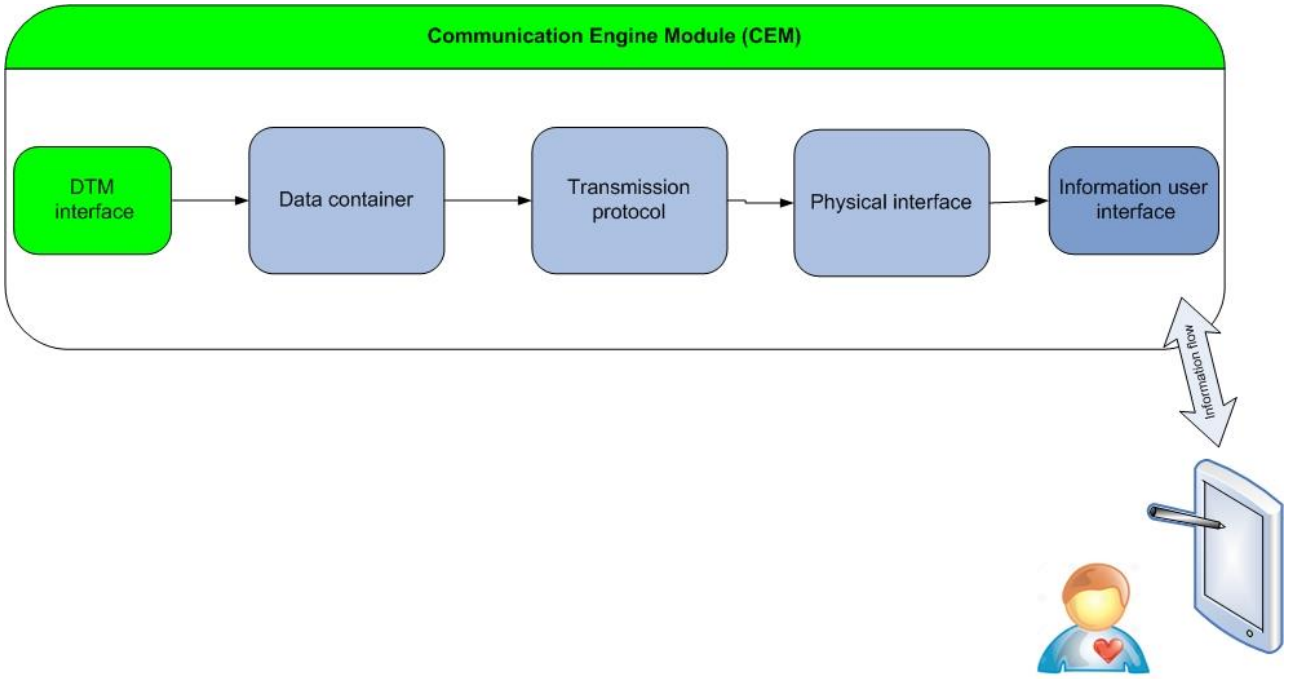


Figure 7. Communication Engine Module detailed diagram.

Communication Engine Module provides physical connection to the CARRE repository. Transmission protocol module covers also specialized XML templates which enable further use of system data transmitted via Aggregator Module.

2.3. Communication between components

The different components making up the CARRE environment can be implemented using a variety of programming languages and tools. To allow flexible communication between these components, all interactions are dealt with by exchanging XML messages over a TCP-IP connection. In order to make this work and to allow for changes in the design and implementation later on in the project, the communication between components needs to be standardized on different levels:

- The (high-level) semantics of the message: what do the message contents mean? This is described using a formalized XML format
- The message XML format: CARRE XML schema. This XML message needs to be “encoded” into a series of bits according to an agreed-upon encoding:
- The (low-level) way in which text is encoded into bits, to allow for exchange between different platforms. For CARRE, the UTF-8 encoding shall be used as this is the dominant way of encoding text, so choosing this format should make interoperability between the different platforms that are used easier.

Illustrated in a schema:

Give me entities close to position (5,7)	The meaning of the message
<CARREmessage>\n<request><position><x>5…	XML using CARRE schema
1001001111010111000101001101110110100 0…	Text encoded using UTF-8

3. CARRE schema interface & semantic repository functionalities

The CARRE repository is at its heart a semantic repository: data stored in it must be available in a format which enables two key functions:

1. Machine inference: with machine-readable representation of the semantics of the data, it becomes possible to carry out sophisticated automated reasoning. For example, data from multiple physical sensors can be inferred as leading to a modification of a patient's health risks by combining them with machine-encoded knowledge from clinical research guidelines.
2. Data linking: there are existing sources of external data which can relate to data stored in the CARRE repository. For example, the daily physical activity levels of a patient may be affected by the weather (heavy rain making a long outdoor walk less likely). It must be convenient to link such external data to the CARRE repository.

By using the principles of Linked Data[1], these abilities are enabled. The CARRE repository will store data using the Resource Description Framework (RDF)[2], which is the global standard for representing semantically-enriched Linked Data[3].

Conceptually, RDF data consists of triples, each of which corresponds to a "<subject> <predicate> <object>" sentence representing the relationship between entities, or between an entity and a value. Complex data can be broken down into triples. The important task in developing RDF schemata is in the choice of vocabularies. In order to get the full benefit of Linked Data, it is important to use standard RDF vocabularies wherever possible, so that, for example, data relating to people is expressed using the same vocabulary regardless of where it comes from, and similarly data relating to weather, biomarkers, and so on. Deliverable.2.4 will present the particular choice of vocabularies required for CARRE in order to represent data relating to biomarkers, environmental, demographic, clinical and lifestyle factors.

The Aggregator design presented here is modular, allowing pluggable components to handle data of different types. It makes sense for the enrichment of raw data with RDF to be located in the modules pertaining to particular types of data. For example, the Data Translator Module (introduced earlier) corresponding to Blood Pressure ought to take care of enriching raw blood pressure measurements with the RDF terms from the D.2.4 schema relevant to blood pressure. By locating this translation process here, it becomes possible to extend the system in a convenient, modular way, without requiring repository changes simply to add a new kind of sensor.

The CARRE semantic repository will be a public RDF store accessible as a RESTful Web Service[4] supporting the standard CRUD (Create, Read, Update, Delete) model. That is to say, by means of the standard operations on the repository URL, it will be possible to create (HTTP POST), read (HTTP GET), update (HTTP POST) and delete (HTTP DELETE) data from the repository. In particular, the standard RDF query language SPARQL5 will be supported by the repository via a standard SPARQL endpoint.

RDF can be represented in a variety of formats: RDF/XML, Turtle/N3, N-Triples/N-Quads, and so on. The repository will support all of these formats for all operations; given that any underlying or existing tools for accessing individual data sources may already use one particular format, and given the very low cost of supporting different RDF formats, it is more convenient to be liberal in the range of accepted formats.

Of course, due to the sensitive nature of much of the data to be held in the CARRE repository, there will need to be tight controls over which agents (users or software) can perform which operations on different subsets of the data. Access to the repository will be solely over encrypted (HTTPS) channels, and solely through a secure authentication mechanism with a fine-grained permissions model. D.4.1 will detail the security and privacy model for access to all CARRE data.

---

1 http://www.w3.org/DesignIssues/LinkedData.html

2 http://www.w3.org/RDF/

3 One can often see the phrase "Linked *Open* Data" in the literature; however, as some of the data in the CARRE repository is clinical and pertains to individual patients, that data will not be open.

4 http://www.ibm.com/developerworks/webservices/library/ws-restful/

5 http://www.w3.org/TR/rdf-sparql-query/

4. Implementation Issues

A Web service[6] can be described as a web API (application programming interface), which enables two applications to communicate using XML over the web, or a network connection. A Web service allows communication in both contexts Client to Server and Server to Server. Thus, a web service can allow questioning a remote database or using a remote business application. A SOAP message (XML) is transmitted in HTTP protocol (see Figure 8).

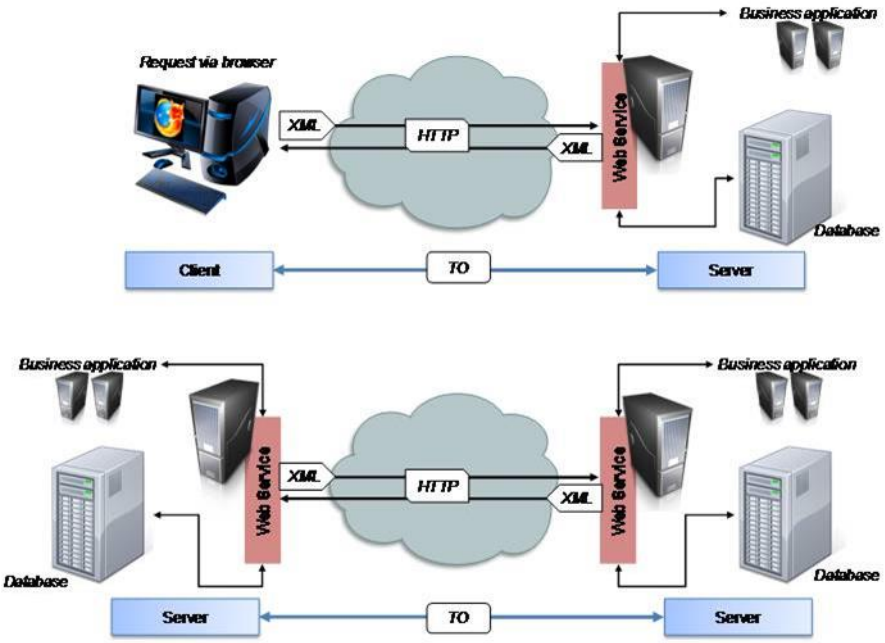


Figure 8. Web Services

Description of the properties and qualities of a SOAP web service:

1. It is based on existing standards:
    a. XML (data format exchange)
    b. HTTP (communication protocol)
    c. SOAP (exchange protocols)
2. As it is suitable for different programming languages and development environments (e.g. Java, Microsoft, Python), a web service allows a large interoperability between different types of business applications, especially between business applications deployed in heterogeneous Operating Systems and languages.
3. It requires the use of an application server (e.g. Apache Tomcat[7]) to be deployed.
4. Functions and data exposed by a web service are defined by a schema definition and translated in WSDL (Web Serviced Description Language). A WSDL file contains information about the different components involved in the web service and their respective message.

---

6 A Web service is defined by the W3C (World Wide Web Consortium) as: “a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically Web Services Description Language WSDL). Other systems interact with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialisation in conjunction with other Web-related standards”.

7 http://tomcat.apache.org/

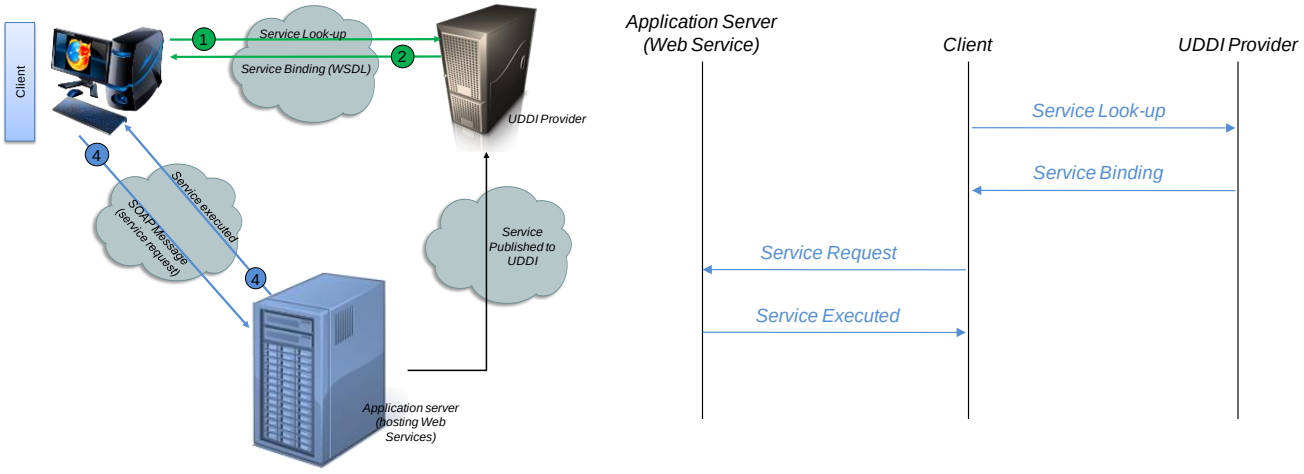
Web Service Description Language (WSDL) is sufficient for the technical details of how a Web service can be accessed and invoked remotely over the Web; Universal Service Description Language (USDL) describes the business and operational aspects of services as well. Hence by using USDL, it becomes possible to describe service properties beyond the technical level which is useful for achieving interoperability at the Operation Level where the business and operational properties of services will be used.

A simpler alternative to SOAP and Web Services Description Language (WSDL) that has gained widespread acceptance across the Web is REST[8], i.e. Representational State Transfer Web services (also called RESTful web services). Key evidence of this shift in interface design is the adoption of REST by mainstream Web 2.0 service providers—including Yahoo, Google, and Facebook—who have deprecated or passed on SOAP and WSDL-based interfaces in favour of an easier-to-use, resource-oriented model to expose their services. REST is an architectural style consisting of a coordinated set of architectural constraints applied to components, connectors, and data elements, within a distributed hypermedia system. REST ignores the details of component implementation and protocol syntax in order to focus on the roles of components, the constraints upon their interaction with other components, and their interpretation of significant data elements.

In CARRE project output form aggregators should be in RDF (any standard RDF format will be fine, whether RDF/XML, N3, etc.). That RDF will need to use the vocabularies from the (semantic) schema currently under development under Deliverable D.2.4.

Figure 9 describes the general overview of a web service in action. It includes:

1. The client which requests a service.
2. The application server which delivers the requested service
3. The UDDI (Universal Description Discovery and Integration) provider which contains services registry with the aim to associate the right service to the request. The UDDI is commonly hosted by the application server.



Illustration

Message Sequence Chart [9]

Figure 9. Web Service in action

The availability of the web service, which is also a security feature, does not depend only on the web service QoS (especially on the availability of application server) but also on the communication availability. The

[8] R. T. Fielding, Architectural Styles and the Design of Network-based Software Architectures, PhD Thesis, University of California, CA, USA, 2000.

[9] http://en.wikipedia.org/wiki/Message_sequence_chart

communication availability depends on the Telecommunication provider and on the type of communication (e.g. Multi Protocol Label Switching MPLS[10]).

The security of data or service in a web service application can be reached using the common solution of secure HTTP with TLS (Transport Layer Security) protocols called HTTPS (used for instance in e-banking and in e-commerce applications) and the authentication of the communication-points (client and server or server and server) thanks to certificates of authentication delivered by a trusted party via a PKI (Public Key Infrastructure)[11].

[10] MPLS Fundamentals, By Luc De Ghein Nov 21, 2006 (ISBN-10 1-58705-197-4)

[11] Adams, Carlisle & Lloyd, Steve (2003). Understanding PKI: concepts, standards, and deployment considerations. Addison-Wesley Professional. pp. 11–15. ISBN 978-0-672-32391-1