FP7-ICT-611140 CARRE

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

D.3.3. Aggregators for Personal Medical & Lifestyle Data

G. Drosatos, R. Kizlaitis, D. Stundys, E. Valys

January 2015

CARRE Contacts

Project Coordinator: Eleni Kaldoudi kaldoudi@med.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 Santariškių Klinikos	Domantas Stundys	Domantas.Stundys@santa.lt
KTU Kaunas University of Technology	Arūnas Lukoševičius	arunas.lukosevicius@ktu.lt
PIAP Industrial Research Institute for Automation & Measurements	Roman Szewczyk	rszewczyk@piap.pl

Disclaimer

This document contains description of the CARRE project findings, work and products. The authors of this document have taken any available measure in order for its content to be accurate, consistent and lawful. However, neither the project consortium as a whole nor the individual partners that implicitly or explicitly participated in the creation and publication of this document hold any sort of responsibility that might occur as a result of using its content.

In case you believe that this document harms in any way IPR held by you as a person or as a representative of an entity, please do notify us immediately.

The content of this publication is the sole responsibility of CARRE consortium and can in no way be taken to reflect the views of the European Union.

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.3.3
Deliverable Title:	Aggregators for personal medical and lifestyle data
Responsible Partner:	VULSK
Authors:	(in alphabetical order) G. Drosatos, R. Kizlaitis, D. Stundys, E. Valys
Input from:	All partners
Peer Reviewers:	S. Daukantas (KTU), S. Roumeliotis (DUTH), E. Liu (BED)
Task:	T.3.3. Aggregators for personal medical and lifestyle data
Task duration:	10 months: 1 April 2014 to 31 January 2015
Work Package:	WP3: Data and metadata harvesting
Work Package Leader:	KTU – Arūnas Lukoševičius
Due Date:	31 January 2015
Actual Delivery Date:	07 February 2015
Dissemination Level:	PU
Nature:	R & D
Files and format:	Deliverable report: 1 pdf file Software source code available from project web site, see Annexes 1-3: (1) CARRE_D.3.3_Aggregators_PersonalData_PHR.zip (2) CARRE_Personal_Medical_Data_Entry_v0.1.zip (3) CARRE_QueryDetector_Firefox_v0.2.xpi CARRE_QueryDetector_Firefox_v0.2.zip CARRE_QueryDetector_Chrome_v0.2.crx CARRE_QueryDetector_Chrome_v0.2.zip CARRE_UserIntentionExtractor_v1.0.zip CARRE_UserIntentionExtractor_src_v1.0.zip
Version:	02
Status:	☐ Draft ☒ Consortium reviewed ☒ WP leader accepted ☒ Coordinator accepted ☐ EC accepted

Document Revision History

Version	Date	Modifications	Contributors
v01.1	19 Jan 2015	Template and content in Section 3 "Web lifestyle data aggregator"	George Drosatos
v01.2	21 Jan 2015	New content in Section 2	Evaldas Valys
v01.3	22 Jan 2015	New content in Section 2, revision of outline	Romualdas Kizlaitis
v01.4	23 Jan 2015	Snapshots of web lifestyle data aggregator	George Drosatos
v01.5	23 Jan 2015	New content in Section 1, Section 2, revision of outline	Domantas Stundys
v01.6	25 Jan 2015	Description of implementations and code metrics of web lifestyle data aggregator	George Drosatos
v01.7	26 Jan 2015	New content in Section 2, revision of outline	Evaldas Valys
v01.8	26 Jan 2015	Specification and discussion of web lifestyle data aggregator	George Drosatos
v01.9	27 Jan 2015	Revision of outline	Domantas Stundys
v01.10	27 Jan 2015	New content in Sections 2.4, 2.5.2, 2.5.4, revision of outline	Romualdas Kizlaitis
v01.11	03 Feb 2015	New Section 2.3 "Code Metrics", revision of outline	Evaldas Valys, Domantas Stundys
v01.12	05 Feb 2015	New content in Section 1, reformatting sections, new Section 3 "Personal medical data entry system", new Annex 1, revision of outline	Evaldas Valys, Domantas Stundys
v01.13	06 Feb 2015	Snapshots of Personal medical data entry system, PHR aggregator Output module architecture specification, update of table of content, list of figures/tables	Evaldas Valys, Domantas Stundys
v01.14	06 Feb 2015	New content in Section 1, revision of outline	Domantas Stundys
v01.15	07 Feb 2015	New content in Section 1, 2.5, 3, 3.3, Annex 1. New Annex 2.	Evaldas Valys, Domantas Stundys
v01.16	07 Feb 2015	New content in Section 4.3, revision of outline	George Drosatos, Domantas Stundys
v02.0	07 Feb 2015	editing for QA conformance	E. Kaldoudi

Table of Contents

List of Figures

List of Tables

Executive Summary

This document is a report of project Task 3.3. “Aggregators for personal medical & lifestyle data deliverable”. It describes the architecture and purpose of aggregators used to extract personal health information and lifestyle related data as well as fetching data from patient’s online interaction.

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, 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 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 (API) is a set of functions and procedures that allow the creation of applications that access the features or data of an operating system, application, or other service
CPU	Central Processing Unit
C#.NET	C# is an elegant and type-safe object-oriented language that enables developers to build a variety of secure and robust applications that run on the .NET Framework. C# programs run on the .NET Framework, an integral component of Windows that includes a virtual execution system called the common language runtime (CLR) and a unified set of class libraries
C#.NET MVC	Model View Controller
DSS	Decision Support System
EC	European Commission
EHC	European Health Card
eHealth	Electronic Health
EHR	Electronic Health Record
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
ICT	Information and Communication Technologies
LM	Language Model
MedLinePlus	The National Institutes of Health's Web site for patients and their families and friends, http://medlineplus.gov.
OAuth	Open Standard to Authorization
Patient ID	Personal Identification Number
PHR	Personal Health Record
PubMed	Free search engine accessing primarily the MEDLINE database www.ncbi.nlm.nih.gov/pubmed/
RDF	Resource Description Framework - a standard model for data interchange on the Web.
SPARQL	RDF query language, that is, a query language for databases, able to retrieve and manipulate data stored in RDF
URL	Uniform Resource Locator
WWW	World Wide Web
XLST	Extensible Stylesheet Language Transformations - a language for transforming XML documents into other XML documents
XML	Extensible Markup Language - a markup language that defines a set of rules for encoding documents in a format that is both human-readable and machine-readable

1. Introduction

Personal patient data 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 themselves via the CARRE platform. Other personal information on patient lifestyle can be harvested from on-line social presence of the patient. Challenges in this source category include semantic integration of personal medical data; information extraction from online user interaction; and data security and privacy issues.

This deliverable presents specific aggregators that have been developed to harvest personal patient data from:

- personal health records;
- manual entry of personal medical data related to CARRE; and
- online user interaction on the web.

The term “personal health record” is not new. The earliest mention of the term was in an article indexed by PubMed dated June 1978[1], and even earlier in 1956 reference is made to a personal health log[2]. The term "PHR" has been applied to both paper-based and computerized systems; current usage usually implies an electronic application used to collect and store health data. In recent years, several formal definitions of the term have been proposed by various organizations[3,4,5]. In terms of content, PHRs may store various medical information in different formats. There are many 3rd party PHR systems in the market, which have their own APIs and data formats. As they are closely coupled to data source, PHR data aggregators are used in order to retrieve and extract the data. If patient’s medical data in electronic form is very scarce or missing, e.g. paper-based medical records, personal medical data may be entered manually either by the patient himself or physician by filling in dedicated web-based forms/fields or by uploading the data from other sources thus appending PHR with relevant medical information. There are many 3rd party PHR systems in the market, which have their own APIs and data formats. As they are close coupled to data source, PHR data aggregators are used in order to retrieve and extract the data.

It is important to note that information on patient’s psychological, behavioural and cognitive aspects is significant while creating personalized healthcare applications. Patient’s motivation and willingness to take active role in disease management (patient empowerment) is also reflected in seeking knowledge on his/hers medical condition, with internet being one of the information sources. As web search engines are one of the most popular uses of the web, among all age groups and have shown to be a good source of user’s psychological and cognitive issues, we decided to use patient's web searches from several most popular search engines to complement directly related medical data, e.g. medical history, current health status stored in PHR.

This report consists of several thematic sections. Section 2 of this deliverable presents aggregators for collecting personal medical data from existing PHRs. Section 3 describes web-based option to enter patient’s medical information manually from other sources or if there is no patient’s data stored in electronic health record, e.g. paper-based medical records. In Section 4 patient’s web involvement and lifestyle aggregators are described in a detailed manner.

Finally, Annexes 1-3 give links for downloading the actual D.3.3 deliverable, which is the software for PHR aggregation, the manual data entry system and the software for web lifestyle data extraction developed in T.3.3 (and described in detail in this report).

---

[1] Computerisation of personal health records, *Health Visitor* **51** (6): 227. Jun 1978

[2] Dragstedt, CA (1956-04-14). "Personal health log". *Journal of the American Medical Association* **160** (15): 1320.

[3] Connecting for Health. The Personal Health Working Group Final Report. July 1, 2003.

[4] AHIMA e-HIM Personal Health Record Work Group (July–August 2005), "The Role of the Personal Health Record in the EHR", *Journal of AHIMA* **76** (7): 64A–D

[5] America's Health Insurance Plans (13 Dec 2006). "What are Personal Health Records (PHRs)?". Archived from the original on 5 Mar 2009. http://www.ahip.org/content/default.aspx?docid=18330

2. Aggregators for personal medical data

PHR aggregator collects personal medical data from different 3rd party PHR systems that provide an API for accessing their records. This data is then consolidated, serialized into a XML document and stored into the repository. The XML file is then parsed, converted into RDF triples and stored into CARRE private semantic repository.

For this task, we chose two sources for medical data aggregation, the PHR systems VivaPort[6] (developed by the EU BSR 2007-2013 Programme funded project "ICT for Health") and HealthVault[7] (a free, open source PHR system provided by a 3rd party vendor).

2.1. Architecture

The main parts of PHR aggregator architecture are presented in Figure 1. As shown in the figure, there are two PHR systems, the aggregator itself, which consists of three parts: **PHR Input modules**, R**esource unification module** and an **Output module**, and one repository used to store aggregated data.

This architecture is extensible because it lets us easily add another PHR system, if necessary by developing a new PHR Input module or another repository by developing a new Output module.

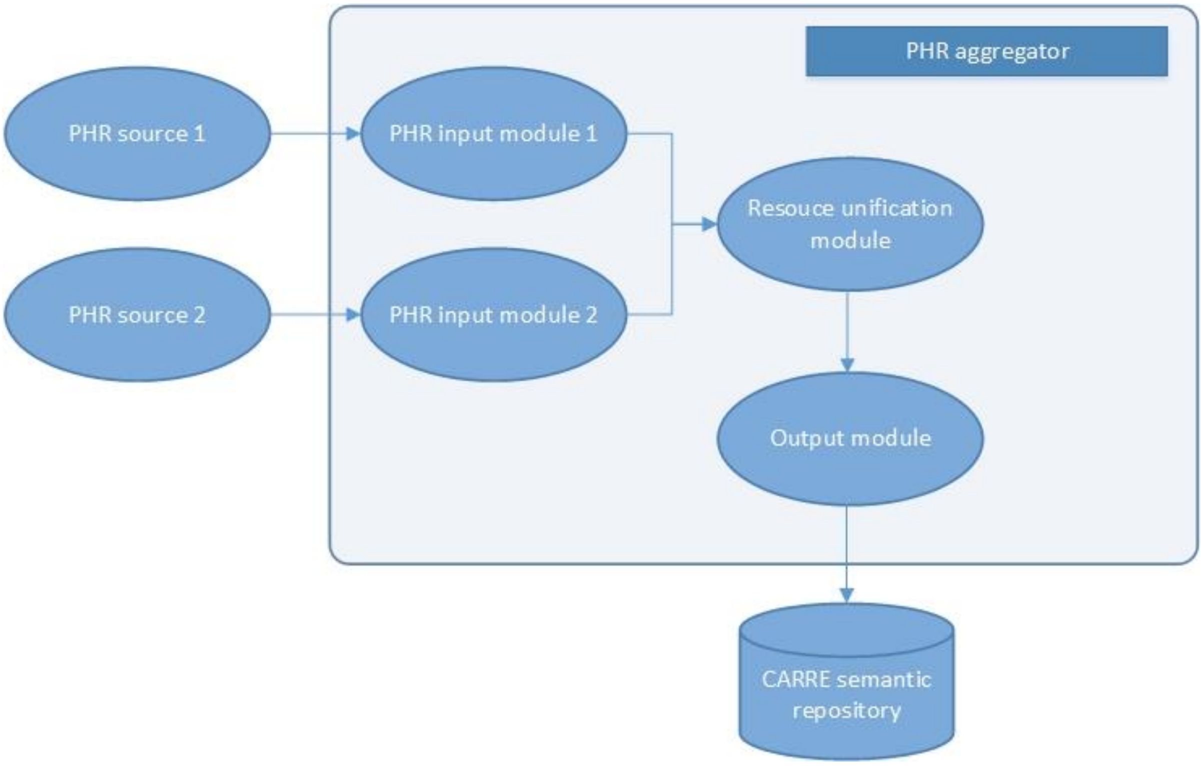


Figure 1. The architecture of a PHR aggregator.

The following subsections provide a detailed description for each module.

6 https://vivaport.eu/

7 https://www.healthvault.com/

2.1.1. Input module

The purpose of the Input module is to interface with and access data in a PHR system, fetch required personal medical data and map[8] it to the internal data structure. Standardized data format is required because there is no standard data format across all existing PHR systems. This and the fact that PHR systems have different APIs has led to current Input module architecture as shown in Figure 2. Each different PHR would in principle require a different, dedicated Input module. The number of deployed Input modules in an aggregator is not limited.

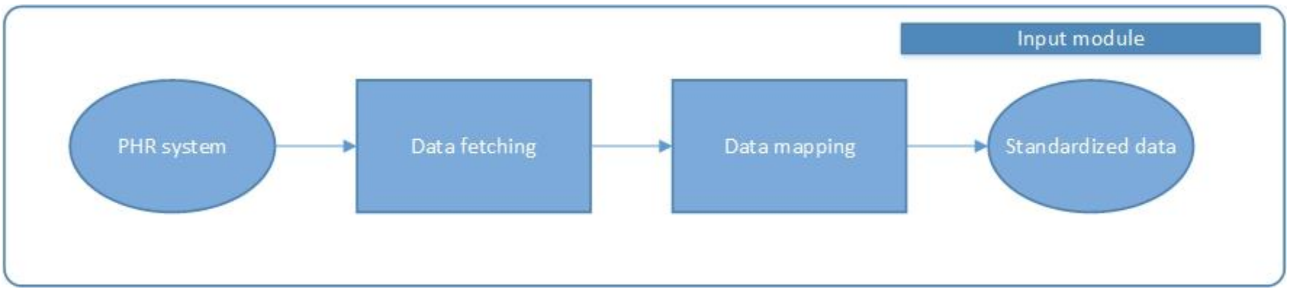


Figure 2. The architecture of Input Module.

2.1.2. Resource unification module

Resource Unification module is responsible for consolidating multiple standardized datasets given by PHR Input modules into one standardized dataset. For example, a patient stores his blood pressure data in VivaPort and the information about his allergies could be found in HealthVault. Input modules fetch this data, standardize it, and Resource unification module consolidates the data to one dataset, which consists of both patient's blood pressure data and information about his allergies. The architecture of Resource Unification module is shown in Figure 3.

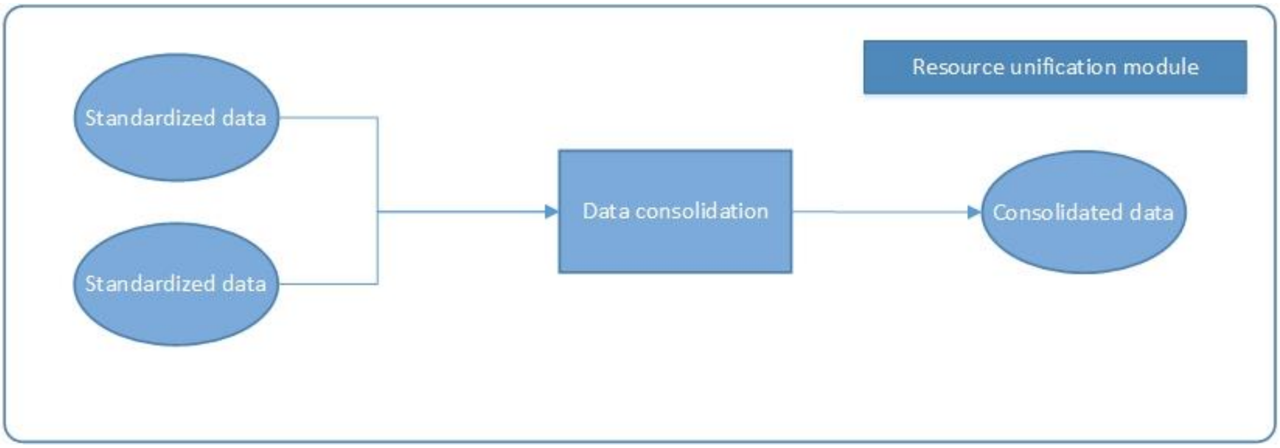


Figure 3. The architecture of resource unification module.

2.1.3. Output module

The Output module takes consolidated personal medical, converts it to RDF triples and stores it to CARRE semantic repository. Currently the Output module turns consolidated data into a XML file according to external rules and then deposits generated XML into the local repository. The local XML is then transformed into RDF triples and inserted into the private CARRE semantic repository. The architecture of the Output module is shown in Figure 4.

---

[8] Data mapping is the process of creating data element mappings between two distinct data models.

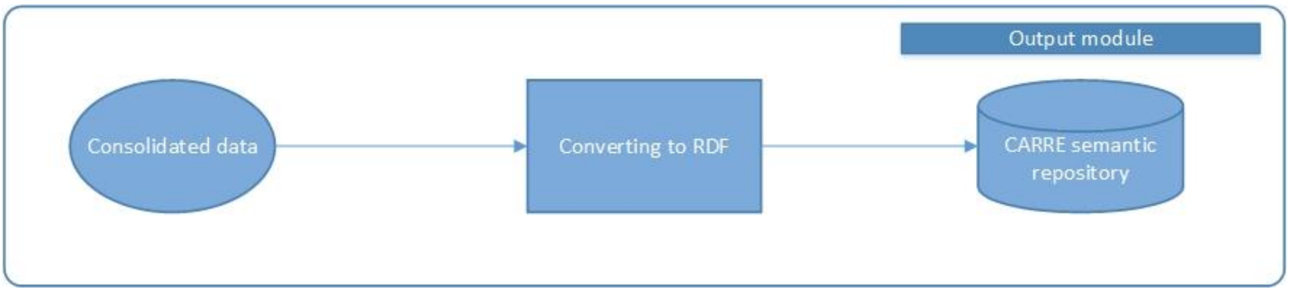


Figure 4. The architecture of Output module.

2.2. Aggregating data from VivaPort

A VivaPort PHR Input module fetches personal medical data in XML format by using VivaPort API and uses an imported VivaPort API library to deserialize it into a C# class. This API exposes patient's personal information as well as personal medical data such as allergies, medications, surgical procedures, blood pressure and weight in XML format. The aggregated data is then standardized and passed to Resource unification module, which then consolidates it to a single data set. This data set is then passed to an Output module that uses XSLT transformations to generate a XML document and store it to the repository.

During the development of the PHR aggregators, we encountered an issue, which is called Master data management[9]. It occurs while working with multiple data sources that contain specific data but recorded values are different. It becomes especially obvious when we do not know which data should be considered as the correct one. For example, a patient is using VivaPort for his blood pressure monitoring while his doctor is using HealthVault. In such case it would remain unclear which data source should be used, e.g. which of aggregated blood pressure values are more reliable. It is also important to consider if the aggregator should merge both data sets when consolidating data from PHR systems, or should master data source be allocated and the rest data disregarded.

2.3. Aggregating data from HealthVault

HealthVault is a cloud-based personal health platform that exposes HTTP/XML based interface. It provides the infrastructure for long-term storage and access to user's health information. For a moment, HealthVault supports 80 data types to enable a range of medical and fitness scenarios. HealthVault API provides more than 40 methods enabling third-part applications to send and retrieve persons clinical and fitness data.

The aggregation of personal medical data from HealthVault is implemented by using Linking Applications model. Patients enable the "link" between the aggregator and their record by authorizing access. From then on, the aggregator has access to patient's HealthVault records. The required medical records are fetched by using HealthVault XML API[10]. Since HealthVault's API returns data in XML format, the whole implementation is similar to VivaPort's – fetched data is standardized, passed to Resource unification module, which consolidates it. Eventually, the data are saved in the Repository as a XML document.

2.4. Implementation

The PHR aggregator for VivaPort and HealthVault is being developed using C# .NET framework. Key parts for a working aggregator are:

- A standardized data format;

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

[10] https://msdn.microsoft.com/en-us/library/jj573971.aspx

- An interface for PHR Input module (with a method that fetches medical data from PHR system and maps it into a standardized format);
- A method that consolidates standardized data into a single data set;
- An interface for Output module (with a method that takes consolidated data, turns it into RDF triples and stores it into the CARRE semantic repository).

Currently there are two PHR Input modules (one for VivaPort and one for HealthVault) and one Output module implemented.

2.5. Code metrics

Table 1 presents the code metrics of HealthVault Input module; **Table 2** describes the Input model of VivaPort. **Table 3** and **Table 4** present code metrics of Resource unification module and Output module respectively.

Table 1. Code metrics of HealthVault Input module.

HealthVault Input module C# code metrics	
Number of C# Files	2
Number of Classes	5
Number of Methods	4
Lines of Code	206
C# Libraries	1

Table 2. Code metrics of VivaPort Input Module.

VivaPort Input module C# code metrics	
Number of C# Files	2
Number of Classes	5
Number of Methods	4
Lines of Code	221
C# Libraries	4

Table 3. Code metrics of Resource unification module.

Resource unification module C# code metrics	
Number of C# Files	1
Number of Classes	1
Number of Methods	4
Lines of Code	63
C# Libraries	6

Table 4. Code metrics of Output module.

Output module C# code metrics	
Number of C# Files	1
Number of Classes	1
Number of Methods	4
Lines of Code	62
C# Libraries	2

3. Personal medical data entry system

It is expected that not all patients will have rich health records in their PHR in an accessible form. Therefore, a browser-based application is being developed to allow patients to record their medical data by entering it manually through the web forms.

3.1. Architecture

The application consists of a set of web forms for different medical data entry. We chose them according to observables[11] that are identified in T.2.2.1. This application is designed so that new web forms can easily be added on demand. The architecture of Personal medical data entry system is shown in Figure 5.

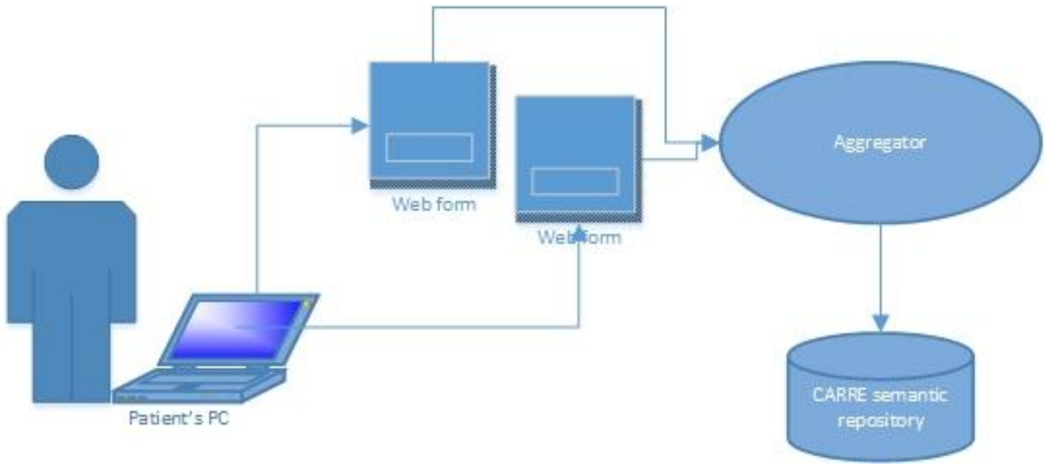


Figure 5. Personal medical data entry system architecture.

3.2. Implementation

The system was developed using C# .NET MVC framework and is similar to VivaPort. The data entered is stored as XML files in the repository and passed to the aggregator. Three web forms are developed:

- Patient personal information form (as shown in Figure 6):
  - Given name (First name)
  - Surname (Last name)
  - Date of birth (Birth date)
  - European Health Card number (EHC)
  - Personal identification number (Patient ID)
  - Home address (Street address, City, Post code, Country)
  - Phone number
  - Email address
  - Insurance (Medical insurance company, Insurance number, Phone number)
  - Photo

[11] http://carre.kmi.open.ac.uk/observables

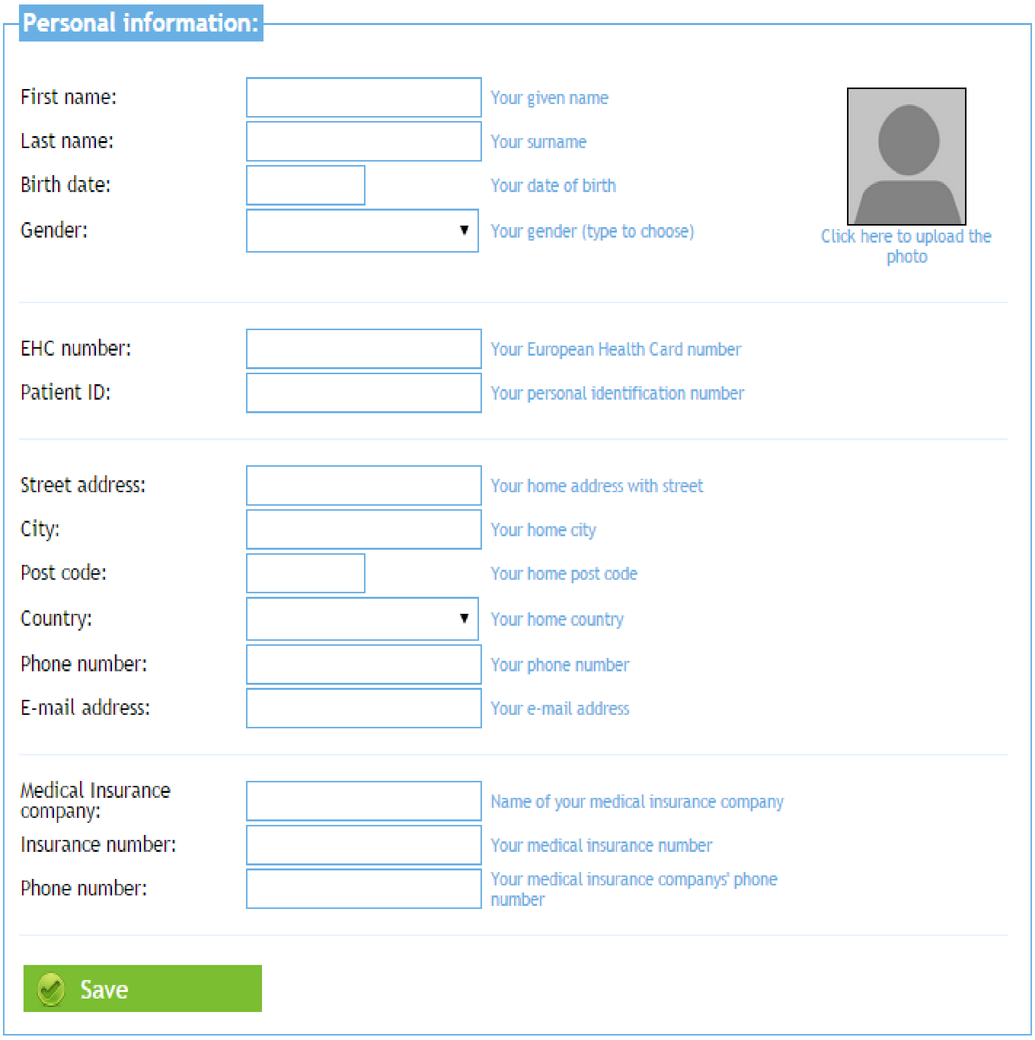


Figure 6. Personal information data entry form.

- Diagnosis form (as shown in Figure 7):
  - ICD code for the diagnosis
  - Diagnosis description
  - Date of diagnosis and recovery
  - Additional information concerning recovery
- Medication form (as shown in Figure 8):
  - Active ingredient of the prescribed medication
  - Strength of the medication
  - Prescribed units per day
  - Prescription start and end dates

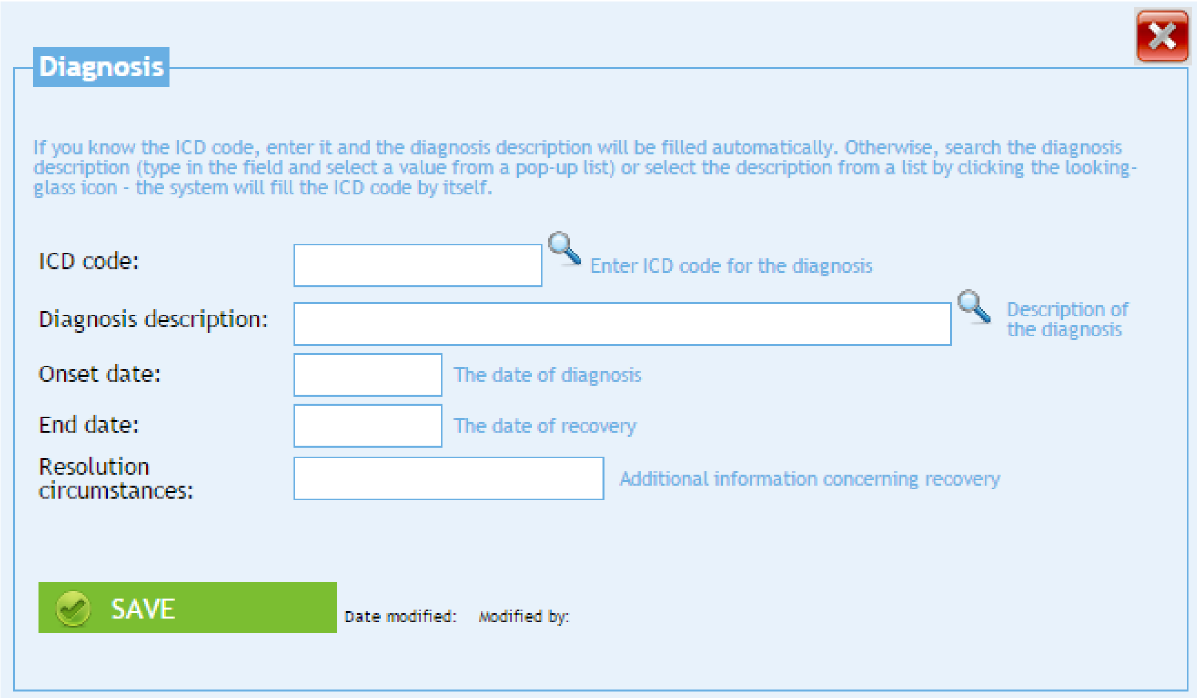


Figure 7. Diagnosis data entry form.

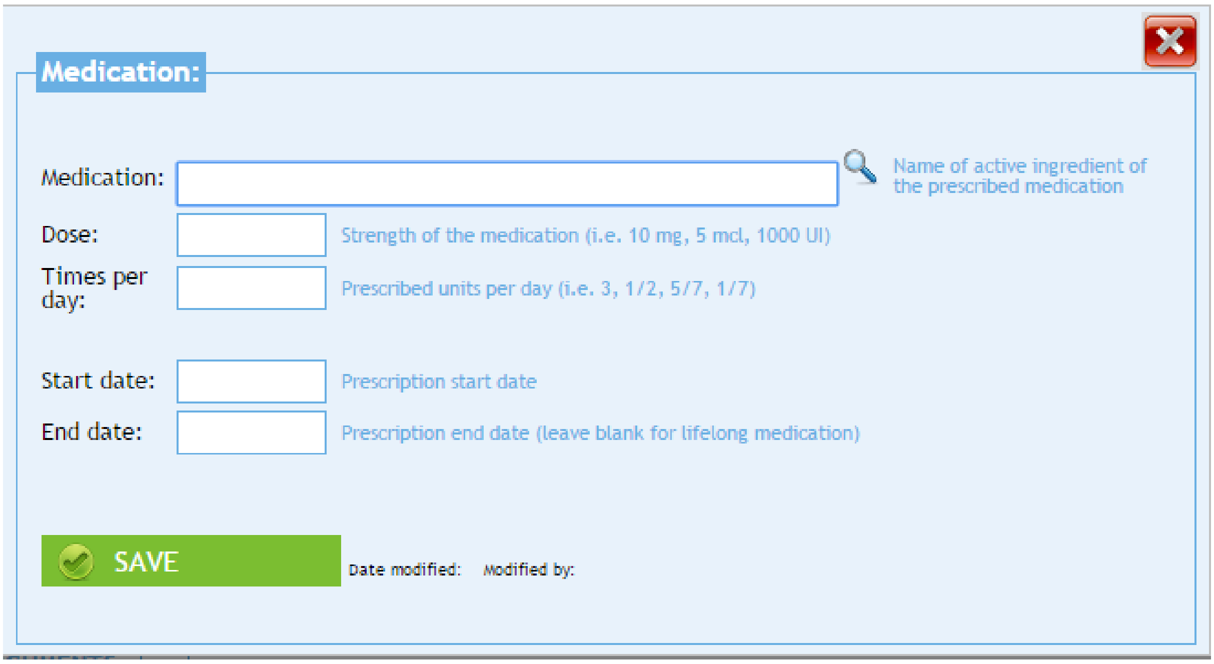


Figure 8. Medication data entry form.

3.3. Code metrics

The code metrics of Personal medical data entry system is presented in **Table 5**.

Table 5. Personal medical data entry system code metrics.

Personal medical data entry system code metrics	
Number of Web Forms	3
Number of Classes	3
Number of Methods	9
Lines of Code	632
C# Libraries	4

3.4. Deployment specifications

The application is deployed on a local server running Microsoft Windows Server 2008 using Internet Information Services.

3.5. Discussion

Diagnosis and Medication data sets are rather standard and there is no discussion about data entry forms of those data types. Regarding personal information form, there are several opinions if we should request more or less personal demographic and contact data entered. We decided to use more complex data form with non-obligatory fields with possibility to decrease number of those fields depending of the future experience of the use of the form and data.

4. Web lifestyle data aggregator

The main challenge in personalized ehealth applications is how to capture personal information in order to customize the generic ehealth application. Such information most often includes in medical history and current biometric readings to deduce current health status. However, of equal importance are psychological, behavioural and cognitive aspects, for example the patients' plans, intentions, attitudes, and behaviour [12]. Personalizing an ehealth application in general refers to customizing the delivery of care for the needs of a particular patient. This is rather broad and can vary considerably with the specific health problem in question.

In the field of patient empowerment, patient education and engagement is the cornerstone. Therefore, personalization here is strongly related to the particular risk or existing pathological condition, but also to the actual interest of the patient in searching for new information. Additionally, in the case of the cardiorenal comorbid patient, an important part of personalization involves two different aspects, namely diet and physical activity, both as embedded in the physical environment. In this context, knowledge on patient's plans to travel is important for the personalization of diet and physical exercise – both need to be customized to weather conditions and physical activity requirements.

The first two types of personal information, medical history and current medical status, although technologically demanding, are more straightforward to capture. In CARRE, such information is harvested from personal health records (previous sections in this document) and from wearable sensors (D.3.2). Capturing information on personal plans, intentions, and other psychological and cognitive issues is rather complicated. So far, such information is generally derived via interviews and questionnaires, which is a rather cumbersome process requiring human intervention. Current research however is increasingly using the person's web involvement as a source for information on psychological and cognitive issues[13].

In particular, web searches have shown to be a good source to reveal user's interests and intentions[14]. In addition, web search engines are one of the most popular uses of the web, among all age groups. For example, a recent US survey[15] shows that 72% of internet users say they looked online for health information of one kind or another within the past year while eight in ten online health inquiries start at a search engine. Therefore, we decided to use the patient's web searches from the most popular search engines that is Google, Bing and Yahoo (based on various 2014 market analyses[16,17,18]).

User web searches can be considered as private information[19], since they can reveal interests, political opinions, religious beliefs, sexual life, and other things a user may like to keep private. All this information according the European Data Protection Directive[20] constitutes sensitive personal data and requires special handling. For this reason, in our proposed approach, user queries are kept and processed at the user-side of the system, e.g. in their personal computing device. Only the relevant user's intentions with the decision support system are then transmitted to the CARRE server.

4.1. Architecture

In this section, we describe the proposed privacy-friendly architecture for extracting user's intentions from her web searches. An overview of this architecture is presented in Figure 9.Figure 9. The architecture of our web lifestyle data aggregator. The main goal of the proposed architecture is to use patient's web searches in order

---

[12] Lapsley P.. Involving patients in their healthcare BMJ. 2012;345.

[13] L. Fernandez-Luque, R. Karlsen, J. Bonander, Review of Extracting Information form the social web for health personalization, J. Med. Internet Res. vol. 13(1), e15, 2011

[14] Ginsberg J., Mohebbi M.H., Patel R.S., Brammer L., Smolinski M.S., Brilliant L. Detecting influenza epidemics using search engine query data, Nature. 2008;457:1012–1014.

[15] Health Online 2013, The Pew Research Center, 2013, http://www.pewinternet.org/2013/01/15/health-online-2013/

[16] http://gs.statcounter.com/press

[17] http://www.torsionmobile.com/2014/04/03/which-search-engine-dominates-mobile/

[18] http://theeword.co.uk/info/search_engine_market/

[19] Arampatzis A., Efraimidis Pavlos S., Drosatos G.. A query scrambler for search privacy on the internet, Information Retrieval. 2013;16:657-679.

[20] European Parliament. Directive 95/46/EC Official Journal L 281. 1995:0031-0050.

to extract her intentions. The patient's intentions are extremely useful information because the can help the doctor to adjust the pharmaceutical and nutritional treatment of patient (and not only). Overall the presented architecture was built by taking into consideration (as much as is possible) the protection of the patient's privacy.

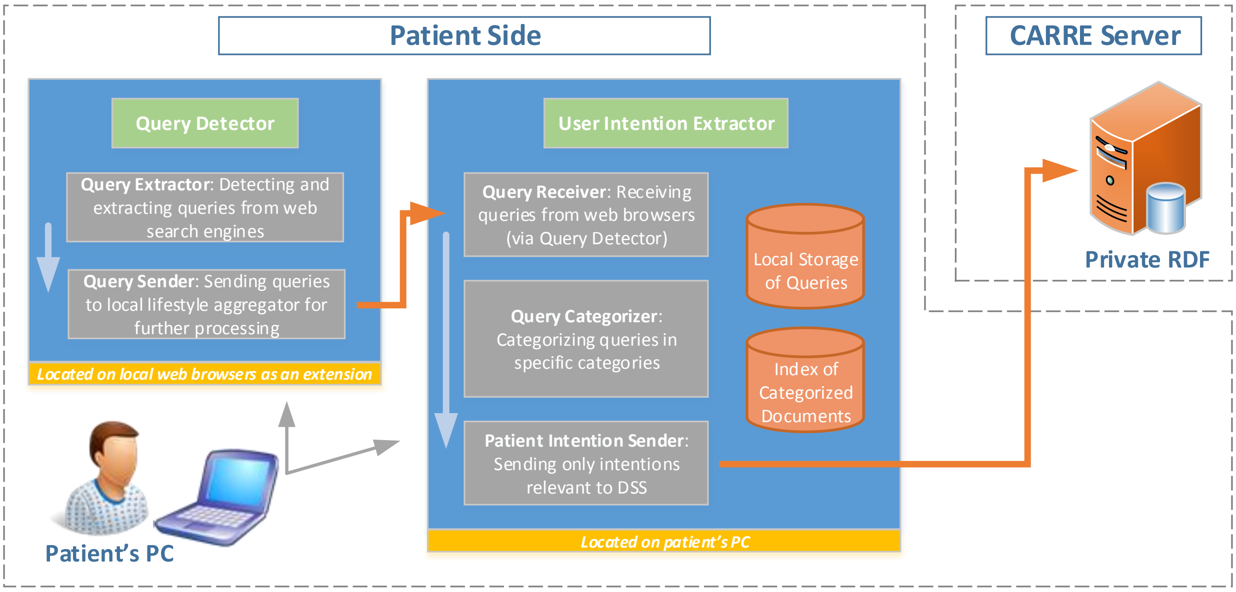


Figure 9. The architecture of our web lifestyle data aggregator.

The main parts of this architecture, as they are shown in Figure 9, are the ***Query Detector*** and the ***User Intention Extractor***. Both of them are located in patient-side and particularly in the personal computer of the patient. In more details, the Query Detector constitutes a browser extension (Firefox add-on and Chrome extension) that is responsible to collect the user's queries from the web browser and send them for further processing to the User Intention Extractor. The components of this part are the following:

a) ***Query Extractor:*** It is responsible for detecting the user's queries in the web search engines (Google, Bing and Yahoo). This component is triggered when the user tries to search something on the Internet. The detection of queries is achieved by parsing the URLs of the opening pages and the steps that are followed are:

   – Detect if a URL is from a search engine.

   – Parse the parameters of URL.

   – Export the query of the parameters.

b) ***Query Sender:*** It is responsible to forward the detected queries over HTTP locally (in localhost) to the Query Receiver of User Intention Extractor for further processing.

Additionally, the User Intention Extractor constitutes a standalone application that runs automatically when the operating system starts and is responsible to store (only) locally the incoming queries and categorize them in specific categories (e.g. travelling, health diseases, etc.) in order to extract the patient's intentions. The components of this part are the following:

c) ***Query Receiver:*** It is responsible for receiving the user's queries from the local browsers' extensions and storing them to its ***Local Storage***. In order to receiving data from a browser extension, it is required to handle HTTP requests.

d) ***Query Categorizer:*** It uses as input the queries from the Local Storage and performs the categorization of them in a specific set of categories. In the categorization process, we apply a web query classification technique[21] that uses documents from the World Wide Web (WWW) to enrich the

---

[21] Agrawal R., Yu X., King I., Zajac R.. Enrichment and Reductionism: Two Approaches for Web Query Classification, in Neural Information Processing;7064 of LNCS:148-157 Springer Berlin Heidelberg 2011.

target categories and further models the web query classification as a search problem. The Query Categorizer consists the backbone of our approach and utilizes the ***Index of Categorized Documents*** to categorize the queries. More details about why we select the particular technique, how we generate the Index of Categorized Documents and how this technique works are presented in the next section.

e) ***Patient Intention Sender:*** It is responsible to send only the relevant intentions (health and travel) to the private RDF repository. The private area of patient apart from her intentions contains data such as sensors' data, demographic elements and personal health records (PHR) that are required from the decision support system (DSS) to be developed in WP6. The communication between the User Intention Extractor and the RDF repository is encrypted (HTTPS) and an appropriate authentication mechanism (OAuth) it is used to identify the patients.

4.2. Query classification process

The extraction of user intention from web queries is usually seen as a categorization problem, and particularly, a classification problem, where queries are classified to predefined categories. These categories represent the user intentions. The query classification is a well-known problem as it is used to identify user search goal in order to improve search engine retrieval[22]. What makes web queries challenging is that they consist only of few words (2-3 words on average[23]) from which user interest must be extracted.

While there is a rich body of literature on query classification[24,25,26,27,28], with comparable results. For this reason, we decided to select an existing method, and adapt it for our approach. The final method should satisfy the following requirements:

– For privacy reasons, the classification process should take place at the user-side—not in a central server. Thus, it should be 'light' on the user's system resources, such as CPU, main memory, disk space, and should run fast.

– The selected method should not require dedicated training/data set because we would like to define specific categories that are relevant to the particular CARRE domain.

Based on the previous two requirements, we adopt the enrichment approach for query classification proposed by Agrawal et al[9]. The basic idea of this approach is to enrich the categories with documents from the web and classify the web queries in these documents. In this way, the query classification task is modelled as a search problem. As shown in the Figure 10, the query classification is a two-step process.

Create Index for Classification. In this process, the category enrichment is achieved by sending the category name as the search query to the index of ClueWeb09_B dataset[29] and using the Indri v5.7 search engine[30]. For each category, we collect the top-m results of documents and associating them with the category label. At the end, all the collected documents are indexed with the Indri search engine and in this way we create the index of categorized documents. The number of top-m documents used per category is set to be m=300 based

---

22 Rose D.E., Levinson D. Understanding User Goals in Web Search, in Proceedings of the 13th International Conference on World Wide Web (WWW '04):13–19 ACM New York 2004.

23 Arampatzis A., Kamps J. A Study of Query Length, in Proceedings of the 31st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '08):811–812 ACM New York 2008.

24 Beitzel S.M., Jensen E.C., Lewis D.D., Chowdhury A., Frieder O. Automatic Classification of Web Queries Using Very Large Unlabeled Query Logs *ACM Trans. Inf. Syst..* 2007;25.

25 Gabrilovich E., Broder A., Fontoura M., et al. Classifying Search Queries Using the Web As a Source of Knowledge *ACM Trans. Web.* 2009;3:5:1–5:28.

26 Lovelyn Rose S., Chandran K. R.. Web knowledge and Wordnet based Automatic Web Query Classification *International Journal of Computer Applications.* 2011;17:23-28.

27 Agrawal R., Yu X., King I., Zajac R.. Enrichment and Reductionism: Two Approaches for Web Query Classification in *Neural Information Processing*;7064 of *LNCS*:148-157Springer Berlin Heidelberg 2011.

28 AlemZadeh M., Khoury R., Karray F.. Query Classification using Wikipedias Category Graph *Journal of Emerging Technologies in Web Intelligence.* 2012;4

29 http://lemurproject.org/clueweb09/

30 http://lemurproject.org/

on the work of Agrawal. The categories that we use are from the SimilarWeb[31] and covers the majority of concepts on the Web. This happened because we want our index to be uniform. Additionally, we added in the category 'health' the following subcategories: 'drugs and therapy', 'diagnostic tests', 'medical equipment and sensors', 'exercise and physical activity' and 'dietary supplements'.

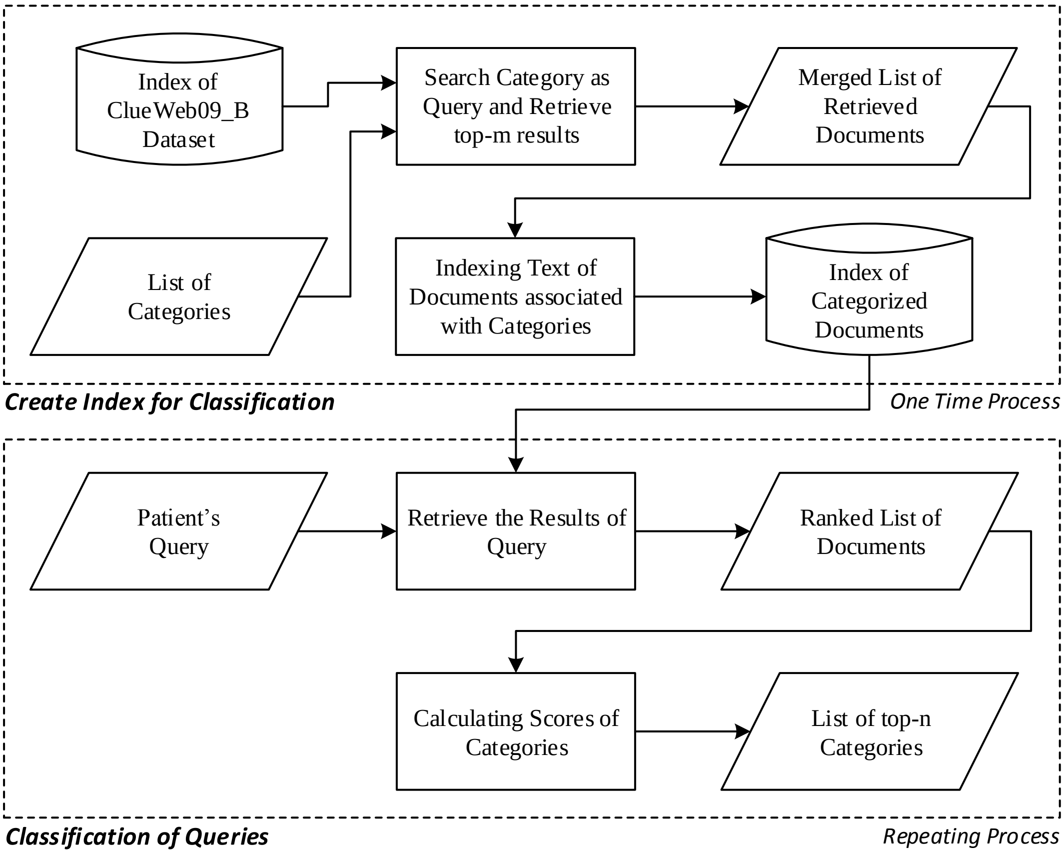


Figure 10. The flowchart of the query classification process.

Classification of Queries. In this process, we perform the actual web query classification task. The web query that needs to be classified is then searched (using Indri search engine) in the index from the previous process and the results of this search includes a ranked list of documents and associated LM (Language Model) relevant scores. Since, each document $\mathrm{d_j}$ is assigned to one to more category from the previous process, we can write the score $\mathrm{Sc(c_i)}$ of a category $\mathrm{c_i}$ given a query as:

$$\mathrm{Sc(c_i)} = \sum_{\mathrm{j:d_j \in c_i}} 2^{s_j}$$

where $d_j \in (-\infty, 0)$ is the Indri LM score of j-th ranked document $\mathrm{d_j}$ when the document belongs on the category $\mathrm{c_i}$. Using the previous equation, the scores for all target categories for a given web query are calculated. From all these categories, we only keep the top-n categories per query when ordered by decreasing scores. The number of top-n categories is set to be n=3 based on the work of Agrawal.

[31] http://www.similarweb.com/category

4.3. Implementation and code metrics

To confirm the feasibility of our approach, we developed prototype implementations of the Query Detector and User Intention Extractor. The Query Detector is implemented in JavaScript as a browser extension for the Firefox v34 (Figure 11) and the Chrome v39 (Figure 12). Accordingly, the User Intention Extractor is implemented in Java as a standalone application and runs in system tray when the operating system starts (Figure 13). Both implementations are platform independent and can work on Windows, Linux and Mac OS X. The development of Query Detector is performed by using a simple text editor (e.g. Notepad++[32]) and the development tools that are provided by the browsers. Finally, the User Intention Extractor is developed by using the NetBeans IDE v8.0.2[33] and some indicated technical details about its components are:

- **Query Receiver**: It is implemented as a tiny HTTP server and it is responsible to store the incoming queries in the local database.
- **Local Storage of Queries**: It is implemented and designed with SQLite[34] v3.8.7 database and is responsible to keep all the required data of categorization process.
- **Index of Categorized Documents**: The collection of documents is achieved with Java by using the ClueWeb09_B[35] dataset (50 Million English documents) and the creation of index is accomplished with the Lemur[36] Toolkit (Indri v5.7). This index is built with 25 root categories that consist from 227 subcategories. Some statistical information about the index is shown in Table 6.

Table 6. Statistical information of index.

Statistical Information of Index	
Number of Documents	65,764
Number of Unique Terms	1,490,312
Number of Total Terms	137,737,203
Size (in MB)	659

- **Query Categorizer**: The development of this component is achieved by using the Java JNI interface of the Lemur Toolkit (Indri 5.7) that provides a powerful search engine.
- **Patient Intention Sender**: It is developed with Java and authenticates the patient to her Private RDF. The patient's intentions are sent to the Private RDF with SPARQL[37] language over the HTTPS protocol.

Figures Figure 14, Figure 15, Figure 16 and Figure 17 show the database structure of the Local Storage and some indicated searches, categories and categorized searches. Also, Figure 18 shows the visualization of the categorized searches in a particular time period, where the size of leafs in the tree indicates the frequency of queries in a particular root category or subcategory. The snapshot in Figure 19 shows the authentication of patient to access the private RDF repository. Finally, Table 7 shows an example of RDF triples that are inserted to Private RDF repository per user's intention. In this example, the user's intention is 'Conditions and Diseases', it is taken at '2015-01-21 19:12:25' and it has weight '3' (this mean that was the first of the top-3 categories).

[32] http://notepad-plus-plus.org/
[33] https://netbeans.org/
[34] http://www.sqlite.org/
[35] http://www.lemurproject.org/clueweb09.php/
[36] http://www.lemurproject.org/
[37] http://en.wikipedia.org/wiki/SPARQL

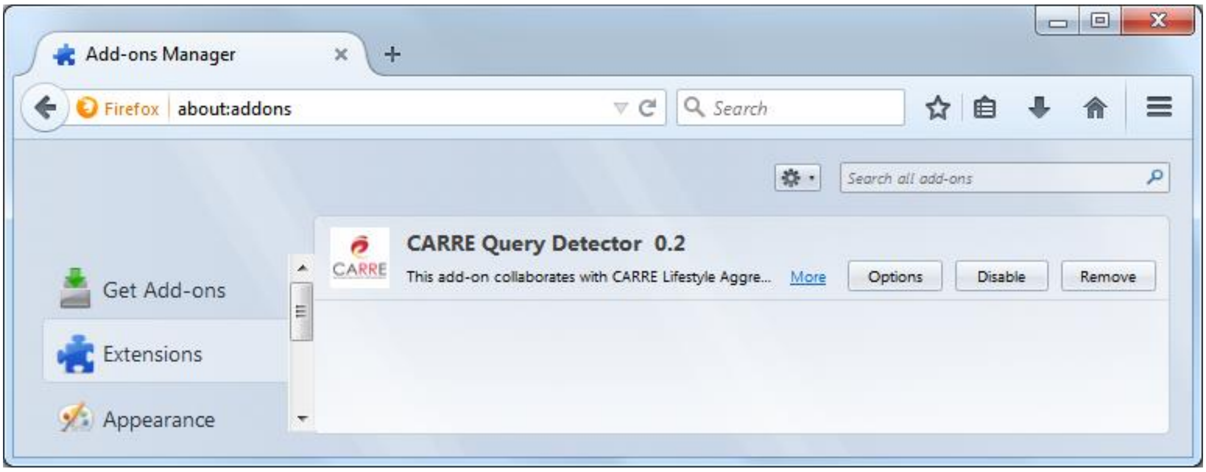


Figure 11. A snapshot of the Query Detector for the Firefox.

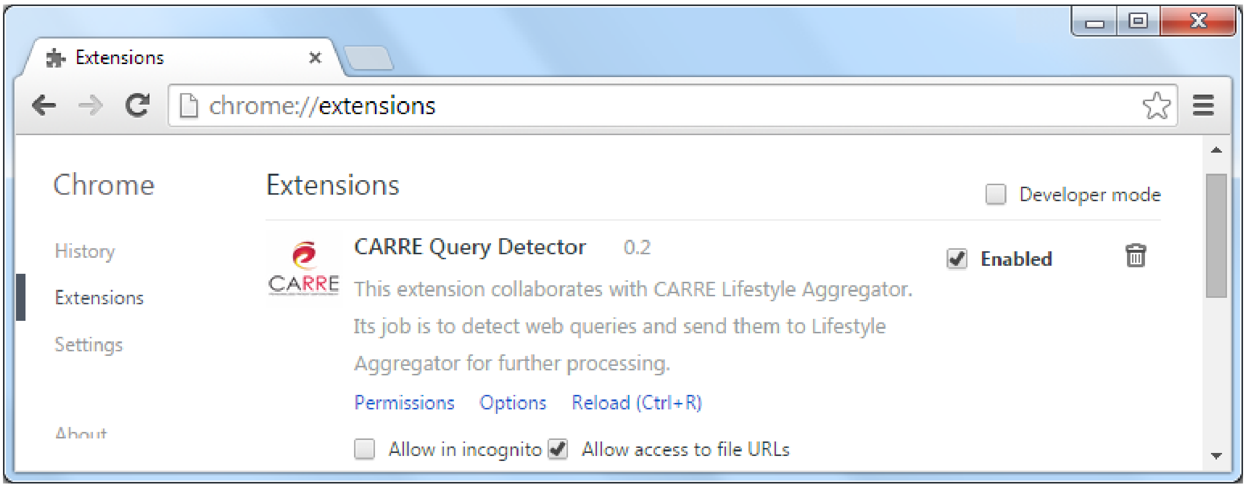


Figure 12. A snapshot of the Query Detector for the Chrome.

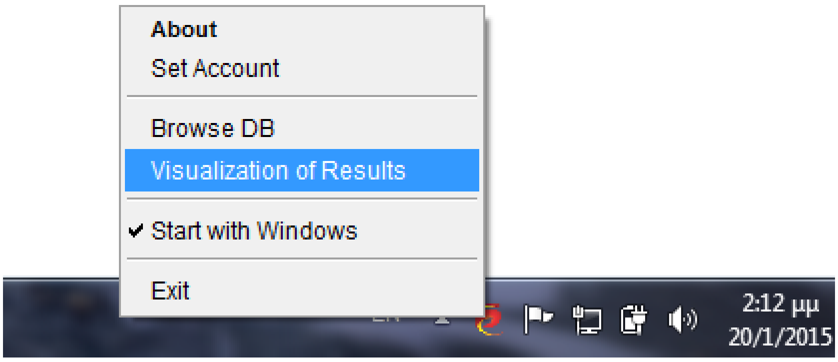
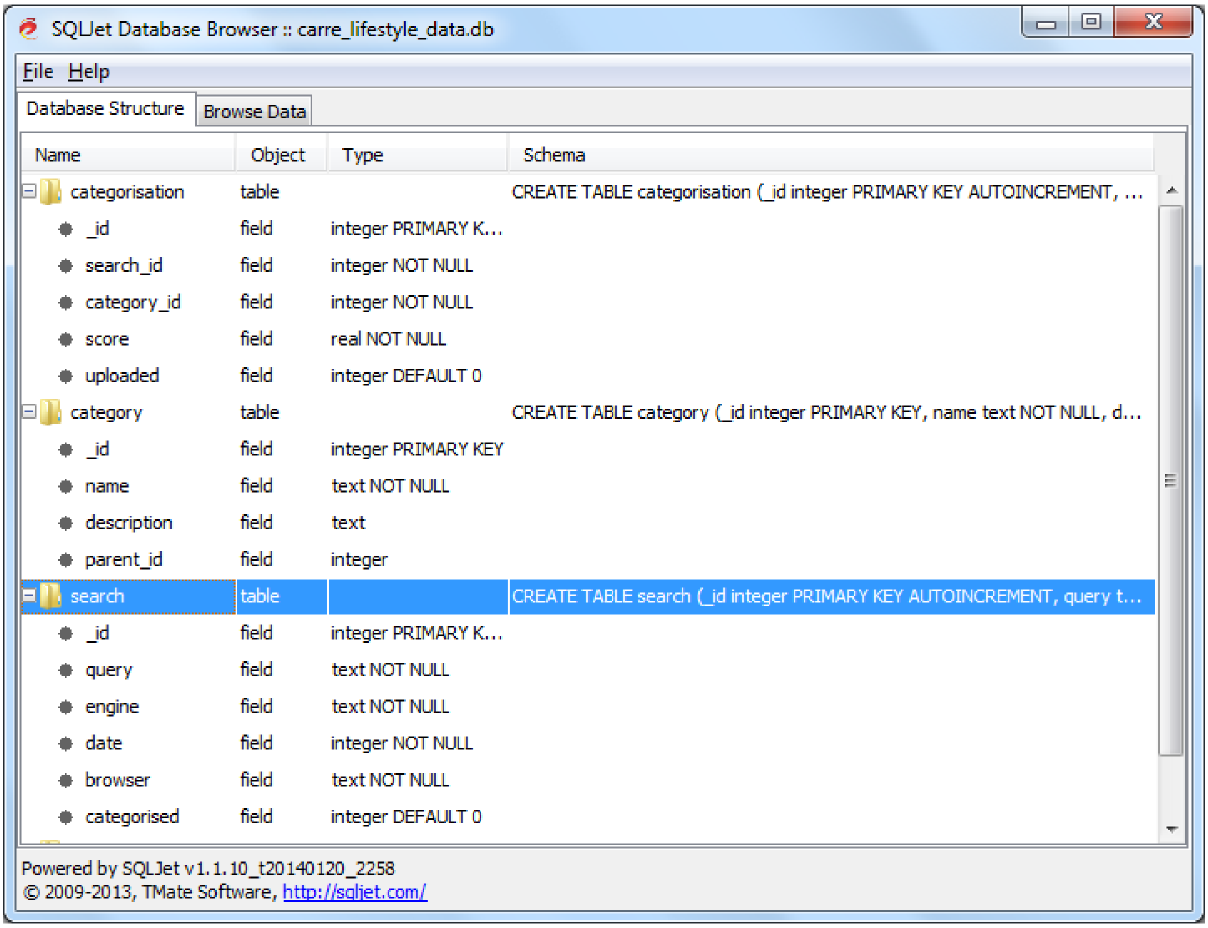
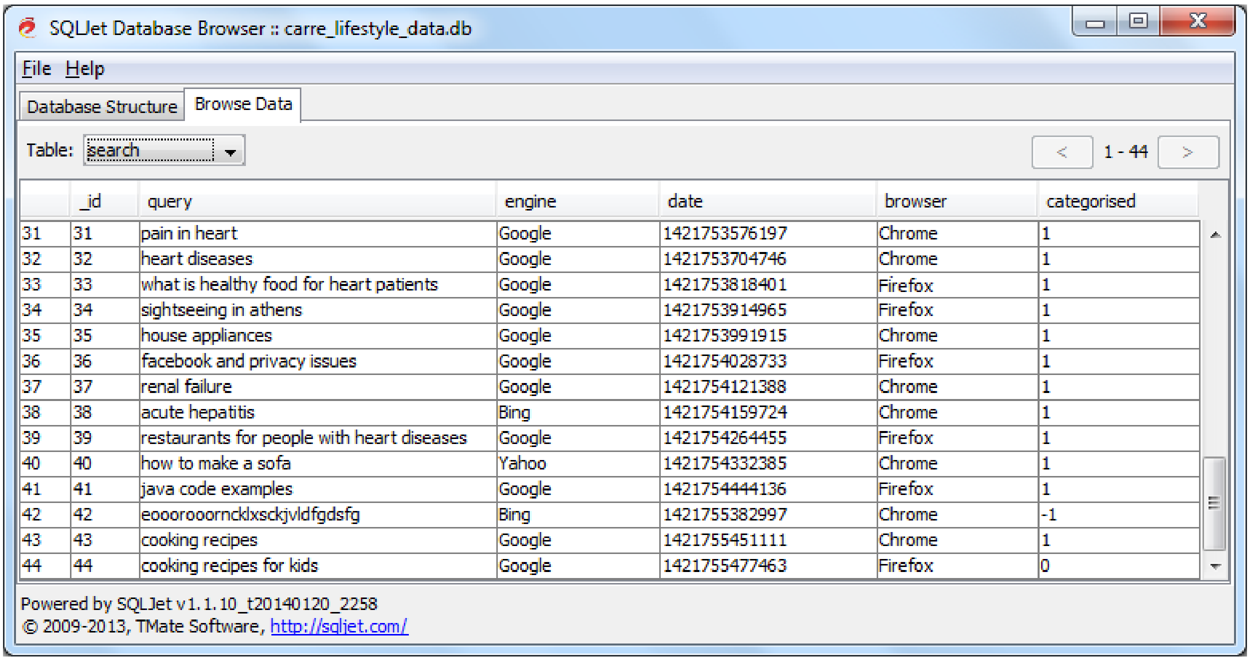


Figure 13. A snapshot of the User Intention Extractor in the system tray.



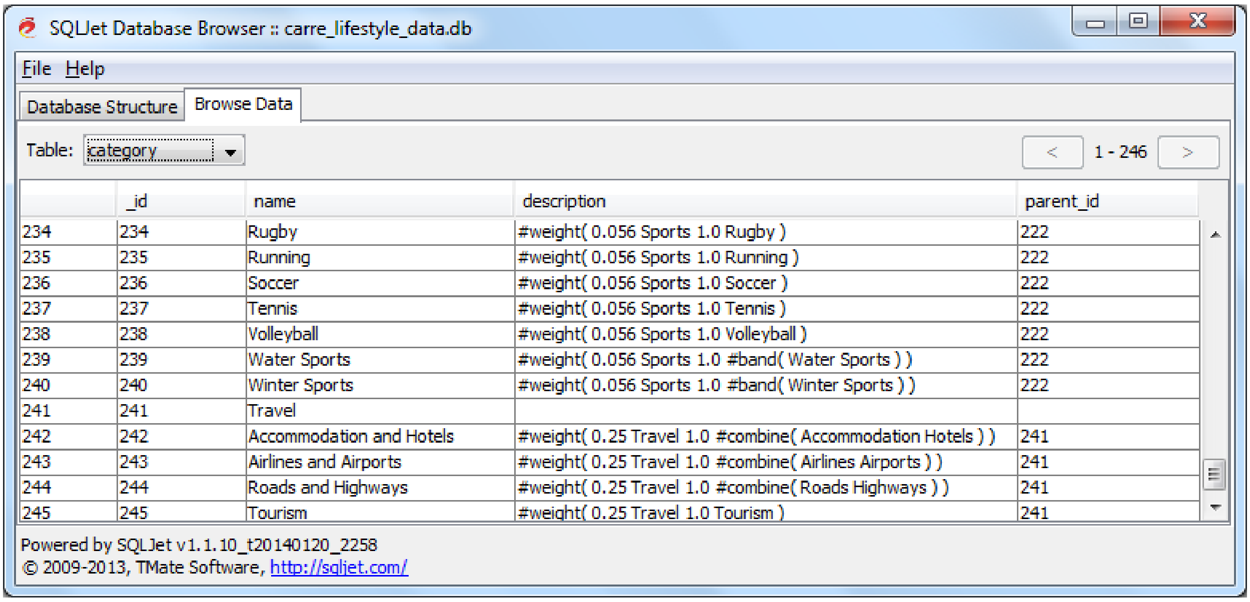
Name	Object	Type	Schema
categorisation	table		CREATE TABLE categorisation (_id integer PRIMARY KEY AUTOINCREMENT, ...
_id	field	integer PRIMARY K...	
search_id	field	integer NOT NULL	
category_id	field	integer NOT NULL	
score	field	real NOT NULL	
uploaded	field	integer DEFAULT 0	
category	table		CREATE TABLE category (_id integer PRIMARY KEY, name text NOT NULL, d...
_id	field	integer PRIMARY KEY	
name	field	text NOT NULL	
description	field	text	
parent_id	field	integer	
search	table		CREATE TABLE search (_id integer PRIMARY KEY AUTOINCREMENT, query t...
_id	field	integer PRIMARY K...	
query	field	text NOT NULL	
engine	field	text NOT NULL	
date	field	integer NOT NULL	
browser	field	text NOT NULL	
categorised	field	integer DEFAULT 0	

Figure 14. The SQLite database scheme of Local Storage.



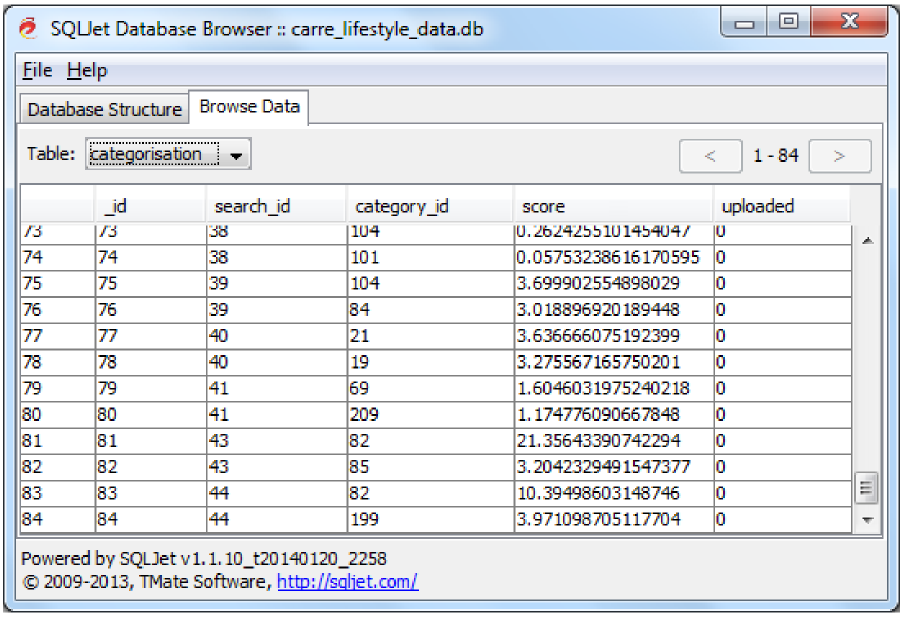
	_id	query	engine	date	browser	categorised
31	31	pain in heart	Google	1421753576197	Chrome	1
32	32	heart diseases	Google	1421753704746	Chrome	1
33	33	what is healthy food for heart patients	Google	1421753818401	Firefox	1
34	34	sightseeing in athens	Google	1421753914965	Firefox	1
35	35	house appliances	Google	1421753991915	Chrome	1
36	36	facebook and privacy issues	Google	1421754028733	Firefox	1
37	37	renal failure	Google	1421754121388	Chrome	1
38	38	acute hepatitis	Bing	1421754159724	Chrome	1
39	39	restaurants for people with heart diseases	Google	1421754264455	Firefox	1
40	40	how to make a sofa	Yahoo	1421754332385	Chrome	1
41	41	java code examples	Google	1421754444136	Firefox	1
42	42	eooorooornckixsckjvldfgdsfg	Bing	1421755382997	Chrome	-1
43	43	cooking recipes	Google	1421755451111	Chrome	1
44	44	cooking recipes for kids	Google	1421755477463	Firefox	0

Figure 15. A sample of the collected searches in database.



	_id	name	description	parent_id
234	234	Rugby	#weight(0.056 Sports 1.0 Rugby)	222
235	235	Running	#weight(0.056 Sports 1.0 Running)	222
236	236	Soccer	#weight(0.056 Sports 1.0 Soccer)	222
237	237	Tennis	#weight(0.056 Sports 1.0 Tennis)	222
238	238	Volleyball	#weight(0.056 Sports 1.0 Volleyball)	222
239	239	Water Sports	#weight(0.056 Sports 1.0 #band(Water Sports))	222
240	240	Winter Sports	#weight(0.056 Sports 1.0 #band(Winter Sports))	222
241	241	Travel		
242	242	Accommodation and Hotels	#weight(0.25 Travel 1.0 #combine(Accommodation Hotels))	241
243	243	Airlines and Airports	#weight(0.25 Travel 1.0 #combine(Airlines Airports))	241
244	244	Roads and Highways	#weight(0.25 Travel 1.0 #combine(Roads Highways))	241
245	245	Tourism	#weight(0.25 Travel 1.0 Tourism)	241

Figure 16. A sample of the categories that are used.



	_id	search_id	category_id	score	uploaded
73	73	38	104	0.2624255101454047	0
74	74	38	101	0.05753238616170595	0
75	75	39	104	3.699902554898029	0
76	76	39	84	3.018896920189448	0
77	77	40	21	3.636666075192399	0
78	78	40	19	3.275567165750201	0
79	79	41	69	1.6046031975240218	0
80	80	41	209	1.174776090667848	0
81	81	43	82	21.35643390742294	0
82	82	43	85	3.2042329491547377	0
83	83	44	82	10.39498603148746	0
84	84	44	199	3.971098705117704	0

Figure 17. A sample of the categorized searches.

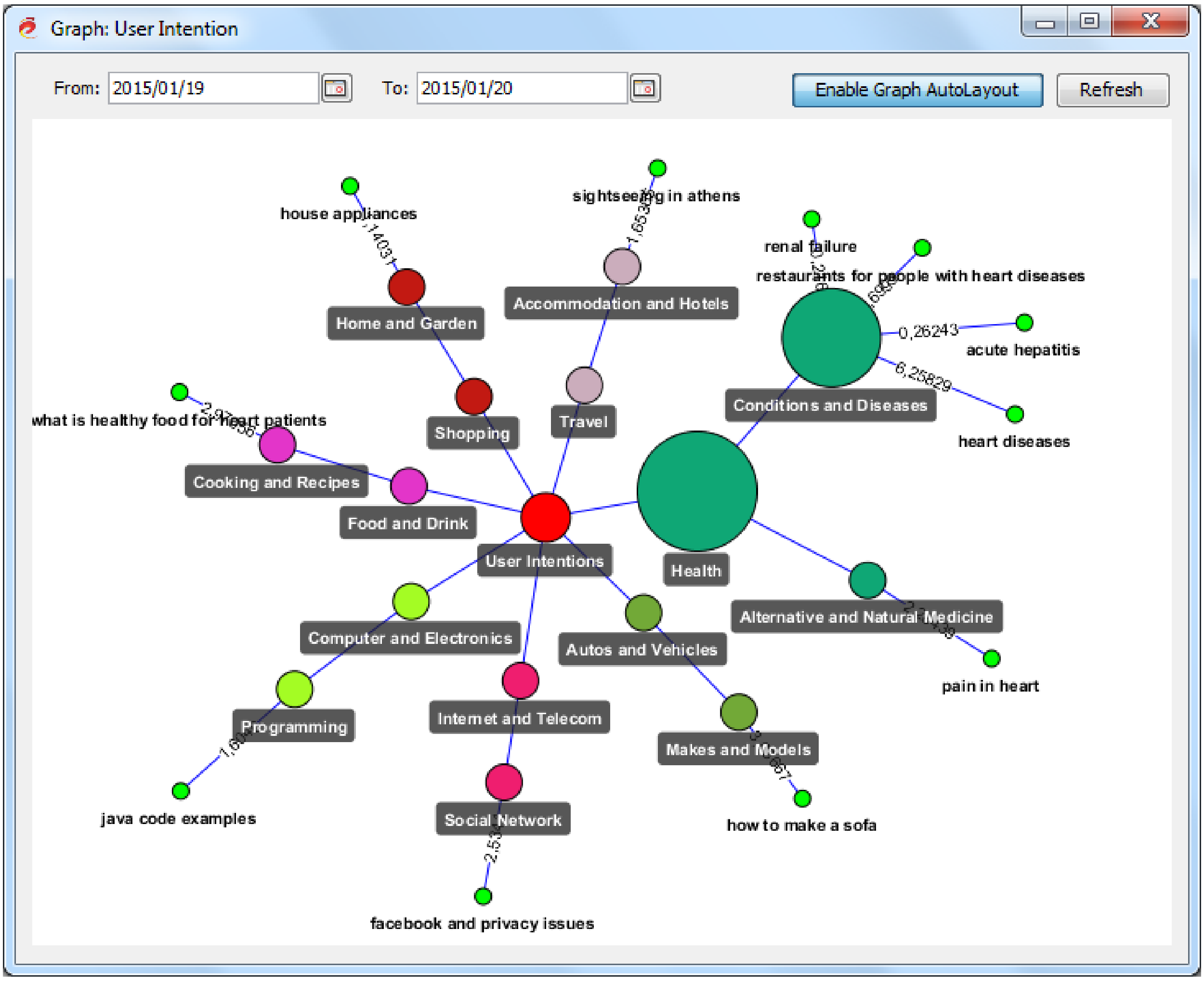


Figure 18. The visualization of the categorized searches in a particular time period.

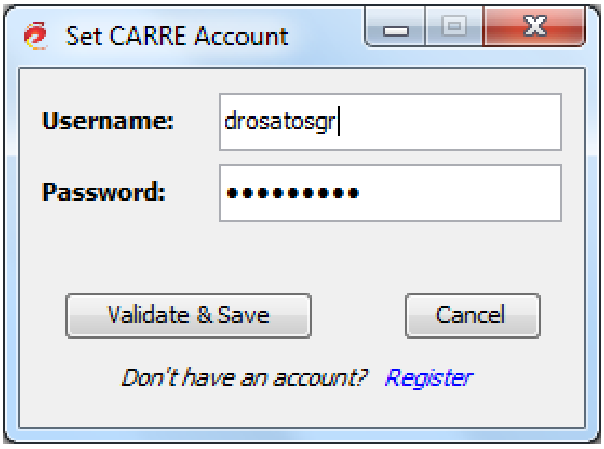


Figure 19. A snapshot of the patient's authentication to her Private RDF repository.

Table 7. Example of RDF triples that are inserted to Private RDF repository per user's intention.

```
PREFIX carreConnection:<https://carre.kmi.open.ac.uk/users/CARRE_USERNAME/connections/>
PREFIX carreMeasure:<https://carre.kmi.open.ac.uk/users/CARRE_USERNAME/measurements/>
PREFIX carreLifestyle:<https://carre.kmi.open.ac.uk/ontology/lifestyle.owl#>
PREFIX carreProvider:<https://carre.kmi.open.ac.uk/providers/>
PREFIX xsd:<http://www.w3.org/2001/XMLSchema#>

carreMeasure:d8f96a15 carreLifestyle:has_date carreMeasure:d8f96a15_date
carreMeasure:d8f96a15_date carreLifestyle:has_value "2015-01-21T19:12:25Z"^^xsd:datetime
carreMeasure:d8f96a15 carreLifestyle:has_weight carreMeasure:d8f96a15_weight
carreMeasure:d8f96a15_weight carreLifestyle:has_value "3"^^xsd:integer
carreMeasure:d8f96a15 carreLifestyle:is_provided_by carreProvider:web_queries
carreMeasure:d8f96a15 carreLifestyle:has_connection carreConnection:706d8a894b9a
carreMeasure:d8f96a15 carreLifestyle:has_intention carreLifestyle:conditions_and_diseases
```

Tables Table **8**, Table **9** and Table **10** present the code metrics of the implemented components (Query Detector and User Intention Extractor) of the Web Lifestyle Data Aggregator.

Table 8. Code metrics of Firefox Query Detector.

Firefox Query Detector JavaScript code metrics	
Number of Folders	3
Number of JavaScript Files	2
Number of Methods	7
Lines of Code	124
JavaScript Libraries	1

Table 9. Code metrics of Chrome Query Detector.

Chrome Query Detector JavaScript code metrics	
Number of Folders	3
Number of JavaScript Files	3
Number of Methods	28
Lines of Code	306
JavaScript Libraries	3

Table 10. Code metrics of User Intention Extractor.

User Intention Extractor Java code metrics	
Number of Packages	18
Number of Classes	50
Number of Fields	203
Number of Methods	243
Number of Static Fields	82
Number of Static Methods	46
Lines of Code	6073
Lines of Comments	1085
Java Libraries	28
C++ Libraries (JNI wrapper)	1

4.4. System requirements

The binaries and the source codes are available at: http://www.carre-project.eu/lifestyle-aggregator. Recommended system requirements for optimal performance are given in Table 11**Table 11**.

Table 11. System requirements of Web Lifestyle Data Aggregator.

	Windows requirements	**Mac requirements**	**Linux requirements**
Operating system	Windows XP SP2+ Windows Vista Windows 7 Windows 8	Mac OS X 10.6 or later	Ubuntu 12.04+ Debian 7+ OpenSuSE 12.2+ Fedora Linux 17
Software	Firefox v34+, Chrome v39+ and Java JRE 8+		
Processor	Intel Pentium 4 or later	Intel	Intel Pentium 4 or later
Free disk space	850 MB		
RAM	768 MB		

4.5. Discussion

The prototype implementation of Web Lifestyle Data Aggregator architecture confirmed the feasibility of our approach. However, there are still open interesting researching issues that we would like to investigate. In particular, the query classification process can be improved by making the required optimizations in health and travelling categories/intentions targeted for the patient decision support services as they will be designed in WP6. Additionally, we will perform evaluation of the results in the context of information retrieval during testing in T.7.2. Also, the Web Lifestyle Data Aggregator application will be evaluated in WP7 to receive the user feedback for: usability of implemented components; privacy concerns of the users; and quality of the detected intentions.

Annex 1

Personal Health Record Aggregator Software

What is CARRE Personal Health Record Aggregator?

The main goal of the CARRE PHR Aggregator is to harvest patient medical related data from various patient health record systems, to convert that data to RDF triples and to store that RDF in the CARRE semantic repository.

The main parts of this aggregator are the **Input Module**, the **Resource Unification Module,** and the **Output Module**.

- The **Input Module** is a medical data retriever designed to fetch patient data stored by PHR systems.
- The **Resource Unification Module** is a service that unifies fetched patient medical data into a single dataset.
- The **Output Module** is a service that converts unified medical data into RDF triples and stores it in the CARRE semantic repository.

Download

CARRE personal Health Aggregator:

Download from: http://www.carre-project.eu/innovation/phr-aggregator/

- Source (107.9 MB): CARRE_PHR_aggregator_v0.1.zip

CARRE Personal Health Record Aggregator is Open Source

Copyright © 2015, CARRE Project, Vilnius University Hospital Santariškių Klinikos (VULSK), Lithuania

Annex 2

Manual Personal Medical Data Entry System

What is CARRE Manual Personal Medical Data Entry System?

The main goal of the CARRE Manual Personal Medical Data Entry system is to allow patients to enter their personal medical data, to convert that data to RDF triples and store RDF in the CARRE semantic repository.

The main parts of this aggregator are personal medical data entry **Web Forms**.

- The **Personal Information Web Form** allows patient to enter his/hers personal information, e.g. first name, last name and email address.
- The **Diagnosis Web Form** allows patient to enter his/hers diagnosis information, e.g. ICD code, description and recovery date.
- The **Medication Web Form** allows patient to enter his/hers medication information, e.g. active ingredient of the prescribed medication, start and end dates of the prescription.

Download

CARRE Manual Personal Medical Data Entry System:

Download from: http://www.carre-project.eu/innovation/phr-manual-data-entry-system/

- Source (60.84 MB): CARRE_Personal_Medical_Data_Entry_v0.1.zip

Access

CARRE Manual Personal Medical Data Entry System:

Access via: http://www.carre-project.eu/innovation/phr-manual-data-entry-system/

CARRE Manual Personal Medical Data Entry System is Open Source

Copyright © 2015, CARRE Project, Vilnius University Hospital Santariškių Klinikos (VULSK), Lithuania

Annex 3

Web Lifestyle Aggregator Software

What is CARRE Web Lifestyle Data Aggregator?

The main goal of Web Lifestyle Data Aggregator is to use patient's web searches in order to extract her intention. The main parts of this aggregator are the **Query Detector** and the **User Intention Extractor**.

- The **Query Detector** constitutes a browser extension (Firefox add-on and Chrome extension) that is responsible to detect the user's queries in the web search engines (e.g. Google, Bing and Yahoo).
- The **User Intention Extractor** is responsible to store (only) locally the incoming queries and categorize them in specific categories (e.g. traveling, health diseases, etc.) in order to extract the patient's intentions. The extracted intentions that are relevant to CARRE are uploaded to the Private RDF Store of patient.

Download

Query Detector v0.2:

Download from: http://www.carre-project.eu/innovation/web-lifestyle-data-aggregator/

- Firefox
  Add-on (175 KB): CARRE_QueryDetector_Firefox_v0.2.xpi
  Source (101 KB): CARRE_QueryDetector_Firefox_v0.2.zip
- Chrome
  Extension (276 KB): CARRE_QueryDetector_Chrome_v0.2.crx
  Source (275 KB): CARRE_QueryDetector_Chrome_v0.2.zip

User Intention Extractor v1.0:

Download from: http://www.carre-project.eu/innovation/web-lifestyle-data-aggregator/

- Binary for Windows x64 (27 MB): CARRE_UserIntentionExtractor_v1.0.zip (Require Java JRE 8)
- Source for NetBeans v8.0.2+ (38 MB): CARRE_UserIntentionExtractor_src_v1.0.zip (Java code)
- Index for categorization (399 MB): Index Similarweb V4 (It is required for the execution of User Intention Extractor and it have to be extracted on folder "files"

Web Lifestyle Data Aggregator is Open Source

CARRE Web Lifestyle Data Aggregator is Open Source and can be freely used in Open Source applications under the terms GNU General Public License (GPL).

Copyright © 2015, CARRE Project, Democritus University of Thrace (DUTH), Greece