Semantic Patient Information Aggregation and Medicinal Decision Support

Pieterjan De Potter\textsuperscript{a}, Hans Cools\textsuperscript{b}, Kristof Depraetere\textsuperscript{b}, Giovanni Mels\textsuperscript{b}, Pedro Debevere\textsuperscript{a}, Jos De Roo\textsuperscript{b}, Csaba Huszka\textsuperscript{b}, Dirk Colaert\textsuperscript{b}, Erik Mannens\textsuperscript{a}, Rik Van de Walle\textsuperscript{a}

\textsuperscript{a}Department of Electronics and Information Systems - Multimedia Lab, Ghent University - IBBT, Gaston Crommenlaan 8 Bus 201, B-9050 Ledeberg-Ghent, Belgium
\textsuperscript{b}Advanced Clinical Applications Research Group, Agfa HealthCare, Moutstraat 100, B-9000 Ghent, Belgium

Abstract

Although the health care sector has already been subjected to a major computerization effort, this effort is often limited to the implementation of standalone systems which do not communicate with each other. Interoperability problems limit health care applications from achieving their full potential. In this paper, we propose the use of Semantic Web technologies to solve interoperability problems between data providers. Through the development of unifying health care ontologies, data from multiple health care providers can be aggregated, which can then be used as input for a decision support system. This way, more data is taken into account than a single health care provider possesses in his local setting. The feasibility of our approach is
demonstrated by the creation of an end-to-end proof of concept, focusing on Belgian health care providers and medicinal decision support.

**Keywords:** Semantic Web, Electronic health care, Data aggregation, Decision support

1. **Introduction**

Multiple studies have illustrated how clinical decision support systems and computerized physician order entry can save lives [1, 2, 3, 4], with several systems already used in practice. These systems require patient information to be in a central database. In certain cases (e.g., when a patient enters a hospital, goes to a general practitioner on duty, or is admitted in emergency care), a lot of information about the patient is not available at a certain location, but is at other locations (e.g., the regular general practitioner or the pharmacist of the patient).

Although patient information is mainly stored in proprietary formats, several standards exist for the exchange of patient information e.g., the Clinical Document Architecture [5], the Continuity of Care Record [6], Kind Messages for Electronic Health Care Record - Belgian Implementation Standard (Kmehr-Bis,[7]). In addition, different coding systems exist to unambiguously represent concepts mentioned in these exchange documents e.g., International Statistical Classification of Diseases and Related Health Problems [8], Logical Observation Identifiers Names and Codes [9], International Classification of Primary Care [10], Code National(e) Kode for medication codes (CNK, [11]).

To enhance interoperability and to enable computerized clinical support,
there is a need for one explicit, unambiguous representation of patient data, where files in different formats can be mapped on.

Different efforts have already been undertaken to make a unique identification of concepts and the modeling of medical knowledge possible. Multiple terminologies, such as the Systematized Nomenclature of Medicine - Clinical Terms [12] and the Unified Medical Language System [13], have been created. Archetypes have been modeled in the openEHR environment [14]. Ontologies, such as the Gene Ontology [15], the Generalised Architecture for Languages, Encyclopaedias and Nomenclatures in Medicine [16], the Unified Medical Language System ontology [17], and other ontologies included in BioPortal [18], have been created.

The major advantage of modeling knowledge using ontologies over other systems is that powerful axioms can be implemented by creating several kinds of links between concepts and applying restrictions to these links and concepts. The information contained in these axioms can be used when data modeled using such an ontology is consumed, for example by a reasoning engine. However, the ontology creation efforts described above often follow a top-down approach, leading to a lot of general concepts and philosophical discussions, or a bottom-up approach, leading to too much detail. Our approach is middle-out: we start with the concepts we need and create extra concepts to link other concepts together if necessary. In the ontology we create this way, we map concepts to multiple terminologies and link them to concepts defined in other ontologies to enhance the semantic value. Like this, data published elsewhere on the Semantic Web can be used as a source of knowledge in our system and vice versa.
Our patient information aggregation and clinical decision support system was developed as part of the IBBT Share4Health project [19]. In this project, a patient centric health care IT platform that addresses the needs of next generation clinical applications was developed. Our work, described in this paper, consisted of creating a proof of concept to demonstrate the feasibility of Semantic Web technologies for enabling interoperability between different health care providers, aggregating data from multiple resources, and providing a medical decision support service.

The remainder of this paper is organized as follows. Related work concerning the use of semantic web technologies for the presentation of patient information and clinical decision support systems is described in 2. In Section 3, we lay out the general architecture of our system. In Section 4, we elaborate on the ontologies we use to represent information in our system. The transformations between different parts of our system are described in Section 5. In Section 6, we describe how information from different resources is aggregated. The decision support system that is used to identify medicinal conflicts is introduced in Section 7. A discussion about the added value of Semantic Web technology is provided in Section 9. Section 8 contains the evaluation of our system. Finally, conclusions and future work are given in Section 10.

2. Related work

In [20], Hussain and Abidi integrate knowledge from multiple sources (e.g., clinical guidelines, clinical pathways, knowledge of practitioners) through the semantic modeling of health care knowledge as ontologies and reasoning
over the ontologies to derive a morphed knowledge object. Their work focuses on the merging of information from multiple sources, while the work in our paper focuses on the aggregation of patient information, assuming that the medical knowledge such as the clinical guidelines and pathways is already strictly defined (but not necessarily using Semantic Web technology).

Bouamrane et al. [21] describe the use of ontologies to create adaptive questionnaires and to represent the information collected during these pre-operative questionnaires. The information they model is limited to a very constrained domain in order to keep the information well defined and manageable. In our paper, an approach to manage more information is introduced, using a multitude of small ontologies. These ontologies can be extended and more ontologies can be created and linked when necessary. Although application in other areas is possible, here we focus on the medicinal history of patients as a use case.

ISABEL [22] is a web-based differential diagnostic aid for paediatrics. It is intended to remind clinicians of diagnoses that might have been missed. Text from multiple textbooks related to different diagnostic labels are added to a database and the labels are allocated to an age group classification. Text entered by a clinician is matched to the database resulting in different diagnosis related to this set of clinical features. This system is purely based on information entered by a clinician and does not take the patient history into account.

The Brigham integrated computing system [23] provides an order entry system, enabling order checking at the time it is written. Examples of interventions are: the suggestion of alternative medications, parameter check and
suggestion (warning of possible overdoses), drug allergy and drug interaction checking. Health care practitioners are divided in classes with separate rights and dependent on the treatment (and dose) cosigning of an order might be required. The system is restricted to a specific system and it is unclear what patient information is used for drug allergy and interaction checking.

In [24], Kataria et al. describe the implementation of an ontology for intelligent hospital wards. It is used to deal with the semantic heterogeneity across different data repositories in a hospital. The (unpublished) ontology seems to be limited to match the databases from which the information needs to be gathered and no links were made with existing ontologies, resulting in a minimum of semantic value in the defined classes and properties. Once the data is mapped on the ontologies, traditional technologies (i.e., JSPs, servlets, session and entity beans) are used to aggregate information and collect alerts for monitoring patients vital signs. In our approach, we maximize the semantic value of the defined classes and properties and demonstrate the use of semantic technologies for the entire end-to-end process from data mapping to decision support.

In [25], we introduced ontologies that model patient information, discussed possible use cases and proposed a possible approach to enable data aggregation and decision support. Our current paper gives a complete overview of our system: a description of the ontologies is given, the aggregation of relevant patient information is described and a method to identify medicinal conflicts is introduced.
3. General architecture

3.1. Cross-enterprise document sharing

As illustrated in Figure 1, a cross-enterprise document sharing (XDS, [26]) architecture is used to access the files from the different health care providers\(^1\). All documents from the health care providers are stored in XDS repositories and registered to the XDS registry. The aggregation service is added as a virtual repository: for each patient a virtual document is registered to the XDS registry; when this document is requested from the XDS repository, the document is created on the fly, based on the information available in the other repositories.

3.2. On the fly aggregation

When a client requests the aggregated medical information of a patient via the portal (1), the URI representing the virtual aggregated document is first retrieved from the XDS registry (2,3), as illustrated in Figure 2. Upon the access of this URI by the portal (4), the aggregation service requests URIs from all available medical information from the registry (5,6). All available documents are then retrieved by the aggregation service (7,8), processed and returned to the portal (9), which displays the retrieved information to the client (10).

\(^1\)KLAV and AVK are Belgian pharmacists associations. Recip-e is a pilot project created to test the roll out of an electronic prescription system in Belgium. Sumehr is a repository containing summarized medical histories maintained by the general practitioner of a patient.
The patient information created by the different health care providers is delivered in Kmehr [7] format. The documents are first transformed to the semantic RDF (resource description framework) format, mapping the data on instances of concepts from our ontologies. Semantic rules are applied to a converted version of these documents in N3 (notation 3) format to obtain the aggregated document.

The aggregated document is transformed to a web page which is presented to the end user and is also used as input for the decision support system. The output of the decision support system is dynamically loaded into the HTML (hypertext markup language) page displayed to the end user using AJAX (Asynchronous JavaScript and XML (extensible markup language), [27]) requests.
4. Ontologies

This section introduces the ontologies developed for the representation of medical information. Firstly, the specifications of the ontologies are given. Secondly, some aspects of the actual development of the ontologies are described. In the last subsection, some of the details about the publication of the ontologies are given.

4.1. Specifications

4.1.1. Coverage

The ontologies cover topics related to health care and more general topics, needed for the definition of these topics. The ontologies elaborate on three topics: medication (i.e., the prescription, delivery, and therapy/administering), the patient summary and Chronic Heart Failure (CHF). In this project,
medication was chosen as a use case, while the elaboration on CHF was done in the CHF project [28].

4.1.2. Knowledge sources

Most data elements defined in the ontologies are obtained from domain experts, literature corpora (e.g., clinical guidelines), coding systems, ontologies and terminologies (e.g., International Classification of Primary Care, International Statistical Classification of Diseases and Related Health Problems, Systematized Nomenclature of Medicine - Clinical Terms), and use cases. These sources give a first formal expression of the domain to be refined in an incremental way.

4.1.3. OWL 2 Full and Notation 3

The ontologies are created in OWL 2 Full (OWL: web ontology language [29]; in the Full sublanguage, RDF-based semantics are used to assign meaning to ontologies), since this allows maximum expressiveness. Because of the very liberal expressiveness in OWL Full it is theoretically possible to create some input for which a reasoner can not terminate the reasoning process [30]. In practice we did not encounter such cases and if there would be any, they can be traced and re-engineered.

The use of OWL Full allows free mixing of OWL with RDF Schema and does not enforce a strict separation of classes, properties, individuals and data values [29]. An example of a non-strict separation of classes and individuals is illustrated in Listing 1\(^2\), which describes the declaration of the

\(^2\)For clearness reasons, certain information (e.g., prefixes, labels) is omitted in the example listings. References to the complete ontologies are given in Section 4.3.
class Monday as a subclass of the class Day and an instance of the class Weekday.

Listing 1: Non-strict separation of classes and individuals (OWL Full)

event:Monday
  a rdfs:Class, event:Weekday;
  rdfs:subClassOf event:Day.

event:Weekday
  a rdfs:Class;
  rdfs:subClassOf event:Day;
  owl:oneOf(event:Monday event:Tuesday event:Wednesday event:
    Thursday event:Friday event:Saturday event:Sunday).

event:Day
  a rdfs:Class;
  rdfs:subClassOf event:Event, [a owl:Restriction; owl:onProperty
    event:hasDuration; owl:hasValue "P1D"^^xsd:duration].

One of the benefits of OWL 2 is the advanced use of properties. Listing 2 illustrates the use of a property chain to declare the birth date of an organism as the begin date of the live of that organism.

Listing 2: Property chain (OWL 2)

organism:hasBirthDate
  a owl:DatatypeProperty, owl:FunctionalProperty;
  owl:propertyChainAxiom (organism:lives organism:beginDate);
  rdfs:domain organism:Organism;
  rdfs:range xsd:date.

organism:lives
The syntax that is used to formalize the ontologies is N3 (notation 3 [31]). RDF/N3 is as complete as RDF/XML, but more human readable. Another advantage of N3 is that it can also be used to express rules and queries.

4.1.4. Quality

Besides its suitability within an application scenario, the quality of an ontology depends on the correctness and (Socratic) completeness of the representation. The quality of the ontologies is assessed on formal criteria. Logical soundness is crucial and is tested throughout the project by the usage of the Euler reasoning engine [32]. Developers on the other hand depend strongly on the help of the domain experts concerning the content. Review by domain experts regarding the correctness of the ontologies (and their entailments) is equally important.
4.2. Development

In this section, some of the aspects that were taken into account during the development of the ontologies are described.

4.2.1. Avoiding redundancy

On the one hand, redundancy in our own ontologies e.g., having the same restriction on a class and a subclass of this class, has to be avoided. On the other hand, redundant work can be prevented by searching and referencing existing ontologies. Classes, properties and instances from ontologies that are already widely used and have proven to be more or less stable are chosen, to provide the basis of our ontologies. Below are some examples.

**FOAF.** The Friend of a Friend (FOAF) [33] project is about describing people and the links between them. FOAF has been developed for Semantic Web use. Although this specification still has unstable elements in it, its usage has grown with the expansion of social network communities [34]. In our ontologies, FOAF is used to describe entities that are able to play a role.

**Contact.** To represent contact information e.g., an address or a phone number of a person who is playing a specific role, the Contact ontology [35] is used.

**NASA SWEET.** The NASA Semantic Web for Earth and Environmental Terminology (SWEET, [36, 37]) ontologies provide a semantic framework for projects on the subject of Earth science. In our ontologies, the SciUnits ontology from the second version of SWEET is used for additional modeling of units. Also other concepts are used, such as the ‘Country’ class from the ‘humanJurisdiction’ ontology.
Dublin Core. For the description of the metadata of the ontologies, the Dublin Core (DC) Metadata Element Set [38] is used. Also, some concepts of the Metadata Terms Set [39] are applied, e.g., to indicate that a concept is a physical resource.

Conflicts and overlap. Since only classes, properties and instances with which we agree are reused, conflicts and overlaps can be avoided. Classes, properties and instances are declared as subclasses and subproperties of existing classes, properties or instances when elaboration is necessary.

A specific example of the reuse of the http://sweet.jpl.nasa.gov/2.0/sciUnits.owl#kilogram (unit:kilogram) instance is given in Listing 3: the units:kilogram instance (from http://eulersharp.sourceforge.net/2003/03swap/units#) is defined as a subclass of unit:kilogram and restricted to be a unit of mass measurement. It is also linked to the Systematized Nomenclature of Medicine - Clinical Terms concept “kilogram” via the concept ID 258683005.

Listing 3: Reuse of existing ontologies

```xml
units:kilogram  
a units:Unit;

rdfs:subClassOf unit:kilogram , [ 
a owl:Restriction; owl:onProperty quant:unitOf; owl:
    someValuesFrom quant:MassMeasurement ];

skos:exactMatch [ 
a skos:Concept; skos:inScheme clisko:sct20080731; skos:
    notation "258683005"^^clisko:sct20080731DT ].
```
4.2.2. Human readable labels

Readable names are given as annotation labels. Also, classes and properties are provided with precise textual definitions, especially where it is not easy to quickly extract enough contextual information from formal declarations.

4.2.3. Thoughtful use of restrictions

Existentially quantified restrictions make very strong ontological assumptions, as they assert the existence of a related entity.

4.2.4. Default values

Default values are not used because this implies non-monotonicity. E.g., one could declare mmHg (millimeter mercury) as the default unit for blood pressure, but when during reasoning no unit is mentioned in the patient data, this unit is not necessarily the right one. Therefore, units must be declared or the corresponding patient data will be discarded. Input data can of course be checked and units that are implicitly used in certain formats or organizations can be added during conversion.

4.2.5. Reusability

The developed ontologies vary strongly, ranging from rather general to domain-specific, from large to small, and from superficial to detailed. By defining a multitude of ontologies instead of one large ontology containing all introduced concepts and relations, enhances the reusability of the resources. The formal description of the concepts declared in the Share4Health project and related projects is currently split up in 74 namespaces along content.
4.3. Publication

The ontologies are published in an open source environment to ensure wide access and to facilitate reuse. The health care specific ontologies can be found at http://www.agfa.com/w3c/2009/, while the more general ones can be found at http://eulersharp.sourceforge.net/2003/03swap/.

5. Transformation

Since in our case, there are not that many different XML formats used, XSL (extensible stylesheet language) transformations are used for the transformation between these formats (Kmehr, RDF, and (X)HTML). In our framework, any format can be used, as long as it can be converted to XML format. However, in case of more different formats, a more generic transformation approach would be necessary, such as the one presented in [40].

The mapping of the input files to the ontology is done by applying XSL transformations on Kmehr data resulting in RDF data (see Figure 2). An instance is created for the person representing the patient, which fulfills the role of a patient and to which the patient code (retrieved from the Kmehr document) is assigned. Then, for each medication delivery, the patient instance is linked to a delivery instance, which is linked to the corresponding date and medication code. Medication prescriptions and therapies are handled in a similar way. This result is then converted to N3 format, aggregated and converted back to RDF format. The result of the aggregation in RDF format is transformed to an HTML document that is displayed to the end user.

An example of a transformed patient file with a single prescription, after
conversion to N3, but before aggregation is represented in Listing 4. Note
that the created instance is a blank node. Since the fos:hasCodeValue and
fos:hasCodeSystem are owl:InverseFunctionalProperty properties, it is possi-
bile to derive which of these instances represent the same patient.

Listing 4: Transformation example (result)

[agent:playsRole [a heca:Patient];
 fos:hasCode [a hecaad:BelgianPatientCode;
 fos:hasCodeSystem "Kmehr Patient"^^rdfs:Literal;
 fos:hasCodeValue "00992556784"^^hecaad:inszPatientCodeDT];
 drug:gotPrescribed [drug:prescribedIn [a drug:Prescription;
  clinproc:hasPrescriptionTimeStamp "2007-12-27T00:00:00"^^
  xsd:dateTime];
 :exactMatch [a :Concept;
   :inScheme clinskosch:cnk1;
   :notation "0044057"^^clinskosch:cnk1DT]]].

6. Aggregation

Using semantic rules and a reasoning engine, data that has a semantic
representation can be processed. We use this method to aggregate data from
multiple resources. In this section, firstly the Euler reasoning engine is briefly
described, secondly the rules that are used to aggregate the data are intro-
duced, followed by the aggregation queries. At last, the rules and query that
assist in post factum decision support are given. The rules and queries dis-
cussed in this section can be found at http://multimedialab.elis.ugent.
be/users/pdpotter/S4H/queries.n3 respectively.
6.1. Euler reasoning engine

Euler is an inference engine supporting logic based proofs. It is a backward chaining reasoner enhanced with Euler path detection. Backward chaining reasoners start from the list of goals that need to be proven and look backwards to see if there is any data available that can support one of these goals. If it is unknown whether this data is true, then this data is added to the list of goals. These steps are repeated until either there is sufficient proof to prove the goals, or there is nothing added to the list of goals, in which case it is not possible to prove the goals. The Euler path detection prevents the same goals to be added multiple times to the list of goals, which would introduce an infinite loop. The project homepage can be found at http://eulersharp.sourceforge.net/.

6.2. Aggregation rules

In our use case, prescriptions (retrieved from a general practitioner), deliveries (retrieved from a pharmacy) and descriptions of medicinal therapies (retrieved from a patient summary maintained by a general practitioner) are taken into account. For a patient, any combination of these appearances can be retrieved combining the documents from the different repositories. The rules for the aggregation of the data of the multiple data sources consist of the following steps.

1. The input data is filtered per patient using the patient ID.
2. The relevant dates of all prescriptions, all deliveries and all therapies related to a certain medication are gathered in separate lists. This is done using the http://eulersharp.sourceforge.net/2003/03swap/
log-rules#findall built-in. This built-in has rdf:List as rdfs:domain and rdfs:range and allows to construct a “select” statement containing one or multiple “where” clauses in a certain “scope”. The “scope” allows to create nested findall constructs.

3. Based on the emptiness of the resulting lists, it is possible to determine in which of the above described appearances a certain drug occurs.

4. As result of the rule, the patient ID, drug ID, applicable situation for that drug, and the dates from the non-empty lists, with a description, are outputted.

These steps are illustrated in Listing 5, demonstrating the rule related to an interaction with a prescription and delivery, without a description of the therapy.

Listing 5: Aggregation rule example

```plaintext
{
?SCOPE e:findall (?timeStamp {[fos:hasCode [fos:hasCodeValue ?patientCodeValue];
   drug:gotPrescribed [skos:exactMatch [skos:notation ?cnk]; drug:
     prescribedIn [clinproc:hasPrescriptionTimeStamp ?timeStamp]]} ?listP).
   event:gotDelivered [skos:exactMatch [skos:notation ?cnk]].
?SCOPE e:findall (?timeStamp {[fos:hasCode [fos:hasCodeValue ?patientCodeValue];
```
6.3. Aggregation query

The aggregation query consists of multiple rules used to filter out results or to add extra data. By adapting this query, it is very easy to impose extra constraints on the results that are returned, in order to get the information that is really wanted. Our aggregation query mostly leaves the data created by the rules unmodified, apart from one addition: the query adds information about a medication if this information is present. The medication information is obtained from a file that contains instances with CNK codes of most of
the products that are currently or were ever in production, with a brief
description of each product. If the information is not present, “No product
information available” is added to the aggregated document.

6.4. Decision support assisting rules

For the post factum medicinal interaction checking, we check whether two
medicinal situations occur close enough after each other to have a possible
interaction. This interaction checking is performed in order to find flaws in
patient history or clinical pathways in the past and to prevent dangerous
situations like these from occurring in the future. Because therapies have a
begin and an end date, while prescriptions and deliveries only have a single
date stamp, three rules are needed:

- evaluate dates between prescriptions and deliveries, and also between
  prescriptions and deliveries among themselves;
- evaluate dates between a prescription or a delivery and a therapy;
- evaluate dates between therapies.

All time stamps are checked to occur within 90 days of each other. This
calculation is based on the maximum half-life of drugs we encountered, which
is 200 hours (for chloroquine [41] and valium (desmethyldiazepam) [42]). To
avoid drug interactions, one has to wait three to five times the half-life of an
administered drug before starting a new drug [43], which results in a period
of 42 days for drugs with the longest half-life. In certain cases (e.g., liver
disease [44]), the half-life of medication can be higher or lower. For this
reason, the interval in which is checked for interactions is expanded to 90 days.

If there is a therapy involved, with a start and an end date, both of these dates are compared to the time stamp of the prescription or delivery, or to both dates of a therapy. In the latter case, it is also verified that the start date of one therapy falls between the start and end date of the other therapy, in order to detect overlaps between them.

6.5. Decision support assisting query

The decision support assisting query leaves the data produced by applying the rules unchanged, apart from some minor renaming.

7. Decision support

Based on the aggregated data originating from multiple health care providers, we can offer decision support that takes more data into account than a single health care provider possesses in his local setting. The detection of medication interactions was chosen as use case for our project.

In this section, the knowledge base that we used to check the interactions is introduced, the SPARQL endpoint (SPARQL protocol and RDF query language, [45]) that we provided for this purpose is described, and the integration of this information source in the user interface is explained.

7.1. Knowledge base for decision support

For this project, we used the Delphi Care Database\(^3\), provided to us by the APB (Association Pharmaceutique Belge - Algemene Pharmaceutische

\(^3\)http://www.delphiplus.be/DutchHTML/delphi_voorstelling.htm (in Dutch)
The Delphi Care Database is a relational database containing information on medication (such as e.g., CNK code, ATC code (Anatomical Therapeutic Chemical, [46]), galenic form) and on interactions between different kinds of medication (such as e.g., type of interaction, effects, actions to take).

7.2. **SPARQL endpoint**

A SPARQL endpoint is created for the Delphi Care Database. The endpoint contains a mapping, using the D2R [47] system, from the Delphi Care Database entities and relationships to formal elements, which are declared in a Delphi Care ontology we created for this purpose. This ontology also reuses concepts we declared in other ontologies for our project. SPARQL queries using semantic relationships can then be used to query the database. These SPARQL queries are converted to SQL queries on the database using the D2R engine.

7.3. **Integration in the user interface**

Since the production of all possible information takes some time, the information that is available at a certain moment is immediately displayed to the end user. Extra information is loaded in the background, the most important information first. Requests from the end user are handled before pending requests for extra background information.

The information that is first displayed consists of the aggregated medicinal information, thus an overview of the different prescriptions, deliveries and

---

4http://www.apb.be/ (in Dutch)
therapies. This information is available immediately after the transformation to HTML format of the aggregated data.

The first piece of information that is dynamically loaded are the post factum medicinal interactions for medications that are prescribed, delivered or taken within 90 days of each other. This check is initiated by an AJAX request to the SPARQL endpoint. The SPARQL endpoint returns the information about possible interactions between the two requested medications. This result in RDF/XML format is transformed by an XSL transformation and added to the HTML page that is displayed to the end user. Possible interactions are added to a list that is shown to the user, on which he or she can click for more information.

Next, information about other combinations of medication, which are separated more than 90 days of each other, are requested and added in the background. In this way, a fast result can be given to the end user if he or she wishes to ask information about possible interactions between two selected medications. In addition, patients are often prescribed medication they already have taken once or more before, so a quicker response can be given if the user wants to check for interactions before he or she prescribes a medication.

The last piece of information that is loaded without user interaction is information about individual medications. This information is displayed to the end user when he or she clicks on a specific medication in the table with aggregated information.

When a user asks for specific information while there is still information loading in the background, the AJAX request that is running is first finished,
after which the AJAX request for the user is immediately processed. After this request, the loading of background information continues.

Besides the post factum medication interaction checking, there is also a possibility for the end user to check the interaction between a certain medication and the medication that has been prescribed, delivered or taken during the last 6 months. Possible interactions are then listed in a table, which can be clicked by the user for more information.

8. Evaluation

The complete system, consisting of the transformations, reasoning engine, virtual XDS repository, Delphi Care database and SPARQL endpoint, was deployed on a virtual machine with one dedicated 2Ghz processor and 1 GB of dedicated RAM for testing purposes. Ten virtual test patients were created by medical doctors, with a medical background that was created based on anonymized information from real patients. The test data per patient consisted of one patient summary containing information about prescriptions and therapies, between 20 and 30 files with prescriptions split up per month and between 50 and 70 files with deliveries of medications. These files were stored on several XDS repositories, as illustrated in Figure 2.

The total time, from the start of the request of an aggregated document for a certain patient to the moment this information is displayed to the user, is in the order of 5 minutes. However, this is mostly because of the lack of optimization of the XDS repositories and registry we had to our disposal. When the files that are normally stored in these repositories are stored locally, the result is obtained within half a minute. Parallelization
during transformation of the input files, transformation of input files on input and the storage of the data in a triple store, or the caching of the fixed files that are used for reasoning would reduce the processing time.

The time needed for more information on a certain medication is about 15 ms on average for the SPARQL request and the transformation of the result. A request for an interaction check between two medications takes about 30 ms on average. Of course, the transmission time needs to be added, but this should be fast enough for application in the real world.

9. Discussion

At this moment the Linked Open Data space is really getting momentum, but to be able to harness the full potential of the Semantic Web, more specific health care data is needed within this Linked Open Data cloud. Community efforts as DBpedia\textsuperscript{5} already contain a lot of information, but for health care applications, more trusted sources are wanted. Governments can and should provide this kind of information and recently some started publishing the data available to them, rightfully adhering to the Semantic Web principles\textsuperscript{6}.

Once information (e.g., the half-life of drugs) becomes available as Linked Open Data, it can immediately be consumed by our system and the provided knowledge can be employed to improve the current decision support system. This also works in the other direction: the knowledge modeled in our ontologies or presented using our ontologies can automatically be used by other systems on the Web.

\textsuperscript{5}http://wiki.dbpedia.org/About
\textsuperscript{6}http://www.data.gov/semantic
10. Conclusions and future work

In this paper we presented a platform that aggregates information from multiple health care providers on one side, and provides clinical decision support based on this aggregated information on the other side. We introduced general and health care related ontologies, developed in close collaboration with domain experts and published them in an open source environment, so this knowledge can be reused in future projects worldwide.

The proof of concept of an end-to-end solution for medicinal decision support, including transformations, ontologies, rules for aggregation of data from multiple sources and the use of a semantic endpoint in a dynamic HTML page, illustrates the possibilities of our platform.

Future work consists of considering more use cases and extending the ontologies and rules towards these use cases (e.g., tailored to specific diseases or trauma situations) in order to obtain the exact information that is requested by these situations. We will also extend our flexible system towards other incorporated data types.

In the use case of medicinal decision support, the knowledge base can be extended with information about allergies and adverse drug reactions, which could be enlisted in the patient summary maintained by the general practitioner of that patient. Other knowledge can lead to better summaries, e.g., the half-life of drugs can be used to calculate a more precise interval in which interactions between different drugs can occur.

Finally, in order to use this system in real-life, a lot of concerns on the subject of privacy of the patients and security of the data need further resolving. Also, when rolled out in emergency care departments, the reliability
and the speed of the system need improvement.

Acknowledgements

This work was supported in part by Ghent University, in part by the Agfa-Gevaert Group, in part by the Interdisciplinary Institute for Broadband Technology (IBBT), in part by the Institute for the Promotion of Innovation by Science and Technology in Flanders (IWT-Flanders), in part by the Fund for Scientific Research-Flanders (FWO-Flanders), and in part by the European Union.

Conflicts of interest statement

None.

References


   URL http://www.w3.org/TeamSubmission/n3/

   URL http://www.agfa.com/w3c/euler/

   URL http://xmlns.com/foaf/spec/


   URL http://www.w3.org/2000/10/swap/pim/contact#

URL http://sweet.jpl.nasa.gov/ontology/

URL http://dublincore.org/documents/dces/

URL http://dublincore.org/documents/dcmi-terms/


