Universität Leipzig
Faculty of Mathematics and Computer Science
Department of Computer Science

Patient-centered Drug Management based on Linked Open Data

Master's Thesis

Leipzig, September 2013

Submitted by
Nitzschke, Marcus
Master's programme Computer Science

Thesis Supervisors:
Prof. Dr. Klaus-Peter Fähnrich
Dipl. Inf. Romy Elze

Abstract

This thesis will present a solution for a simple way to answer several drug related questions, like “What are the side effects of drug X?” or “Do drug X and drug Y interact with each other?”. The knowledge foundation used for these queries are Linked Data knowledge bases which allow comprehensive queries over semantic data. The presented approach provides an Application Programming Interface (API) for the different questions that allows an usage without any background knowledge about the data sources and their structures. Additionally the predefined questions support the “Drug Management life cycle” introduced by the World Health Organization. Finally the presented solution will be evaluated by integrating it in the Dispedia and DrugMan project.

Keywords Semantic Web, Linked Open Data, Drug Management

Contents

1 Introduction

1.1 Subject

The prescription and application of drugs in health care is one of the most important parts in the healing process of a patient. If this process is not managed accurately consequences from financial losses to human harms are possible. In Germany estimations state that annually 25,000 patients die because of medication errors [1] (status 2010). Managing drugs means to support the different phases drugs passes during the healing process. The *World Health Organization* (WHO) published a Drug Management Cycle which defines four different phases [2]. Details about this Drug Management Cycle are given in section 2.1. An important base to implement a proper drug management and therefore to avoid the problem of medication errors is that all available knowledge about drugs and their characteristics is freely accessible and usable. Especially in the biomedical domain the role of ontologies for structuring such knowledge is still growing since decades. And in the last years a trend towards open access policies that on the one hand make the respective knowledge available freely and on the other hand allows the further use of these ontologies and the provided data is noticable. Newer examples for such projects that either provide these ontologies or use the free access for other use cases is the *Linking Open Drug Data* (LODD) [3] or BioPortal [4] project. LODD is a project that links the knowledge of several vocabularies by recognizing common entities. The additional gained knowledge in form of so called "linked data" is again provided freely under an open access policy. More details about this approach are given in section 2.3. The second project – BioPortal – offers a platform where people or projects serve their biomedical ontologies and knowledge bases. These knowledge bases can in turn be mapped so that the same entities in different

vocabularies are identified. Although it is possible to build third-party applications on top of this data, such applications are very rare at this time. In the case of LODD only small projects exist like DiseaseCard [5] or Pharmer [6]. The former one is graph-based browser that visualizes associated entities of other knowledge bases regarding a certain disease. The latter one – Pharmer – is a proof of concept for the authoring of semantic prescriptions.

Besides these third-party projects mainly clinical applications implement specific drug related functionalities. For example a *Patient Information System* stores all the details about the prescribed drugs of a certain patient. Therefore a nurse could be supported in the process of drug application by offering information about the route of application or the maximum dosage. Another application, like a *Computerized Physician Order Entry System* that selects and prescribes drugs, could check for interactions of the drugs that are going to be prescribed. And for both of the applications the side effects of a certain drug may be of interest. These examples show that many different medical applications implement drug related functionalities. And many of them also share the same requirements regarding the used data sources because there is a set of common questions, like *"What are the side effects of drug X?"*. But ignoring these shared requirements, many of the applications provide their own – mostly proprietary – knowledge bases.

A comparable situation on the data level of another domain was recently solved. The Wikidata project "centralizes access to and management of structured data" [1]. It solves the problem that many different Wikipedia provided common data about the same entities, e.g. the population of countries. Now Wikidata centrally provide this data through a common interface and the different Wikipedia refer to this source. In consequence there exist only one place where the data has to be updated and maintained.

[1] `https://www.wikidata.org/wiki/Wikidata:Main_Page` (last access Sept 18, 2013)

1.2 Problem

In section 1.1 the diversity of applications that implement drug related functionalities was mentioned. This goes along with the fact that multiple knowledge bases are used which actually should provide the same data. This leads to several problems.

The first problem – insufficient integration – follows from the different scopes and the different expressivity of the knowledge bases. This means there are many knowledge bases for special domains like side effects or alternative medicine. But if one knowledge base wants to use the data of another special domain this is usually not possible, for instance because of an insufficient semantic integration. This means that term x in knowledge base X does not implicitely refer to the same object as term x in knowledge base Y. Therefore the knowledge bases have to provide the data on their own.

This is the starting point of the next problem: redundancy. Many drug related knowledge bases include a set of common facts that are essential for this domain, like the code of the *Anatomical Therapeutic Chemical Classification System* (ATC). So this is a common *"Don't repeat yourself"* problem. A side effect of this problem is the higher chance that a given statement of a knowledge base is wrong or not up-to-date anymore.

So the third problem is the correctness of the knowledge. Proprietary knowledge bases generally have to deal with a strong limited number of people validating the data. In contrast open projects like Wikipedia are considered as "self-healing" information systems because of the high number of volunteers that report and correct data errors. To transfer this phenomenon to medical data it is unimaginable that everyone could edit such important information, but reporting errors would be a strong impact nonetheless.

Section 1.1 presented two third-party applications using open data as their knowledge sources. A common problem using these data sources is the extensive schema knowledge that is required to use the data efficiently. Besides the information about the usage of the given interface the developers are often forced to understand the structure of the data to perform their respective queries. This can be a deal-breaker

for developers and therefore for innovative applications.

1.3 Motivation

If the given problems of section 1.2 could be solved this would lead to a better data quality in the first place. Data quality contains in this context the expressivity, completeness and correctness of the knowledge. Medical applications would benefit from this improvement by getting a proper knowledge foundation. Following from that this would lead to unified, more well-grounded decisions for drug related questions. By supporting physicians, nurses, pharmacists and even the patient itself this would hopefully avoid some of the prescription errors that are made during the healing process of a patient.

Solving the problem that developers often have to know the schema of a certain knowledge base the consequence would be a reduced starting barrier for application developers. Instead of knowing about the whole schema of a knowledge base, only the knowledge about using the given interface would be required. Probably this would lead to many interesting applications on top of the given data. And by a growing number of applications using these data sources more and more feedback would be flow back to improve the knowledge bases in turn.

1.4 Objectives

Emerged from the motivation, this thesis shall proof that Linked Open Data is an appropriate knowledge source for drug management tools. For this purpose a web-based *Application Programming Interface* (API) will be provided. This API will support the process of answering drug related questions according to the WHO drug management life cycle. This is achived by offering several API endpoints – e.g. for drug-drug interactions – that gather information from several semantic knowledge bases and return the merged results. One design goal is the simplicity of the API to reduce the starting barrier of developing new applications based on the given interface. More details about the implementation process are described in chapter 3.

This thesis shall also provide two use cases where the usage of the API will be demonstrated and evaluated. The first use case is the integration in Dispedia – an information system in the complex field of rare diseases [7]. The main purpose here is the integration of the drug-drug interaction endpoint. The second use case is a personal drug management portal built around this API to support privat persons managing their prescribed drugs. This use case shall demonstrate the other available endpoints of the provided interface. Chapter 4 will offer more information about this evaluation process.

2 Preliminaries

2.1 Drug Management

To get a better understanding what the term Drug Management stands for it is essential to clearify the term Management at first. In general management comprises the non-executing tasks of a certain domain to accomplish certain goals. These non-executing tasks are divided in the four stages: *planning*, *organizing*, *leading* and *controlling* [8]. The term Drug Management was adopted by the WHO in their publication about managing drugs at health centres [2]. The term management is defined in their publication as follows:

> Management is the act or art of being responsible or in charge and conducting or supervising something (e.g. a health centre pharmacy, business, public undertaking) with a degree of skill and address. It is the judicious use of means to accomplish an end (i.e. public health).

With this definition in mind the question can be answered why it is useful to manage drugs. Refering to the WHO there are three main reasons:

1. "Firstly, drugs are part of the link between the patient and health services."
   This is the most trivial reason that addresses the direct influence of drugs to the patients health state. If drugs are not managed properly the success of the treatment process is imperiled.
2. "Secondly, poor drug management [...] is a critical issue, but major improvements are possible that can save money and improve access."
   Besides the medical benefits, a proper drug management can also result in financial advantages. These can be achived by an improved selection and procurement of the required drugs.

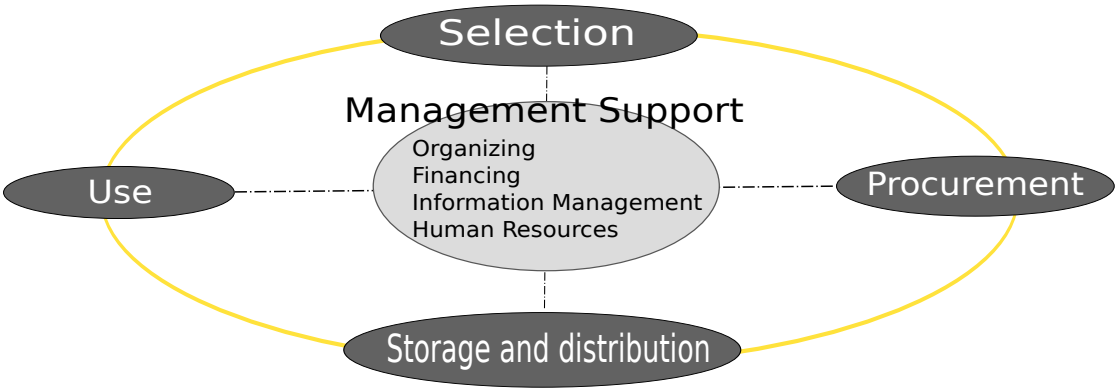

Figure 2.1: WHO drug management life cycle

3. "Finally, drugs are no longer the responsibility of health workers only."
   This fact is based on a progress in the last couple of years that led to a broader self-determination of patients. Therefore the claim to be involved in the drug selection process and to have an overview of the currently prescribed drugs requires an adequate drug management of all involved players.

The WHO proposes a drug management life cycle which is the foundation to implement a proper drug management at a certain institution. This life cycle which is shown in figure 2.1 contains four phases which are described shortly in the following listing:

- Selection
  Selection comprises the process of choosing the right drug regarding the patient's symptoms and other circumstances like already prescribed drugs or drug allergies. If all possible circumstances are considered properly, this phase can be the most comprehensive of the cycle.
- Procurement
  After the right drugs where chosen in the selection phase the next step is to procure these drugs. In this phase drug management can help to chose the best, e.g. cheapest vendors. Otherwise it may be the case that more expensive vendors are chosen just because of the large sector of drug companies.
- Use
  Many adverse drug events occur because of incorrect delivered drugs. Examples

are too high dosages or incorrect routes of administration. The goal of the third phase therefore is to provide the right information about the usage of a drug at the right place to the right time.

- Storage and distribution

  The fourth phase manages how drugs are stored. This is important to prevent disfigurations of labels and to "maintain integrity of packaging and so guarantee quality and potency of drugs during shelf life". Further the patient needs help in cases of losing the drug packaging or similar cases.

According to the WHO these phases "are interlinked and are reinforced by appropriate management support systems (i.e. tools)". The API which is developed by this thesis is settled up in this category of support systems.

One point has to be remarked regarding the scope of the WHO recommendations. Althought the cited publication addresses health centres, recommendations like the life cycle are easily adoptable to other areas, e.g. private healthcare.

2.2 Semantic Web

The ongoing growth of digital data in the past years has revealed many shortcomings that occur with these amounts of data. Fields of research like *Big Data* also show the importance of managing these amounts of data efficiently. One key issue since years is the distinction between syntactically and semantically processable data. Tim Berners-Lee proposed in 2001 a concept to describe specific data statements in a better way [9]. This model behind this approach was called *Ressource Description Framework* (RDF) [10] and is one important part of the Semantic Web Stack. This stack combines several technologies and standars that are necessary to transfer the syntactic web to a semantic web. The Semantic Web Stack is illustrated in figure 2.2. The advantages of a semantic web are numerous. The main advantage is that the information are not only machine readable but machine processable. This entails other improvements like reasoning that transfers implicit knowledge into explicit.

The following paragraphs will describe the most important technologies and standards of the Semantic Web Stack that are necessary for this thesis:

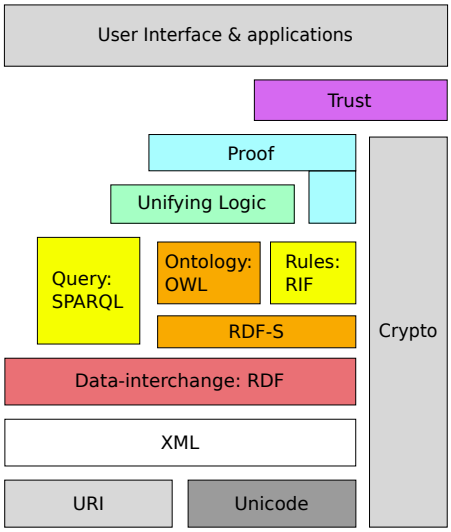

Figure 2.2: Semantic Web Stack

At the lowest level of the stack there is the concept of the Uniform Resource Identifier (URI). A URI identifies a specific resource, e.g. a car. For several other domains there already exists such concepts as URIs. For example the Uniform Resource Locator (URL) identifies web sites or an International Standard Book Number (ISBN) identifies books. The URI itself is just a set of characters that can be additionally divided into five segments: scheme, authority, path, query and fragment. An example of such a URI would be: `http://example.com/Alice`. This URI would describe the resource "Alice".

Based on the identification of resources, the already mentioned Resource Description Framework has the goal to describe these resources. Therefore RDF is one of the most important concepts of the Semantic Web Stack. The approach of RDF to model knowledge is to express statements as triples. This is tight to how natural language is mostly constructed. Obviously that is why the components of the triple use the well known names of *Subject*, *Predicate* and *Object*. Figure 2.3 shows an example triple. The subject of such a triple has to be always an URI, same belongs to the predicate. An exception is the object which can be additionally a literal. A literal is a direct information typed in a specific way, e.g. as date or simply text, which refers to no other resource. For more advanced models it is possible to describe the subject and object

Figure 2.3: Example of a RDF triple

in a anonymous way, without a concrete entity. These anonymous resources are called blank nodes, but will be without broader relevance for this thesis. At the end one important distinction has to be made about RDF. Often RDF is getting confused with RDF/XML [11] – a notation of RDF statements. While RDF itself only describes how to model information, several RDF notations exist that express these information in several ways. Examples for such notations besides the already mentioned RDF/XML is N-Triples, Turtle [12] or JSON-LD [13].

On top of RDF the Semantic Web Stack levels SPARQL, an RDF query language. SPARQL enables comprehensive queries over a knowledge base built on RDF. SPARQL itself uses many well known keywords from SQL, but because of the graph structure of RDF and therefore of the triple stores the languages naturally differ.
Given the following two triples in Turtle notation:

```
@prefix ex: <http://example.com/>.
ex:anna ex:name ‘‘Anna’’.
ex:bob  ex:name ‘‘Bob’’.
```

Listing 2.1: Example Turtle triples

Then the following simple SPARQL query would return the result “Bob”.

```
SELECT ?name
WHERE {
  ex:bob ex:name ?name.
}
```

Listing 2.2: Example SPARQL query

With SPARQL 1.1 so called *Federated Queries* were introduced. These Federated Queries are Statements that are queried against multiple (federated) SPARQL endpoints. This is a key feature for an easy integration of several RDF graphs.

2.3 Linked Open Data

As described in the previous section RDF statements spans a graph of triples. Thereby, each knowledge base spans their own graph. Examples for large knowledge bases are DBpedia[2], LinkedGeoData[3] or Drugbank. *Linked Data* in general now describes the situation that entities of one knowledge base link to entities of another knowledge base. For example an entity of DBpedia about a drug could link to the same entity in Drugbank for further information. In this case both knowledge bases would describe the same entity. In other cases Linked Data is used to refer to different entities. In the example above this could mean that the drug entity of Drugbank is refering to an entity of a chemical knowledge base to describe the compund of the drug in more detail. Listing 2.3 shows an example given in Turtle notation where an RDF triple links from one knowledge base to another. This triple states that the DBpedia entity "Metoprolol" describes the same subject as the Drugbank entity "Metoprolol".

```
@prefix drugbank: <http://drugbank.bio2rdf.org/>
@prefix dbpedia: <http://dbpedia.org/resource/>
@prefix owl: <http://www.w3.org/2002/07/owl#>

dbpedia:Metoprolol owl:sameAs drugbank:Metoprolol.
```

Listing 2.3: Example triple for linking two entities of different knowledge bases

If the mentioned knowledge bases are freely available and usable then this is called *Linked Open Data* (LOD).

Tim Berners-Lee proposed some rules[4] that state whether a data source is Linked Open Data, or not. These rules are:

1. Use URIs to denote things.
2. Use HTTP URIs so that these things can be referred to and looked up ("dereferenced") by people and user agents.
3. Provide useful information about the thing when its URI is dereferenced, lever-

[2] `http://dbpedia.org` (last access Sept 18, 2013)
[3] `http://linkedgeodata.org` (last access Sept 18, 2013)
[4] `http://www.w3.org/DesignIssues/LinkedData.html` (last access Sept 18, 2013)

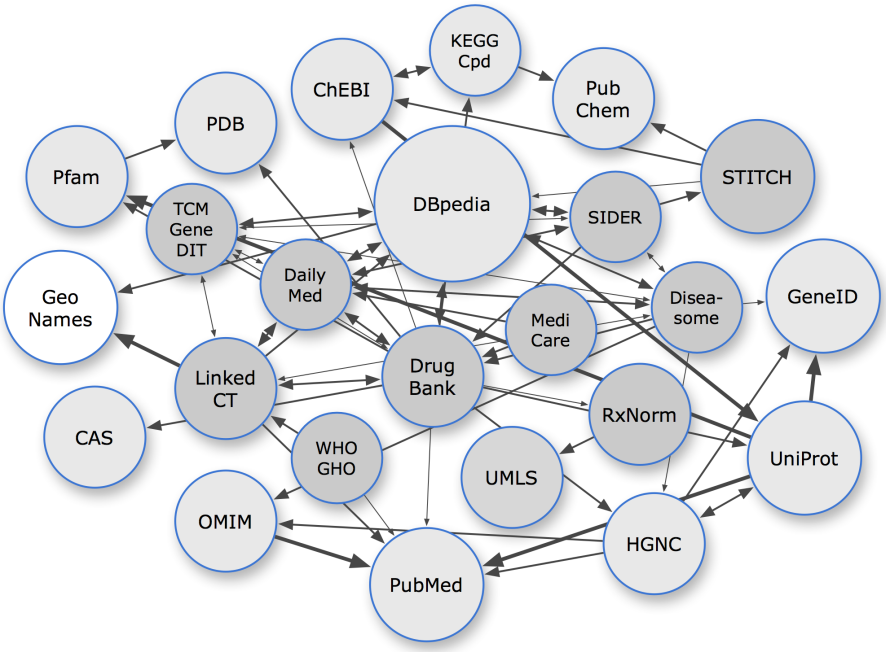

Figure 2.4: Biomedical data sets of the Linked Open Data cloud

aging standards such as RDF, SPARQL.

4. Include links to other related things (using their URIs) when publishing data on the Web.

Further, Berners-Lee categorizes Linked Open Data on the basis of a five star rating. Therefore the more a knowledge base follows these principals the more stars it earns.

- **1 star** - Available on the web (whatever format) but with an open licence, to be Open Data
- **2 stars** - Available as machine-readable structured data (e.g. excel instead of image scan of a table)
- **3 stars** - as (2) plus non-proprietary format (e.g. CSV instead of excel)
- **4 stars** - All the above plus, Use open standards from W3C (RDF and SPARQL) to identify things, so that people can point at your stuff
- **5 stars** - All the above, plus: Link your data to other people's data to provide context

In conjunction with SPARQL this leads to large federated collections of knowledge that are comprehensively queryable. Figure 2.4 gives an overview over the biomedical domain of the Linked Open Data cloud in 2010. But this is only a small section of the complete Linked Open Data cloud and produces such graphs for the complete cloud is going to be unreadable in the last years.

3 Methods

This chapter describes the background of the developed interface for integrating several drug related knowledge bases. The project is called Semantic Drug Interface (SEDRI) and will be explained by the following sections in detail.

3.1 Data integration

Before the architecture and used data sets of SEDRI will be explained, this section will discuss the different possibilities of integrating semantic data sets.

At first there is a distinction between virtual and local (also: materialized) integration [14]. Virtual integration means that several federated data sets are queried and the results will be integrated virtually at query time independently of the source data sets. In contrast, the local integration deploys an own copy of the data sets locally and integrates them based on certain criteria. The main criteria that decides which approach is more suitable are query performance versus currentness of the data.

Virtual integration provides always up-to-date data because the knowledge bases are queried directly. In the majority of cases this also leads to slower query times compared to local integration. This is naturally caused by network times and the delay of the integration process. Additionally it is not uncommon that knowledge bases are not available for a given time.

The local integration approach only queries a local target and therefore it is more performant in most of the cases. The downside of this approach is that the data has to be updated regularly or the integration component will provide outdated data very quickly.

Example engines for the virtual integration of semantic knowledge bases are SPARQL 1.1

federated queries or tools like FedX [15] that support also SPARQL < 1.1. With SPARQL 1.1 it is possible to query multiple tagets with one statement by using the `SERVICE` keyword. Listing 3.1 gives an example of a federated query over two knowledge bases.

```
PREFIX foaf: <http://xmlns.com/foaf/0.1/>

SELECT ?name
FROM <http://example.org/myfoaf.rdf>
WHERE
{
  <http://example.org/myfoaf/I> foaf:knows ?person .
  SERVICE <http://people.example.org/sparql> {
    ?person foaf:name ?name . }
}
```

Listing 3.1: Example of SPARQL 1.1 federated query

These federated queries are an easy yet powerful way to perform queries over multiple knowledge bases without the use of external tools. The only problem is that the specific endpoints have to support SPARQL 1.1 which is currently not given in all cases.

An example of a local integration is the Linked Life Data project.[5] This project integrates multiple knowledge bases like Drugbank or Diseasome. A disadvantage of the project is the license model[6] which states that free access is limited to only one query per 30 seconds and only annual updated data sets. Otherwise the commercial enterprise license is required which then would allow umlimited query execution and monthly updated data sets.

Which approach the SEDRI project uses will be explained in the following section.

[5] Linked Life Data: `http://linkedlifedata.com` (last access Sep 04, 2013)
[6] Linked Life Data – license model: `http://linkedlifedata.com/about` (last access Sep 04, 2013)

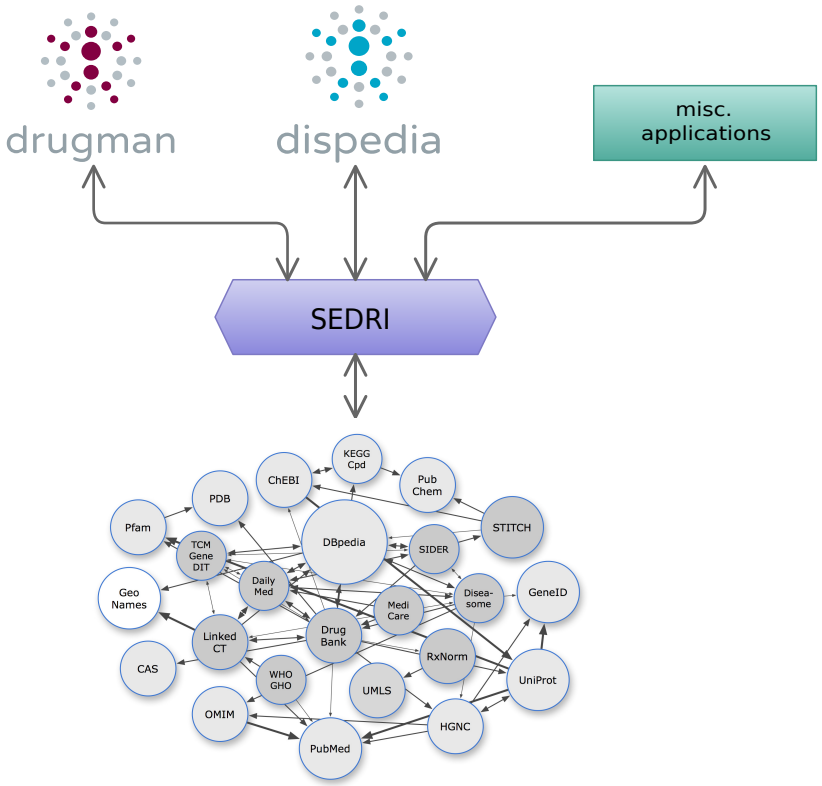

Figure 3.1: Overview of the SEDRI infrastructure

3.2 Architecture

As figure 3.1 shows, the Semantic Drug Interface acts like an abstraction and integration layer between the source data and the end user applications.

Section 3.1 described the different possible approaches for integrating semantic knowledge bases. SEDRI currently uses a virtual integration approach. The reasons for this decision is the simplicity, the low costs of the approach and the fact that possible applications expect always up-to-date knowledge. Especially in the biomedical domain this is an important requirement. SEDRI does not use the SPARQL 1.1 federated query possibility because many of the used data sets, or more specific the respectiv SPARQL endpoints, doesn't support this feature yet. Therefore the different SPARQL endpoints are queried and the results merged. The results are queried by a SPARQL `CONSTRUCT` or `DESCRIBE` statement. This means that always full RDF resources

are returned and not only single entities or literals that are returned by a `SELECT` query. These RDF resources can be returned by SEDRI in several representation formats, like *Turtle*, *RDF/XML* or *JSON-LD*. By default JSON-LD will be returned, because JSON in general is the dominating data exchange format of modern web API's. Other formats can be requested by setting the HTTP `Accept` Header to the appropriate MIME type.

To abstract from the required background knowledge about the data sources, SEDRI predefines some API endpoints that are easily usable by only providing the necessary drug codes. As such drug codes currently the Anatomical Therapeutic Chemical Classification System (ATC) code or the Drugbank ID are supported. These IDs are required to identify a drug unambigiously. The API endpoints are designed to cover each phase of the WHO drug management life cycle. The following paragraphs describe all available API endpoints that are provided by SEDRI and the respective SPARQL queries and show an example request for each endpoint.

Drug-drug interactions

This endpoint expects at least two drug codes and checks these drugs for possible drug-drug interactions. In detail each possible combination of the given drugs is calculated and tested for interactions. The maximum number of drug codes is unlimited. This use case is mainly placed in the *Selection* phase of the WHO drug management life cycle. Here it may help the respective parties, e.g. a physician or pharmacist to prevent that they prescribe drugs that possibly cause harm.
The following code listings show example queries to retrieve the interactions of Ibuprofen and Metoprolol as well as in the first example additionally Metoprolol and Bisoprolol.

```
PREFIX db:  <http://bio2rdf.org/drugbank:>
PREFIX dbv: <http://bio2rdf.org/drugbank_vocabulary:>

DESCRIBE ?ddi
WHERE {
  ?ddi a dbv:Drug-Drug-Interaction .
```

```
  {
    db:DB01050 dbv:ddi-interactor-in ?ddi .
    db:DB00264 dbv:ddi-interactor-in ?ddi .
  }
  UNION
  {
    db:DB00612 dbv:ddi-interactor-in ?ddi .
    db:DB00264 dbv:ddi-interactor-in ?ddi .
  }
}
```

Listing 3.2: Example Drugbank query for drug-drug interactions between Ibuprofen – Metoprolol and Bisoprolol – Metoprolol

```
PREFIX dikbD2R: <http://dbmi-icode-01.dbmi.pitt.edu:2020/vocab/resource/>
PREFIX dbv:     <http://bio2rdf.org/drugbank_vocabulary:>
PREFIX owl:     <http://www.w3.org/2002/07/owl#>
PREFIX rdfs:    <http://www.w3.org/2000/01/rdf-schema#>

CONSTRUCT {
  ?s a dbv:Drug-Drug-Interaction.
  ?s rdfs:label ?desc.
  ?drug1 dbv:ddi-interactor-in ?s.
  ?drug2 dbv:ddi-interactor-in ?s.
}
WHERE {
  ?s rdfs:label ?desc.
  {
    ?drug1 owl:sameAs db:DB01050 .
    ?drug2 owl:sameAs db:DB00264 .
    {
      ?s dikbD2R:ObjectDrugOfInteraction ?drug1.
      ?s dikbD2R:PrecipitantDrugOfInteraction ?drug2.
    } UNION {
      ?s dikbD2R:ObjectDrugOfInteraction ?drug2.
```

```
      ?s dikbD2R:PrecipitantDrugOfInteraction ?drug1.
    }
  }
}
```

Listing 3.3: Example DIKB query for drug-drug interactions between Ibuprofen and Metoprolol

Listing 3.4 shows an examplary HTTP request url for the drug-drug interaction endpoint.

```
GET http://sedri.de/ddi/?drug1=C07AB02&drug2=C07AB07&drug3=C01EB16
```

Listing 3.4: Example request for the drug-drug interaction endpoint

Side effects

The side effects endpoint expects one drug code and returns all available side effects of this drug. This component supports primarily the phase *Use* where it could support the treating parties for example if the respective patient shows signs of an adverse effect. But this component is also applicable for the *Selection*. In this case drugs with a specific side effect could be excluded.
The following code listings show two example queries that retrieve the side effects of Ibuprofen from SIDER and PhramGKB.

```
PREFIX owl: <http://www.w3.org/2002/07/owl#>
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX sider: <http://wifo5-04.informatik.uni-mannheim.de/sider/resource/sider/>
PREFIX db: <http://wifo5-04.informatik.uni-mannheim.de/drugbank/resource/drugs/>

CONSTRUCT {
  ?se rdfs:label ?label.
  ?se a ?type.
}
WHERE {
```

```
  ?drug owl:sameAs db:DB01050 .
  ?drug sider:sideEffect ?se.
  ?se rdfs:label ?label.
  ?se a ?type.
}
```

Listing 3.5: Example SIDER query for side effects of Ibuprofen

```
PREFIX pharmgkbv:  <http://bio2rdf.org/pharmgkb_vocabulary:>

CONSTRUCT {
  ?side rdfs:label ?label.
  ?side pharmgkbv:p-value ?p.
  ?side a pharmgkbv:Side-Effect.
  ?side pharmgkbv:event ?event.
}
WHERE {
  ?drug pharmgkbv:xref <http://bio2rdf.org/drugbank:DB01050>.
  ?drug pharmgkbv:xref ?chem.
  ?side pharmgkbv:chemical ?chem.
  ?side a pharmgkbv:Side-Effect.
  ?side pharmgkbv:in-sider "false".
  ?side pharmgkbv:event ?event.
  ?side pharmgkbv:p-value ?p.
  ?event rdfs:label ?label.
}
```

Listing 3.6: Example PharmGKB query for side effects of Ibuprofen that are not listed in SIDER

Listing 3.7 shows an examplary HTTP request url for the side effects endpoint.

```
  GET http://sedri.de/sideeffects/?drug=C07AB02
```

Listing 3.7: Example request for the side effects endpoint

Procurement

The procurement component returns the possible manufacturers and packagers for a given drug code. Trivialy this component supports the *Procurement* phase of the drug management life cycle.

```
PREFIX dbv: <http://bio2rdf.org/drugbank_vocabulary:>
PREFIX db: <http://bio2rdf.org/drugbank:>

CONSTRUCT {
  db:DB01050 dbv:packager ?packager.
  ?packager ?p ?o.
  db:DB01050 dbv:manufacturer ?manufacturer.
}
WHERE {
  db:DB01050 dbv:packager ?packager.
  ?packager ?p ?o.
  db:DB01050 dbv:manufacturer ?manufacturer.
}
```

Listing 3.8: Example Drugbank query for procurement meta data of Ibuprofen

Listing 3.9 shows an examplary HTTP request url for the procurement endpoint.

```
  GET http://sedri.de/procure/?drug=C07AB02
```

Listing 3.9: Example request for the procurement endpoint

Drug proposals

This endpoint proposes several drugs to a given disease. To specify the disease currently only the Medical Subject Headings (MeSH) ID is supported. As results the respective URIs to the drugs are returned. With these URIs more details can be retrieved or other endpoints of SEDRI may be used. This component covers the *Selection* phase where it could support the physician by chosing an appropriate drug to a given

disease. Important is the fact that these are only recommendations to relieve the physiscian – the final decisision has to be made by the expert.

In example 3.12 `diseasome:561` is the appropriate Diseasome URI of "Hypertension" which was converted from the MeSH ID (D006973) by DBpedia in a previous request.

```
PREFIX diseasome <http://www4.wiwiss.fu-berlin.de/diseasome/resource/diseases/>

CONSTRUCT {
  diseasome:561 diseasome-instance:possibleDrug ?o.
}
WHERE {
  diseasome:561 diseasome-instance:possibleDrug ?o.
}
```

Listing 3.10: Example Diseasome query for possible drugs of Hypertension

Listing 3.11 shows an examplary HTTP request url for the drug proposal endpoint.

```
  GET http://sedri.de/proposal/?code=D006973
```

Listing 3.11: Example request for the drug proposal endpoint

Pharmacokinetic meta data

This component returns several meta data for a given drug code. The results contains mainly pharmacokinetic information like dosages, the affected organism or the route of elimination. In the WHO drug management life cycle this component covers mainly the *Use* phase but also in parts the *Selection* phase.

```
PREFIX db: <http://bio2rdf.org/drugbank:>
PREFIX dbv: <http://bio2rdf.org/drugbank_vocabulary:>
PREFIX dc: <http://purl.org/dc/terms/>

CONSTRUCT {
  db:DB01050 dbv:absorption ?absorption.
```

```
  db:DB01050 dbv:affected-organism ?organism.
  db:DB01050 dbv:dosage ?dosage.
  ?dosage rdfs:label ?dosage_label.
  db:DB01050 dbv:food-interaction ?food.
  db:DB01050 dbv:indication ?indication.
  db:DB01050 dbv:mechanism-of-action ?mechanism.
  db:DB01050 dbv:route-of-elimination ?elimination.
  db:DB01050 dc:description ?description.
}
WHERE {
  db:DB01050 dbv:absorption ?absorption.
  db:DB01050 dbv:affected-organism ?organism.
  db:DB01050 dbv:dosage ?dosage.
  ?dosage rdfs:label ?dosage_label.
  db:DB01050 dbv:food-interaction ?food.
  db:DB01050 dbv:indication ?indication.
  db:DB01050 dbv:mechanism-of-action ?mechanism.
  db:DB01050 dbv:route-of-elimination ?elimination.
  db:DB01050 dc:description ?description.
}
```

Listing 3.12: Example Diseasome query for possible drugs of Hypertension

Listing 3.13 shows an examplary HTTP request url for the meta data endpoint.

```
  GET http://sedri.de/details/?drug=C07AB02
```

Listing 3.13: Example request for the meta data endpoint

3.3 Data sets

The following Table gives an overview of the data sets that are used by SEDRI for the different API endpoints. The table also contains a short description of the data set and how it is used by SEDRI.

Data set	Description	Usage
DBpedia	DBpedia retrieves meta data form Wikipedia pages that are often shown in info boxes. DBpedia therefore provides many cross domain information and takes a central role in the Linked Open Data cloud. In 2011 DBpedia contained descriptions about more than 3.64 million things including 5,400 diseases.	The data set is used for converting the given ATC codes of drugs to Drugbank IDs for a better processing. The second use case is the proposal endpoint where DBpedia is used for the assignment of the Diseasome URI to a given MeSH code.
Drugbank	Drugbank is a large knowledge base that contains many different information about drugs and their meta data such as interactions, dosages or manufacturers. It has a comparable central role like DBpedia but in the biomedical part of the Linked Open Data cloud.	Drugbank is the main knowledge base used by SEDRI. It provides information for nearly each given endpoint, except the side effects.

SIDER	The main domain of SIDER are adverse drug reactions. "The information is extracted from public documents and package inserts"[7] and includes "side effect frequency, drug and side effect classifications as well as links to further information, for example drug–target relations".	SIDER is only used by the side effects endpoint.
Drug Interaction Knowledge Base (DIKB)	The DIKB is an evidence-focused knowledge base that provides information about drug drug interactions. "The DIKB contains quantitative and qualitative assertions about drug mechanisms and pharmacokinetic drug-drug interactions"[8]	DIKB is used as additional source for the interactions endpoint.
Pharmacogenomics Knowledge Base (PharmGKB)	"PharmGKB curates primary genotype and phenotype data, annotates gene variants and gene-drug-disease relationships via literature review, and summarizes important PGx genes and drug pathways."[16]	PharmGKB is used as additional source for the side effects endpoint.

[7] `http://sideeffects.embl.de/` (last access Sept 19, 2013)

[8] `http://dbmi-icode-01.dbmi.pitt.edu/dikb-evidence/front-page.html` (last access Sept 09, 2013)

Diseasome	"[...] Diseasome publishes a network of 4,300 disorders and disease genes linked by known disorder-gene associations for exploring all known phenotype and disease gene associations, indicating the common genetic origin of many diseases. The list of disorders, disease genes, and associations between them was obtained from the Online Mendelian Inheritance in Man (OMIM) [...]"[9]	Diseasome is used for proposing drugs to a given disease implemented in the proposals endpoint.

Table 3.1: Description and usage of the relevant SEDRI data sets

[9] `http://diseasome.eu/about.html` (last access Sept 09, 2013)

4 Evaluation

The evaluation of SEDRI has the goal to test how well it can be integrated in different projects with different use cases, based on different programming languages and so on. This thesis will evaluate SEDRI by integrating it in two projects: Dispedia and DrugMan.

4.1 Dispedia

4.1.1 Introduction

"The Dispedia Framework is an information system in the complex field of rare diseases. The goal of the system is to harmonize social care conditions and health care conditions with the focus on personalization and patient autonomy." [7]
Dispedia is based on several ontologies and uses Linked Open Data e.g. for additional information about diseases or drugs. The following paragraph will describe some of the core concepts of Dispedia which are essential for the integration of SEDRI.

Maybe the most important core concept are *Proposals*. A proposal is a possibility of a physician to offer several therapeutical activities to a patient. In the Dispedia model a proposal represents a frame for multiple proposal components. A proposal component in turn contains the mentioned therapeutical activities like prescribing drugs, medical devices or other services. Proposal components act like bricks which can be assigned to different proposals. This is a flexible way to create treatment offers. Proposals are assigned to patients. The relationship proposal – patient is $n : m$ which means that it is possible to assign a proposal to multiple patients and that a patient can hold multiple proposals.
Besides these proposals a patient can receive several products or services, like drug

products or a physiotherapy. This is relevant for the integration of SEDRI because in that way there are two ways how a patient can take drugs. First by accepting a proposal and second by "receiving" a drug product on their own.
The third important concept is the distinction between a drug and a drug product. As described in the last sentences a patient can receive a drug product. A drug product is a concrete commercial product distributed by a pharma company. These products include an active ingredient which is the "drug" in the Dispedia sense. An example of the distinction may be the drug product "Ibuflam" and the drug "Ibuprofen".

4.1.2 Integration of SEDRI

The integration point of SEDRI in Dispedia is the possibility to propose a specific drug to a patient. In that case SEDRI is used to check the proposed drugs for possible drug-drug interactions. In detail there are three integration points which are described in the following listing in more detail:

- *Check A*: The first integration is when a proposal component is created. Each proposal component can contain multiple drugs. If such a proposal component is created or modified then the containing drugs are tested for possible interactions. In this case Dispedia has nothing to do than calling SEDRI with the appropriate drug codes that are stored in the system and to process the results.
- *Check B*: The second integration is implemented in the logic of proposals. As the introduction about Dispedia described each proposal can contain multiple proposal components which thereself contains drugs. In this scenario it is possible that a physician proposes a specific proposal including multiple components where each component itself contains no interactions but the components among each other contains interactions. To prevent such a scenario the components of a proposal are tested for interactions on creation and modification of the proposal.
- *Check C*: The third integration of the drug-drug interaction endpoint is implemented in the patient proposal allocation. If a physician allocates several proposals to a patient, these proposals will be checked against the current medication of the patient. This prevents patients to get interacting drugs in their medication

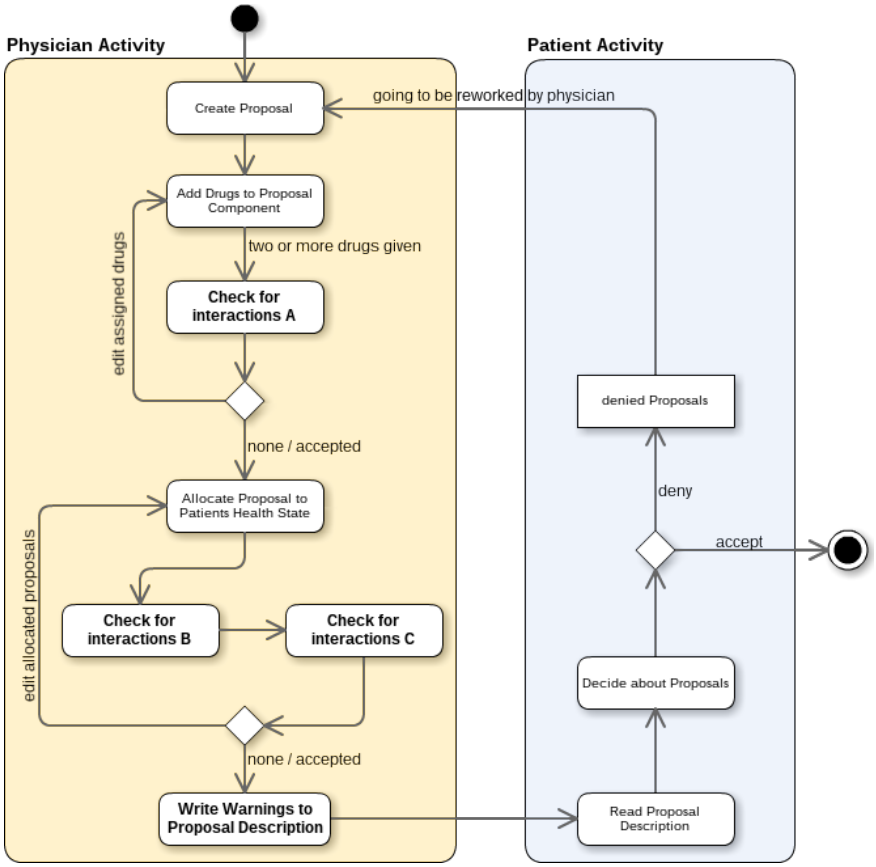

Figure 4.1: Activity diagram of Dispedia proposal workflow

plan. In this third case one difference to the other two implementations has to be considered. Because the items of the medication plan of the patient are drug products in the meaning of Dispedia the active ingredient of this drug product has to be received. In the other two cases the drugs could be directly compared because the proposals and proposal components contain no drug products but drugs.

If in one of those cases an interaction is detected the details of the interaction are saved in relationship to the corresponding instance of the proposal or proposal component. This allows the system to provide these information in other use cases. For example when a patient decides to accept or decline a proposal the possible interaction information of the third check can be provided. In this case an additional call of SEDRI is not necessary.

Figure 4.1 gives a graphical overview of the typical proposal workflow in Dispedia and where SEDRI is used. TODO: grafik

The workflow is divided in the physician's activity and the patient's activity. The beginning of the workflow is the creation of a new proposal by a physician. If there are already existing proposal components then several of these components can be added to the new proposal. Otherwise or additionally it is possible to create new proposal components. These proposal components may contain drugs or other therapeutical activities as described in the previous introduction section. If a new proposal component was created the included drugs will be tested for drug-drug interactions by SEDRI (*Check A*). After all desired proposal components are added to the proposal and the proposal is saved, the second call of SEDRI (*Check B*) is executed. Now the new proposal can be allocated to several patients. In this step SEDRI executes the third check (*Check C*). With the completed proposal allocation the activity of the physician is finished. The patient's activity starts with taking note about the given drug-drug interactions in their allocated proposals. Based on these information the patient can accept or decline a given proposal. In case of accepting the proposal the workflow ends. Otherwise the physician is notified that a proposal has been declined and the physician is now able to rework the proposal. This means the workflow starts the next iteration.

Figure 4.2 gives an overview of the information flow between Dispedia, SEDRI and the Linked Open Data cloud. It illustrates how Dispedia is calling SEDRI for two use cases, interactions and side effects. The latter one was not in the scope of this evaluation and is not implemented yet. The calls to SEDRI are synchronous which means Dispedia has to wait until the results are delivered by SEDRI to work on further. SEDRI itself calls different Linked Data sources asynchronously. After the results are delivered by SEDRI, the diagram shows that Dispedia continues processing these results. In this case Dispedia calls the DBpedia Spotlight annotation service[10] which extracts several entities from a given text and links them to their DBpedia resources. This can improve the given warning messages by much more information.

[10] `http://dbpedia-spotlight.github.io/demo/` (last access Sept 09, 2013)

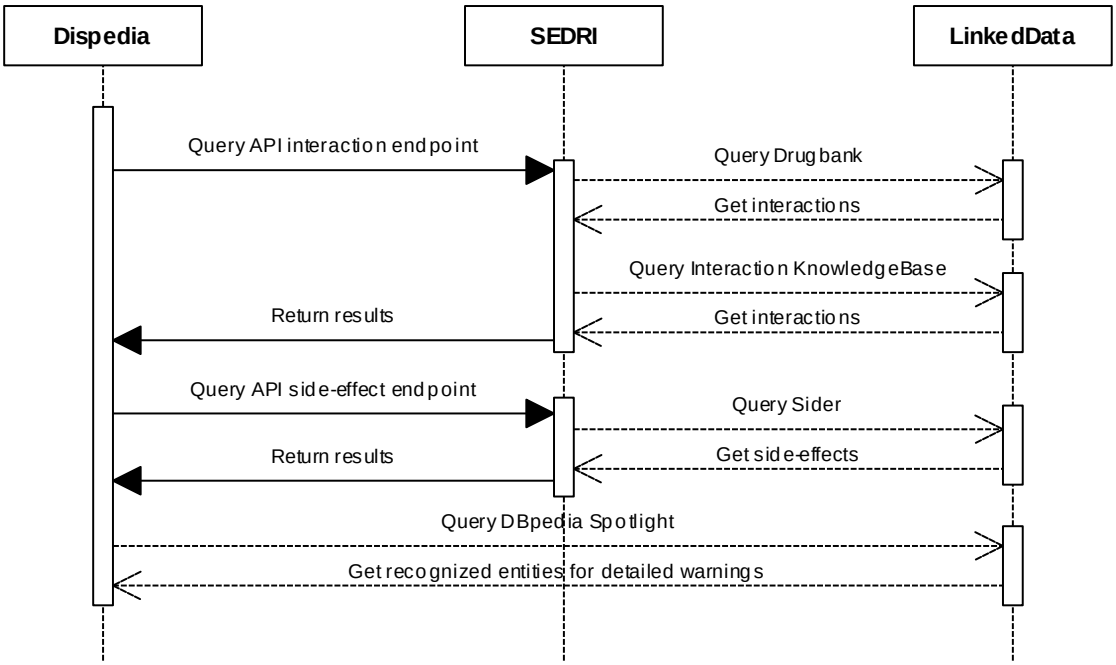

Figure 4.2: Sequence diagram of Dispedia and SEDRI information flow

4.2 DrugMan

4.2.1 Motivation

Spoken for Germany there is a static rise in the last years of multiple morbidities of patients. These often imply automatically that the regarding patient takes also multiple drugs. In the case of five or more medications this is called *Polypharmacy* [17]. Besides the fact of the multiple medications this term stands also for the unnecessary prescription of drugs which may lead to several issues.

A general issue of polypharmacy is that it is hard for patients as well as physicians or pharmacists to keep track of all the currently prescribed medications. Additionally the patient can buy several prescription free drugs that also could lead to issues with the other prescribed drugs. If this overview of all the medication can not be preserved then different physicians may prescribe independently their drugs what often leads to overlapping prescriptions or interacting drugs in the medication of the patient.

Another problem that DrugMan is facing is the low self-determination of people

during pharmacotherapy. This self-determination covers the confirmability and information procurement in the process of pharmacotherapy. One example where these two values are lacking are drug-drug interactions. If a patient is taking five drugs, which is the lower limit of polypharmacy, then there are ten possible drug-drug interactions that a patient has to test if he wants to know whether his medication includes interactions or not. These information about possible interactions are provided by the instruction leaflet. But these information – in the case of Germany – doesn't name any specific drug names but only drug classes. So without any expert knowledge it is nearly impossible for a patient to get any concrete information about possible drug-drug interactions. And in the case that the information would be available it is cumbersome for a private person to test all the possible combinations of drugs.

4.2.2 Features

Derived from the two problems described in the previous section, the goal of DrugMan – a personal drug management portal – is to provide a modern approach for supporting the patient during the pharmacotherapy. This includes the possibility to capture the full medication of the patient and several actions that support the self-determination of the patient, e.g. testing the medication for drug-drug interactions. DrugMan was developed as a side project of this thesis. It therefore has a strong connection to SEDRI.
The following paragraphs will describe the core features of DrugMan.

Medication plan

The central part of DrugMan is the medication plan. Each registered user can add several drugs to their medication plan. This allows the user always to have the overview of the current taken drugs. A drug item of the medication plan consists of a label and an active ingredient.

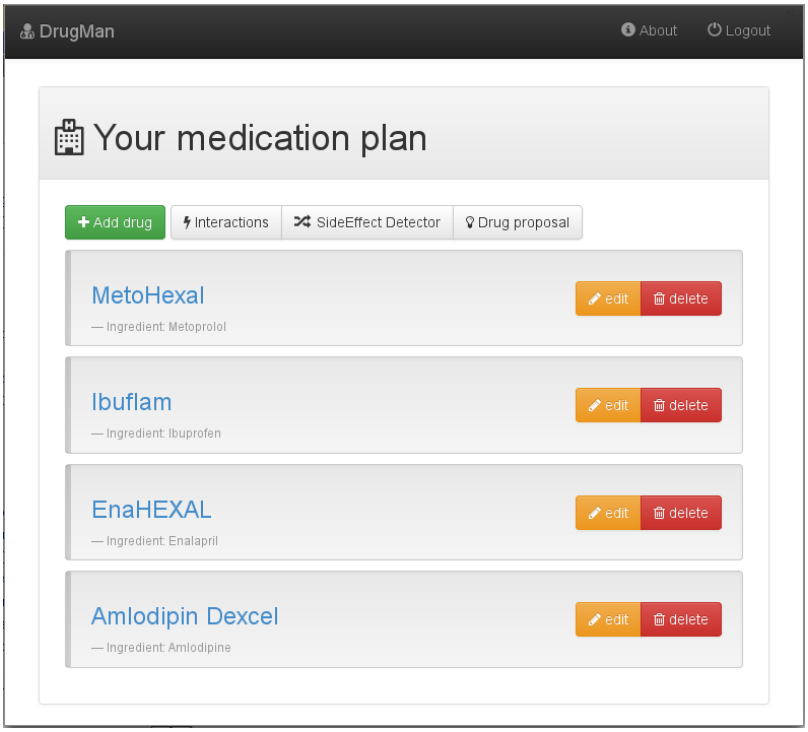

Figure 4.3: Screenshot of DrugMan

Drug details

The details page about a drug gathers different information about the drug and their active ingredient. One example for such details are pharmacokinetic information about the active ingredient. Among others these information contains the absorption, route of elimination and the affected organism. Other examples for details are side effects or available packagers and manufacturers.

Interactions

The first of several actions that can be executed on top of the medication plan is a drug-drug interaction check. This action will test all the drugs that are included in the medication plan for possible drug-drug interactions. These checks are based on the active ingredient of each drug. In case of existing interactions the affected drug-drug combination and some details about the interaction will be communicated by a notification.

Side effect detector

Another feature is the "Side effect detector". By specifying several side effects, the side effect detector will return all the drugs of the medication plan that may cause these side effects. The process of specifying the side effects is supported by an autocomplete function that gathers side effects from the SIDER knowledge base. The matching is done by comparing the given side effect URI's of the user with all the URI's of possible side effects of the drugs inside the medication plan.

Drug proposals

The drug proposal feature supports the user by informing about possible drugs to a given disease. After specifying a disease the user will get a selection of possible drugs. Based on this selection the user can read some details about the drug including side effects, add this drug to the medication list or check whether the drug suits to the other drugs of the medication list in terms of possible drug-drug interactions.

4.2.3 Integration of SEDRI

Since DrugMan was developed as a side project of this thesis the project is tightly coupled with SEDRI. The internal architecture of DrugMan can be grouped in the management component and the drug specific component.

The former one is responsible for the general management features like user management including registering new users, login, logout etc. and the medication plan features including the registration, modification and deletion of drug items.

The latter component is responsible for the drug-specific features and is completely based on SEDRI. This means all the funtionality described in Section 4.2.2 except the medication plan is built on top of SEDRI.

An example that SEDRI is usable for frontend information as well as backend processing is the Side effect detector. This feature uses the side effects endpoint of SEDRI in the backend for the internal comparison of the given side effects and all the possible side effects of the current medication. On the other hand it uses the same endpoint in

the frontend by displaying all available side effects in the detail page of a drug.

If desired the application could request different return formats for the different use cases. But in the case of the Side effect detector JSON-LD is used for both processes and the evaluation showed that this suits well.

5 Discussion

Data level improvements

As chapter 3 showed, SEDRI queries several knowledge bases that mainly provide drug specific data. Althought much information is given by the semantics of RDF there are also some syntactic information inside the knowledge bases. For example the Drugbank knowledge base provides for each drug-drug interaction a label with the appropriate message what the interaction may cause. A big problem with such information is the fact that it is mostly only available in English. This limits the use of these labels in many systems. A workaround for this problem may be Natural Language Processing (NLP) like Named Entity Recognition which would detect important entities. These entities may exist in other languages in other knowledge bases like DBpedia. But in many cases this approach would lead to information loss.

This is a general problem that many knowledge of the Linked Open Data cloud only provides English data. In terms of language as well as the information itself. This has two main reasons: First, because the Linked Data domain is still mainly a research topic most of the researchers build English applications to reach a broader community. The second reason is the patent policy of many countries regarding the pharmacology. For example in Germany only a fraction of pharmaceutic information is available compared to the USA. To get all available information about a drug you have to pay a license fee. But this doesn't enable one to use these information freely. So a further use, e.g. the transformation to RDF is hard to implement. Therefore – especially on the national level – it still has to be done much work to get comparable knowledge bases as the US.

As always, there are many other improvements on the data level imaginable. In case

of the granularity of information in the knowledge bases an concrete example are the drug-drug interactions of Drugbank. Currently there is no severity of the interactions. This means an interaction that occurs in 1 of 1,000 cases has to be processed in the same way as an interaction that occurs in 1 of 10,000 cases. This directly leads to an information overload that in turn could lead to ignorance by the physician or any other involved party.

Application level improvements

Besides these improvements on the data level also the application level – which concretely means SEDRI – could be improved further in the future. Independently from the evolution of more specific drug related questions that will be implemented other improvements are imaginable. One of those is a switch from the self implemented federated query processing to a tool like FedX, that was already introduced in section 3.1. This would especially lead to an easier maintenance of the code base. Whether such a change would also lead to a performance improvement has to be evaluated.

Another improvement that is coupled to the virtual integration approach would be a caching mechanism. This would compensate the disadvantage that queries are not processable in case of an unreachable knowledge base.

A third advancement of SEDRI could be the integration or implementation of a more generic SPARQL/ReST converter. This could mean it would be possible to formulate arbitrary SPARQL queries which would be automatically converted to a web interface endpoint for an easier processing. The closest approach for such a requirement is the *Linked Data API*[11]. The Linked Data API is a specification for a configuration vocabulary that defines several API endpoints for a triple store or as a proxy in front of a SPARQL endpoint. With Elda[12] and Puelia[13] exist two implementations for this specification. The former one is the implementation for a Java environment, the latter one for PHP.

[11] `http://code.google.com/p/linked-data-api/` (last access Sept 19, 2013)
[12] `http://code.google.com/p/elda/` (last access Sept 19, 2013)
[13] `http://code.google.com/p/puelia-php/` (last access Sept 19, 2013)

Other projects like *Restpark*[14] which aimed to be a "minimal RESTful API for RDF triples" are either not in a stable version or not being developed any further.

A last possible improvement would be a mechanism for instance matching. This means that if you query two knowledge bases, e.g. for drug-drug interactions and each of the knowledge bases contains information about the same interaction then these two instances would be recognized as identic. But currently there exist no perfect solution for this problem, because most of the matching technologies regarding ontologies are focused on the schema level [18].

Additional improvements are also imaginable for the two evaluation projects Dispedia and DrugMan and the integration of SEDRI. One possible advancement regarding Dispedia would be the integration of more endpoints that are provided by SEDRI, e.g. the side effects or drug proposal endpoints. The former one could support the physician by chosing the right drugs when creating new proposals or the patient by deciding whether he wants to accept or decline a proposal.

Drug Management improvements

Other improvements regarding drug management in general are possible especially in the process of determining the drug name. Currently Dispedia as well as DrugMan use a simple text input with a given autocompletion helper. Nonetheless this approach requires much background knowledge which may be given for a physician but not for a patient who wants to manage his drugs. A possible solution could be the detection of the drug by a given barcode. Several European countries like France or Germany are already testing or even using such an approach by labeling their drug products with a GS1 DataMatrix code[15]. This code contains an unique identifier for the drug like the "Pharmazentralnummer" in Germany. In the consequence by dereferencing such an identifier in a given knowledge base all needed information about the drug could be fetched automatically. Besides the unique identifier in most cases the codes contain more additional data like the expiration date. Especially in patient-centered

[14] `http://lmatteis.github.io/restpark/` (last access Sept 19, 2013)

[15] `http://www.gs1.org/barcodes/technical/bar_code_types#data_matrix` (last access Sept 18, 2013)

applications like DrugMan this approach could lead to a huge benefit for the users – in case of the required effort to determine all drugs as well as the final outcome.

Another drug management advancement for DrugMan would be the implementation of import and export features. There are several possibilities for import and export formats. A general solution would be the support of HL7 CDA [19]. But there are many other country specific solutions, like the German recommendation "ABDA Medikationsplan" [20]. This approach encodes a full medication plan of a patiant as 2D code which can be easily decoded by other interfaces. The main difference between this solution and HL7 CDA is the transport medium of the data. While HL7 CDA is mainly developed for the electronical exchange of the documents the German approach uses the patiant for the exchange. In the latter case the patient ships his medication plan to the physician or pharmacist who decodes the barcode and in case of the prescription of new drugs encodes and prints the new medication plan for the patient.

6 Summary

This thesis presented an approach for answering several drug related questions based on Linked Open Data knowledge bases. Section 1.4 stated the objective to provide an easy to use interface. This objective could be achived what is proofed by the evaluation projects. Therefore the developers or later the respective users have nothing else to know than the appropriate drug names or drug codes. All the required knowledge about the structure of the different knowledge bases is abstracted by SEDRI. How easy the usage of SEDRI is showed the two evaluation projects. They both showed also that SEDRI is usable independently of the programming language or other implementation details of the projects. This is due to the integration and component characteristics of a web-based HTTP interface. Thereby the two main objectives of section 1.4 are achieved.

The problems that were worked out in section 1.2 could be avoided mostly by the infrastructure of SEDRI. The first problem was insuffient integration which is mostly fixed by the Linked Data approach. Having that said the integration could be further improved by implementing Entity Matching mechanisms or the like. By using Linked Data as a knowledge foundation also the second problem – redundancy – is mostly fixed. This is due to the several domain specific knowledge bases where only a small part of the data is redundant. The last problem of possibly incorrect data is rather not applicable to Linked Open Data. The reason is that most of the biomedical knowledge bases are converted from already existing data bases, eg. Drugbank. Therefore the correctness of the knowledge depends on the original data sets.

On the downside especially the evaluation by Dispedia showed some points that are missing to reach a state where the information provided by SEDRI are ready for the everyday professional use. Belonging to these points is the fact that most of the

information in the knowledge bases are only available in English. Another point is the granularity of the provided data, for example the missing severity of drug-drug interactions as carved out in section 5. By regarding the additional downtimes of some knowledge bases every now and then – which might be fixed at application level – the question whether Linked Open Data is an appropriate knowledge foundation for drug management has currently rather to be answered negatively. But it has to be mentioned that this applies to the professional use with the current state of the data. Private persons might already have a benefit by getting the information that their medication contains drug-drug interactions independently of any severity. Also the language problem mentioned aboved does not apply for completely English applications. Therefore in this case the current state of the data might be sufficient.

These points and the other outlined possible improvements of chapter 5 show that the domain of drug management in combination with semantic data contains enough potential for further work and theses in the next years.

Bibliography

[1] Pharmazeutische Zeitung. Medikationsfehler: 25.000 Tote jährlich. `http://www.pharmazeutische-zeitung.de/index.php?id=32557`. abgerufen am 02.12.2012.

[2] World Health Organization. *Management of Drugs at Health Centre Level - Training Manual*, 2004.

[3] Anja Jentzsch, Jun Zhao, Oktie Hassanzadeh, Kei-Hoi Cheung, Matthias Samwald, and Bo Andersson. Linking open drug data. In *Triplification Challenge of the International Conference on Semantic Systems*, pages 3–6, 2009.

[4] Patricia L Whetzel, Natalya F Noy, Nigam H Shah, Paul R Alexander, Csongor Nyulas, Tania Tudorache, and Mark A Musen. Bioportal: enhanced functionality via new web services from the national center for biomedical ontology to access and use ontologies in software applications. *Nucleic acids research*, 39(suppl 2):W541–W545, 2011.

[5] José Luís Oliveira, Gaspar Dias, Ilídio Oliveira, Patrícia Rocha, Isabel Hermosilla, Javier Vicente, Inmaculada Spiteri, Fernando Martin-Sánchez, and António Sousa Pereira. Diseasecard: A web-based tool for the collaborative integration of genetic and medical information. In *Biological and Medical Data Analysis*, pages 409–417. Springer, 2004.

[6] Ali Khalili and Bita Sedaghati. Pharmer–towards semantic medical prescriptions. In *eTELEMED 2013, The Fifth International Conference on eHealth, Telemedicine, and Social Medicine*, pages 9–14, 2013.

[7] Romy Elze and Klaus-Peter Fähnrich. The dispedia framework: A semantic model for medical information supply. In *ICONS 2013, The Eighth International Conference on Systems*, pages 59–63, 2013.

[8] Stephen P Robbins and David A De Cenzo. *Fundamentals of management: essential*

concepts and applications. Pearson Education India, 2007.

[9] Tim Berners-Lee, James Hendler, Ora Lassila, et al. The semantic web. *Scientific american*, 284(5):28–37, 2001.

[10] Graham Klyne, Jeremy J Carroll, and Brian McBride. Resource description framework (rdf): Concepts and abstract syntax. *W3C recommendation*, 10, 2004.

[11] Dave Beckett and Brian McBride. Rdf/xml syntax specification (revised). *W3C recommendation*, 10, 2004.

[12] David Beckett and Tim Berners-Lee. Turtle-terse rdf triple language. *W3C Team Submission*, 14, 2008.

[13] Markus Lanthaler and Christian Gütl. On using json-ld to create evolvable restful services. In *Proceedings of the Third International Workshop on RESTful Design*, pages 25–32. ACM, 2012.

[14] Maurizio Lenzerini. Data integration: A theoretical perspective. In *Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 233–246. ACM, 2002.

[15] Andreas Schwarte, Peter Haase, Katja Hose, Ralf Schenkel, and Michael Schmidt. Fedx: a federation layer for distributed query processing on linked open data. In *The Semanic Web: Research and Applications*, pages 481–486. Springer, 2011.

[16] Micheal Hewett, Diane E Oliver, Daniel L Rubin, Katrina L Easton, Joshua M Stuart, Russ B Altman, and Teri E Klein. Pharmgkb: the pharmacogenetics knowledge base. *Nucleic acids research*, 30(1):163–165, 2002.

[17] SC Montamat, B Cusack, et al. Overcoming problems with polypharmacy and drug misuse in the elderly. *Clinics in geriatric medicine*, 8(1):143, 1992.

[18] Silvana Castano, Alfio Ferrara, Davide Lorusso, and Stefano Montanelli. On the ontology instance matching problem. In *Database and Expert Systems Application, 2008. DEXA'08. 19th International Workshop on*, pages 180–184. IEEE, 2008.

[19] Robert H Dolin, Liora Alschuler, Calvin Beebe, Paul V Biron, Sandra Lee Boyer, Daniel Essin, Elliot Kimber, Tom Lincoln, and John E Mattison. The hl7 clinical document architecture. *Journal of the American Medical Informatics Association*, 8(6):552–569, 2001.

[20] Koordinierungsgruppe zur Umsetzung und Fortschreibung des Aktionsplanes zur Verbesserung der Arzneimitteltherapiesicherheit in Deutschland. Spezifikation für einen patientenbezogenen Medikationsplan, 2012.

List of Figures

List of Tables

Declaration

This master's thesis is the result of my own work. Material from the published or unpublished work of others, which is referred to in the thesis, is credited to the author in the text. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this thesis and the degree examination as a whole.

Leipzig, 14. March 2013 Signature