Architecture for assessing and managing medical knowledge

Peter Winnberg

November 17, 2009
Bachelor's Thesis in Computing Science, 15 ECTS credits
Supervisor at CS-UmU: Helena Lindgren
Examiner: Per Lindström

Umeå University
Department of Computing Science
SE-901 87 UMEÅ
SWEDEN

Abstract

The medical knowledge that is represented in clinical guidelines and protocols contains natural language and informal diagrams. They can contain ambiguous, implicit, and imprecise language. It has been proven difficult to create a formal representation of this knowledge. The Argument Interchange Format (AIF) aims to solve interoperability issues related to argumentation theory. It has been suggested that AIF together with Semantic Web technology could work as a way to formalize clinical guidelines.

This bachelor's thesis describes the design and implementation of a prototype Semantic Web application that uses a modified AIF ontology to represent clinical guidelines. The goal is to support the knowledge engineering process by providing expert physicians with a tool for interpretation of medical knowledge into a computable structure in a knowledge system. Once entered and evaluated, this structure be exported and used in a clinical decision support system.

Contents

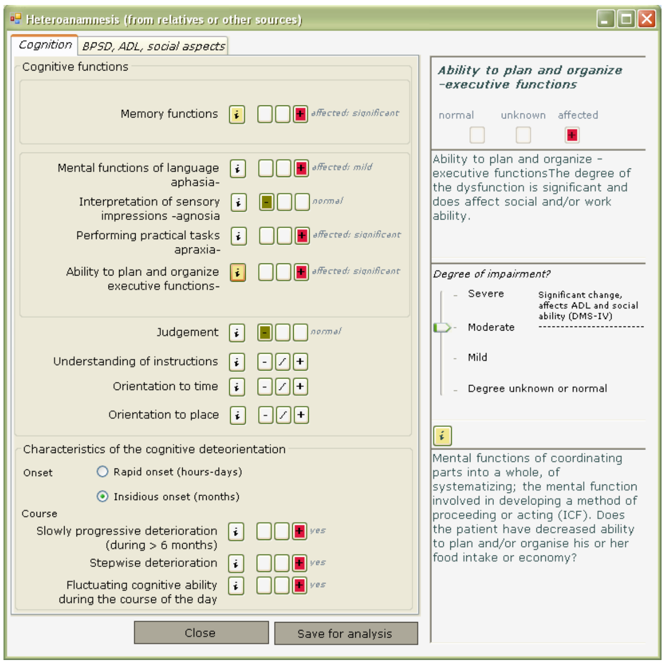
List of abbreviations

AIF	Argument Interchange Format
AJAX	Asynchronous JavaScript and XML
AN	Argument Network
ASCII	American Standard Code for Information Interchange
CA-node	Conflict application node
CDSS	Clinical Decision Support System
DLB	Dementia with Lewy Bodies
DMSS	Dementia Management and Support System
DSM-IV	Fourth edition of the Diagnostic and Statistical Manual of Mental Disorders
FTD	Frontotemporal Dementia
HTML	Hyper Text Markup Language
HTTP	Hypertext Transfer Protocol
I-node	Information node
IE	Internet Explorer
JSON	Javascript Object Notation
MCI	Mild Cognitive Impairment
N3	Notation3
OWL	Web Ontology Language
PA-node	Preference application node
PNG	Portable Network Graphics
RA-node	Rule of inference application node
RDF	Resource Description Framework
S-node	Scheme node
SNOMED CT	Systematized Nomenclature of Medicine – Clinical Terms
SPARQL	SPARQL Protocol and RDF Query Language
SVG	Scalable Vector Graphics
VaD	Vascular dementia
VML	Vector Markup Language
W3C	World Wide Web Consortium
WAI-ARIA	Accessible Rich Internet Applications
WCAG	Web Content Accessibility Guidelines
WWW	World Wide Web
XHTML	Extensible HyperText Markup Language
XSL	Extensible Stylesheet Language
XSLT	XSL Transformations

Chapter 1

Introduction

Dementia Management and Support System (DMSS) is a clinical decision support system that is developed at the department of Computing Science at Umeå University. The goal of this system is to support physicians when diagnosing dementia and one step in this process can be seen in Figure 1.1. The knowledge in the system is based on several guidelines including the fourth edition of the Diagnostic and Statistical Manual of Mental Disorders (DSM-IV) , consensus guidelines for Dementia with Lewy Bodies (DLB), Frontotemporal Dementia (FTD), and Mild Cognitive Impairment (MCI) (Lindgren, 2007).


Heteroanamnesis (from relatives or other sources)
Cognition
BPSD, ADL, social aspects
Close
Save for analysis

Cognitive functions

Memory functions	affected: significant
Mental functions of language aphasia-	affected: mild
Interpretation of sensory impressions -agnosia	normal
Performing practical tasks apraxia-	affected: significant
Ability to plan and organize executive functions-	affected: significant
Judgement	normal
Understanding of instructions	- / +
Orientation to time	- / +
Orientation to place	- / +

Characteristics of the cognitive deteorientation

Onset ○ Rapid onset (hours-days) [✓ Insidious onset (months)]

Course

Slowly progressive deterioration (during > 6 months)	yes
Stepwise deterioration	yes
Fluctuating cognitive ability during the course of the day	yes

Ability to plan and organize -executive functions

normal unknown [✓ affected]

Ability to plan and organize -executive functionsThe degree of the dysfunction is significant and does affect social and/or work ability.

Degree of impairment?

- Severe — Significant change, affects ADL and social ability (DMS-IV)
- [✓ Moderate]
- Mild
- Degree unknown or normal

Mental functions of coordinating parts into a whole, of systematizing; the mental function involved in developing a method of proceeding or acting (ICF). Does the patient have decreased ability to plan and/or organise his or her food intake or economy?

Figure 1.1: A screenshot from the research version of DMSS. Created by Helena Lindgren and used with permission.

The medical knowledge represented in these guidelines contains natural language and informal figures. As such they can contain ambiguous, imprecise, and implicit knowledge. Because of this they can be hard to formalize into a formal, computable structure. Lindgren (2009) argues that the Argument Interchange Format (AIF) could be a way to structure medical knowledge like clinical guidelines. This format is built on argumentation theory and allows modelling of premises, conclusions, and argumentation schemes as a directed graph. Argumentation schemes are important in argumentation theory and they are dialogue patterns that have a set of premises and a conclusion. AIF describes these concepts and the relationship between them in a ontology. Lindgren has created a modified AIF ontology that includes concepts needed to formalize medical knowledge connected to dementia diagnosis. The problems with formalizing medical knowledge and how the AIF could be used as a possible solution to this is described in more detail in Chapter 3.

To be able to add knowledge to DMSS both physicians and knowledge engineers needs to be involved. These experts could be distributed throughout the world and the knowledge could be in different natural languages. A visualization of the structure and reasoning process is also missing. The goal is to create a prototype that is able to create and modify data using the modified AIF ontology. The prototype should be able to visualize the data and handle multiple natural languages. A physician should be able to enter medical knowledge into the system and evaluate it by using patient cases. Chapter 2 describes these goals in more detail.

The ArgDMSS prototype web application is created using the Java programming language and a number of software packages. The data is stored using the Sesame triple store and is queried using the SPARQL Protocol and RDF Query Language (SPARQL). The visualization of the structure is created using the Graphviz package that produces Scalable Vector Graphics (SVG) that is embedded into the Extensible HyperText Markup Language (XHTML). Why these software packages were used is described in more detail in Chapter 4. Chapter 5 shows different parts in the prototype that has been created. Some limitations and some possible future work connected to this prototype is described in Chapter 6.

Chapter 2

Problem description

The first section describes some of the enhancements that are needed compared to the existing software. The second section describes the goals of this project and the third section describes a possible usage scenario.

2.1 Problem statement

To create or modify rules in DMSS (Lindgren, 2007) requires that experts from both the medical and the computer science domain are involved. This can prove difficult when the users are spread out geographically throughout the world. Another challenge connected to this is handling the translation of the data to several languages in the system. Better support for local (and individual) preferences are also important in a clinical decision support system (Lindgren, 2008).

Larsson (2008) suggests a number of changes to DMSS-R, the research version of DMSS. One of the suggestions is visualizing the reasoning process. Larsson uses the Araucaria[1] argumentation tool developed at the University of Dundee as an example of a system that has this. The suggested graphical display of the reasoning is a graph where different colors and edge styles help the user tell different types of nodes apart (e.g., missing or contradicted nodes). Larsson also suggests using hyperlinks to important parts of the guidelines.

2.2 Purpose

The purpose of this thesis is to design and implement a prototype that:

– Is able to create, manipulate, and visualize an argumentation structure based on a modified AIF ontology specified using the Resource Description Framework (RDF) and the Web Ontology Language (OWL).

– Uses a centralized architecture.

– Handle data in several natural languages.

– Allow a physician to analyze guidelines and create schemes.

[1]`http://araucaria.computing.dundee.ac.uk/`

– Allow a physician to create formal representations of the schemes by creating rules.

– Allow a physician to evaluate and validate the interpretations and rules, partly by applying them to patient cases.

2.3 Usage scenario

A physician has a clinical guideline written in natural language that he wants to test against the other information that is in the system. First he identifies the important parts and critical questions and enters these in a scheme. The next step is to create a rule by selecting the scheme and add the available arguments as premises or as a conclusion. Once this rule is saved the physician can use the patient cases in the system to test it. By doing so the system will also visualize conflicts with other rules. In this way, alternative ways to interpret a guideline can be identified and dealt with as well as different ways to implement the guideline in formal rules.

Chapter 3

Formalizing medical knowledge

To be able to diagnose and treat patients accurately physicians need to deal with a very large medical knowledge body. To complicate things further this body of knowledge is under constant change as new diagnoses and treatments are found. Improving the quality of care is the main goal of medical knowledge like clinical guidelines and they can for example be used in education or provide up to date information to physicians.

Patel et al. (2001) argues that to convince physicians overcome unwillingness to use the guidelines they need to be integrated into Clinical Decision Support Systems (CDSSs). The clinical guidelines often include a mix of natural language and informal diagrams and implicit knowledge. The language can be ambiguous and imprecise. In some cases there are several guidelines available for a certain diagnosis and a physician could prefer one guideline over another one. Formalizing the knowledge in the guidelines for use in a CDSS is difficult because of this. Eccher et al. (2008) argues that formalizing another type of medical information, clinical protocols, shares similar problems.

But how does physicians reach a diagnosis? How do humans reach a conclusion? Argumentation theory deals with this. A set of premises leads to a conclusion. Argumentation schemes are dialogue patterns with premises and conclusions that have been identified by researchers. These schemes can be domain independent and one example of such a scheme is argument from expert opinion (Walton et al., 2008, p. 310).

> "Major premise: Source E is an expert in subject domain S containing proposition A."
> "Minor premise: E asserts that proposition A is true (false)."
> "Conclusion: A is true (false)."

For example,

> John Smith is a medical doctor.
> John Smith thinks that M is the best treatment for disease D.
> According to the scheme this leads to the conclusion that M is the best treatment.

Critical questions can be associated with a scheme. The answers to these questions will help decide how credible an argument is. The following questions are examples of critical questions that are associated with the argument from expert opinion scheme (Walton et al., 2008):

"Is E an expert in the field that A is in?"
"Is E personally reliable as a source?"

The second example is a question about trustworthiness and could have subquestions like "Is E honest?" and "Is E biased". If we try this using the John Smith example then, yes John Smith clearly has a degree in a field that is relevant to know something about treatments for diseases. But is John biased? If John works for the corporation that manufactures the M treatment it clearly could reduce the credibility of the argument. But again if John's opinion is based on evidence it is possible that the credibility could be raised again.

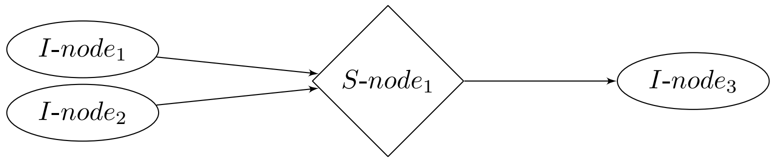


Figure 3.1: An example argument network with three I-nodes and one S-node.

Argumentation theory is an interdisciplinary field where there are many different methods and computer applications to work with arguments, visualize the arguments, and testing hypothesises. The lack of a common standard for interchanging data has lead to interoperability issues. The Argument Interchange Format (AIF) is an attempt to solve this. An initial draft of the ontology was presented by Chesñevar et al. (2006). Ontologies are a description of concepts in a domain and the relation between these concepts. Shadbolt et. all argues that ontologies are important for interoperability and that they can remove ambiguity (Shadbolt et al., 2006).

The AIF models arguments as a directed graph, informally known as an argument network (AN), with two types of nodes, information nodes (I-nodes) and scheme nodes (S-nodes). S-nodes have three subtypes, rule of inference application nodes (RA-nodes), conflict application nodes (CA-nodes) and preference application nodes (PA-nodes). Nodes can have properties (e.g., weight or author) however these properties are not part of the core ontology behind AIF.

Edges between different types of nodes have different meaning. One restriction is that I-nodes can not have edges to other I-nodes. Figure 3.1 shows an example of a basic AN. Edges from I-nodes to RA-nodes means that the I-nodes are used in the inference application. Edges from RA-nodes to I-nodes represent a conclusion, or result, of the inference application. Figure 3.2 shows the John Smith example using an argument network. One of the examples that Chesñevar et al. (2006) brings up is representing the AIF ontology using Semantic Web technology and techniques.

The Semantic Web aims to add meaning to the World Wide Web (WWW). To do this many different methods and techniques have been proposed. One of these is the Resource Description Framework (RDF)[1]. By using RDF it is possible to make statements about a resource. These statements have three parts, a subject, a predicate, and an object (and are also known as a triple). A basic example of a RDF triple is "Anna knows Marie." Several formats have been created to handle the graph that a collection of RDF triples creates, these includes RDF/XML and N3. RDF Schema build on RDF to create a minimal language for representing ontologies.

[1]http://www.w3.org/RDF/

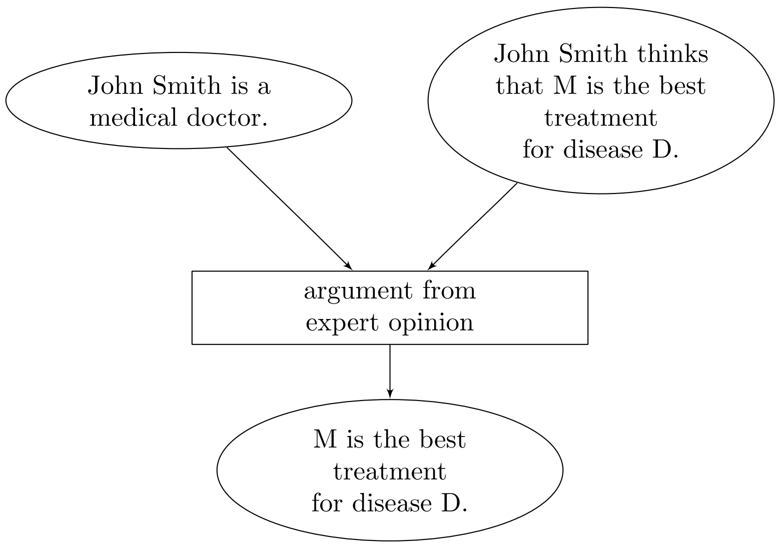

John Smith is a medical doctor.
John Smith thinks that M is the best treatment for disease D.
argument from expert opinion
M is the best treatment for disease D.

Figure 3.2: The John Smith example in an argument network.

The work related to AIF and the Semantic Web has been taken further by Rahwan et al. (2007) that have worked on representing AIF in RDF Schema and implementing this in a prototype application called ArgDF. Work has later been done to represent the AIF ontology, and take advantage of the greater expressive power, in the Web Ontology Language (OWL) (Rahwan and Banihashemi, 2008).

Lindgren (2009) argues that AIF would be useful for representing medical knowledge like clinical guidelines and presents an extended AIF ontology for this purpose. The ontology is using a RDF/OWL representation and has been developed to represent data already in use in the DMSS application. The concepts modelled in this ontology can be seen in Figure 3.3. The idea is that a physician analyses a guideline and identifies the important parts of the guidelines and translates this into a scheme. A formal implementation of this scheme could then be created.

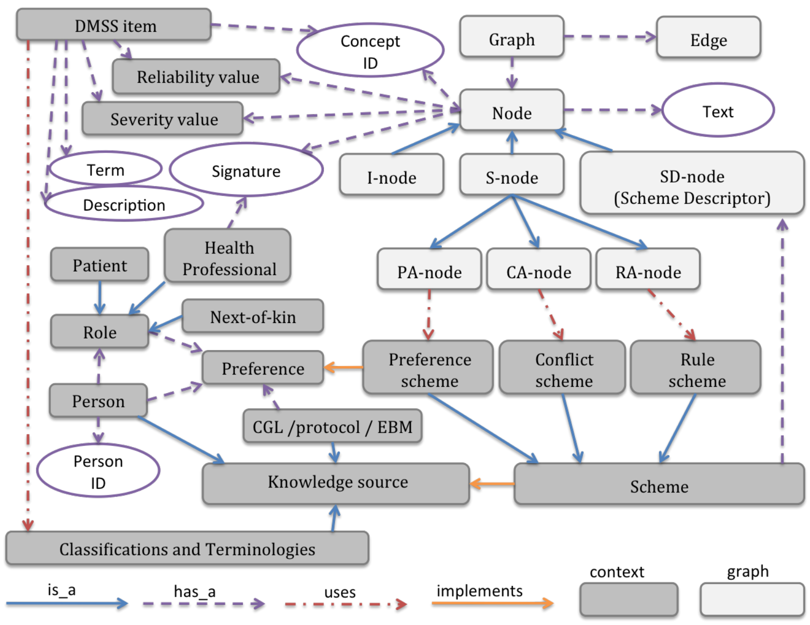


Figure 3.3: The modified AIF ontology described in Lindgren (2009). Created by Helena Lindgren and used with permission.

Chapter 4

Design

This chapter describes the design and development process. It also motivates why certain design choices were made. One of the first design choices that was made was to make a web application. This also affected many of the choices in this chapter.

The development of the prototype has been done using an iterative development process that has included a number of meetings between two computer scientists and two expert physicians. These meetings included both demonstrations and user testing. The user testing showed that the physicians managed to identify ambiguities in the guidelines and managed to take this into account when creating schemes. They also managed to use the system to create rules.

4.1 User Interface

A number of issues came up during meetings with the physicians including the language use in the prototype and how to present the existing data to the user. The labels and text in the prototype contain both graph related terms (e.g. nodes), computer related (e.g. browse), and other terms (e.g. DMSS item) that a physician that is going to use the system will understand right away.

To add premises and a conclusion to a rule a list of all available I-nodes in the system was used (with roughly 100 items). An observation made during one of the meetings with the physicians was that this is a very tedious way to find the wanted I-node because of the way the text description connected to the I-nodes is displayed. Take two examples, "VaD is present" and "Probable VaD" (VaD is short for Vascular dementia and both I-nodes also point to the same DMSS item because of this), two closely related I-nodes would not be close to each other in the list if it was sorted in alphabetic order.

Two redesign options were looked at, searching for I-nodes by using a string and searching for I-nodes using the DMSS item. The first option was ruled out because the ontology lacks support for synonyms and abbreviations. The support for keyword searching in the triple store is also basic. The second option was implemented so that instead of one list with all I-nodes two lists were used. To get an I-node one would first select the DMSS item, submit the form, and then a second list of I-nodes connected to the DMSS item would appear. In this case the second list would contain 4 items. However this would mean that twice as many forms to submit and page reloads would be needed (in some cases) to add I-nodes.

Using a set of web development techniques that is known as AJAX (Garrett, 2005) it is possible to reduce the number of form submits by fetching information in the background without reloading the page. This technique was implemented in the prototype so that when the user selects something in the first list of DMSS items the system loads the next list in the background.

However (Thiessen and Chen, 2007) argues that using this technique could mean that the web application will have accessibility issues but also suggests that Accessible Rich Internet Applications 1.0[1] (WAI-ARIA) could be used to avoid such issues.

4.2 Language & character encoding

The project should be able to handle instances with data in several different languages. This includes languages that use an alphabet other than the latin alphabet. To be able to handle this, the data in the system is stored using the UTF-8 character encoding. When a page is requested by a user information about the character encoding is sent with the Hypertext Transfer Protocol (HTTP) header.

Web browser	`_charset_` support
Firefox 3.0.11	Yes
Internet Explorer 8.0.6001.18702	Yes
Opera 9.64	Yes
Safari 4.0	No

Table 4.1: Support for the `_charset_` field in common web browsers.

To modify the data in the system web forms are used and to ensure that the data sent by these forms use the right encoding (or indicate which encoding that is used so the data can be converted) two methods are used. The first is the `accept-encoding` attribute on the form element. This attribute is part of HTML 4.01 and makes it possible to specify a list of character encodings that the browser should use when sending the data. The other one is a hidden form field that is named `_charset_`. If included the web browser sets the value of the field to the character encoding that the data was sent in.

The `_charset_` field is not part of any specification however work is begin done to include it in HTML5[2]. Also worth noting is that if the `accept-charset` attribute specifies a character encoding that the data does not fit in (e.g. "åäö" using the ASCII character encoding) the data will be converted to numeric character references (e.g. "åäö" becomes "`&#229;&#228;&#246;`").

[1]http://www.w3.org/TR/wai-aria/
[2]http://www.w3.org/TR/html5/

```
<form ... accept-encoding="UTF-8">
...
<!--
Note that the hidden field has no value.
It is added by the web browser on submit
-->
<input type="hidden" name="_charset_"/>
...
</form>
```

Listing 4.1: HTML template to handle the character encoding issues.

As seen in Table 4.1 the support for the `_charset_` field is not supported by all major web browsers. Listing 4.1 shows a template with both the `_charset_` field and the `accept-encoding` attribute. The prototype uses the `_charset_` field, if available, to convert the data to the right character encoding (i.e. UTF-8). If it is not available the data is assumed to use the UTF-8 character encoding.

According to the Web Content Accessibility Guidelines 2.0[3] (WCAG) the default natural language in a document should be specificed so that it can be determined programmatically. If the language is changed within the document it is also important to specify that so the change can be detected.

4.3 Visualization

There are several techniques available to visualize graphs in web applications. These include Java applets, Adobe Flash, Scalable Vector Graphics (SVG), HTML5's canvas element, and regular bitmap images each with its own advantages and disadvantages.

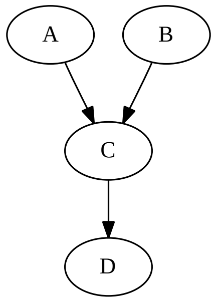


Figure 4.1: A simple graph created using Graphviz.

The Graphviz[4] graph visualization software takes a graph written in the DOT language and supports several different output formats including Portable Network Graphics (PNG), SVG, and HTML5's canvas element (by using CanViz[5]). A basic example of a graph created using Graphviz is shown in Figure 4.1. It is possible to assign URLs to the nodes and if the output format supports it make them clickable links. For example Graphviz can create a PNG image with a client-side image map to make it clickable.

[3]http://www.w3.org/TR/WCAG20/
[4]http://www.graphviz.org/
[5]http://code.google.com/p/canviz/

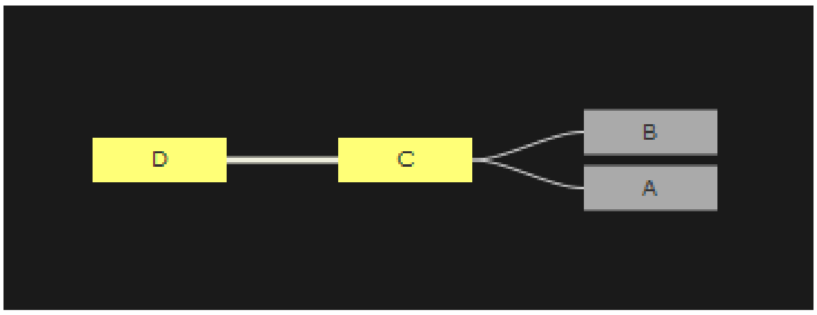


Figure 4.2: A simple graph created using the Javascript InfoVis Toolkit and rendered using HTML5's canvas element.

The Javascript InfoVis Toolkit[6] can load data in JavaScript Object Notation (JSON) and output it on HTML5's canvas element. Figure 4.2 shows a simple example of the Spacetree visualization method included in the package. The graph is interactive and the graph can be explored by clicking on nodes in the graph to focus on that graph and display its children.

There are also work done that is focused on visualizing AIF structures directly, the OVAview[7] is such a tool and it is using Adobe Flash.

Johnson and Jankun-Kelly (2008) describes and compares Java applets, SVG and the canvas element with a focus on performance and scalability. The conclusion they reach after doing a number of tests is that Java has the best performance. SVG showed good performance up to a medium dataset. The dataset that needs to be visualized in this project can not be very large to make sure that the graphs are easy to understand.

To avoid having physicians install additional software on their computers to be able to use this tool both Java-based and Flash-based solutions were ruled out. But the support for SVG and HTML5's canvas element varies between web browsers and this could also lead to that another web browser needs to be installed to be able to use the tool. The excanvas[8] and svgweb[9] projects can be used to add support for these visualization methods to more web browsers. The excanvas Javascript library tries to add support for the canvas element in Internet Explorer (IE) by translating the content of the canvas element to the Vector Markup Language (VML). The svgweb Javascript library tries to add support for SVG in IE by letting Flash render the image.

Because Graphviz supports multiple output methods it was chosen for this project. Both SVG and the canvas element can be used to create interactive visualizations but SVG was choosen because it is a World Wide Web Consortium (W3C) recommendation and that the support for the SVG output method is more stable in Graphviz. Both XHTML and HTML5 allow SVG to be embedded directly into the markup and allow hyperlinks to be embedded in the SVG image. Letting Graphviz create a new image on each request could become a performance problem and because of this a cache mechanism is needed in the prototype. This would allow the server to load pregenerated images from the cache and only generate new images when it is really needed.

[6]`http://thejit.org/`
[7]`http://www.arg.dundee.ac.uk/?page_id=143`
[8]`http://excanvas.sourceforge.net/`
[9]`http://code.google.com/p/svgweb/`

4.4 Storage

Triple stores are repositories that allow persistent storage of RDF triples. The ontology that is used in this project is already available in RDF/XML and using a triple store would allow direct storage of it without translating it to a different format. The prototype uses Sesame[10] and it is triple store that supports both the SERQL and SPARQL[11] query languages. In this prototype SPARQL was chosen over SERQL because it is endorsed by the World Wide Web Consortium (W3C). A very basic example of SPARQL can be seen in Listing 4.2. To create or manipulate instances in the repository Sesame's XML-based transaction data format is used, an example of how to add data using this format is shown in Listing 4.3.

```
SELECT ?node WHERE {
  ?node a argdmss:I-Node
}
```

Listing 4.2: An example of a query to get all I-nodes using SPARQL

```
<?xml version="1.0"?>
<transaction>
  <add>
    <uri>http://example.com/#abc</uri>
    <uri>http://www.w3.org/2000/01/rdf-schema#label</uri>
    <literal xml:lang="en">Abc</literal>
    <contexts>
      <null/>
    </contexts>
  </add>
</transaction>
```

Listing 4.3: An example of how Sesame's transaction format can be used to add a triple to the RDF repository.

Using a triple store does not necessarily mean that a relational database is not used. Sesame can store the triples in a relational database as well as in regular files. Sesame also have support for named graphs (but Sesame calls the feature contexts), a technique to make it possible to name a part of a RDF graph in a single repository. Then it is possible to query just that named part of the repository and not the whole repository. The result of SELECT queries and ASK queries are commonly encoded using SPARQL Query Results XML Format[12].

4.5 Software architecture

The prototype is a web application and uses a client-server architecture. Because of this bug fixes and feature enchantments can be installed on the server and distributed to the clients that connect to it.

There are both examples of where new architectures have been designed directly around RDF triples (e.g., Hera-S (van der Sluijs et al., 2006)) and where it has been

[10] http://www.openrdf.org/
[11] http://www.w3.org/TR/rdf-sparql-query/
[12] http://www.w3.org/TR/rdf-sparql-XMLres/

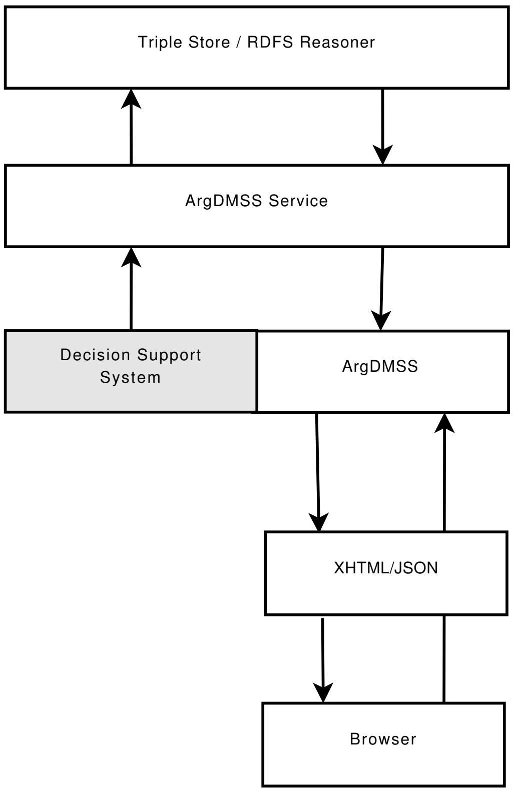


Figure 4.3: Overall view of the system.

integrated into existing frameworks (e.g., ActiveRDF[13]) like Ruby on Rails[14].

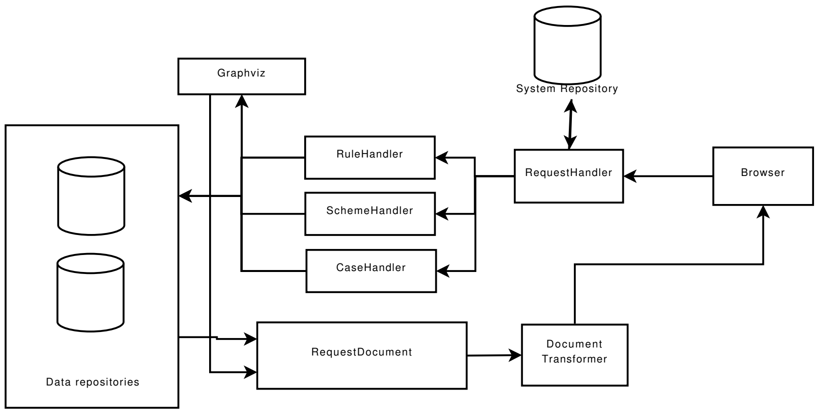


Figure 4.4: More detailed view of the system.

In the initial design the prototype connected directly to the triple store. However since it was possible that several applications (that does not have to be web applications) was going to connect to the same system an additional ArgDMSS Service layer in the design was added and this is shown in Figure 4.3. Applications talk to this service layer using HTTP and XML. This is also important as an additional abstraction layer to avoid that big changes need to be made if the triple store is changed to another one.

The dataflow in the design is shown in Figure 4.4. When a user connects the system looks up the user in the system repository. User information is stored in the system repository along with access permissions (e.g., can the user delete I-node A on repository B?) and translatable system messages. There can be several data repositories. Each data repository does not have to share the exact same ontology but have to be based on the AIF. The result of the SPARQL queries in the Handlers are stored in the RequestDocument. This data can then be transformed using XSLT before it is sent.

[13]`http://activerdf.org/`
[14] `http://rubyonrails.org/`

Chapter 5

The ArgDMSS prototype

The prototype is a web application built using the Java programming language. It uses a number of software packages including:

- Apache Tomcat[1]
- Graphviz[2]
- jQuery[3]
- Sesame[4]
- URL Rewrite Filter[5]

The prototype has three main functionalities:

Schemes identifies and translates important parts of a clinical guideline into a scheme.

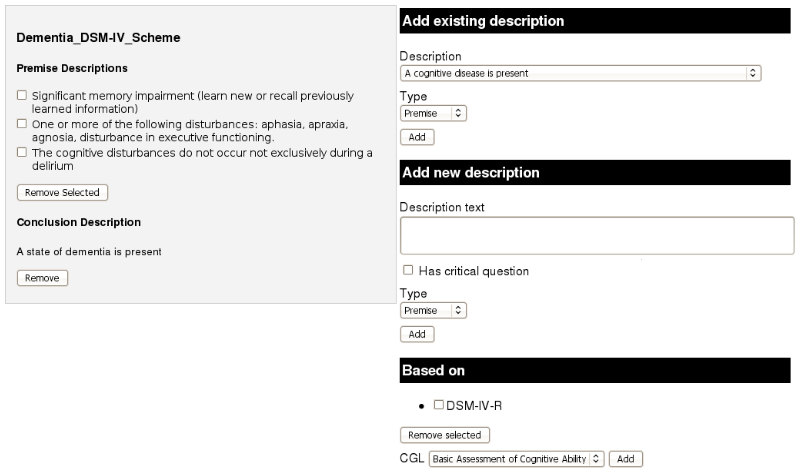
Rules creates formal representation of a scheme.

Patient Cases evaluates the knowledge by using patient cases.

5.1 Schemes

The scheme creation interface (a screenshot can be seen in Figure 5.1) allows you to translate clinical guidelines into a set of schemes and add descriptors. A descriptor is a short description of one premise or conclusion that is found in a guideline. Existing descriptors can be reused in other schemes.

[1]`http://tomcat.apache.org/`
[2]`http://www.graphviz.org/`
[3]`http://jquery.com/`
[4]`http://www.openrdf.org/`
[5]`http://tuckey.org/urlrewrite/`



Dementia_DSM-IV_Scheme

Premise Descriptions

- ☐ Significant memory impairment (learn new or recall previously learned information)
- ☐ One or more of the following disturbances: aphasia, apraxia, agnosia, disturbance in executive functioning.
- ☐ The cognitive disturbances do not occur not exclusively during a delirium

Remove Selected

Conclusion Description

A state of dementia is present

Remove

Add existing description

Description: A cognitive disease is present

Type: Premise

Add

Add new description

Description text

☐ Has critical question

Type: Premise

Add

Based on

- ☐ DSM-IV-R

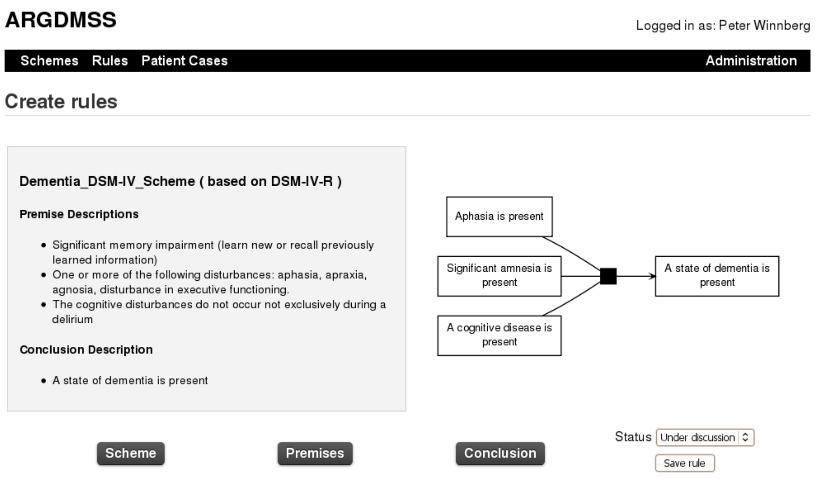
Remove selected

CGL: Basic Assessment of Cognitive Ability — Add

Figure 5.1: A screenshot of the scheme creation interface.

5.2 Rules

When a rule is created a scheme is selected, followed by a number of premises that matches the premise descriptors in the scheme and a conclusion that matches the conclusion descriptor. A screenshot of the user interface can be seen in Figure 5.2. With the scheme and rule next to each other missing premises are easy to catch.



ARGDMSS

Logged in as: Peter Winnberg

Schemes Rules Patient Cases Administration

Create rules

Dementia_DSM-IV_Scheme (based on DSM-IV-R)

Premise Descriptions

- Significant memory impairment (learn new or recall previously learned information)
- One or more of the following disturbances: aphasia, apraxia, agnosia, disturbance in executive functioning.
- The cognitive disturbances do not occur not exclusively during a delirium

Conclusion Description

- A state of dementia is present

Aphasia is present
Significant amnesia is present
A cognitive disease is present
A state of dementia is present
Scheme
Premises
Conclusion

Status: Under discussion

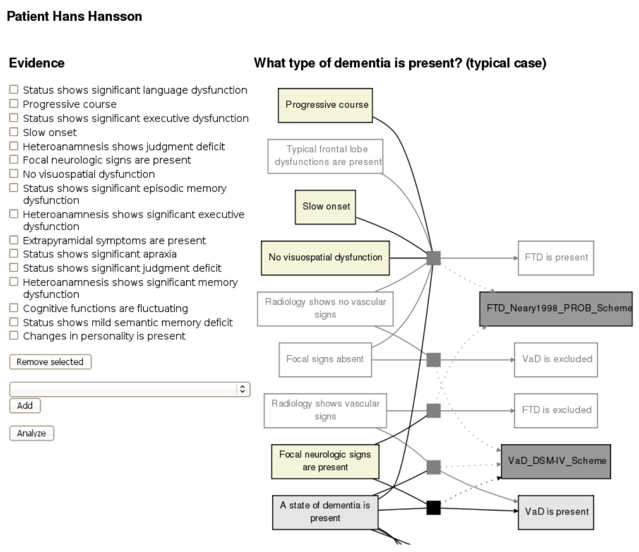
Save rule

Figure 5.2: A screenshot of the rule creation interface.

5.3 Patient Cases

By using the reasoning functionality it is possible to create patient cases, select symptoms and match this against rules in the system. There are 4 different steps:

- Is there a cognitive disease?
- What type of cognitive disease is present?
- What type of dementia is present? (typical case)
- What type of dementia is present? (atypical case)


Patient Hans Hansson

Evidence

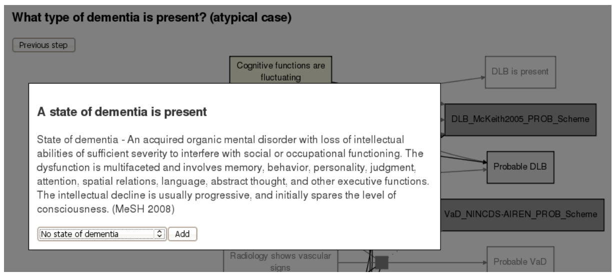
- Status shows significant language dysfunction
- Progressive course
- Status shows significant executive dysfunction
- Slow onset
- Heteroanamnesis shows judgment deficit
- Focal neurologic signs are present
- No visuospatial dysfunction
- Status shows significant episodic memory dysfunction
- Heteroanamnesis shows significant executive dysfunction
- Extrapyramidal symptoms are present
- Status shows significant apraxia
- Status shows significant judgment deficit
- Heteroanamnesis shows significant memory dysfunction
- Cognitive functions are fluctuating
- Status shows mild semantic memory deficit
- Changes in personality is present

What type of dementia is present? (typical case)

Remove selected
Add
Analyze
Progressive course
Typical frontal lobe dysfunctions are present
Slow onset
No visuospatial dysfunction
Radiology shows no vascular signs
Focal signs absent
Radiology shows vascular signs
Focal neurologic signs are present
A state of dementia is present
FTD is present
FTD_Neary1998_PROB_Scheme
VaD is excluded
FTD is excluded
VaD_DSM-IV_Scheme
VaD is present

Figure 5.3: A screenshot of a patient case.

Each one of these steps is connected to a set of schemes. The set of schemes associated with a certain step is visualized together, the way Figure 5.3 shows.


What type of dementia is present? (atypical case)

A state of dementia is present

State of dementia - An acquired organic mental disorder with loss of intellectual abilities of sufficient severity to interfere with social or occupational functioning. The dysfunction is multifaceted and involves memory, behavior, personality, judgment, attention, spatial relations, language, abstract thought, and other executive functions. The intellectual decline is usually progressive, and initially spares the level of consciousness. (MeSH 2008)

Previous step
Cognitive functions are fluctuating
DLB is present
DLB_McKeith2005_PROB_Scheme
Probable DLB
VaD_NINCDS-AIREN_PROB_Scheme
No state of dementia
Add
Radiology shows vascular signs
Probable VaD

Figure 5.4: Additional information and options that appear when clicking on a node in a patient case.

Nodes that have been asserted have beige color, infered nodes a light grey color, and schemes a dark grey color. It is possible to click on the nodes for additional information and options like Figure 5.4 shows.

Chapter 6

Conclusion

A prototype web application has been developed that is able to manipulate data from the ontology presented in Lindgren (2009). The system can store information in different natural languages. Based on the limited user testing done during the project physicians were able to create schemes, rules and could do basic evaluations of the rules in the system using the patient case view.

6.1 Limitations & Future work

The development described in this thesis has focused on RA-nodes. More work is needed to support CA-nodes and PA-nodes. One possible way to handle PA-nodes could be a feature that allows users to attach comments to schemes and rules. The Patient Case view can not yet handle detection and explicit visualization of conflicts between different rules.

The accessibility in the application should be evaluated. The development of Accessible Rich Internet Applications (WAI-ARIA) 1.0 should be followed and necessary changes should be made to make sure that the application is accessible.

The prototype has made use of existing data that is essential for diagnosing dementia. Entering the data needed to support new areas could be a tedious process. Existing collections of medical terminology exists (e.g. SNOMED CT) and ways to import and use this data should be investigated.

The development of the ArgDMSS Service is not yet complete. To fully enable other applications to share the same repository this should be completed and another system should be adapted to use it in order to test it. Other parts of the software architecture, the visualizer and its cache mechanism, also needs further testing.

Keyword search in parts of the user interface could be very useful. However to do this both better support for abbreviations and synonyms in the ontology should developed. Another thing that needs to be looked into in order to implement this is more advanced support for keyword searching in the triple store.

Implementing a revision control system for the rules in the system could help users by providing undo functionality and could also provide data for future evaluations.

How to handle critical questions is something that should be investigated further because it could be an important part of the process to assess the knowledge. One possible solution to this is to engage the physician in a dialogue while a patient case is analyzed. The result of this dialogue could then affect the analysis of the patient case.

Rules where logical or is involved is currently handled using multiple rules. For example to draw a conclusion Premise1 must be present and one of the following premises Premise2, Premise3 Premise 4 (i.e., $Premise1 \wedge (Premise2 \vee Premise3 \vee Premise4)$. This would be represented using 3 rules. A possible question for further work is if logical or was handled by the underlying ontology would it be possible to create a good user interface that is intuitive to the users (compared to only using logical and).

Physicians with different level of experience need a different level of support. Further user testing needs to be done with a larger number of physicians to decide how to handle this and what changes that is needed to support different experience levels. Which terms that are most intuitive for physicians of different levels should also be investigated.

Acknowledgements

I like to thank my supervisor Helena Lindgren for giving me the opportunity to work on this project, always being interested in trying new solutions, and providing many interesting articles to read. I would also like to thank the expert physicians that provided valuable insights during the evaluations of the prototype and the reviewers that provided valuable feedback on the report. Finally I would also like to thank my family for being there, for providing feedback about the prototype, and helping me extend my understanding of health care in general.

References

Chesñevar, C., McGinnis, J., Modgil, S., Rahwan, I., Reed, C., Simari, G., South, M., Vreeswijk, G. and Willmott, S. (2006). Towards an Argument Interchange Format, *The Knowledge Engineeing Review* **21**(4): 293–316.

Eccher, C., Rospocher, M., Seyfang, A., Ferro, A. and Miksch, S. (2008). Modeling clinical protocols using semantic MediaWiki: the case of the Oncocure project, *ECAI 2008 Workshop on the Knowledge Management for Healthcare Processes (K4HelP)*, University of Patras, pp. 20–24.

Garrett, J. J. (2005). Ajax: A New Approach to Web Applications.
URL: *http://adaptivepath.com/ideas/essays/archives/000385.php (visited 2009-05-15)*

Johnson, D. W. and Jankun-Kelly, T. J. (2008). A scalability study of web-native information visualization, *GI '08: Proceedings of graphics interface 2008*, Canadian Information Processing Society, Toronto, Ont., Canada, Canada, pp. 163–168.

Larsson, L. (2008). Activity Theory based evaluation of DMSS-R. MSc thesis, UMNAD 744/08, Department of Computing Science, Umeå University, Sweden.

Lindgren, H. (2007). *Decision Support in Dementia Care–Developing Systems for Interactive Reasoning*, PhD thesis, Umeå University, Sweden.

Lindgren, H. (2008). Decision Support System Supporting Clinical Reasoning Process – an Evaluation Study in Dementia Care, *Stud Health Technol Inform* **136**: 315–320.

Lindgren, H. (2009). Towards Using Argumentation Schemes and Critical Questions for Supporting Diagnostic Reasoning in the Dementia Domain, *Proceedings of CMNA 2009*.

Patel, V., Arocha, J., Diermeier, M., How, J. and Mottur-Pilson, C. (2001). Cognitive psychological studies of representation and use of clinical practice guidelines, *International Journal of Medical Informatics* **63**(3): 147–167.

Rahwan, I. and Banihashemi, B. (2008). Arguments in OWL: A Progress Report, *in* P. Besnard, S. Doutre and A. Hunter (eds), *Computational Models of Argument: Proceedings of COMMA 2008*, IOS Press, pp. 297–310.

Rahwan, I., Zablith, F. and Reed, C. (2007). Laying the foundations for a World Wide Argument Web, *Artificial Intelligence* **171**(10-15): 897–921. Argumentation in Artificial Intelligence.

Shadbolt, N., Hall, W. and Berners-Lee, T. (2006). The Semantic Web Revisited, *Intelligent Systems* **21**(3): 96–101.

Thiessen, P. and Chen, C. (2007). Ajax live regions: chat as a case example, *W4A '07: Proceedings of the 2007 international cross-disciplinary conference on Web accessibility (W4A)*, ACM, New York, NY, USA, pp. 7–14.

van der Sluijs, K., Houben, G.-J., Broekstra, J. and Casteleyn, S. (2006). Hera-S: web design using sesame, *ICWE '06: Proceedings of the 6th international conference on Web engineering*, ACM, New York, NY, USA, pp. 337–344.

Walton, D., Reed, C. and Macagno, F. (2008). *Argumentation Schemes*, Cambridge University Press.