Design of an inference engine for the semantic web

This document is the proposal for the Master Thesis made as a part of my Master Degree in Computer Science Education (Software Systems) at the Open University of the Netherlands . The work will be done in collaboration with the research department of the company Agfa in Mortsel Belgium.

Student data

Name	Guido Naudts
Student number	831708907
Address	Secretarisdreef 5
	2288 Bouwel
Telephone work	0030-2-542.76.01
Home	0030-14-51.32.43
E-mail	Naudts_Vannoten@yahoo.com

Introduction

What is the semantic web ? Very short : the semantic web is the automation of the internet .What does this mean ? On the internet an enormous amount of information is available mostly in HTML or other human readable formats . The semantic interpretation of the material has to be done by a human being and cannot be done by a computer . These pages therefore are only in a small degree accessible for computer manipulation (something can be done by interpreting e.g. the embedded links) .

Now if we want to make these pages available for processing by the computer we can do two things :

- we can let the computer analyse the contents of the pages . This brings us into the realm of artificial intelligence . I will not speak further of this possibility .
- we can add information to the pages that is meant to be for usage by a computer . This information is called meta-information . As always if this information has to be useful standards are needed . So W3C devised the standard RDF . RDF stands for Resource Description Framework and W3C stands for World Wide Web Consortium , an organism whose task is to make standards for the internet . RDF gives the possibility for adding declarative meta-information to the web pages . It is this second possibility I will follow in this proposal .

The standard representation of RDF is in XML format . I will not use this representation but instead , for ease of use, I will use the RDF representation named N3 . This is a syntax for RDF that is more easy to use and work with than the XML syntax .

Thus basically RDF is a set of triples : subject verb object . These are binary relations where the verb can be considered to be a predicate (but not completely). This set of triples constitutes a directed graph with arrows going from subject to object; the arrows represent the verb .

With RDF alone we can go a long way towards the automation of the internet . But eventually our system that is now composed of the first two layers of fig. 1 will have to be extended because some of the things we wish to be done by computers still are not within their reach . One standard extension exists already : RDFS or RDF Schema . Here RDF is extended with notions like : class, property, constraint,... Another extension is in preparation (in the Webont working group of the W3C) where an ontology for RDF will be specified .

But we are not satisfied yet : on top of these four layers (see fig. 1) we want a logical layer so that we can define rules to be used by inference engines . Sometimes we want to prove something too e.g. rule 1 says : x is the author of y . Rule 2 (on another website) says : the author of y is director of w3c . Then we want to prove : x is director of w3c .

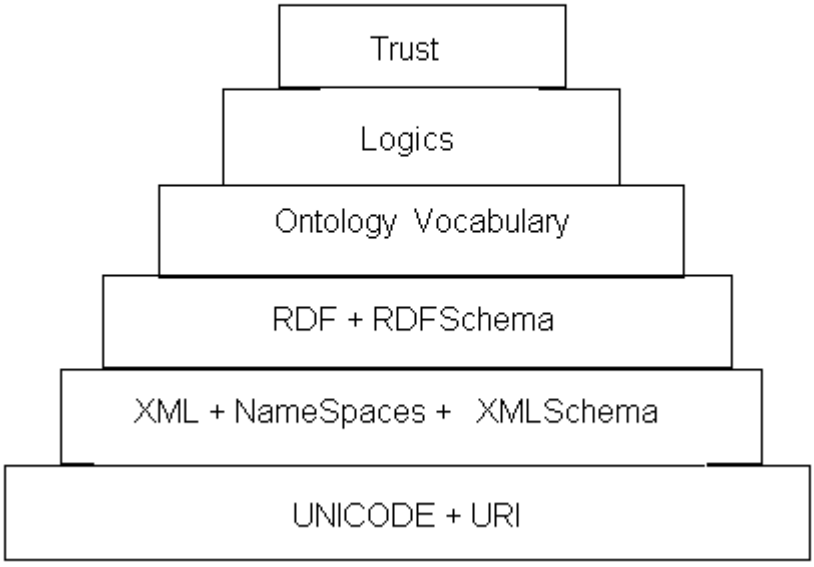

Fig. 1 The different layers composing the semantic web .
Explanation of fig. 1 :
At the bottom we have the basic mechanisms for defining and using information on the internet : unicode and URI = Universal Resource Locator .
At the second level we find the well known XML (Extensible Markup Language) with its namespaces and XMLSchema , the successor of DTD (= Document Type Definition) , a language for describing XML objects .
At the third level we find RDF (Resource Description Framework) [RDFMS] and RDFSchema (see text) [RDFSC] .
At the fourth level we find the definition of an ontology . Where RDF offers a syntax for describing a document , the ontology will offer the specific terms needed for describing it .
At the fifth level we find the logic layer . This layer will make it possible to generate rules and make queries based on these rules .
Finally, at the sixth level we find the definition of trust . In order for people and computers to trust each other we must establish a web of trust . This must be based on

rules (logic) and cryptographic mechanisms . Besides trust we can find other applications at this level .

It is the logic layer and the proof layer that will be the subject of this thesis .

Research question :

The problem :

For the semantic web experiments have been done with two programming systems that are inference engines : CWM (abbreviation for Closed World Machine) [SWAP/CWM] and Euler [DEROO] . These programs are based on the RDF and RDFS standards and on top of these are using logical enhancements and ontology features . However these program systems are experimental and were developed ad-hoc . The need has come up now for an inference engine that is adaptable , well specified and as minimal as possible as Tim Berners-Lee points out [DESIGN] .

A well defined engine can help a lot in defining consistency and procuring efficiency .Indeed it is difficult to reason about something which is not well specified .

So there really is a need to develop an inference engine (or family of engines) that are founded on a more formal basis . So the fundamental question arises : **What is the best way for realising an inference engine so that the requirements that are posed by the semantic web (and the internet) are met ?** .

What requirements are posed by the internet ? I see following points :

- the data are distributed . This has as a consequence that it is difficult to avoid inconsistencies .
- heterogeneity of goals and purposes must be taken into account .
- we have to deal with a relatively slow transmission medium .
- the scale of everything can vary from small to extensive .

My first idea was that the development of a WAM machine might bring enough formalism to build an efficient inference engine . However as was pointed out at a meeting in Heerlen by prof.dr. Jeuring a WAM machine is too low level for efficiently reasoning on the characteristics of the deductive engine (meta-reasoning) . So the formalism can better be made on a higher level . I agree with this (though a WAM-machine might give considerable gains in the programmatic implementation) when we are talking about the logical mechanisms involved who seem to be more important than the programmatic issues .

There are however other arguments . There is the argument of evolvability (a general requirment for software development : see [GHEZZI]) . The internet is not a static phenomenon neither is the semantic web . So we must take into account continuous development and changes to an inference engine deployed on the scale of the internet . There is the argument of variability . In a system with millions of users

we might expect that not all user communities present the same needs and can be satisfied with the same characteristics of an inference engine . What does this imply ? It implies that the engine must be build in a modular way . This again leads to the structure of a basic engine which is kept as small as possible . This basic engine is enhanced with a set of logic specifications comprising facts and rules . However there has to be a formal specification of the engine too . Such a specification can be made with a functional language .

To be able to reason about the inference engine a specification of the engine has to be made . One way of doing this is by using meta-logical frameworks . Hence the question of research can be reformulated : **can meta-logical frameworks be used for specifying the inference engine of the semantic web ?**
However more generally other systems exist that could be used for the purpose of defining logical systems : ALF, GANDALF, lambda-prolog ...

Once a description on a meta-level has been made we might want to reconsider the two points mentioned above namely : optimisation and checking of inconsistency . Different mechanisms might exist for optimising a proof process . One is especially worth mentioning : reordering (this can be done as well on the inter-clause level (reordering of the sequence of clauses) as on the intra-clause level (reordering of the premises)). Other principles might exist (e.g. failure should be detected as soon as possible) . **So the question is : what optimisation techniques can be used ?**

Consistency check is also important on the internet . It interferes in a certain degree with optimisation in a sense that inconsistencies should be detected as soon as possible . We can define inconsistencies by logical rules . However another possibility is : if we have defined the inference engine by a metalogical specification then we can metalogically check the correctness of the proofs whereby thus inconsistencies will be detected . **The question is : how can we best avoid inconsistencies ?**

As we speak about deduction we also speak about logic . Some assumptions can readily be made concerning the logic when a system on internet scale is involved :

- no closed world : we can not accept that the information will be complete when we gather and merge files on the internet .
- paraconsistent logics : when we encounter contradictions we would acquire strange results if we accept that from a contradiction everything can be deduced .
- should reasoning be monotonic ?

The question is : which logic system should be followed on the internet ?

Another question is : what is the interpretation of the logic i.e. what is its semantics ?

Solution framework :

We propose the following framework as a guiding structure for arriving at an answer to the questions posed above :

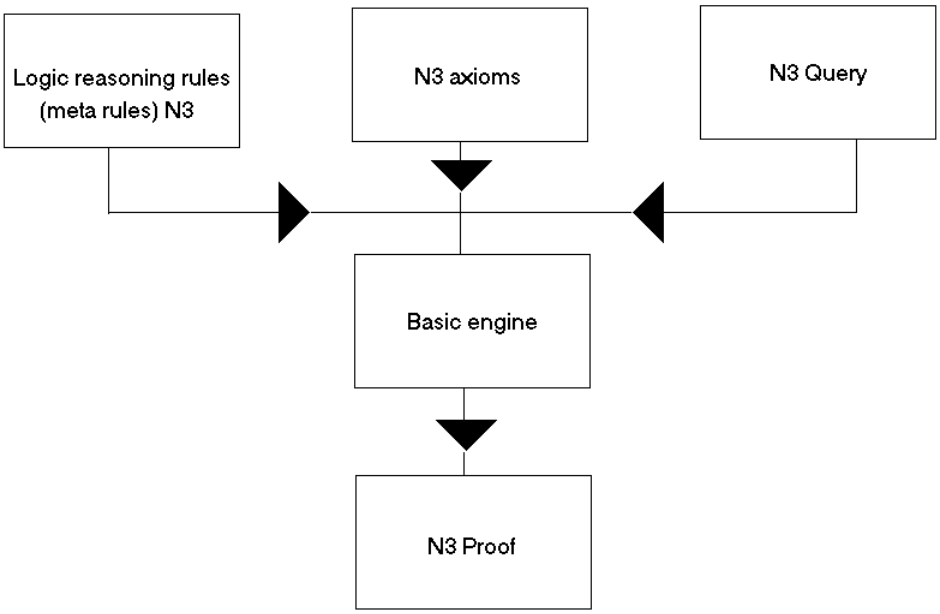

Fig.2. Guiding framework for the thesis .

The basic engine is based on the Robinson resolution algorithm and consists basically of unification and backtracking . The engine is kept as simple as possible . Functions which are not considered to be basic will be defined in the meta-rules in N3-format . An example of this is the definition of a property as being a transitive property in RDFSchema . Possibly also optimisations and strategies for the engine can be defined here ?

What must be built in and what not ?

Should the meta-part be treated by the engine in the same way as the N3 axioms or not ?

The axioms represent input data and rules . They and the query are also in N3-format as well as the proof .

Goal of the thesis

The goal is to specify and test a basic inference engine in a functional language ; study how the engine can be extended with meta-statements in particular statements that are concerned with the efficiency of the deductive process and the consistency of the input data ; test the engine using test cases of the semantic web . An answer will be given to the questions asked above : if the answer cannot be complete , I will strive to highlight the most important points .

Eventually a working engine will be built in Python that can use the meta-rules in N3 syntax .

Planning

Specification of a parser and an inference engine in a functional language (presumably Haskell) : 300 hours .

Testing the use cases and building a set of meta-statements for the engine :

200 hours .

Study of the literature : 150 hours .

Writing the text of the thesis : 150 hours .

Coaching and graduation committee

Chairman: prof dr J. Jeuring
Secretary : ir. F. Wester
Coach: ir. J. De Roo

Literature

[AIT] Hassan Aït-Kaci
WAM A tutorial reconstruction .
* could be more readable but still a good intro in Prolog engines .

[BENTHEM] Van Benthem e.a.
Logica voor informatici
Addison Wesley 1991.
* a very good introduction in logic . In Dutch .

[DESIGN] http://www.w3.org/DesignIssues/
* *Tim Berners-Lee's site with his design-issues articles* .

[DEROO] http://www.agfa.com/w3c/jdroo
* *the site of the Euler program*

[GANDALF] http://www.cs.chalmers.se/~tammet/gandalf
* *Gandalf Home Page*

[GHEZZI] Ghezzi e.a.
Fundamentals of Software Engineering
Prentice-Hall 1991 .

[HILOG] http://www.cs.sunysb.edu/~warren/xsbbook/node45.html

[LAMBDA] http://www.cse.psu.edu/~dale/lProlog/
* *lambda prolog home page*

[MCGUINESS] Deborah McGuiness
Explaining reasoning in description logics
1966 Ph.D.Thesis
* a very readable text

[OTTER] http://www.mcs.anl.gov/AR/otter/

* *Otter Home Page*

[PFENNIG1] Pfennig F.
Computation and deduction Draft April2 1997 .
* difficult to read

[PFENNIG2] Pfennig F .
Logic frameworks 1999
* difficult to read

[RDFM] *RDF Model Theory* .Editor: Patrick Hayes
<http://www.w3.org/TR/rdf-mt/>
* readable, important .

[RDFMS] *Resource Description Framework (RDF) Model and Syntax Specification*
<http://www.w3.org/TR/1999/REC-rdf-syntax-19990222>

[RDFSC] *Resource Description Framework (RDF) Schema Specification 1.0*
<http://www.w3.org/TR/2000/CR-rdf-schema-20000327>

[SWAP/CWM] http://www.w3.org/2000/10/swap
http://infomesh.net/2001/cwm
* CWM is another inference engine for the web .

[TBL01] Berners-Lee e.a.
The semantic web
Scientific American May 2001
* the famous article in Scientific American

[TWELF] http://www.cs.cmu.edu/~twelf
* *Twelf Home Page*

[UNCERT] Hin-Kwong Ng e.a.
Modelling uncertainties in argumentation
Department of Systems Engineering & Engineering Management
The Chinese University of Hong Kong
http://www.se.cuhk.edu.hk/~hkng/papers/uai98/uai98.html
* some ideas about handling uncertainties

[WESTER] Wester e.a.
Concepten van programmeertalen
Open Universiteit Eerste druk 1994
* Important for a thorough introduction in Gofer . In Dutch .

Abbreviations

ALF : Algebraic Logic Functional Programming Language

CWM : Closed World Machine
An experimental inference engine for the semantic web

DTD : Document Type Definition , a language for defining XML-objects .

HTML : Hypertext Markup Language

RDF : Resource Description Framework

RDFS : RDF Schema

W3C : World Wide Web Consortium

WAM : Warren Abstract Machine
Probably the first efficient implementation of prolog .

XML : Extensible Markup Language .
The difference with HTML is that tags can be freely defined in XML .