Ontology-Based Query and Answering in Chemistry: OntoNova @ Project Halo

J. Angele, E. Moench, H. Oppermann, S. Staab, and D. Wenke

Ontoprise GmbH, Amalienbadstraße 36,
76227 Karlsruhe, Germany
{angele, moench, oppermann, staab, wenke}@ontoprise.de

Abstract. The Project Halo has the long term objective of developing a *Digital Aristotle*, i.e. a knowledge system that is able to answer questions in a particular domain and give explanations for its answers. In this paper we report about the Ontoprise contribution to the Halo Pilot Project, in which various competing ontology engineering methodologies and knowledge system capabilities have been investigated. Concerning the first, we describe how we dealt with engineering a significant set of laws from chemistry that we had to let interact at different levels of generality and in varying orders. With regard to the latter, we report on the ability of our system to produce coherent and concise explanations of its reasoning. The importance of these two aspects can hardly be underestimated in the Semantic Web, as with future growth the interaction of large sets of laws will require dedicated management as well as the ability to let the user explore the trustworthiness of the ontology and the underlying data sources.

1 Introduction

The Halo Pilot Project[1] is the first phase of a projected multi-phase effort by Vulcan Inc.[2] whose ultimate goal is the creation of a "Digital Aristotle", an expert tutor in a wide variety of subjects. The Halo Pilot was a six-month effort intended to assess the state-of-the-art in question answering with an emphasis on deep reasoning. The effort was structured around the challenge of responding to variants of AP Chemistry questions that focused on a portion of the "Advanced Placement test: Chemistry".[3]

Our system, OntoNova, answers questions from this AP test. OntoNova justifies its answers in detail giving natural language explanations. The answers have been evaluated by expert graders according to the directives of the Educational Testing

[1] http://www.projecthalo.com
[2] http://www.vulcan.com
[3] The Advance Placement Program gives students the chance to try college-level work while still in high school. If one gets a "qualifying" grade on the AP Exam, many colleges give credit or advanced placement for these efforts (http://apcentral.collegeboard.com/program).

Services. Besides Ontoprise two other contenders, Cycorp Inc[4]. and SRI International[5], have built competing systems.

There are a number of issues that make the Halo Pilot Project as a whole and OntoNova as one of its parts extremely relevant for the Semantic Web:

1. The Halo Pilot constitutes a good match for some of the objectives that the Semantic Web community wants to achieve:
   a. Exploiting a rich ontology,
   b. Integrating various sources of knowledge,
   c. Giving justifications for deduced results.
2. The Halo Pilot Project was a controlled experiment with a cleanly defined setting, independent evaluators, freely available resources and re-produceable results from. Such settings are useful to
   a. Quantify costs, and
   b. Quantify adequacy of methods.
3. While the Halo Pilot still produced knowledge bases that appear to be too costly, it also showed that encoding knowledge by experts is "just" one order of magnitude more costly than writing the natural language text itself.

While the description of the overall Halo Pilot Project is currently put together by a joint committee consisting of representatives of Vulcan Inc. and the three teams, we here focus on number 1 shown in the list above and in particular on the description of OntoNova and the conclusions we could draw from our experiences building it. To do so, we first describe our technical approach to allow for the deduction of answers and justifications. In chapter 3, we present the architecture of our knowledge base that produced the answers and justifications. Chapter 4 briefly compares the overall results of the project as far as necessary to frame the OntoNova system. We conclude with some lessons learned.

2 Technical Approaches

Our technical approach builds on two main pillars. First (cf. Section 2.1), we build on the OntoBroker® technology that we have developed and that we have continuously improved for over a decade now [Ang93,DEFS99]. A complex ontology with rules, multiple representations of objects, and call-out functionality to particular problem-solving methods has been modeled in F-Logic [KLW95] to allow for infering answers posed to the system. Second (Secion 2.2), the deduction process has been extended and logged in F-Logic according to a particular trace ontology. This result could then be used by a second, but otherwise identical, inference engine in order to reason why a particular result had been achieved.

2.1 Representing the chemistry domain in F-Logic

Conceptual (or Ontology) modeling deals with the question of how to describe in a declarative and abstract way the domain information of an application, its relevant

[4] http://www.cyc.com
[5] http://www.sri.com

vocabulary, and how to constrain the use of the data, by understanding what can be drawn from it. Corresponding abstract representation languages support the understanding of such descriptions, their rapid development, their maintenance and their reuse.

F-Logic ("F" stands for "Frames") combines the advantages of a conceptual high-level approach typical for frame-based languages and the expressiveness, the compact syntax, and the well defined semantics from logics. The original features of F-Logic [KLW95] include signatures, object identity, complex objects, methods, classes, inheritance, and rules. Our implementation of F-Logic by OntoBroker [DEFS99] introduced extensions and restrictions of the original features to make F-Logic a powerful and efficient language for many respective intended application domains.

Basic concepts of chemistry are represented as concepts in F-Logic and are arranged in an isa-hierarchy:

```
Molecule::Root.
Reaction::Root.
Ion::Molecule.
Anion::Ion.
Cation::Ion.
AlkaliMetalCation::Cation.
AlkalineEarthMetalCation::Cation.
PrecipitationReaction::Reaction.
GaseousReaction::Reaction.
```

Properties of these concepts and relations between these concepts are represented by methods. E.g.:

```
IonicMolecule[
      hasName=>>STRING;
      hasAnion=>>Anion;
      hasCation=>>Cation;
      hasAnionCoefficient=>>NUMBER;
      hasCationCoefficient=>>NUMBER;
      hasFormula=>>STRING].

HomogeniusMixture[hasComponent=>>PureSubstance].

Reaction[
      hasReactants=>>Molecule;
      hasReactantnames=>>STRING;
      hasProducts=>>Molecule;
      hasProductnames=>>STRING;
      hasEquation=>>STRING;
      fromMixture=>>Mixture;
      …
      ].
```

Complex chemical relationships and axioms are represented by rules:

```
rule burnhydrocarbon: FORALL F,V1,V2,V3
burned(F):CombustionReaction[hasReactants->>{"O2",F};
          hasProducts->>{"H2O","CO2"}] <-
          burn(F) and hydrocarbon(F).
```

This rule states that if a formula *F* represents a hydrocarbon and is burned then the reaction is identified as a combustion reaction with the reactants O_2 and *F* and the products H_2O and CO_2 of the reaction equation .

Within these rules very often the arithmetic built-in functions of OntoBroker have been used:

```
FORALL R,C,C1,M,Re,Pr,L,L1,L2,Z
    R[coefficients->>C;
          hasReactantsList->>L1;hasProductsList->>L2]
          <-    reactants(R,Re) and products(R,Pr) and
                and concatlists(Re,Pr,L) and
                matrixbycols(L,M) and lessolve(M,C1)
                wholenumbered(C1,C).
```

In this rule the reactants and products of a reaction are taken as lists, a matrix for a linear equation system is generated, solved and from the solutions vectors with integer coefficients are generated. Using this approach the problem to balance a chemical equation is solved.

The input for the answering process is either given as ground facts or as constants used in the query. For instance the input "methanol CH_3OH is burned" is given by the F-Logic fact:

```
burn("CH3OH").
```

Afterwards we can pose a question for the balanced reaction equation

```
FORALL Eq,R <- R:Reaction[hasEquation->Eq].
```

which results in the following answer:

```
R = burned("CH3OH")
Eq = "2O2 + 3CH3OH -> 4H2O + 2CO2"
```

As a conclusion it turned out, that F-Logic is very well suited for capturing the chemical domain. In contrast to the other teams OntoNova provided the most compact encoding of the knowledge base to cover the domain.

OntoBroker provides means for efficient reasoning in F-Logic. OntoBroker performs a mixture of forward and backward chaining based on the dynamic filtering algorithm [KL86] to compute (the smallest possible) subset of the model for answering the query. During forward chaining not only single tuples of variable instantiations but sets of such tuples are processed. It is well-known that set-oriented evaluation strategies are much more efficient than tuple oriented ones. The semantics for a set of F-Logic statements is then defined by a transformation process of F-Logic

into normal logic (Horn logic with negation) and the well-founded semantics [GRS91] for the resulting set of facts and rules and axioms in normal logic.

2.2 Answer Justification

There are many reasons that users and applications in the Semantic Web need to understand the provenance of the information they get back from applications. One major motivating factor is trust. Trust and reuse of retrieval and deduction processes is facilitated when explanations are available. Ultimately, if users and/or applications are expected to trust, use and reuse application results, potentially in combination with other information or other application results, users and agents may need to understand where the derived and source information came from at varying degrees of detail. This information, sometimes called provenance, may be viewed as meta information about information told. Provenance information may include source name, date and author(s) of last update, authoritativeness of the source, degree of belief, degree of completeness, etc.

Even more, additional types of information may be required if users need to understand the meaning of terms or implications of query answers. If applications make deductions or otherwise manipulate information, users may need to understand how deductions were made and what manipulations were done. Information concerning derived or manipulated information may include term or phrase meaning, term inter-relationships (ontological relations including subclass, super-class, part-of, etc.), the source of derived information, reasoner description and others. In [McG03] an overview about requirements for answer explanation components may be found.

Recognition of the importance of explanation components for reasoning systems existed in a number of fields for many years. For example, expert systems researchers understood the need for systems that understood their reasoning processes and could generate explanations in a language understandable to its users [SHORT76]. A more detailed description about requirements and the necessity of answer explanation components can be found in [McG96], literature on explanations for expert systems [SWA91], theorem proving explanations [FeMi87] or [BOY95].

2.2.1 Approach

Our approach for answer justification is based on meta-inferencing. While processing a query the inference engine is producing a log-file of the proof tree for any given answer. This proof tree itself is represented in F-Logic. It contains the instantiated rules that were successfully applied to derive an answer. This file acts as input for a second inference run, where answers are produced, that are explaining the proof tree in natural language and by that how the answer to the original query was inferred.

We shortly will illustrate this approach with a sample question. The question asks for the K_a value if *mole* and *pH* of a substance is given

Encoded question:

```
<- exists Ka MPhKa(0.2,3.0,Ka).
```

An extract from the log-file of the proof tree:

```
a15106:Instantiation[ofRule->>kavalueMPhKa;
     dependsOnInstantiation->>{a15092};
     instantiatedVars->>{i(M,0.2),i(PH,3.0),
                  i(Ka,4.99E-6),i(H,0.0010),i(HH,1.0E-6)}].
a15092:Instantiation[ofRule->>phvaluephH;
     instantiatedVars->>{i(H,0.0010),i(Ph,3.0),i(R,-3.0)}].
```

The first statement means the rule *kavalueMPhKa* had been applied at the point in time logged here. Then, the variables *M* had been instantiated by 0.2, *PH* by 3.0, *Ka* by 4.99E-6, *H* by 0.0010 and *HH* by 1.0E-6 respectively.

Rules which are important for justifying results were named like “kavalueMPhKa” or “phvaluephH”. For these rules certain explain rules are formulated which will be applied in the second, the meta-inference run. Frequently the named rules corresponded to important chemistry laws, which would also be found in the text book, while less important rules were typically required for technical reasons, e.g. in order to translate between two alternative representations of the same content, but were not important for the human to understand the solution proposed by the system.

For the first named rule of the proof tree, the corresponding explanation-rule is:

```
FORALL I,M1,PH1,Ka1,H1,HH1,EX1,Ka1R
explain(EX1,I) <-
     I:Instantiation[ofRule->>kavalueMPhKa;
     instantiatedVars->>{i(M,M1),i(PH,PH1),
     i(Ka,Ka1),i(H,H1),i(HH,HH1)}]and
     dround(Ka1,2,Ka1R) and EX1 is
     ("The equation for calculating the acid-dissociation
     constant Ka for monoprotic acids is <b>Ka=[H+][A-
     ]/[HA]</b>. For monoprotic acids the concentrations for
     hydrogen [H+] and for the anion [A-] are the same:
     <b>[H+]=[A-]</b>. Thus, we get <b>Ka= "+H1+" * "+H1+" /
     "+M1+" = "+Ka1R+" </b>for a solution concentration
     <b>[HA] = "+M1+" M</b>.").
```

An explanation-rule is specified by its reference to the rule (*kavalueMPhKa*), by accessing the instantiations of the variables of the answer producing rule, e.g. *M1* accesses the instantiation of variable *M*, and a text referring to these variables which is presented in the explanation.

These explanation rules are applied to the proof tree of our example and produce the following justification output:

- The equation for calculating the acid-dissociation constant Ka for monoprotic acids is **Ka=[H+][A-]/[HA]**. For monoprotic acids the concentrations for hydrogen [H+] and for the anion [A-] are the same: **[H+]=[A-]**. Thus, we get **Ka= 0.0010 * 0.0010 / 0.2 = 5.0E-6** for a solution concentration **[HA] = 0.2 M**.
  - The equation for calculating the pH-value is **ph=-log[H+]**. Thus we get pH-value **ph = 3, H+ concentration [H+] = 0.0010**.

2.2.2 Advantages of the Approach

This two-step process for creating explanations allows for applying the full power of inferencing to generate explanations. For OntoNova, we have developed an entire knowledge base for this purpose. Thus, one may (i) integrate additional knowledge, (ii) reduce redundancies of explanations, (iii) abstract from fine-grained explanations, and (iv) provide personalized explanations.

(i), when the proof tree does not contain enough information in itself, additional information will be needed to create comprehensible explanations. In our context, e.g., the proof tree contains the formulae of the different substances. However, in the explanations the names of the formulae are often necessary to produce understandable justifications. The generation of names from formulae then requires additional information not found within the proof tree.

(ii), in knowledge bases different paths often lead to the same result. For instance the acidity order of two substances may be determined by the pH-values that may be computed from the amount of substances in aqueous solutions – or they may be given by qualitative criteria like the number of oxygen elements in the formulae of the substances. Both criteria lead to the same ordering. As a consequence there are two different explanation paths for the same result. An explanation rule may rank these two explanation paths and ignore one of them when formulating the justification. Thus, redundancies in generated explanation may be reduced.

(iii), often it is necessary to come to different abstractions of explanations. E.g. for the purpose of debugging the knowledge base a very fine grained explanation level is necessary while for an expert a low grained explanation level is sufficient. The single inference steps and thus the proof tree which is a representation of these single inference steps provide the finest resolution of explanations. Additional rules may summarize these steps and may thus produce higher abstracted explanations. E.g. in our domain such rules may summarize the three different steps of a reaction like the (a) decomposition of the substances into their ionic compounds, (b) the composition of an insoluble substance out of these compounds and (c) the precipitation of the resulting substance from the solution as a precipitation reaction.

(iv), the three means (i)-(iii) just mentioned may all be used for generating specialized explanations for different contexts and for generating personalized explanations. Domain experts need other explanations than freshmen or novices and different types of explanations have to be given in different functional situations and contexts like science, research or in AP exam, respectively.

3. Encoding and Knowledge Base Architecture

3.1 Encoding of the KB

For testing purposes in the dry run Vulcan Inc. provided 50 Syllabus questions for each team. The encoding of the corpus (80 pages from [BLB2003]) has been done in three different phases. During the first phase the knowledge within the corpus has been encoded into the ontology and into rules, without considering the 50 Syllabus questions. This knowledge has been tested with some 40 exercises of [BLB2003]. For

each of these exercises an encoded file and an output file by pair have been created. These pairs have been used for automatically testing the knowledge base.

In the next phase the Syllabus questions have been tested with a covering of around 30% of these questions. During this phase the knowledge base has been refined until coverage of around 70% of these questions has been reached. Additionally, in parallel during this phase, the explanation rules have been encoded. In the so-called dry run itself the encoded Syllabus questions have been sent to Vulcan, to test the installed systems. The remaining time to the challenge run then has been used to refine the encoding of the knowledge base and the explanation rules. During the entire process the library of test cases has been extended and used for automatic testing purposes. This ensured stability of the knowledge base against changes.

3.2 Question Encoding

The Syllabus contained two major types of questions: multiple choice questions and detailed answer questions.
Multiple Choice. An example for a multiple choice question is the following:

Which of the following compounds will produce a gas when HCl is added to the solid compound? HCl is a strong acid producing a yellow-green colored gas above the acid solution.
a. $Ba(OH)_2$ (s)
b. $CaCO_3$ (s)
c. $CuSO_4$ (s)
d. Na_3PO_4(s)
e. NaCl(s)

For multiple choices section two different encoding schemas have been used. The first schema encoded the multiple choice questions where our knowledge base could conclude the correct answer. E.g.:

```
//input facts
m1:Mixture[hasComponents->>{"HCl","Ba(OH)2"}].
m5:Mixture[hasComponents->>{"HCl","NaCl"}].
// encoding of the options
answer("A") <- exists P P:GaseousReaction[fromMixture->>m1].
answer("E") <- exists P P:GaseousReaction[fromMixture->>m5].
// query for the correct option
FORALL X <- answer(X).
```

The second schema encoded multiple choice questions where all but one of the options could be excluded by the knowledge base:

```
// all options
possibleanswer("A").
…
possibleanswer("E").
// encoding wrong options
FORALL X wronganswer("A") <- checkequation(ra,X).
…
```

```
FORALL X wronganswer("E") <- checkequation(re,X).
// exclude wrong options
FORALL Answer <-
     possibleanswer(Answer) and not wronganswer(Answer).
```

Detailed Answer Questions. The detailed answer questions, for which an exact answer like the pH-value had to be delivered, have been encoded as direct queries to the knowledge base. E.g.:

```
// give the pH-value of the buffer solution
FORALL Ph <- m1:BufferSolution[hasPHValue->>Ph].
```

3.3 Architecture of the Knowledge Base

The knowledge base has a three level architecture as shown in figure 1.

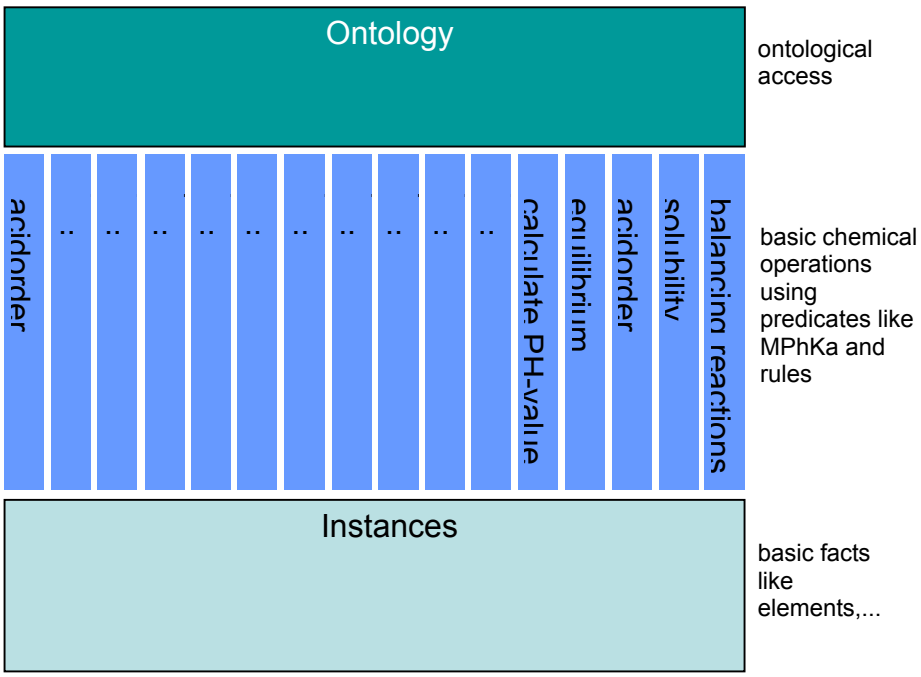

Figure 1: Architecture of the knowledge base

The upper level contains the ontological information about concepts like elements, reactions, substances and mixtures together with their attributes and relationships. This layer provides the domain vocabulary and the domain structure and is target of the queries posed to the system. Let us illustrate this by an example.

The ontology for instance defines the concept *mixture* with attributes *hasPHValue, hasComponent, hasMole, and hasQuantitiy*. These attributes describe the formulae, the moles and the quantities of the components of the mixture. The attribute

hasPHValue represents the pH-value of the mixture. Given such a mixture a query like

```
FORALL X <- m1:Mixture[hasPHValue->>X].
```

results in the pH-value of the mixture.

For the different types of mixtures different rules describe this pH-value. For instance, for strong acid-base titrations the pH-value is described by the following rule which represents one basic chemical operation (variable quantifications are skipped):

```
strongacidstrongbasetitration(Acid,AcidQuantity,AcidMole,
Base,BaseQuantity,BaseMole,PH)
<-
    A:StrongAcid[hasFormula->>Acid]
    and B:WeakBase[hasFormula->>Base] and
    multiply(BaseQuantity,BaseMole,OHMole) and
    multiply(AcidQuantity,AcidMole,HMole) and
    greater(HMole,OHMole) and
    add(AcidQuantity,BaseQuantity,Volume) and
    add(HAfterReaction,OHMole,HMole) and
    multiply(HConc,Volume,HAfterReaction) and
    phH(HConc,PH).
```

Please remark that this rule uses in its body another basic chemical operation by the predicate *phH*.

This basic chemical operation is now attached to the ontology and thus defines the attribute *hasPHValue* of a mixture:

```
M:Mixture[hasPHvalue->>PH]
<-  M:Mixture[hasComponent->>{F1,F2}; hasQuantity->>c(F1,Q1);
    hasQuantity->>c(F2,Q2); hasMole->> c(F1,M1);
    hasMole->> c(F2,M2)] and
    strongacidbasetitration(F1,Q1,M1,F2,Q2,PH).
```

The advantages of this three level architecture are twofold. First the basic chemical operations are nearly independent from each other. This strongly reduces complexity of the knowledge base and the different chemical operations are decoupled and can be developed and tested independently. So each chemical operation is a stand-alone knowledge chunk.

Accessing the values using the ontology on the other hand frees the encoder from knowing all the specialized predicates and on the other hand enables an access to the different information pieces in a way that is much closer to natural language than the predicates of the basic chemical operations.

3.4 Encoding Sensitivity

Related to the methodology discussion above, it is worth to note that answer justification showed some sensitivity against our encoding. This is the logical consequence of our approach. As an example one may consider the following

example of two different encodings for the same problem with, consequently, different justification outputs:

Original question:

```
The pH of a 1.0M solution of HCl is:
     a. 1.0
     b. 0.1
     c. 0.0
     d. less than zero
     e. between 0 and 1
```

Encoded question version a)

```
m1:Mixture[hasComponents->>"HCl"; hasMolarity->>c("HCl",1)].
answer("A") <- m1[hasPHValue->>1.0].
answer("B") <- m1[hasPHValue->>0.1].
answer("C") <- m1[hasPHValue->>0.0].
answer("D") <- exists Ph m1[hasPHValue->>Ph]
                  and less(Ph,0.0).
answer("E") <- exists Ph m1[hasPHValue->>Ph]
                  and less(Ph,1.0) and greater(Ph,0.0).
FORALL Answer <- answer(Answer).
```

Here we defined a mixture of *HCl* to be able to determine the pH value. Then we compared the calculated pH to the given values.

Answer version a): **C**

Justification version a)

- The equation for calculating the pH-value is **ph=-log[H+]**. Thus we get pH-value **ph = 0, H+ concentration [H+] = 1**.
- In the given mixture **the concentration of H3O-molecules(atoms) is 1 M**.
    - **HCl is a strong monoprotic acid** that completely splits into H+ and Cl-. A 1-molar mixture of HCl has therefore a 1-molar concentration of H+ and Cl- molecules.

Encoded question version b)

```
answer("A") <- phvalue(1,"HCl",1).
answer("B") <- phvalue(0.1,"HCl",1).
answer("C") <- phvalue(0,"HCl",1).
answer("D") <- exists Ph phvalue(Ph,"HCl",1)
                  and less(Ph,0.0).
answer("E") <- exists Ph phvalue(Ph,"HCl",1)
                  and less(Ph,1.0) and greater(Ph,0.0).
FORALL Answer <- answer(Answer).
```

In this encoding approach, we used the basic operation *phvalue* to compute the pH value of the given substance in an aqueous solution. Then we compared the calculated pH to the given values.
Answer version b): **C**

Justification version b)

- The pH-value of an acid is calculated by **ph=-log[H+]**. For **HCl** and the given data this leads to pH-value **ph=0,**H+ concentration **[H+]=1,**Molarity **M=1.**
    - As **HCl** is a strong acid (see page 122, table 4.2) and fully dissociates in water, the calculation of [H+] or molarity M is based on formula **[H+]=1*M**. With the given data this leads to H+ concentration **[H+]=1,** Molarity **M=1**
        - § molecule **HCl** splits into **cation H** and into **anion Cl**

4 Challenge Results

After having encoded four chapters of [BLB2003] the competing systems have been sequestered by Vulcan Inc. and every team got 100 challenge questions which had to be encoded in each teams own representation formalism. After 14 days the encoding documents have been sent to Vulcan Inc. and each teams' system had to process these questions. In the following the results of this challenge run are presented.

4.1 Scores

In general, all teams showed up with similar good scores in the challenge run. There was no time limit for the answering process, so that the teams' systems did not take an AP exam under original conditions. The aim of the run was to test the general ability of the systems to answer questions of a complex domain.

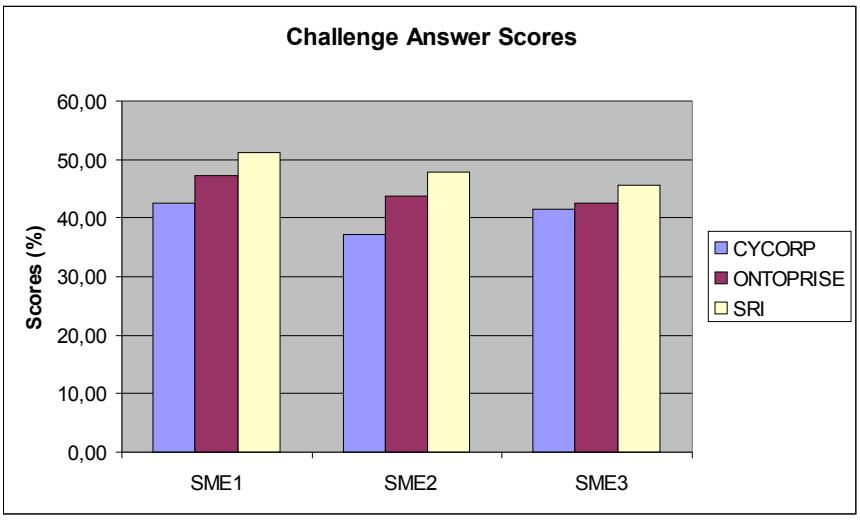

Figure 2: Challenge answer scores

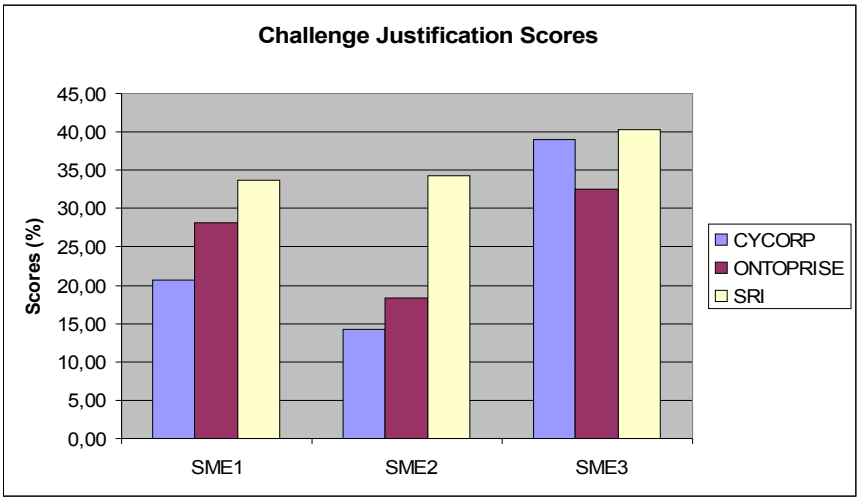

Figure 3: Challenge justification scores

4.2 Fidelity

The project participants joined a fidelity committee which had the task to judge about the fidelity of the encodings. Most of the encodings were not matter of fidelity concerns. Nevertheless, some encodings not directly encoded the original information of/in the challenge question. Most violations simply ignored irrelevant information that was not encoded. Some cases encoded the formula of substances instead of their names and vice versa, as the substance name was not available but the formula could be still processed to derive an answer. As far as the question was encodable and the question's subject was covered by the KB, no major fidelity conflicts occurred. An example for a fidelity concern was stated for encoding the multiple choice question 12. In this case the knowledge engineer has interpreted that if a substance reacts with oxygen the corresponding reaction is a combustion reaction. Thus additional knowledge from the knowledge engineer is used here and a fidelity concern occurred:

```
When methane, CH4, gas reacts with oxygen, the following
     changes occur
burn("CH4"). // has been interpreted as combustion reaction
```

4.3 Coverage

Coverage means to what extend the questions could be encoded in the representation language. Coverage of challenge questions strongly correlates with the type of question. We had strong coverage in sections that we experienced in the sample questions for training purposes. We had 100% coverage in the MC-section and some

90% in the Detailed Answer Section (DAS). Sparse coverage was shown with unknown question types like in the Free Answer Section, where we only covered some 20% of the questions.

4.4 Brittleness

One outcome of the project is a brittleness taxonomy which classifies failures to different failure classes. As a summary one can state that the major part of failed answers belongs – in the case of OntoNova – to brittleness type "Modeling Error" (knowledge engineer fails to capture domain information properly in their modeling). This is true for answer generation and answer justification, respectively.

For example, we received the correct answer for the multiple choice question no. 19 (MC 19), but lost points for missing explanations. Though the explanation rule in general existed, but there was a typo in the rule that prevented the firing of the rule. There exist some other examples, where explanation rules were just missing, like in MC 2. MC 11 is an example, where parts of the knowledge base had slight bugs, but explanations fit pretty well. MC 23 is seen as an example where the corresponding part of the domain to answer the question was just not modeled at all. All these examples are modeling errors.

The second, major class of errors was the "Unexpected Question Type". This is especially true for the Free Form Section. The reason for the inability to answer those types of questions is based in the modeling of the knowledge base. In these cases the knowledge engineer did not forget to model a part of the domain, but lacked in modeling the different kinds of usage and application of the knowledge base. So in these cases one could often enough observe that certain operations could be done in a certain direction, but could not be operated in the other direction. For example, the calculation of the pH-value could be done by given ka-values but **not** vice versa:

```
Ascorbic acid, H2C6H6O6, is a diprotic acid
     with a Ka1 value of 8.9 x 10-5.
The pH of a 0.125 M solution of ascorbic acid is 2.48 and the
     concentration of C6H6O62- is 1.6 x 10-12 M.
Determine the value of Ka2.5
```

4.5 Performance

Processing performance of OntoNova has been much faster than its competitors, depending on the systems used to perform the challenge run. Our system for the official challenge run required about an hour to process the encodings. For a second, optional and not sequestrated run we slightly improved the knowledge base and brought down processing time to under 10 minutes.

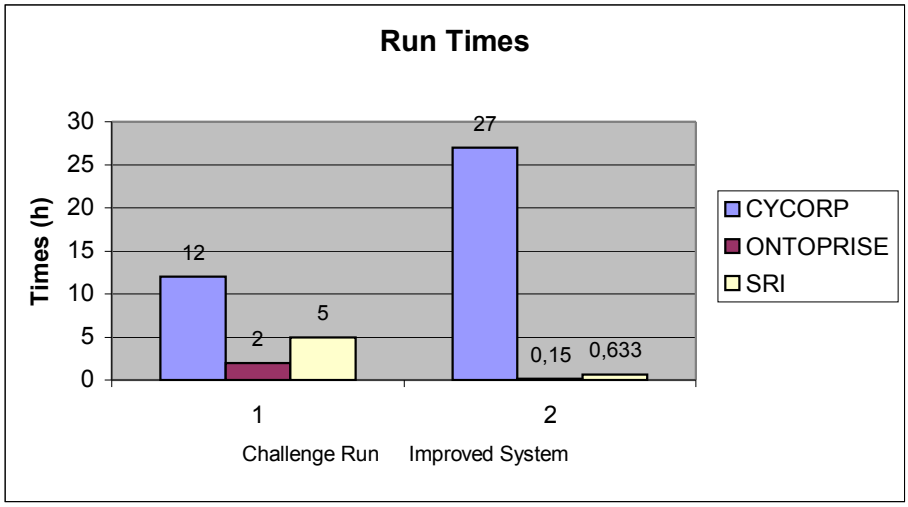

Figure 4 : Run times

5 Conclusion

From the development of OntoNova we gained a range of insights valuable for traditional knowledge systems – as well as for the Semantic Web.

1. F-Logic appeared as a representation language which is very well suited to be used in complex rule intensive domains. One obviously required addition to this language could be a package construct which divides the knowledge base into different parts. Our addition of namespace constructs is one step into F-Logic towards this objective (cf. [Onto03]).
2. Our inference engine OntoBroker has performed very well in answer and justification generation.
3. We also noticed that we have to improve our knowledge modeling environment by a better support for rule editing and rule debugging.
4. Another insight was that at the end of the knowledge base modeling domain experts should be able to extend and modify the knowledge base on their own. For that purpose a two-step process will be necessary. First the basic building blocks (like basic representation of formulas, basic chemical operations etc.) must be developed by a knowledge engineer in a general development environment like OntoEdit. In a second step such an environment must be specialized for a tool which comprises the special view of the domain expert and gives him the elementary building blocks as modeling primitives.

Concerning the evaluation results Ontoprise finished second according to the scores in answer generation and justification, respectively (see 4.1 Scores). This result seems very formidable, because we used just 2 person years instead of the 4 person years consumed by the other two teams and by that showed up with half the costs for encoding. Compared to other teams, our encodings are very compact and do not

require specific hardware resources to be processed (experiments were performed on a standard current version of a Linux machine).

Processing performance of OntoNova has been faster than its competitors by factors between 5 and 30, depending on the systems used to perform the challenge run. The challenge run could have been done in some 3-4 minutes using a special twin-server that would have done answer generation and justification in parallel instead of sequentially. Additionally, we could have applied several optimization strategies to improve performance, so that there is much more potential for improvement – and will be needed for the Semantic Web to scale in size, scope and trust – matters advanced by the OntoNova in the Halo Pilot Project.

Acknowledgements

The research described in this paper has been financed by Vulcan Inc. We gratefully acknowledge their support as well as the fruitful interaction with all the other members of the Halo Pilot Project, in particular Noah Friedland (Vulcan) & Oliver Roup (previously at Vulcan), David Israel & Vinay Chaudhri (SRI), Bruce Porter (Univ. of Texas), Peter Clark (Boeing), G. Matthews, Michael Witbrock, Doug Lenat, (Cyc).

References

[Ang93] Jürgen Angele: Operationalisierung des Modells der Expertise mit KARL. DISKI, Infix Verlag, 1993.

[BLB2003] T.L. Brown, H.E. LeMay, B.E.Bursten, J.R.Burdge: Chemistry The Central Science, Ninth Edition, Prentice Hall, 2003.

[BOY95] Robert Boyer, Matt Kaufmann, and J. Moore. The Boyer-Moore Theorem Prover and Its Interactive Enhancement, Computers and Mathematics with Applications, 29(2), 1995, pp. 27-62.

[DEFS99] S. Decker, M. Erdmann, D. Fensel, and R. Studer. Ontobroker: Ontology based access to distributed and semi-structured information. In R. Meersman et al., editor, *Database Semantics: Semantic Issues in Multimedia Systems*. Kluwer Academic, 1999.

[FeMi87] Amy Felty and Dale Miller. Proof Explanation and Revision. University of Penn, Tech. Report MSCIS8817, 1987 http://cm.bell-labs.com/who/felty/abstracts/proof87.html.

[FHK97] J. Frohn, R. Himmer, P.-Th. Kandzia, C. Schlepphorst: How to Write F-Logic Programs in FLORID. Available from fpt://ftp.informatik.uni-freiburg.de/pub/florid/tutorial.ps.gz, 1997.

[GPQ97] Hector Garcia-Molina, Yannis Papakonstantinou, Dallan Quass, Anand Rajaraman, Yehoshua Sagiv, Jeffrey D. Ullman, Vasilis Vassalos, Jennifer Widom: The TSIMMIS Approach to Mediation: Data Models and Languages. JIIS 8(2): 117-132 (1997)

[GRS91] A. Van Gelder, K. A. Ross, and J. S. Schlipf. The well-founded semantics for general logic programs. *Journal of the ACM*, 38(3):620–650, July 1991.

[KL86] M. Kifer and E. Lozinskii. A framework for an efficient implementation of deductive databases. In *Proceedings of the 6th Advanced Database Symposium*, pages 109–116, Tokyo, August 1986.

[KLW95] M. Kifer, G. Lausen, and J.Wu. Logical foundations of object-oriented and framebased languages. *Journal of the ACM*, 42:741–843, 1995.

[McG03] Deborah L. McGuinness and Paulo Pinheiro da Silva. Inference Web: Portable and Shareable Explanations for Question Answering ". In the Proceedings of the American

Association for Artificial Intelligence Spring Symposium Workshop on New Directions for Question Answering. Stanford University, Stanford, CA. March 2003.

[McG96] Deborah L. McGuinness. 1996. Explaining Reasoning in Description Logics. Ph.D. Thesis, Rutgers University, Technical Report LSCR-TR-277.

[Onto03] How to Write F-Logic Programs
http://www.ontoprise.de/customercenter/support/tutorials

[SHORT76] Edward Hance Shortliffe. Computer-Based Medical Consultations: MYCIN. Elsevier/North Holland, New York, 1976.

[SWA91] W. Swartout, C. Paris, and J. Moore. **"**Explanations in Knowledge Systems: Design for Explainable Expert Systems". In IEEE Intelligent Systems, June 1991. http://www.computer.org/intelligent/ ex199/x3058abs.htm**.**