# Design document of the ISSA parser

Steven Bos and Remco Jansen

Delft University of Technology, Faculty of EEMCS
steven@stevenbos.com entreco@gmail.com

**Abstract.** This paper describes our early finding to a new approach on parsing a sentence and extracting semantical information. The Infinite Sentence Semantic Analyser (ISSA) parser uses both syntactical information based on a PCFG, and a semantic grammar (SG). The parser uses the roots of a syntax parse tree, to deteremine multiple sentences in one. It appears that this shallow analysis in combination with a SG is sufficient to extract basic understanding of a sentence in the 'EWI-project' domain. Results show that the ISSA parser is able to extract some information when it faces unknown words by its context. More importantly it can extract all intended meaning of a domain model. The most time consuming tasks are finding quality rules for the semantic grammar and testing the grammar. The parser could easily be extended with a WordNET feature to further increase the understanding, because WordNET references on lexical level are already present in our SEMISUSANNE-based PCFG. Other extensions like the usage of 'parts of speech' and a automatic learning engine based on Hidden Understanding Models could also increase the meaning extraction of this parser

## 1 Introduction

Extracting semantical information is key in the domain of Natural Language Understanding. Fields and applications who depend on the accurate (deeper) meaning of a sentence are various. For example in the field of Machine Translation, an automatic translator could overcome the language barrier when searching (scientific) information on the Internet or could support multi-lingual negotiations where instant translation is absolut crucial. Another field is that of Dialogue Systems. The 'TU Delft EWI-project' is an application where the 3D virtual avatar named 'EWI' is equipped with a dialogue system so that it (ultimately) can communicate on a very basic human level.

Previous work has shown that there currently is no single semantical analyser that suits all NLU fields and applications [1]. The ISSA parser we developed is no exception, it is designed for the EWI-project and therefore focusses on the field of Dialogue Systems. Within this field, one of the widely applied approaches of doing semantic analysis is with help of semantic grammars (SGs) [2]. One of the great advantages of SGs is that the semantics of a sentence can be directly read of the parse tree. One of the major drawbacks is the limited reusability of a SG, it is knowledge domain dependent [3]. The EWI avatar will initially

operate in one knowledge domain, so this drawback shouldnt pose a problem and thus justifies our choice for SG over approaches like 'Template matching information extraction' or 'syntax-driven semantical analysis'. The latter has a theoretical advantage that allows it to capture domain independent information with one grammar. Unfortunately this approach requires a syntax grammar extention, with semantic annotations on a rule-to-rule basis. Not only is this time consuming job more suitable for linguistic experts, it is still very theoretical and has its own trade-offs.

We acknowledge the importance of knowing the syntax of sentence; tests by Gibson [in 1998] show that humans do look at the syntax when extracting the meaning. We therefore looked for a way to use the syntactic information and combine it with a semantic grammar to extract more accurate meaning. This resulted in the ISSA parser who's workings and algorithms are desribed in chapter 3. Information on the workings and background of the PCFG parser, which produces our syntactic information, can be found in the paper of Adriaan de Jong [in 2008]. In the next section we describe the other input component of our ISSA parser; the semantic grammar.

## 2 Developing a Semantic Grammar

The most important input for the ISSA parser is the semantic grammar. Building a semantic grammar is a delicate proces; you need to find rules that capture knowledge in a specific domain. The following subsections explain how we found our rules.

### 2.1 Development approach

When doing some research on building semantic grammars, we found an approach which is stated by Gavalda as the 'the traditional Semantic Grammar development approach' [3]. It consist of 5 phases, namely data collection, design domain model, development kernel grammar, expansion grammar coverage and deployment. Because our ISSA parser approach is a experimental one, we thought to only focus on the first three steps. Step five is just building a ready-made package for an application. Step four, which is what makes the system future proof (because a kernel grammar is usually too limited to cover the domain) requires a study of its own. Gavalda builded a system called GSG [4] which semi-automatically learns new semantic mappings. She mentions in her paper that experimenting with Hidden Understaning Models might be a better way to learn mappings fully automatically by just giving enough training data. The approach with HUMs (also referenced as 'stochastic models') was also mentioned independently by our instructor dr. ir. Pascal Wiggers. Next we discuss our three steps to build our kernel grammar.

### 2.2 Phase1: Data collection

We started with a domain investigation. Fortunately, Remco helped first year students growing accustomed to the way things work on TU Delft as well as answering questions. We then sat down and made a list of 5 questions, which the semantic parser should initially be able to parse. The construction of the ISSA parser is part of the EWI-project and therefore the parser should also be able to handle pragmatics. We started by including 2 basic sentences, to make our total data collection 7 sentences. The results in presented here:

**1. Statement**

- 'My name is Steven' or 'My name is Remco'

**2. Request**

- 'What is the difference between TU Delft and TU Eindhoven'
- 'What is the difference between MKE and Computer Science'
- 'When does the first semester start?'
- 'What courses are given at the tudelft'

**3. Statement+Request**

- 'I like Mathematics, what faculty would you recommend'

**4. Opening**

- 'Hello Computer'

### 2.3 Phase2: Domain model

The next step is distinguish key concepts in the sentences of our data collection. Examples of the found concepts were: location, person and time. After that we developed a model with hierarchical relations between key concepts and made up new higher level and lower level concepts like: point (in time) and possible sentence starts like what, where, how, etc. Under the root of a tree is a dialogue act according to the DAMSL scheme by Allen and Core, 1997 [**?**] like a request or a fact (statement/claim made by the speaker). When building the domain model we realised that we used a wrong name convention so that it might appear that a concept 'name' of a university is the same 'name' as of a person, they are of course different. A better name convention would be universityName and personName. The domain model has four levels: intial (lvl 1),continuation (lvl 2),concept (lvl 3) and terminal (lvl4). We present here a small part of the domain model of all levels:

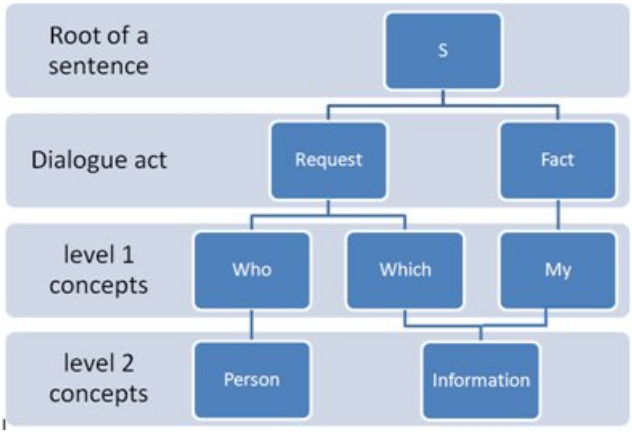


**Fig. 1.** DM part: level 1,2

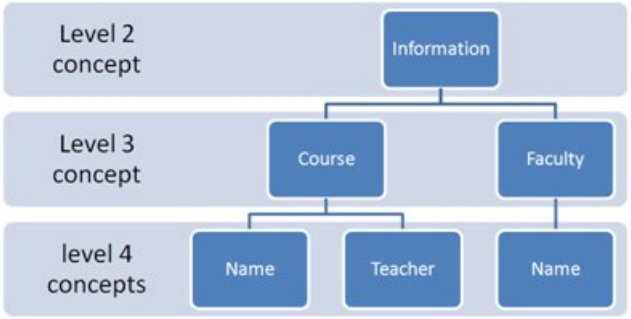


**Fig. 2.** DM part: level 2,3,4

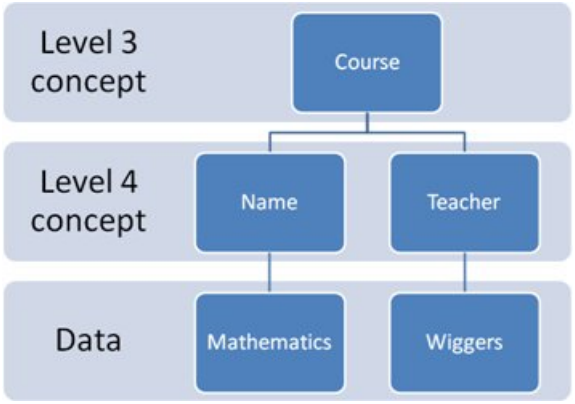


**Fig. 3.** DM part: level 3,4,data

### 2.4 Phase3: Kernel grammar

With the domain model we can infer production rules out of the relations. Every concept in the domain model must be reached. Note that the wrong name convention also appears in the rules (to keep it consistent). All the rules together form the kernel grammar (the basic grammar) in contrast to the user grammar which complements the kernel grammar with rules found during for example grammar expansion activities. The following is a part of our kernel grammar. The complete grammar can be found in our application directory. Rules:

1. [information] → [location];[person];[faculty];[course]
2. [location] → [university];[person];[city]
3. [person] → [student];[teacher];[EWI];[dean]
4. [time] → [course];[semester]

Four different types of rules have been inferred. The first kind of rules are initial rules. Which allow for an access point in the grammar. They should quickly recognize parts of a sentence and determine the kind of dialogue act. Next there are continuation rules. Once an initial rule has been found, the continuation rules can help to analyse the sentence on a deeper level. The continuation rules ultimately lead to concept rules. Those are rules about the different concepts defined in our domain model. Concepts such as 'university' and 'course' are key concepts in our domain model and the concept rules describe how these objects are composed of smaller parts. For example a 'university' consists of a name, facilities and a city. Of course, in real life a university consists of many more things. But then it should also be included in the domain model, which could lead to very large models. Therefore it is important to think about introducing new concepts and the relevance of it in a specific knowledge domain.

Finally we have the terminal rules. Terminal rules contain the highest level of detail. The terminals are actual words that define our grammar. They define for instance that a name of a course can be 'mathematics'. While the initial rules are quite abstract in the sense that they refer to general objects like 'information' or 'location'.

## 3 Semantic Grammar Parser Prototype

Goal of the semantic parser is to extract meaning of the words that are being used by the user. In an ideal situation the semantic parser, ISSA, knows the meaning of all possible words and also knows when people are being for example sarcastic, or joking etc. For a computer to know all this information is needed about the context. The ISSA parser is not context aware, it is however, able to extract as much information as possible from a sentence, using a SG and syntactic information.

### 3.1 Development approach

Key to creating a semantic parser is the ability to extract information from sentences. Since a relatively small amount of words are defined in the grammar, the most important feature of the semantic parser should be the ability to deal with incomplete information, or so to say, deal with unknown words. Since we cannot add all information in the rules, we need to extract information from the sentences in a clever way. Ultimately the EWI-project should be able to take part in a conversation. In order to comply with this, goal of the prototype is to parse sentences real-time. By looking at one word at a time, it is possible to parse a complete sentence by looking at a single word. As soon as we find something interesting, we can parse the remainder of the sentence. Thus we can increase the level of details by going deeper in the semantic parse tree. There is a small trade-off in our real-time approach and that its how multiple words that belong together (ie. with spaces) like 'TU Delft' are parsed. We choose to use a 'space' to indicate the next word. Momentarily TU Delft is written as tudelft to be recoqnised. A simle solution could be a pre-parser who translates words with spaces to words without spaces.

### 3.2 General workings

Like mentioned in Chapter 2, the rules have a different level op complexity. When a sentence is being parsed, the first thing to do is to find an entry rule to begin traversing the semantic grammar. After that we try to find a terminal, assuming that the terminal belongs to the entry rule of the grammar. This technique is generally called template matching and rule fireing. The part of the sentence from the entry to the terminal is being replaced by a general concept extended by the remaining words of the sentence. This newly created sentence serves as the beginning of a new sentence.

We try to look at one word at a time and try to find appropriate rules. When we find an appropriate rule, we know what type of dialogue act we are dealing with and we also know what rules might match. We can look at the rest of the sentence, and try to extract more information from the remaining sentence. This process can be repeated until no more information can be extracted. The figure below shows the process of analysing the sentence.

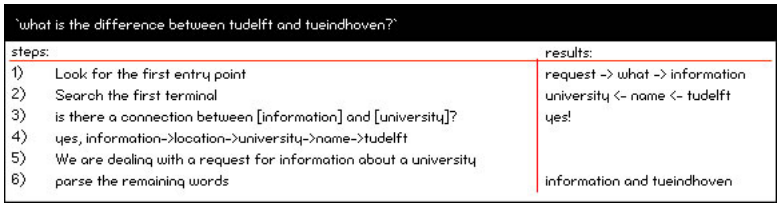


**Fig. 4.** First step: 'What is the difference between TU Delft and TU Eindhoven'

Basically we look for parts within a sentence that have been defined in our grammar. When we see an entry point and a terminal we can make a connection. In the example above we can make a connection from a request for information to a university with the name 'TU Delft'. From the university concept we also know what parts make up the university, so we can look for more information in the remaining sentence to complete the university slot. When no information about the university is present, we continue our search.

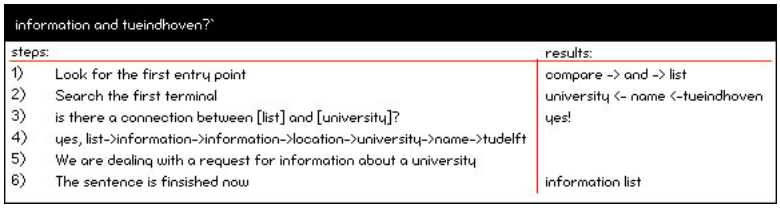


**Fig. 5.** Second step: '[information] and TU Eindhoven'

Now, we arrive at a new entry point. We leave the slots of the university empty, and continue from the word 'and' to conclude that we are now comparing things. We find a new terminal ('TU Eindhoven') and look for information to complete the TU Eindhoven concept. However, at this point we see that there are no more words in the original sentence, so we stop searching.

### 3.3 Prototype Algorithm

A visual representation of the algorithm is shown below.

On top you can see the part of the GUI that serves as the user input. The user input is passed through the syntactic probabilistic CYK parser (more details about the parser can be found in the paper of Adriaan de Jong). Based upon the syntactic information the number of sentences is determined. Now all of these sub sentences are being analysed by the semantic parser. The semantic parser 'consults the Domain Model' and goes as deep as possible in parsing the semantic meaning. When no more information can be found with the present rules, the semantic analyser produces a frame-based output, which can be sent to a dialogue manager.

### 3.4 Screenshots

The figure below contains information about the different aspects of the GUI. The area marked with 1. is the parse tree generated by the CYK Parser (see Adriaan de Jong [2008]). Area 2. contains the user input and a clear button. The area marked by 3. is the output frame.

The figure below shows a screenshots of the ISSA parser in action.

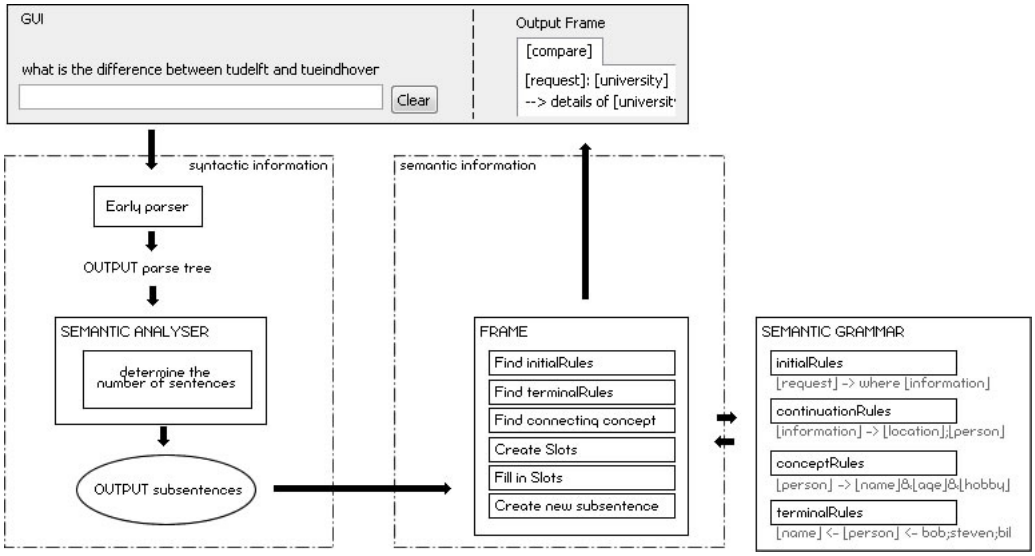


**Fig. 6.** The algorithm

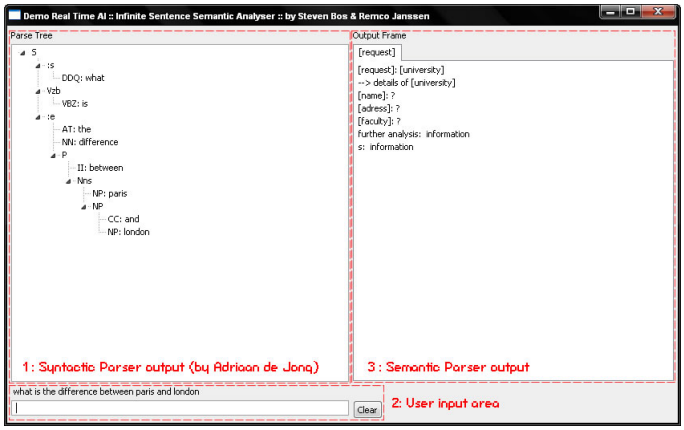


**Fig. 7.** The GUI and its main elements.

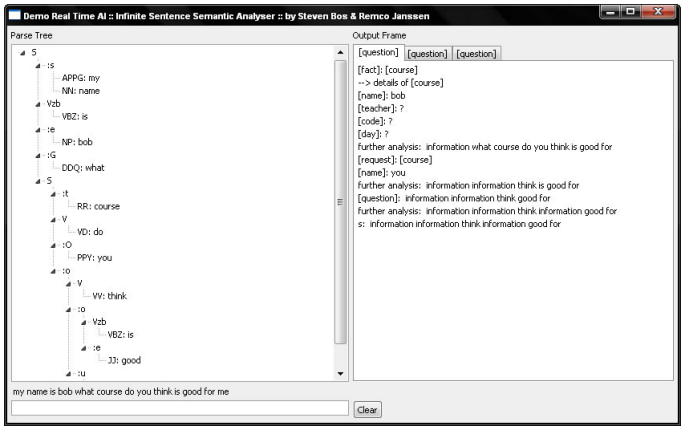


**Fig. 8.** The ISSA parser in action.

The output in the frame on the right should the kind of output a potential Dialogue Manager could expect as an input. Based upon the information in the output frame such a Dialogue Manager where to execute queries on Databases and to select answers to reply to the user.

## 4 Results

Testing is important to get an idea of the functionality of our ISSA parser. It is important to test how the ISSA parser behaves, performs and what it's capabilities are. Within this chapter we try to figure out how the parser performs, what its limitations are and where improvements can be made.

### 4.1 Test procedure and Test set

To test the workings of the ISSA parser we use three scenarios. Although we developed the parser for the 'EWI-project' domain, we also state questions for different domains as well as some known and unknown small talk (pragmatics). The test is not bias-free nor is it complete to compare with other parsers. The test set compromises of the following questions:

**1. Questions that are in the FAQ**

- The FAQ questions are stated in Chapter 2.

**2. Questions that are in the FAQ but deviate one word**

- I like Physics, what faculty would you recommend
- Hello computer, my name is Jan

– When does the first lecture start

**3. Questions that are not in the FAQ**

– I like to apply, how can I do that
– How are you doing
– Tell me more about that

### 4.2 Results

**Scenario 1: Questions that are in the FAQ** The first set comprises of questions that are expected to be posed by potential users. The ISSA parser is designed to handle these questions and therefore should be able to capture the semantics of them. Aslo important to recoqnise is the type of dialogue act. The figure below shows an example of the frame output for the sentence: 'my name is Steven'.

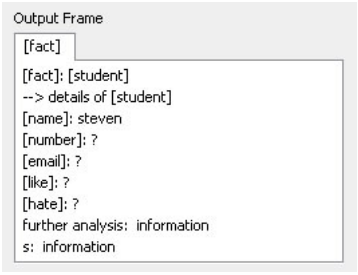


**Fig. 9.** Output of the sentence: 'My name is steven.'

As can be seen from the example above, this sentence is parsed correctly by the parser. Correct in this case means that the ISSA parser is able to determine what kind of dialogue act we are dealing with and to categorize the sentence. It determines that what is being said is a fact(statement). And the fact is that there is a student whose name is Steven. This, of course is not true in general, but it is true when looking at this sentence from our domain model perspective. Our domain model assumes that the person that is speaking to 'EWI' is a (potential) student.

Our domain model is created around the questions within the FAQ test set, there are some difficulties with the questions. Below is the output for the sentences: 'What courses are given at the tudelft?' and 'I like mathematics, what faculty would you recommend?'.

The first output frame ('What courses are given at the tudelft') is lacking some important information. It can be seen that we are looking for information about a university called 'tudelft' but there is no indication about what we would

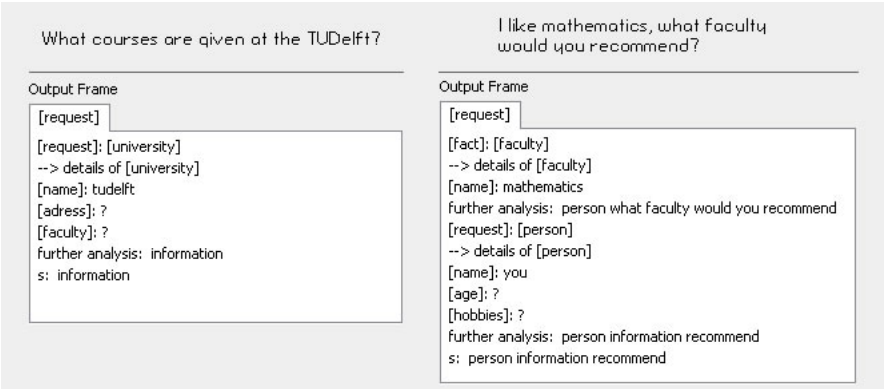


**Fig. 10.** Output of two sentences that are not interpreted well

like to know from that specific university. Reason for this loss of information is that the word 'courses' is not in our semantic grammar. The singular version 'course' in fact is, but in our semantic grammar we only defined singular words and there is no implementation that takes plural words into account. This causes the sentence to be categorized by some kind of request for information about the university.

The figure on the right misses some important information due to other issues. The word 'faculty' is a part of our semantic grammar. However, it does not occur in the SEMI-SUSANNE based syntactic grammar. Therefore this sentence cannot be parsed by our syntactic parser, and syntactic information about the sentence is missing. To be able to parse at least something (like humans do aswell!), we pass the complete sentence through our semantic parser. It now has no information whatsoever about the strucure of a sentence. What we get is one very long sentence, without recoqnising smaller sentences.. The first part is classified as being a fact about the faculty mathematics. The remainder of the sentence (in this case 'would you recommend') is being classified as a fact. In advance the idea would be that this kind of sentence would been split up. Because our current semantic grammar is relatively small (compared to our syntactic grammar) we can only process a limited number sentences when there is no syntactic information. For example in an extensive grammar there could be rules such as:

1. [person] → [name];[age];[hobbies];[likes];[dislikes]
2. [likes] → [course];[person];[city]

With such rules it might be possible to draw the conclusion that we are dealing with a fact about a person, and a request for information.

**Scenario 2: Questions that are in the FAQ but deviate one word** Next part of the test set, are sentences that look a lot like the sentences defined

within the FAQ but who contain a single word that is not defined within our grammar. We already know that our grammar does not deal with singular and plural differences and that our semantic grammar is relatively small. Both issues can be resolved relatively easy by spending more time in developing the domain model and its corresponding rules. However, it is impossible to include every excisting word or self-invented word, so unknown words are bound to show up. It is interesting for us to see what happens when a completely unknown word is being used. Below is the output for sentences containing completely undefined words.

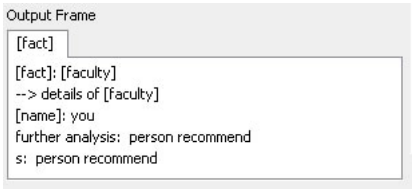


**Fig. 11.** Parsing an unknown word in our grammar

In this case, the sentence has no terminal word so it cannot find a complete path through our semantic grammar rules. However, this issue is not inherent to the way we parse the sentences. The only reason for this flaw is that for practical reasons mentioned earlier, no syntactic part of speech rules have been implemented in our domain model. So a semantic grammar, developed with the knowledge of linguistic experts, might cause this problem to be solved. Unfortunately we where not able to test this hypothesis due to time constraints. We can imagine a rule that specifies for instance that when we see a fact, that the corresponding information is always the complete ':o' (see SEMI-SUSANNE manual, [5] ) element following the fact.

Although, some information is lost when unknown words are being used, the semantic parser can parse unknown words. Especially when they are combined, the semantic analyser can parse acceptable results. Compare for example the two sentences: I like physics, and I like the course physics. The first causes the parser to lose essential information. While the second still produces acceptable results because it now knows it is dealing with a course. Therefore, in the case where information is lost, the dialogue manager can ask for additional information to help the classification of the unknown term. Moreover, since the parser knows what a course consists of, it can ask direct questions about the course to gain additional information.

**Scenario 3: Questions that are in not in the FAQ** Finally we take a look at sentences that are outside the scope of our domain model. So sentences that have no direct relation to our model. First we observe the sentence: 'I would like

to apply, how can i do that.' We have no rules for filing an application. Neither do we have rules with regard to questions about certain procedures. The figures below show the results for our frame:

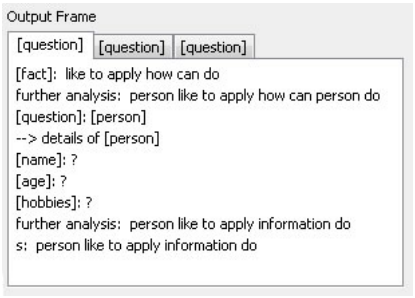


**Fig. 12.** Questions beyond the scope of our domain model

Even though we are dealing with concepts and questions that we did not expect when we designed our domain model, it is clear that some basic concepts indeed are being recognized. The result for the indicated sentences is that there is a person that wants to apply (whatever that might mean) and the person wants information about do (whatever 'do' might mean). In case of an severely more detailed semantic grammar, even more information might be recognized allowing a dialogue manager to ask more specific questions about the unknown things in the sentence. Basically we already see that it is a good idea to implement many rules with different levels of abstraction (high level semantic mappings). When a lot of details are known, those details should be captured by 'concrete' rules. On the opposite, when there is a lot of missing information, the 'abstract' rules should capture the most basic elements.

## 5 Recommendations

The ISSA parser and its (semantic) performance are in its most early stage. Our approach to use specific high level syntactic input based on PCFG and combine it with a semantic grammar has proven to be a working combination, but how good it is compared to proven approaches, like pure semantic grammars or the classical syntax-driven semantic analysis, has yet to be seen. In the short development process of seven weeks we encountered several opportunities to improve the functionality and performance of the parser. The following list of ideas might prove to be useful for teams who like our approach and would like continue working on it.

1. **Improve meaning extraction** The parser is dependent on two components; the syntactic input and the semantic grammar. To improve the syntactic in-

put, we like to refer to the paper of Adriaan de Jong. To improve the semantic part we suggest:

- Redo steps 1 to 3 of the classical semantic grammar development to capture more domain data, which results in more rules (knowledge) in the kernel grammar and allow more sentences to be recoqnised.
- The WordNet information can be interesting because a kernel grammar simply cannot capture all the rules to parse a random sentence in a complex domain. WordNet can solve this partly by providing a database with synonyms. Even better; WordNet has semantical information on several levels! see Chapter 16 (lexical semantics) of the book by Jurafsky[2000]. We havent worked it out because of other priorities with the parser, but our raw idea was the following: First a sentence pops up with one unknown word. The sentence can be only be parsed if this word is known and this word does not effect the meaning of the whole sentence. Second the parser looks up the word in WordNet or another thesaurus and see if the word is a synonym for an existing word in the kernel grammar. Third, the parser has to look up in a exception 'grammar' to see if the synonym might change the meaning of the sentence. If not then the sentence can be parsed and the unknown word can be added to the user grammar. A well developed exception 'grammar' is crucial. One of the most important rules of this grammar will be that a synonym is the same part of speech as the existing word where it is a synonym of and thus avoid choosing the wrong homonym. The WordNet feature we spoke about here will probably only work for single unknown words (the exception 'grammar' would become to big otherwise, because two words are more likely to change the meaning of a sentence). Nevertheless it makes the parser not hopeless when it finds an unknown word. This automatic feature is not the same as the (user-interupting) learning engine feature mentioned below.
  On a side note: Note that we didnt do anything with the fact that WordNet contains semantical information on several levels and thus based on a WordNet entry more information then in our kernel grammar can be gathered see page 619 of Jurafsky [in 2000].
- The shallow analysis of the syntactic input should be developed further. Currently it only uses few, simple rules. But when more and more complex rules are added, the syntactic analysis could prove to be even more valuable to the semantic grammar.

**2. Improve functionality** The parser was hastly build and only minimal functionality was implemented. Great opportunities are:

- Usability. Develop a GUI that is friendly out of the box.
- Option to see n-best trees for debug reasons, with n to be chosen by user.
- Extend parser to also look in user grammar. The user grammar contains new semantic mappings (stage 4 of the classical semantic grammar development[3]). Of course a learning engine who detects similarities and verifies this with a developer or end-user must also be developed. This

semi-automatic system may be similar to the GSG engine [4]. A full automatic learning engine is also possible. Oppertunities here are Hidden Understanding Models (see examples, [3]). The learning engine could be a subsytem of the 'EWI dialogue system'.

3. **Improve performance and code enhancement** Although output is generated real-time, the code is not optimized. Optimized code is especially usefull when the parser is part of the bigger system EWI-system, where maybe multiple sentences are given as input.
4. **Collaborate with linguistic experts** Our work has been made from a computer scientist perspective, inspired on the research of Jurafsky and Gavalda. The results mentioned should also be verified by linguistic experts.

## 6 Conclusions

Meaning extraction with semantic grammars and syntactical information works for our small data collection. It is however, a very tedious job to find enough rules to be able to cover the entire 'EWI-project' domain. Already we faced problems with overhead. A structured way of working is absolut neccesary to deal with the grammar complexity. Validation, verification and evaluation (=testing) are not mentioned in the classic SG development approach, but these phases inevitably show up before or after each classical step in order to produce quality rules. They consume major chunks of time when developing a SG.

The ISSA parser and its way of converting SG and syntactic input is really in its earliest stage. The raw power of combining the two is by far not exploited, especially using parts of speech. Collaboration with a linguistic expert could be a great help to improve this syntactical part.

The robustness of the parser was tested with three scenario's: sentences of the data collection, sentences which deviate one word and which deviate almost in total. Results show that the parser can handle most of the sentences of scenario 1. The sentence it couldnt handle was because of the absence of a plurar form of the word. This again emphasizes the need for a larger SG. The second scenario could also be handled very well, depending on the importance of a word in a sentence (a noun or verb is often more important than a adjective). We recommend using the WordNet extension to at least recoqnise the word on a word level and have a chance that it is a synonym for a recognised word. Another possibility would be to develop a learning engine where rules are learned automatically (by means of Hidden Understanding Models) or semi-automatically (by means of automatic rule inference and verification by humans). The last scenario is especially interesting. Results show that the parser can extract some knowledge even with a small grammar. We expect results in this category to improve automatically when the grammar is extended, but cannot say anything on how usefull it will be.

As a final remark: the working of the ISSA parser depends on the SEMI-SUSANNE PCFG. When a word is not recoqnised by the syntactic parser it provides an empty syntax tree. The semantic parser then treats the sentence as one and is unable to detect multiple sentences in one.

## Acknowledgements

This project is part of the 'EWI multimodal conversational agent' project. The full scope of the project includes speech recoqnition, natural language understanding and dialogue management. For the master course 'Realtime AI and Speech Recoqnition' we choose to work on the NLU domain. Together with the syntactical parser of Adriaan de Jong, the ISSA parser is able to cover this domain in its greatest simplicity. We appreciate the collaboration with Adriaan. We also like to thank the initiator of the EWI project and our mentor dr. ir. Pascal Wiggers. Many thanks also go to the founders of the GSG approach; reseachers Marsal Gavalda and Alex Weibel. Their work was highly inspirational for us.

As a final note: this paper only described our research on the subject of semantical parsers and extracting information. Ethical issues, which inevitably will rise when developing the applications mentioned in the introduction, deserve a paper of their own.

## References

1. Jurafsky, D., Martin, J.H. In: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Prentice Hall PTR, Upper Saddle River, NJ, USA (2000) chapter 15
2. Jurafsky, D., Martin, J.H. In: Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition. Prentice Hall PTR, Upper Saddle River, NJ, USA (2000) 546
3. Gavaldà, M., Waibel, A.: Growing semantic grammars. In: Proceedings of the 36th annual meeting on Association for Computational Linguistics, Morristown, NJ, USA, Association for Computational Linguistics (1998) 451–456
4. Gavaldà, M.: Epiphenomenal grammar acquisition with gsg. In: ANLP/NAACL 2000 Workshop on Conversational systems, Morristown, NJ, USA, Association for Computational Linguistics (2000) 36–41
5. Sampson, G.: Semi-susanne corpus. http://www.grsampson.net/SemiSueDoc.html (2000)