ANTWERP PAPERS IN LINGUISTICS
nr. 6, 1976

P R O D U C I N G   N A T U R A L   L A N G U A G E
F R O M   S E M A N T I C   I N F O R M A T I O N

Luc Steels

UNIVERSITEIT ANTWERPEN

UNIVERSITAIRE INSTELLING ANTWERPEN

Departementen Ger. en Rom. Afdeling Linguïstiek

Universit eitsplein,l B-2610 Wilrijk(Antwerpen) Tel 031/28.25.28

Abstract

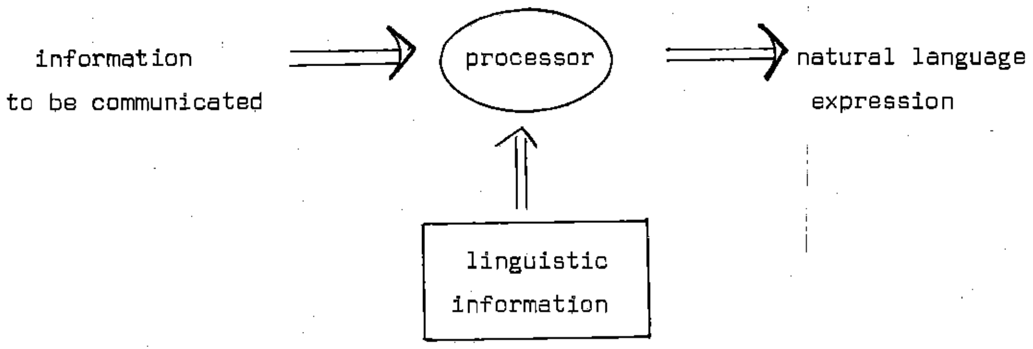
The problem of producing natural language sentences expressing given semantic information is briefly discussed in terms of a general processor which takes linguistic information as a possible 'program' and produces natural language sentences for input expressions containing information in a neutral format.
The most remarkable aspect of the approach is that the production process is directed by semantic factors and based on case frames instead of the usual phrase structures.

The goal of the research reported in this paper is to construct a theory about the way in which natural language is being produced.

By the production of natural language we do not mean the generation of a sentence from an initial symbol by successively applying the derivation relation on the basis of some generative grammar, but rather the realization of a mapping from information contained in a store into sentences of some natural language.

To realize this goal we will not only construct an abstract model but we will try to design and implement a general language producing mechanism, which takes as data linguistic information about a target language, and as input information in a language free format. The output is a sentence in the target language expressing the information from the input. Seen in this way the production mechanism is a processor which takes a grammar as a possible program.

Schematically:



The paper is organized as follows. First we describe the linguistic assumptions and the organization of the whole system. Then we deal with each aspect in some more detail. Finally we discuss experimental results of the implementation.

CONTENTS

----

1. LINGUISTIC ASSUMPTIONS

Language is produced in order to convey information about objects in some universe of discourse. Information in this context will mean to specify _ _ _ _ properties of the objects under consideration, so we make abstraction from externalizing purely semantic information.

Let us call a 'bundle' of properties a concept and let a predicate be the name of a concept. We furthermore distinguish between

(i) concepts being attributes (unary predicates)

(ii) concepts being relations (n-ary predicates), where the elements involved in a relation are said to fill the case slots associated with the relation.

As a communication consists of a series of interrelated properties, it must be decided

(i) how each property will be externalized, that is the wordform corresponding to the predicate,

(ii) in what order the properties will be realized.

It turns out that both points are depending from particular factors, and we will deal with these factors now.

(1) Communicative function

The first key to the realization of a concept in natural language is the fact that each concept that occurs in a communication process has a particular role or function in this communication process.

We see three main roles:

(i) a concept can be used to introduce (= name) an object

(ii) a concept can be used to specify more information about an object

(iii) a concept can be used to further modify or amplify other concepts.

Let us call such a role a communicative function. To know the communicative function of a concept is important because it will determine (partly) the way in which the concept will be realized. In particular it determines

(i) the part of speech of the concept (we call this the functional signal)

(ii) (for modifiers) when the concept will occur in the sentence (before or after the unit it is modifying)

(iii) (for relations) when the associated case slots will be realized (before or after the predicate)

(iv) (for modifiers) whether or not the features of the unit which it is modifying have to be taken over. (It is so that with each concept a list of features is associated, containing information that is expressed by means of the wordform, such as gender, number, etc... . This list is passed to other wordforms in certain situations. In English e.g. from the concept realizing the subject to the concept realizing the main verb, In German this happens e.g.

from the concept introducing an object (as a noun) to the concept modifying this (as an adjective).)

To make the issue more clear, we give some examples of the relation between the communicative function and the part of speech/word order, mostly for English.

(i) The part of speech used to introduce an object is a noun, case slots attached to nouns follow the noun itself. (The predicate is therefore in prefix position)

(ii) The concept introducing a noun can be modified by

(a) an attribute, then an adjective is used as part of speech and it comes (in English) before the noun it is modifying, compare

1. a written text (was input for the system)

and not

2.*a text written (was input for the system)

(The star indicates ungrammaticality)

(b) a relation, then a preposition or participle is used and it comes (in English) after the noun it is modifying. The case slots which are associated with this relation come after the predicate, compare

3. a text written by John (was input for the system)
4.* a written by John text " " " "
5.* a text by John written " " " "

Note that these matters are language dependent.

e.g. 1. In Dutch (and in German) relations modifying a noun can be placed before the noun as in

6. een door Jan geschreven tekst ....
(a by John written text) ...

The case slots should here come before the predicate.

e.g. 2.In French the attributes are not necessarily placed before, but rather after the noun as in

7. un texte écrit ....

and not

8. un écrit texte ... .

(iii) Modifications of attributes modifying nouns are expressed in terms of adverbs and they are placed before the attributes they are modifying, as in

9. a well written text (...)

Concepts modifying attributes modifying a noun, which are themselves relations, do not seem to occur.

(iv) Modifications of relations modifying nouns can take both forms

(a) being attributes adverbs are used and placed before the predicate.

(b) being relations, prepositions are used and placed after the predicate. Compare:

10. a text well written by John

11. a text written by John during the summer

(v) Normally in a communication process one chooses a main modification for the object which is considered the 'starting point' (= topic) of the communication. This modification is expressed by a main verb and special rules hold for modifying the main verb. The main verb is placed after the object it is modifying.

(vi) Attributes modifying the main verb are expressed in terms of adverbs and placed before (but also after) the main verb , as in

12. John recently wrote a letter to Jane

Relations modifying the main verb are expressed by means of prepositions and placed after the verb as in

13. John wrote a letter to Jane on the 14th of july.

Of course this sketchy outline is neither complete nor accurate, but it should give at least some idea about the relation between communicative function and part of speech/word order.

(2) Informative function

This brings us to a second major decision factor in the realization process. It turns out that not only the role a concept plays in the communication process is important but also in what way the concept is involved in the information. We call the way in which a concept gets involved in the information the informative function of the concept. As the relation between an element and the predicate to which this element is attached is called the case relation, the informative function is given in terms of case relations.

Consider e.g. the concept TRANSLATE and the agent, result and source case. Then we can use 'translate' e.g. to modify as a relation the agent as in

'John translating a book ...'

or to modify as a relation the source as in

' a book translated by John ...' .

Note the difference in the wordform of the predicate and in the case signals that are used, but the equality of the part of speech and the word order.

As a second examples let us take as function the introduction of objects, then we can do this in connection with the agent as in

'the translator of a text ... '

or in connection with the result as in

'the translation of a text by John ...'.

As a third example let us take as function the main modification of an object. We can do this in connection with the agent as in

'John translates a text ...'

or in connection with the source as in

'A text was translated by John ...'

Note again the difference in case signals and the form of the predicate but the equality in position and part of speech.

Note that there are certain restrictions on the way in which a concept can get involved in the information given a certain function. E.g. we can (in English) not directly introduce the source of a translation , the result case cannot be expressed if we use TRANSLATE to introduce the agent of the concept ,etc ... . These restrictions are probably again language dependent although we could not yet find a clear cut example .

It should be clear from the examples that the involvement of the concept in the information turns out to determine another part of the morphological appearance of the predicate (e.g. or-ending vs. ion-ending in the second example, active vs. passive in the third one). Let us call that part 'the involvement signal'. It should also be clear that the informative function determines the outlook of the (surface) case frame attached to the predicate: what case slots can be expressed and what signals are to be related to each case signal.

Together with the informative function we have two special 'functions' associated with a particular communicative function

(1) Mood

It turns out that the 'top' modifier (expressed in the main verb) can take several forms such as 'imperative', 'question', 'probable assertion', etc.. We call this the mood-function of the concept. It tells how the whole propositional concent should be used in the communication process.

(2) Quantification

Also it is possible to refine the introducing function of some concepts. These refinements are normally discussed in terms of quantifiers, and we add therefore a quantification function for each concept introducing objects.

(The necessity for such a function was pointed out by Remko Scha)

(3) Weight function

To make matters even more complicated a third dimension should be taken into account and that is the ability of the communicator to stress certain aspects of the information. This is reflected (for written texts) in the word order, based on the idea that if something is different from the way it is expected to be, then it more strongly affects our attention.

The change of normal word order due to stress is such a complicated matter that we hardly dare to discuss it. Just to give one example from Dutch. Given as normal word order

'Jan schreef een brief'

(John wrote a letter) ,

stress on 'een brief ' (a letter) leads to

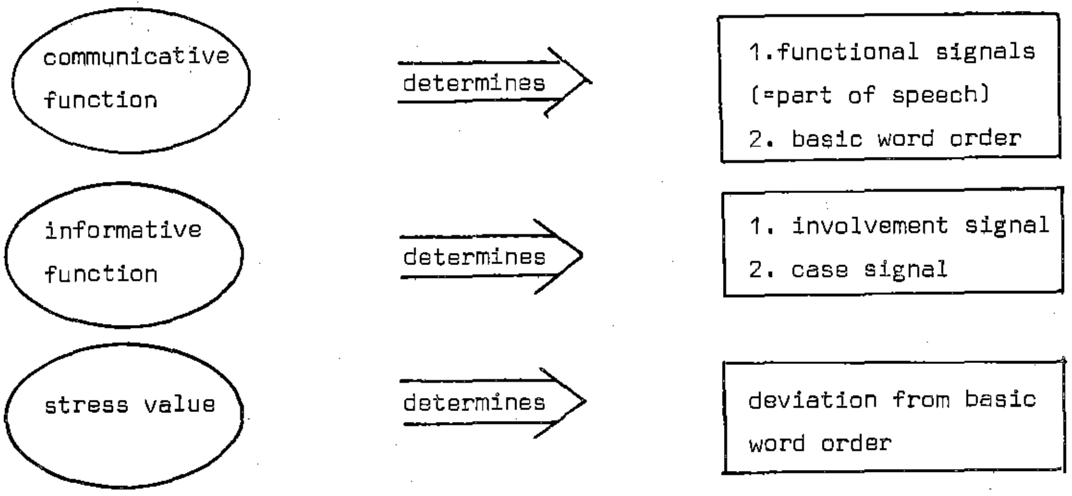
' een brief schreef Jan'

(a letter wrote John)

Note that **not only** the noun phrase which is stressed is brought in front of the sentence but the position of the subject as regards the main verb of the sentence is reversed. This illustates that the change in word order can affect the whole pattern involved.

(4) Conclusions

From the above discussion it becomes more or less clear what knowledge should be produced by the grammar, and what factors will determine the process. Schematically:



Note that the position that meaning based functions affect the way in which the sentence is produced, is absent from the 'generative' treatment of language, where a particular configuration is obtained by applying the derivation relation by trial and error so to say. There is no 'control' provided for the generation process or for the application of transformations. The idea that there are certain factors stemming from the communication situation which determine the formation of a sentence, should be credited to the Prague linguists. We consider however the notion of 'communicative dynamism' to affect strongly the extraction of information (i.e. the order in which objects and concepts are extracted from the memory). What we study is what happens once it is known how a particular extraction operation is going.

2. OVERVIEW OF THE PROCESS

Let us now turn to the process that may be capable of producing natural language given the above linguistic assumptions.

The production process clearly starts by extracting information, i.e. a number of interrelated concepts, from a store (the memory). The concepts (and the various links between them) are then brought in contact with the target language by relating them to predicates, i.e. names of concepts and the caseslots associated with these predicates.We call a predicate together with its various case slots an abstract or deep case frame. According to the communicative , informative and weight function of the predicate a surface case frame is constructed and related to this deep case frame. The surface case frame contains information about the part of speech of the predicate, the order in which the frame is to be realized, the case slots that are allowed to be expressed, the case markers that will be issued to express each case relation, etc... . These surface case frames are the basis for the actual realization. This includes chosing the actual word forms for the predicates and adding the words to the target sentence in the correct order.

So, the following main tasks can be distinguished:

(i) extract the information from the memory

(ii) relate the information to the language specific patterns (the deep case frames)

(iii) decide how the case frames are to be realized, i.e. convert them to surface case frames

(iv) realize the words and add them to the target sentence.

It is easy to see that each of these tasks draws upon certain knowledge, in particular

(i) during the extraction of information the system consults the memory

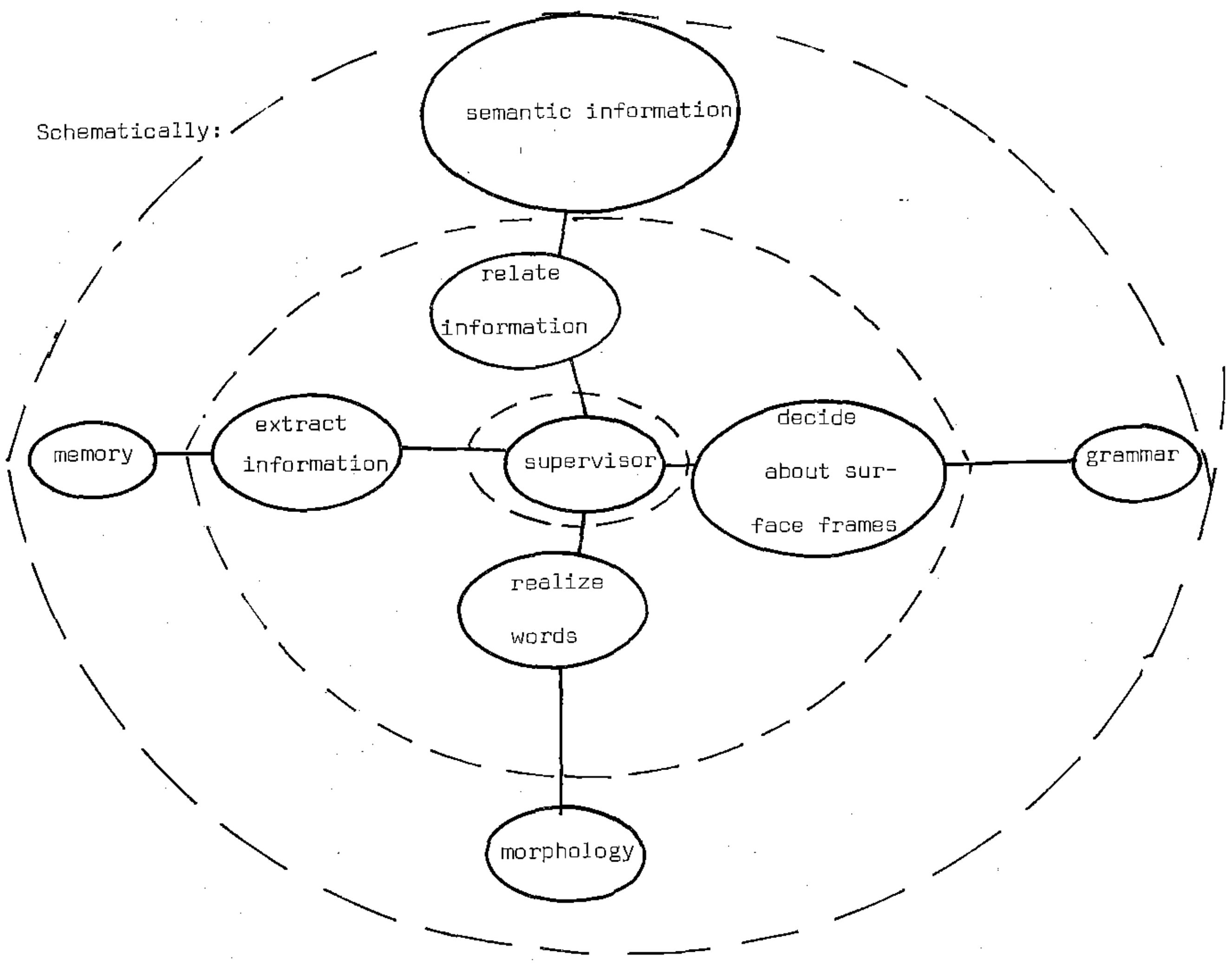
(ii) during the binding of information, the system consults a list of the abstract case frames, let us call such a list the semantic information about the target language.

(iii) when deciding how to realize case frames, information about what the surface case frames are is consulted, let us call such information the grammar of the target language

(iv) when realizing the words, a morphology of the target language is consulted.

It is also easy to see that to accomplish each linguistic activity, there must be a way to apply this knowledge, these are the algorithms associated with each task.

Schematically:



The inner circle contains the general supervisor which sends tasks to the appropriate modules, the second circle contains the procedure for each component and the outer circle contains the knowledge consulted by each procedure. Starting from an initial task, new tasks are created by each component during the execution of previous tasks, when no tasks are left, the process ends. So what we have is a task oriented system, that will work by interaction of all components. The procedures are defining the mapping from one configuration into one (or more) other configurations.

Seen in this way the knowledge is static, that means all the knowledge that is ever going to be needed must be there explicitly. This is bad for the following reasons (i) too much storage will be necessary, this will slow down the consultation process and it will cost a lot of space, (ii) obvious generalities are not captured.
It seems therefore appropriate to add another dimension to the whole system by introducing active processes that create or change the knowledge for each component.

This is very obvious for some knowledge bases such as the memory for which the dynamics are normally expressed in terms of inference mechanisms. It is also obvious for the morphological information where ideally not each word form is stored but principles to construct the word forms.
The idea of dynamic knowledge base is however not very common on the level of the grammar or the abstract case frames that are used. We feel that it is just as natural (and as necessary) here as on the other levels.

3. SUBSYSTEMS

We now discuss the different components and the knowledge structures that are used in each component. We concentrate here on the 'linguistic' aspects, especially the actual realization process, rather than the extraction of information from memory.

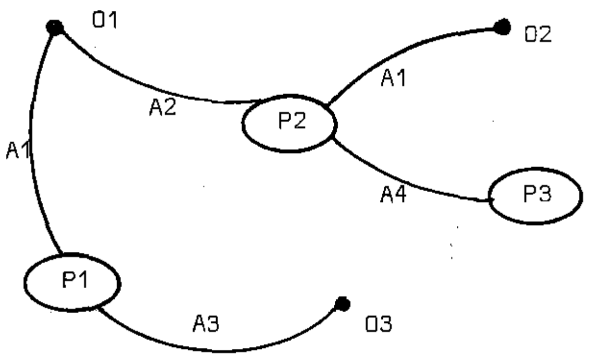
3.1. Memory and the extraction of information

Let us describe very roughly how the memory itself is supposed to be structured. Note that we will only deal with information from episodic memory, i.e. the properties of objects in a particular universe of discourse or the factual knowledge, rather than the communication of purely semantic knowledge. There are many other proposals for the structuring of memory, mostly stemming from cognitive psychology or artificial intelligence research. The one given here is comparable to the model of **Rummelhart** (1972).

1. A universe of discourse consists of a set of objects and particular properties (possibly relations) of the objects. Let us assign to each object a unique node and label it for ease of reference. Besides object nodes we must have a way of representing the properties. For this purpose we introduce other nodes and call them property nodes. We label these nodes with a signal indicating what property is contained in the node. The object nodes are brought in contact with the concept nodes by connecting them by lines. As a particular object node has a particular relation to a property, we will label these lines also. The labels are called case indicators.
Finally we can bring properties in contact with other properties by connecting their respective nodes by lines and labelling them also.

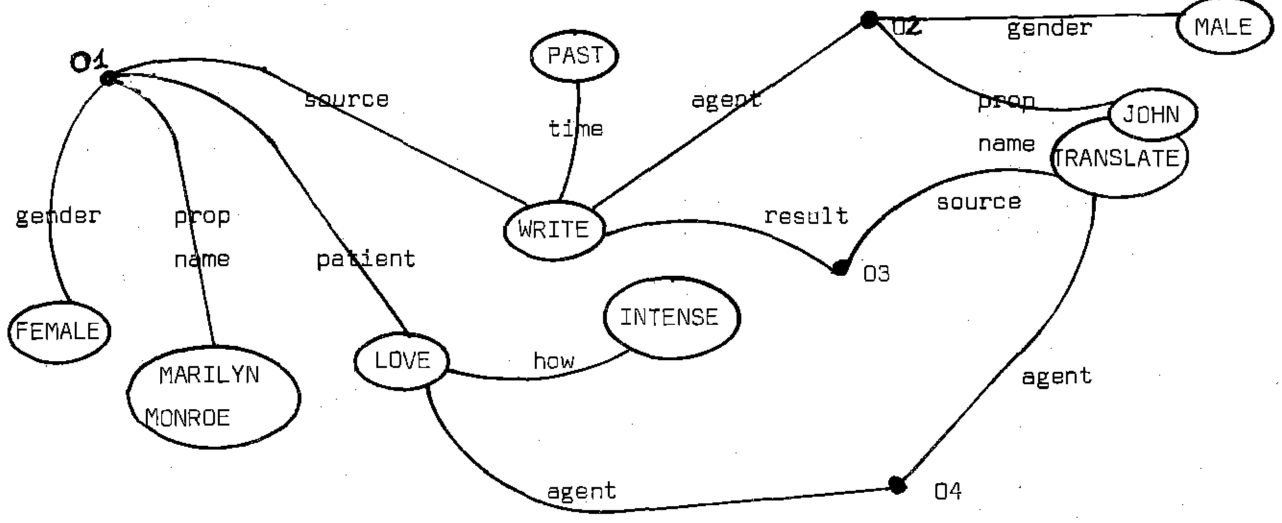
Example 1.

Let P1, P2 and P3 be labels for properties, O1, O2, O3 labels for object nodes and A1, A2, A3, A4 the case indicators then we can construct the following memory structure:



Example 2.

Using English-like words for the labels of properties one can construct the following example



Note
1. Although we use natural language words as labels for the properties, they should in no way be considered as such. Rather one should consider them as expressions in some conceptual language, as e.g. used in the conceptual dependency graphs used by Schank (1975).

2. Do not take these memory structures as a representation of the content of a sentence. (In most systems there is no difference between the memory representation and the representation used to specify what meaning will be conveyed in a particular sentence.)

2. The extraction of information is guided by various processes, in particular cognitive or other psychological machinery (starting with a stimulus to communicate), pragmatic knowledge such as to whom is the message addressed, what is the speaker supposed to know about the subect matter, etc... (see Bruce (1976) for a discussion of these aspects in the context of a production mechanism).
As a consequence the extraction process can only be made operational by embedding it in another task environment such as a question/answering system (or a problem solver), where there is a need to communicate particular information.
However as we want to concentrate on the production process itself we will have to find a way out of this. The solution that we have taken is to simulate the extraction process by giving the result of it, expressed in some format, as input to the production mechanism.
There are some questions that should first be dealt with
(1) Is the extraction of information guided by the linguistic information or in other words is there a'normal way' of doing it for a particular community of language producers, and if so what is the way,
(2) Is the extraction process language dependent ? That means does a speaker of English extract in another way information from his memory than a speaker of German ?
To be honnest for both questions we have to say that we do not (yet ?) know it.
Some speculations however, the first question is likely to be answered affirmitively, because e.g. the case slots appear in a definite order . Also the modifications of a verb e.g. appear in a particular order. (As long as no stress is involved of course). E.g. in English you seem to mention first the place indication and then the time indication as in

'I went to the park this morning' ,

whereas

' I went this morning to the park'

is only appropriate when 'morning' is stressed.
Also adjectives in front of the noun seem to appear in a definite normal order as in

' she has beautiful long red hair'

whereas

' she has red long beautiful hair'

is less appropriate.

Now for the second question, it is clearfrom observation that there are differences among languages as regards the preferential word order. The normal way of saying

'I went to the park this morning' in Dutch is

' Ik ben vanmorgen naar het park geweest'

and not

' Ik ben naar het park vanmorgen geweest'

(unless there is stress on'park').

Does this mean that Dutch speaking people extract information about the time first and then about the location, whereas the English speaking ones do it in a reverse way ? Or are there some mechanisms that take the same list but reverse it either for English or for Dutch ?

We will take the position that the order in which the information is extracted is indeed fixed and unversal although it will be quite a problem to say for particular series what the general principle of extraction is. (It is very difficult for the adjectives e.g., see De Schutter (1976) for an attempt to solve that problem for Dutch). We come back on these issues in forthcoming papers.

(Remark:the whole issue of word order is currently the focus of interest for many linguists, mostly stemming from a framework provided by Greenberg (1965))

Now we start our discussion about how the <u>result</u> of an extraction process might be expressed, we give first an example of a possible extraction process for the memory structure given in example 2.

Example 3.

'Let us say something about the object node 02, first we decide how to introduce 02, let us do that by means of its proper name, then we decide about the basic topic to be discussed in connection with 02; WRITE. With WRITE several other case slots are connected. We decide to realize the result case. Also we realize the concept PAST with WRITE; Now we have to choose a way of introducing 03. For this purpose we pick one of the properties attached to 03, namely TRANSLATE.With 'TRANSLATE' another case slot is being associated in which the object 04 is located. To introduce 04 we use the concept LOVE. With LOVE we realize the patient case which yields 01. To realize 01 we use its proper name which is 'MARILYN MONROE'. The sentence resulting from this extraction process might be the following one: 'John wrote a text which was translated by someone who loves Marilyn Monroe'.'

Resulting from <u>other</u> extraction processes many other sentences are possible for the same piece of memory. E.g.:

'John wrote about Marilyn Monroe',
' The translator of a text written by John loves Marilyn Monroe'
' The author of a text about Marilyn Monroe is called John'
' Marilyn Monroe is being loved by someone',etc... .

Now we design a format in which the result of the extraction process can be expressed. Note that other formats are possible and this depends of course fully on the structure of the factual knowledge in the episodic memory.

Let us call the result of an extraction process a source expression (as it is input for the production mechanism). The format of source expressions is defined by means of a context-free grammar (in BNF-notation) as follows

(a) Simple patterns

From previous discussion it becomes clear that a source expression will contain various properties. Also for each property it will be necessary to specify how it is related to the object node, i.e. we have to specify the case indicator under consideration. (the informative function) .
This yields:

```
<pattern>   ::=      ( <case indicator>  <property label> )
```

A pattern is associated with an object node in the following way:

```
<source>   ::=    ( <object label>  <pattern> )
```

Example:

```
(O2   (Propname    JOHN ))
```

(b) First extension

It turns out that a number of additional properties are associated with each concept. These additional properties are not realized as seperate words (although that could be) but incorporated in the natural language word corresponding to the concept. Examples of additional properties are gender, time, number, etc..
The list of additional properties contains also the mood-function for the main modifier and the quantification-function for the concepts introducing an object.

We write this list of additional properties after the concept label. If there are no additional properties we write NIL (the null list).

```
⟨pattern⟩ ::=   ( ⟨case indic⟩ ⟨concept label⟩  ⟨addit prop.⟩ )
```

Example:

```
    (O2    (AGENT  WRITE   (MALE))
```

(Additional properties are also a way to distinguish subconcepts, although we did not yet work this out already)

In addition we want to associate different case slots with each concept. This is done by adding a list of case slots to the pattern, where the first item in the list is the key word CASES and the rest of the list contains pairs of case indicators and labels for object nodes.

```
⟨pattern⟩  ::=    ( ⟨case indic⟩ ⟨concept label⟩ ⟨addit prop.⟩
                                 (CASES   ( ⟨case indic⟩ ⟨object label⟩  )*  )
```

(The * is the Kleene operator)

Example:

```
(i) (O2  (AGENT WRITE  (MALE)
                (CASES   (SOURCE O1)  (RESULT O3))))
(ii) (O1 (PATIENT LOVE  NIL   (CASES   (AGENT O4))))
```

(c) Second extension

As a concept can be used to modify another concept, we add other key words to a pattern with a list of the concepts involved. For the time being we add only one particular key word namely MOD (from modification)
So we get

```
⟨pattern⟩  :: =   ( ⟨case indic⟩ ⟨concept label⟩ ⟨addit properties⟩
                          (CASES     ( ⟨case indic⟩ ⟨object label⟩ )* )
                          (MOD ⟨pattern⟩* ))
```

(note the recursion on ⟨pattern⟩)

Example:

The extraction process given previously (as example 3) corresponds to the following source expressions:

```
(02 (PROPNAME JOHN  NIL (MOD   (AGENT WRITE (PAST)  (CASES ( RESULT 03))))))
(03 (SOURCE  TRANSLATE  NIL  (CASES (AGENT 04))))
(04 (AGENT LOVE NIL ( CASES ( PATIENT 01))))
(01  (PROPNAME  (MARILYN MONROE) NIL))
```

(Technical remark: as a general rule we write a unit between brackets if there is more than one element, e.g. MARILYN MONROE is a concept with two elements, so we write it between brackets, the same holds e.g. for the list of additional properties and other lists that will be met with. This is of course a matter of defining the list processing activities accordingly)

The last extension is one specifying the main modification of an object, this is done by having a special pattern under the keyword TOP.

(d) summary

The rules of the complete grammar defining source expressions are then:

```
<source expression> ::=    ( <object label> <pattern> )

 <pattern>   ::=

              ( <case indic> <concept label> <addit prop>

                     [(CASES   (  <case indic> <object label>   )*  )]
                     [( MOD  <pattern> *  )]
                     [(TOP  <pattern>  )] )
```

where square brackets denote optional elements.

Note that source expressions are actually list structures in list notation.

3.2. Relating concepts to predicates of the target language

Now we come to the second sort of tasks, those where concepts are related to predicates (i.e. names for those concepts) on the basis of semantic information

Let us define an abstract case frame to consist of

(i) a predicate (i.e. the name of a sequence of properties)

(ii) a set of case slots (i.e. the particular relations that can hold between objects and the predicate involved), seen from the semantic interpretation point of view, these case slots are the various arguments for the procedures associated with the predicate

(iii) for each case slot we also specify the <u>type</u> of the arguments that can fill the slot, i.e. a value restriction. We define the type of the argument by specifying a sequence of semantic properties that the objects which might fill the case slots are supposed to have (not necessarily but preferentially). These semantic properties could be called the selection restrictions for that case slot.

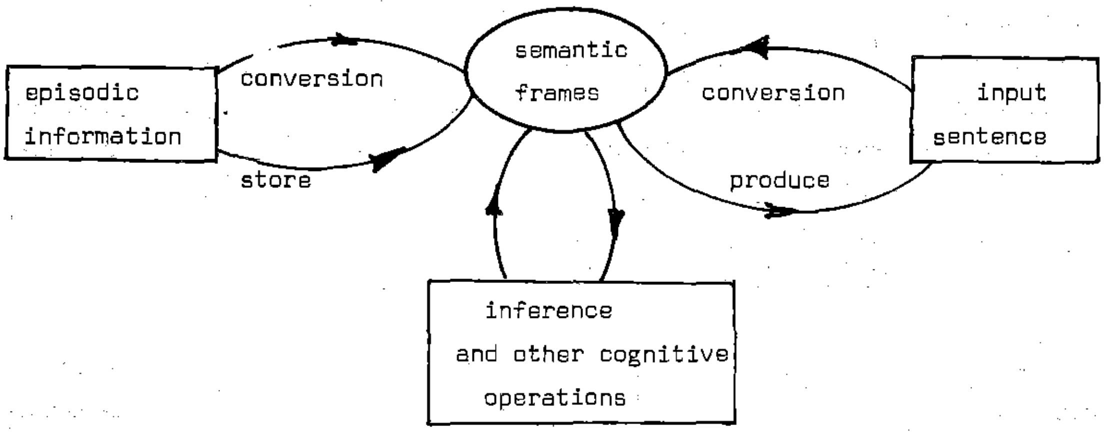
(Note that for the predicates we will again use English words although further processing is necessary before we reach the actual word of the language)

The association of a case frame with a concept consists of a matching process between a sequence of properties in the memory and a series of properties associated with a predicate. Also the different case relations that occur in the memory are matched against the case relations found in the case frames, and the various objects depending on these case relations are associated to their corresponding argument place or case slot in the case frame.

(As B. Rieger pointed out to me, it is probably so that 'fuzzy' principles are guiding these matching processes.)

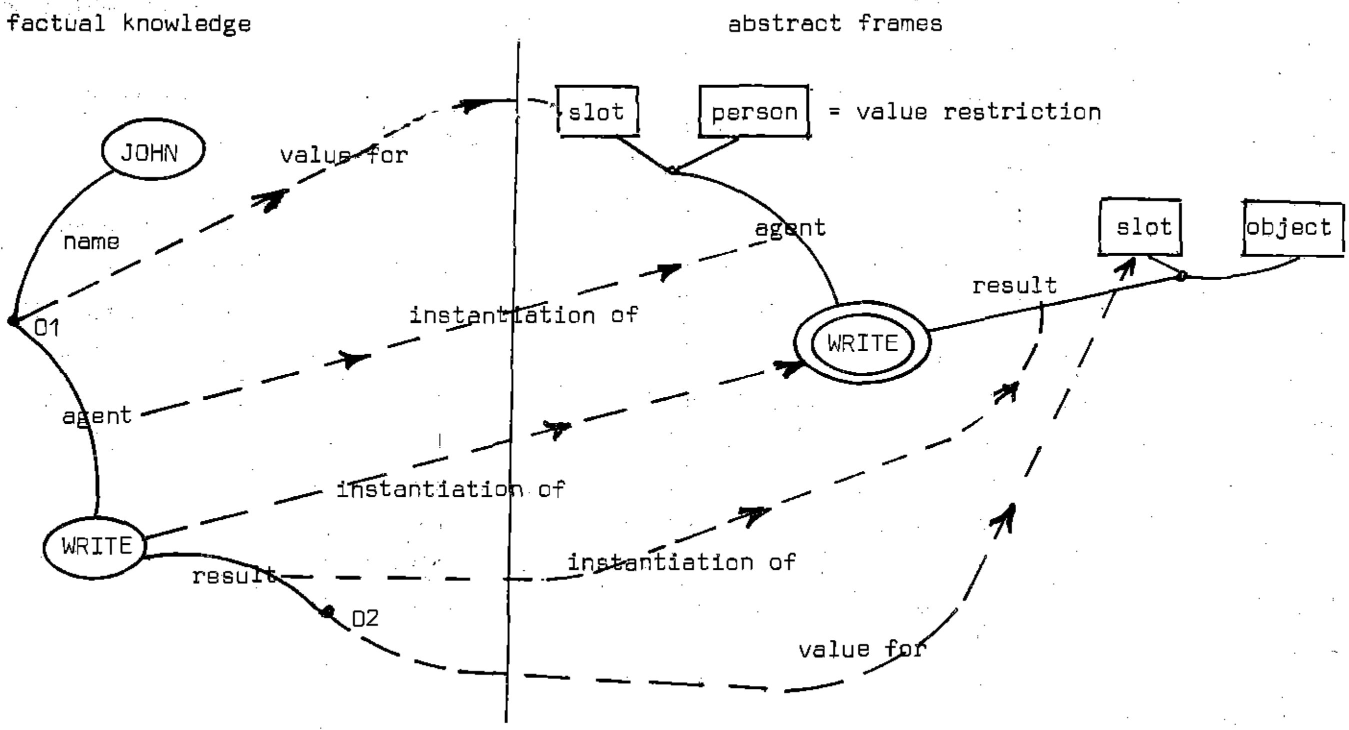
The last process can be compared to the process of lambda-conversion (as it is used in Church's lambda-calculus (Church, 1941)) and in the programming language LISP). Also here one starts from 'abstracted' forms or frames containing a function name and various slots for arguments (the bound variables). The bound variables are then brought into contact with the actual arguments by pairing the values of the actual arguments to the bound variables on the association list.

Moreover the analysis process might also be regarded as such a conversion process, so , we obtain a two-way convertibility of the deep case frames, one way from the memory and another way from the language input. This suggests interesting perspectives for the design of analysis routines and we will come back on this in forthcoming work . Note that for other tasks such as inference making the information has also to be bound to the abstract frames, seen in this way the abstract frames are really the'filter' through which all activities pass:



Another way to express what happens when the abstract frames are related to factual knowledge is to consider the memory structures as instantiations of the concepts in the abstract case frames and the main task is then to find abstract case frames such that particular information can be regarded as an instantiation.

Schematically:



3.3. From abstract to surface case frames

Given such a deep case frame and a temporary association to objects in the memory, it must then be decided what signals will be used to externalize the case frame. As we discussed earlier this goes in such a way that for each concept the informative, communicative and weight function will lead to a decision on

(i) the part of speech of the predicate
(ii) the occurrence (before or after the unit which the concept is modifying)
(iii) the occurence of the other case slots (before or after the predicate)
(iv) additional morphological signals
(v) what case slots may occur
(vi) the signals of the case slots.

With a dynamic knowledge base, w e can state the above information in an abstract manner, and apply it to yield the desired surface rules. The knowledge about how a case frame should be realized is applied by creating tasks for each aspect of the frame as we will see in section 5.

3.4. Morphological information

A predicate takes different external appearances depending partially on its communicative and informative function. Other aspects that determine the outlook of a predicate are

(1) additional properties not realized by separate words
(2) signals following from the surface case frames (such as affixes which act as case markers)
(3) in addition some predicates take signals from predicates that they further modify (e.g. adjective from noun or mainverb from subject)

During the production process such features are gradually collected on a so called feature list. The morphological information will then consist of associations of feature lists with word forms for each predicate. As a rule the wordform which embodies most features is chosen. In an ideal situation this mapping from feature lists to word forms is expressed as a procedure, but in our (first) experiments we will define the mapping explicitly.

4. The whole system

We now describe a first experimental version of a production mechanism that was implemented and tested for small linguistic data in 4 languages: English, German, French and Dutch.

The program consists of 3 modules and a control module. Communication of data and flow of control goes by means of tasks. The control module constructs the initial task, takes successively tasks from the task list and sends them to the three other modules, except when the task is to simply add a word to the target sentence. When no tasks are left, the control module returns to the main program to read another source expression.

The tasks consists of 4-tuples W1, W2, W3, W4 each element of which may contain a particular sort of information depending on the sort of tasks. There are 1 sorts of tasks:

(1) Tasks to find a source expression related to a particular object node label. This sort of tasks is issued when a case indicator has been found with an object node filling the case slot.

In this case

W1 = the keyword CASES

W2 = a pointer to the relevant part of the source expression prèvious worked upon

W3 = a pointer to the frame in the grammar associated with the predicate from which the case is depending;

W4 = a feature list already assoċiated with the object node.

Such a task (which we call a case task) is processed by MODULE 1 . After processing a new task is created of the second type.

(2) The second type of tasks is meant to search in the case frames of the target language to find out how a particular concept should be expressed, that means (i) to search for a predicate related to the concept, (ii) to find relevant surface case frames, (iii) to find the word form for the feature list of the predicate. We call this sort of tasks concept tasks they are processed by MODULE 2

In this case:

W1 = the function of the predicate

W2 = a pointer to the relevant part of the source expression

W3 = empty

W4 = feature list associated with the predicate

After processing new tasks are created of type 3.

The third module (MODULE 3) starts working after module 2. It performs the process of task building. It takes a concept task(the same as was processed in module 2) and creates new tasks for depending case relations, modifications, etc... on the basis of the source expression.

Special facilities:

In addition to the usual flow of control we provide the following special facilities:

(i) if an object node has already been realized, it will the second time be realized as a pronoun. If so, the morphology is consulted with the feature list then available and with as predicate a sign for the pronouns. Also if the pronoun refers to the subject, the feature 'self' is added and realized morphologically.

(ii) there are two special cases where features in the feature list of a certain predicate are realized as words, this is the case for auxiliaries and determiners.
This is noticed during task building and the appropriate tasks are created to realize these words and to realize the rest of the features later on.

Examples

To see more clearly how a production process is going, we give some output of the program simulating the process. We do this for an English sentence and give examples for French, German and Dutch afterwards. The reader is advised to study the comments given by the program.
A technical description of the program will appear later when a second version of the implementation is complete.

```
INPUT:
******

IN :
(ENGLISH
((O1)(AGENT WRITE (DEF)
  (TOP MAIN AFFIRM (AGENT WRITE FUTURE
    (CASES (RESULT O2 ))))))
(O2 (RESULT WRITE (UNDEF FICTIVE)
   (CASES (SOURCE O3))
    (BEFORE (RESULT WRITE NIL
      (BEFORE (HOW GOOD NIL))
))
))
(O3(PROPNAME (MARILYN MONROE)))
)

STEP  1
--------
FIRST TASK CREATED
ENTER MODULE 2 * CURRENT TASK WORKS ON THE FOLLOWING CONCEPT :

OUT:
WRITE
ADDITIONAL PROPERTIES ADDED
WORDSTEM FOUND
PATTERN FOUND
FEATURES FROM CASE SLOT ADDED
LIST OF FEATURES  ALREADY ASSOCIATED WITH WORDSTEM:

OUT:
(DEF PERSON )
FUNCTIONAL SIGNAL IS :

OUT:
NOUN
TASK BUILDING * ENTER MODULE 3
TASK VOR VERB CREATED
REALIZATION TASK FOR PREDICATE CREATED
REALISATION TASK FOR DETERMINER CREATED

STEP  2
--------
WORD ADDED , PARTIAL RESULT :

OUT:
(THE )

STEP  3
--------
WORD ADDED , PARTIAL RESULT :

OUT:
(THE AUTHOR )

STEP  4
--------
ENTER MODULE 2 * CURRENT TASK WORKS ON THE FOLLOWING CONCEPT :

OUT:
WRITE
ADDITIONAL PROPERTIES ADDED
WORDSTEM FOUND
PATTERN FOUND
FEATURES FROM CASE SLOT ADDED
LIST OF FEATURES  ALREADY ASSOCIATED WITH WORDSTEM:

OUT:
(DEF PERSON FUTURE )
FUNCTIONAL SIGNAL IS :

OUT:
MAINV
```

```
TASK BUILDING * ENTER MODULE 3
AUXILIARY FOUND

OUT:
(WILL INFIN AFTER )
TASKS CREATED FOR TOP LEVEL
REALIZATION TASK FOR PREDICATE CREATED

STEP  5
--------
WORD ADDED , PARTIAL RESULT :

OUT:
(THE AUTHOR WILL )

STEP  6
--------
TASK BUILDING * ENTER MODULE 3
TASKS FOR AUXILIARIES ADDED

STEP  7
--------
WORD ADDED , PARTIAL RESULT :

OUT:
(THE AUTHOR WILL WRITE )

STEP  8
--------
CURRENT TASK ON FILLING THE CASE SLOT * ENTER MODULE 1
CASE INDICATOR FOUND IN GRAMMAR
OBJECT NODE FOUND
NEW TASK CREATED TO REALIZE OBJECT NODE

STEP  9
--------
ENTER MODULE 2 * CURRENT TASK WORKS ON THE FOLLOWING CONCEPT :

OUT:
WRITE
ADDITIONAL PROPERTIES ADDED
WORDSTEM FOUND
PATTERN FOUND
FEATURES FROM CASE SLOT ADDED
LIST OF FEATURES  ALREADY ASSOCIATED WITH WORDSTEM:

OUT:
(OBJECTIVE UNDEF FICTIVE OBJECT )
FUNCTIONAL SIGNAL IS :

OUT:
NOUN
TASK BUILDING * ENTER MODULE 3
TASK WITH CASES ADDED
REALIZATION TASK FOR PREDICATE CREATED
NEW TASK WITH BEFORE MODICIATION
REALISATION TASK FOR DETERMINER CREATED

STEP 10
--------
WORD ADDED , PARTIAL RESULT :

OUT:
(THE AUTHOR WILL WRITE A )
```

```
STEP 11
--------
ENTER MODULE 2 * CURRENT TASK WORKS ON THE FOLLOWING CONCEPT :

OUT:
WRITE
WORDSTEM FOUND
PATTERN FOUND
FEATURES FROM CASE SLOT ADDED
LIST OF FEATURES  ALREADY ASSOCIATED WITH WORDSTEM:

OUT:
(OBJECTIVE UNDEF FICTIVE OBJECT )
FUNCTIONAL SIGNAL IS :

OUT:
(PAST PARTIC )
TASK BUILDING * ENTER MODULE 3
REALIZATION TASK FOR PREDICATE CREATED
NEW TASK WITH BEFORE MODICIATION

STEP 12
--------
ENTER MODULE 2 * CURRENT TASK WORKS ON THE FOLLOWING CONCEPT :

OUT:
GOOD
WORDSTEM FOUND
PATTERN FOUND
FEATURES FROM CASE SLOT ADDED
LIST OF FEATURES  ALREADY ASSOCIATED WITH WORDSTEM:

OUT:
NIL
FUNCTIONAL SIGNAL IS :

OUT:
ADV
TASK BUILDING * ENTER MODULE 3
REALIZATION TASK FOR PREDICATE CREATED

STEP 13
--------
WORD ADDED , PARTIAL RESULT :

OUT:
(THE AUTHOR WILL WRITE A WELL )

STEP 14
--------
WORD ADDED , PARTIAL RESULT :

OUT:
(THE AUTHOR WILL WRITE A WELL WRITTEN )

STEP 15
--------
WORD ADDED , PARTIAL RESULT :

OUT:
(THE AUTHOR WILL WRITE A WELL WRITTEN NOVEL )

STEP 16
--------
CURRENT TASK ON FILLING THE CASE SLOT * ENTER MODULE 1
CASE INDICATOR FOUND IN GRAMMAR
PREPOSTION ADDED
OBJECT NODE FOUND
NEW TASK CREATED TO REALIZE OBJECT NODE
```

```
STEP 17
--------
ENTER MODULE 2 * CURRENT TASK WORKS ON THE FOLLOWING CONCEPT :

OUT:
(MARILYN MONROE )
TASK BUILDING * ENTER MODULE 3
REALIZATION TASK FOR PREDICATE CREATED

STEP 18
--------
WORD ADDED , PARTIAL RESULT :

OUT:
(THE AUTHOR WILL WRITE A WELL WRITTEN NOVEL ABOUT MARILYN MONROE )

STEP 19
--------

FINAL RESULT (OBTAINED AFTER    3.656 SECONDS OF P ROCESSING)

OUT:
(THE AUTHOR WILL WRITE A WELL WRITTEN NOVEL ABOUT MARILYN MONROE )
```

Or without the comments on the production :

```
INPUT:
******

IN :
(ENGLISH
((O1)(AGENT WRITE (DEF)
  (TOP MAIN AFFIRM (AGENT WRITE FUTURE
     (CASES (RESULT O2 ))))))
(O2 (RESULT WRITE (UNDEF FICTIVE)
   (CASES (SOURCE O3))
    (BEFORE (RESULT WRITE NIL
       (BEFORE (HOW GOOD NIL))
))
))
(O3(PROPNAME (MARILYN MONROE)))
)

FINAL RESULT (OBTAINED AFTER    0.945 SECONDS OF P ROCESSING)

OUT:
(THE AUTHOR WILL WRITE A WELL WRITTEN NOVEL ABOUT MARILYN MONROE )
```

Now we given the German example with the same source expression, note the case affixes, the different prosition of the case slots attached to the concept realized by the main verb.

```
INPUT:
******

IN :
(GERMAN
((O1)(AGENT WRITE (DEF)
  (TOP MAIN AFFIRM (AGENT WRITE FUTURE
     (CASES (RESULT O2 ))))))
(O2 (RESULT WRITE (UNDEF FICTIVE)
    (CASES (SOURCE O3))
     (BEFORE (RESULT WRITE NIL
       (BEFORE (HOW GOOD NIL))
))
))
(O3(PROPNAME (MARILYN MONROE)))
)

FINAL RESULT (OBTAINED AFTER    1,039 SECONDS OF P ROCESSING)

OUT:
(DER AUTOR WIRD EINEN GUT GESCHRIEBENEN ROMAN UEBER MARILYN MONROE SC
HREIBEN )
```

Dutch example:

```
INPUT:
******

IN :
(DUTCH
((O1)(AGENT WRITE (DEF)
  (TOP MAIN AFFIRM (AGENT WRITE FUTURE
     (CASES (RESULT O2 ))))))
(O2 (RESULT WRITE (UNDEF FICTIVE)
    (CASES (SOURCE O3))
     (BEFORE (RESULT WRITE NIL
       (BEFORE (HOW GOOD NIL))
))
))
(O3(PROPNAME (MARILYN MONROE)))
)

FINAL RESULT (OBTAINED AFTER    0,859 SECONDS OF P ROCESSING)

OUT:
(DE SCHRIJVER ZAL EEN GOED GESCHREVEN ROMAN OVER MARILYN MONROE SCHRI
JVEN )
```

Now the French example, note that the time indication realized in the main verb is notexternalized as an additional word, but by the morphological signals in the verb itself.

```
INPUT:
******

IN :
(FRENCH
((O1)(AGENT WRITE (DEF)
  (TOP MAIN AFFIRM (AGENT WRITE FUTURE
     (CASES (RESULT O2 )))))))
(O2 (RESULT WRITE (UNDEF FICTIVE)
    (CASES (SOURCE O3))
        (BEFORE (HOW GOOD NIL))
))
(O3(PROPNAME (MARILYN MONROE)))
)

FINAL RESULT (OBTAINED AFTER    1,297 SECONDS OF P ROCESSING)

OUT:
(L" ECRIVAIN ECRIVERA UN BON ROMAN SUR MARILYN MONROE )
```

5. CONCLUSIONS

We briefly sketched a way of producing natural language. The main deviations from the usual treatment are

(i) the production process is guided by meaning- based principles such as the informative, communicative and weight function of each concept.

(ii) the production is based on case frames rather than phrase structure rules

(iii) the knowledge consulted by all components is dynamic

(iv) the system is task-oriented and works by interaction of all components, this leads to a clear 'from left to right' production.

(v) no intermediate structures are produced, the necessary information is solely distributed via the items contained in a task

For parsing systems one often makes the distinction between syntax-directed systems (such as Woods' ATN-parser, Woods(1973)) and semantics-directed systems (such as Wilks'analyzer (Wilks,1975) and Riesbeck's parser (Riesbeck (1975)). Clearly what we have tried to do is to design a semantics-directed producer, rather than a syntax-directed producer based on a ps-grammar or ATN-grammar (as e.g. the producer by Goldman (1975)).

Further references

We tried (in cooperation with D. Vermeir) to formalize the notion of the (surface) grammar used in the production system, in terms of formal language theoretic notions. The resulting type of grammar has been termed 'completion grammar' and results appeared in Steels (1976a),Steels (1976b) and Steels and Vermeir (1976c). We are also working on an analysis procedure based on the same linguistic assumptions.

Acknowledgement

Many thanks to prof. De Schutter whose valuable comments contributed considerably to the approach and Luc Baert who helped in the preparation of the computer experiments. Furthermore we thank all people who have shown interest in our work and were willing to discuss it.

References

Bruce, B. (1976) Discourse Influences on language generation. Paper presented at Coling conference 76, Ottawa.

Church, A. (1941) The calculi of lambda-conversion . Princeton University Press, Princeton.

Goldman, N. (1975) Conceptual generation. in: Schank (1975), 289-371.

Greenberg, C. (1966) Some universals of grammar with particular reference to the order of meaningful elements. in: Greenberg (ed) (1965) Universals of Language. Cambrdige M.I.T. Press.

Rummelhart, D. et.al. (1972) A process model for long-term memory. in: Tulving, E. and W . Donaldson (1972) Organization of memory. Academic Press, N.Y. and London.

Riesbeck, C.K. (1974) Computational understanding: analysis of sentences and context. Institute for semantics and cognitive studies. Castagnola microfiche.

Schank, RC. et.al. (1975) Conceptual information processing. North-Holland Amsterdam.

Schutter, G. DE (1976) De beschrijving van adjektieven in het Nederlands APIL, forthcoming.

Steels, L. (1976a) A formalism for case grammar. Paper presented at the Artificial intellingence and simulation of behavior summer conference Edinburgh, 1976.

Steels, L. (1976b) Completion grammars and completion automata revisited. To appear in ITL.

Steels, L. and D. Vermeir (1976c) Some results on the relation of word order to grammatical complexity. Forthcoming.

Wilks, Y. (1975) An intelligent analyzer and understander of English. Comm. of the ACM, 18, 1975.

Woods, W. (1973) An experimental parsing system for transition network grammars. In: Rustin (ed) 1973. Natural language processing. Algorithmics press. New York.

Issues of Antwerp papers in Linguistics

1 - 1975 Luc Steels: Parsing systems for Regular and Context-free languages;

2.- 1975 Johan Van der Auwera: Semantic and pragmatic Presupposition

3 - 1975 Luc Steels : Completion grammars and their applications

4 - 1976 Georges De Schutter & Eddy Kockx : Meervoudsvorming en vervoeging in het Nederlands. Een morfofonologische proeve.

5 - 1976 Luc Steels & Dirk Vermeir: On the formal properties of completion grammars and their related automata.