Interpretation of iconic utterances based on contents representation: Semantic analysis in the PVI System

Pascal Vaillant

Natural Language Engineering **1** (1): 1–24

Interpretation of Iconic Utterances based on Contents Representation: Semantic Analysis in the PVI System

Pascal Vaillant

Thomson-CSF LCR, Computer Science Group
Domaine de Corbeville, 91404 ORSAY CEDEX, FRANCE
E-mail: vaillant@lcr.thomson.fr

(*Received 20 March 1997*)

Abstract

This article focuses on the need for technological aid for agrammatics, and presents a system designed to meet this need. The field of Augmentative and Alternative Communication (AAC) explores ways to allow people with speech or language disabilities to communicate. The use of computers and natural language processing techniques offers a range of new possibilities in this direction. Yet AAC addresses speech deficits mainly, not linguistic disabilities. A model of aided AAC interfaces with a place for natural language processing is presented. The PVI system, described in this contribution, makes use of such advanced techniques. It has been developed at Thomson-CSF for the use of children with cerebral palsy. It presents a customizable interface helping the disabled to compose sequences of icons displayed on a computer screen. A semantic parser, using lexical semantics information, is used to determine the best case assignments for predicative icons in the sequence. It maximizes a global value, the "semantic harmony" of the sequence. The resulting conceptual graph is fed to a natural language generation module which uses Tree Adjoining Grammars (TAG) to generate French sentences. Evaluation by users demonstrates the system's strengths and limitations, and shows the ways for future developments.

1 Introduction

It is clear how greatly the quality of life of many disabled people has been improved by the technical aids invented in recent decades. Most of these aids were anything but superfluous, giving the disabled access to abilities that most people regard as being fundamental to their lives. Some types of deafness, for example, were overcome by cochlear implants, for deaf people who would otherwise have remained completely and definitely deprived of hearing.

The same is happening in the field of Augmentative and Alternative Communication (AAC), a technological market whose existence was made possible by the advent of the computer as a mass product and the availability of information processing to the average citizen. AAC is opening up language to people who formerly would simply have been compelled to stay mute. Via a suitably adapted computer,

they may compose messages and have them pronounced by voice synthesis devices; thus gaining access to no less than communication with others.

The research presented in this paper is located in the field of AAC. Its focus is the development of specific techniques to build up a comprehensive communication-aid software for people with both cognitive and motor impairments. We first argue that some types of language impairment are barely addressed by classical AAC systems, and propose a taxonomy of AAC systems where new methods may be placed (Section 2). We then explain how specific user needs determined the approach in the PVI system (Section 3). We describe the system's architecture and interface (Section 4); its analysis mechanism, based on semantic parsing (Section 5); and its sentence generation module, using TAGs for French (Section 6). Lastly, evaluation and results will be presented and discussed (Section 7).

Working towards technical assistance for the disabled is a vital task, for the first and obvious reason that acknowledging the principles of human rights also implies trying to restore those rights wherever they are hindered (disability can in some way be said to deprive a person of the right to see, move, speak... all of which we take for granted). It is also a stimulating scientific challenge, connected inevitably with progress in the knowledge of how human abilities work. Furthermore, the importance of the techniques developed in such research frequently ranges well beyond the scope of the particular disability being considered: these techniques may get applied to other systems, other problems, and eventually become profitable. It should be noted that the ubiquitous remote control, now in use with many electronic products, originally came out of a research program for disabled people. These are three reasons why we think that research projects of this type are to be encouraged and developed.

2 Different Systems for Different Needs

2.1 Levels of Language Disorders

Speech and language disorders can be of many types. They can show many different symptoms, and stem from many different causes. So, it is not so easy to design intelligent AAC devices well adapted to their uses—and users. One of the first necessary tasks is to characterize thoroughly the users' deficits and needs.

Two main factors have to be considered when trying to overcome a communication impairment: (1) the level in the speech and language production chain that is affected by the impairment, and (2) the nature and cause of the impairment. In other words—what is going wrong where, and why.

The first element obviously determines the location of the problem and its possible remedy. The second one is critical in determining how to tackle the problem. In particular, it gives the answer to some important questions which guide the selection of a communication strategy: can the affected speech or language function be healed or rehabilitated? Is it a consequence of a more general behavioural disorder? Is it partially or wholly damaged? Has it ever once worked *normally*? Has it to be artificially performed, or replaced by another function?

If the answer to the first of these questions is no, the language impairment is permanent. Some permanent speech and language disorders may require the use of AAC. We have given a few examples of different types of these permanent disorders in Figure 1, where we have collected data from some literature (in particular (Gérard 1993; Eustache and Lechevalier 1993); the classification of psychiatric troubles follows the A.P.A. taxonomy (A.P.A. 1987)).

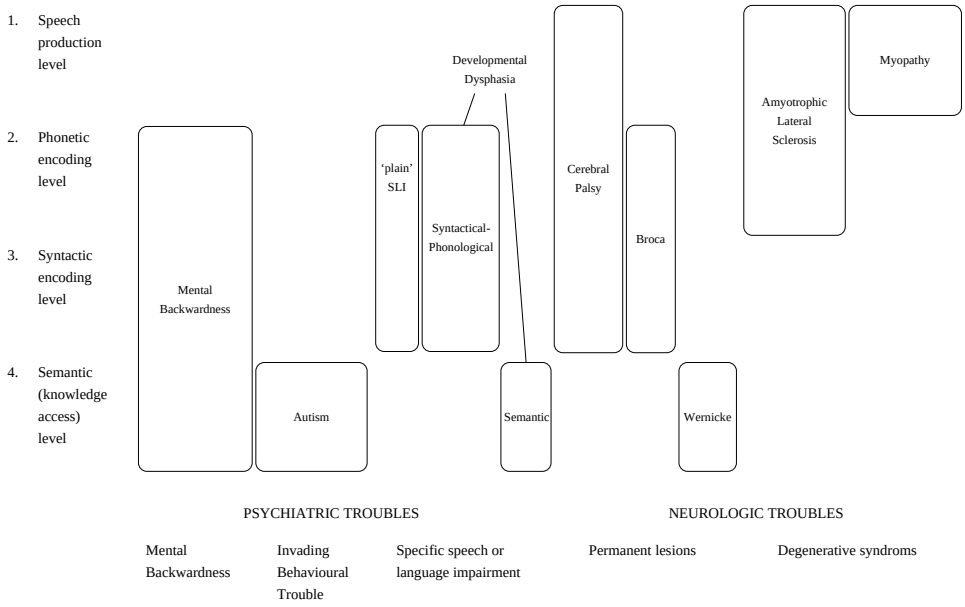

Fig. 1. Permanent language disorders. The vertical axis represents the different levels at which the speech and language production may be affected (Factor 1, above). The horizontal axis represents different possible disorders' nature (Factor 2, ibid.).

The different levels at which the impairment may occur are:

1. Speech production level: neuromotor troubles affecting the articulatory organs prevent the subject from fluently uttering spoken words;
2. Phonetic encoding level: the subject has not mastered the phonological system of his mother tongue. He may use a reduced set of phonemes, or show systematic or non-systematic deviance in his choice of phonemes;
3. Syntactic encoding level: the subject is partially or totally unable to use the morphological or syntactic structures of his mother tongue to encode the information of his messages;
4. Semantic level: the subject cannot access lexical knowledge normally. He may miss words, show improper categorization, improper use of existing words, or in some cases use made up words without noticing it.

We shall be particularly concerned in this paper with the language problems

affecting the syntactic encoding of messages. These problems bear a generic name: *agrammatism*.

Saffran *et al.* (1980) define the agrammatic deficit saying that:

> "agrammatic speech is generated without underlying structures that represent logical relations (...) The agrammatic selects the salient elements of a cognitive representation and maps them into a surface structure of the N-V-N form, ordering the NPs on the basis of cognitive factors like animacy or potency. The order of the NPs carries no semantic significance" (p.278).

Agrammatism is not a syndrome, it is a functional deficit which may be linked to different ætiologies. The best known example is Broca's aphasia; but agrammatism is also linked to developmental dysphasia (or to a lesser degree to children's plain Specific Language Impairments [SLI] (Van der Lely 1994)), and to Cerebral Palsy (see Figure 1).

2.2 The Need for Intelligent Systems

Most AAC systems today tackle the problem mainly at the level of speech production. They are meant for subjects who cannot talk, but ideally have the linguistic and cognitive abilities that would allow them to do so. If these abilities are impaired, their problem is not addressed.

When a subject is unable to talk, what is the first thing we think he might manage instead? Writing. This is why most AAC systems however are typewriters with sophisticated input systems:

1. In the simplest case, the speech impaired individual is capable of writing his mother tongue. The AAC device then comes down to a typewriter, with voice synthesis output in the most sophisticated case.
2. If the user has an imperfect command of the written language's orthography, the letter input is replaced by a phonetic or syllabic input. This second solution is fairly similar to the first, except that it takes the difficulties of orthography away. A good phonetic command of the mother tongue is still required.
3. Lastly, if the impairment is not limited to a speech deficit but also includes a language deficit—preventing the user from accessing either written or phonetic word representation—, then icons are used as input symbols. This solution has for example proved to be useful for aphasic patients in the C-VIC system (Steele *et al.* 1989). It is otherwise widely used in many AAC systems, with such various visual symbol sets as PIC, Blissymbolics, or Minspeak.

Saying that communicating with visual symbols is somehow a sophisticated form of typewriting might seem to be a provocative assertion. However, what is simply meant here is that in most implemented systems, even those employing icons, the basic principle of augmented communication is that the user selects symbols which correspond to natural language words, and that those symbols are eventually displayed or transmitted in the same order, with no further interpretation.

Let us take a closer look at three widespread visual symbol systems, commonly

used in the field of AAC: PIC, Blissymbolics and Minspeak. These three languages are particularly interesting, because each one of them, one could argue, has a predominant representation mode distinct from the others: pictographic in PIC, ideographic in Bliss and morphographic in Minspeak.

PIC symbols are typical pictograms: they are little drawings which "look like" the things they stand for. Here the semiotic function is massively iconic; this of course involves strong limitations on the representation of abstract notions.

Blissymbolics (Hehner 1980) is an ideographic sign system: this means that every symbol stands for a concept. Some Blissymbols purportedly have an iconic ground; but with a type of iconicity which involves a high degree of stylization, and the frequent use of rhetoric symbolizations (e.g. a triangle for 'creation'). The basic graphic components used in Bliss, should they have an iconic origin or be borrowed from older, arbitrary sign systems (like figures or arrows), are combined to obtain more complex ideograms, fairly much like in the Chinese writing system by which Bliss has originally been inspired. In principle, Blissymbolics also has a specific morphology (e.g. indicators) or grammar (word order, relative clauses), which should make it into a proper language.

Minspeak uses a minimal set of signs which are not concepts by themselves; but the combination of two or three of them, within the specific context it creates, makes up a complete word. The Minspeak system is essentially morphographic. What we mean by this is that the predominant type of mnemonic assembling is the combination of meaning features however iconically present in the elements (e.g. 'apple' + 'night' $\rightarrow$ 'dinner', where 'apple', in this context, evokes the meaning feature *food*, and 'night' the meaning feature *late*, the combination of the two yielding the idea of *dinner*). Other mnemonic grounds are used, like rebus or homophony (Van Tatenhove and Micher 1997), but the meaning combination is the predominant one, making the elementary icons somehow equivalent to the *morphemes* of natural language — although they do not correspond to English morphemes. At the grammar level, on the contrary, when it comes to inflecting or assembling words, Minspeak is an exact, word to word copy of English grammar.

These three symbol systems could in theory be processed in their own specific way; Bliss in particular could be processed as an independant language having its own syntax. However, except in a few remarkable exceptions (see Subsection 2.3 below), the elements transmitted, displayed or recognized by the message receiver are English words left with no further processing.

This is why we argue that AAC has mainly tackled speech production disorders, with also some attempts to fight alexia (semantic level) by using icons instead of words.

AAC systems are also used in the common case where the neuromotor problems causing the speech disorder are also affecting hand movements. Possible solutions in such cases are a mechanical pointer making use of a fairly well controlled body part (e.g. headpointer), or a virtual keyboard displayed on the computer screen, with special selection strategies allowing a simple switch device to be used.

However, in most cases, the intelligence needed to compose and interpret the

message is entirely left to the care of the human sender and receiver. Language disorders involving agrammatism are not addressed.

2.3 The Role of NLP Techniques

There are three ways intelligence and linguistic knowledge can be added to an AAC system to improve its efficiency.

The first one consists in feedback from the system output to the symbol selection process: properly analysed, a partially composed message may give information on which letters/words are likely to come next. Word prediction techniques are now available on fairly reliable implemented systems (see for example the use of syntactic and semantic information in the ILLICO system (Pasero and Sabatier 1997)). These techniques allow faster word and sentence composition for reading subjects, but their helping agrammatic subjects is dubious (how do they overcome the first steps of sentence composition, when no guiding can yet take place but syntactic well-formedness is already expected?)

The second way consists in taking the grammatical order for granted, and inflecting the words according to their deduced grammatical role in the sequence. Systems based on this principle have been designed for French (SAHARA, MUTAVOX) and for Swedish, English and French (BLISSTALK). They all adopt the target natural language's syntax as the input syntax of the symbol sequences. A comparative study of the respective grammars of Bliss, Swedish, English and French ((Hunnicutt 1986)) has shown that using pure Bliss grammar is feasible, but that eventually natural language syntax seems to be preferred by the disabled—and at least by their circle. Systems of this type provide help at the morphological level; they can address the problem of subjects who cannot read or write, but have at least a rough knowledge of spoken language grammar. For the specific problem of the agrammatics, there is still the need for an alternative approach.

The third way consists in trying to *interpret* an ill-structured sequence. This is the only suitable approach to language deficits involving agrammatism. Here we want to process the user's utterance in order: (1) to understand its intended meaning on the basis of semantic contents—since no grammatical clues may be relied upon; and (2) yield a well-formed natural language sentence. This approach is still mainly at a research stage, and has led to prototypes only. A good example is the COMPANSION system, which translates lists of uninflected words into English sentences (Jones *et al.* 1991; Demasco and McCoy 1992). The PVI system will be described in Sections 4, 5 and 6 of this paper.

2.4 Towards an Additional Augmentative Communication Layer

Lloyd *et al.* (1990) proposed a model for AAC strategies which is a fairly good reflection of the "classical" trend in AAC. In their view, augmentation techniques may take place at three levels in the utterance production process: (1) at the symbol level, (2) at the selection level, and (3) at the transmission level. At each of these levels, communication may be unaided or aided.

1. Symbol production is unaided when the symbols used during the communication are produced by the user through controlled movements of his proper body (e.g. spoken words, signing). If artifacts need to be used (e.g. pictograms, printed letters or words), it is aided.
2. If communication is aided at the symbol level, it becomes relevant to distinguish between aided or unaided symbol selection. For example, if pictograms are used, selection is unaided when the user points at them using his finger, but aided if the user needs a mechanical pointer.
3. Lastly, symbol transmission is compulsorily unaided when symbol production is unaided (spoken words are simply perceived by the receiver's ear, signs by the receiver's eye); it is compulsorily aided when symbol production is aided (e.g., pictograms are displayed on a communication board or a computer screen).

This AAC interface model is represented in Figure 2.

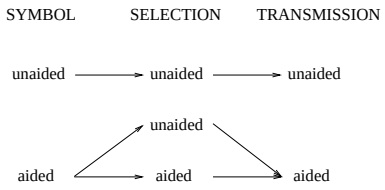

Fig. 2. The possible combinations of technical aid at three levels (based on (Lloyd *et al.* 1990), Figure 3, p. 176).

Firstly, we would argue that the *symbol level* and *transmission level* of this unaided/aided taxonomy are logically equivalent, and hence redundant. By this we mean that it is no coincidence that in all instantiated AAC systems, both levels are at the same time aided or unaided, but that aided transmission, in Lloyd *et al.*'s terminology, is merely a side effect of aided symbol production. As a matter of fact, the use of an artificial medium to engrave the symbols obviously implies that the reading of the symbols, at least, should take place on this artificial medium. For this reason, we suggest that the "transmission level" is not relevant for AAC systems characterization.

Secondly, we propose to add a new level to the AAC taxonomy, where aided communication techniques may take place: the *translation* level. Let us define aided translation as the use of Natural Language and Knowledge Representation techniques for automatic processing of the sequences of symbols produced by the user.

A new AAC interface model could then be such as represented on Figure 3.

Using this taxonomy, we could characterize communication situations in the following fashion:

1. U-U-U: natural communication, e.g. natural spoken language or sign language;
2. A-U-U: pointing at icons or photographs in a picture book, loose-leaf file or communication board;

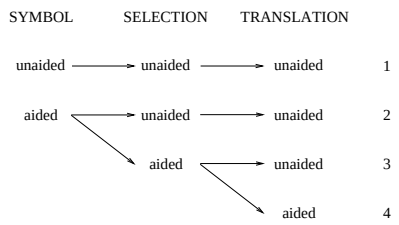

Fig. 3. The proposed AAC interface model

3. A-A-U: pointing at icons on a mechanic or electronic system, using specific pointing devices;
4. A-A-A: pointing at icons on a computer screen—the computer then translates the icon sequences into natural language sentences.

Many classical AAC experiments would fit into categories 2 and 3 of this model. The system we wish to describe falls into category 4.

3 User's Needs in the PVI project

The PVI (*Prothèse Vocale Intelligente*: Intelligent Voice Prosthesis) system was developed in the context of a research project involving 4 users. These were children aged $8\frac{1}{2}$, 10, 14 and $17\frac{1}{2}$ years, suffering from cerebral palsy, in a French-speaking environment.

Cerebral palsy involves permanent lesions of the central nervous system caused by pre- or perinatal accidents. These lesions cause neuromotor problems and may also cause cognitive deficits. The latter are all the more difficult to fathom as they are hidden by the former: global parameters (e.g. I.Q.) are significantly lower than in the average population, but it is difficult to evaluate the impact of the communication deficit on test results. The problem is the same when it comes to evaluating language disorders. Nevertheless, it has been shown (Gérard *et al.* 1987) that the language disorders are similar in their symptoms to the most common type of developmental dysphasia: the syntactical/phonological form, affecting levels 2 and 3 in the speech production process (see Figure 1). Data collected in speech therapy sessions with the subjects of this study seemed to give further evidence of agrammatism (short, telegraphic-style messages).

So, the system designed for the use of these subjects ideally ought to provide technical aid at the three levels of AAC interface: (1) aided symbol production, since they are unable to articulate speech—and preferably iconic symbols, since their ability to learn reading and writing skills is impaired; (2) aided sequence composition, since motor impairments hinder normal pointing gestures; and (3) aided translation of mixed, syntactically impoverished utterances into well-formed natural language sentences.

4 Architecture of the PVI System

These requirements were met by using a modular architecture, involving two parallel processes: a customizable interface adapted to the motor disability, and a translation program adapted to the agrammatic deficit (Figure 4). The two communicate on a client/server basis, the interface program addressing requests to the translation program.

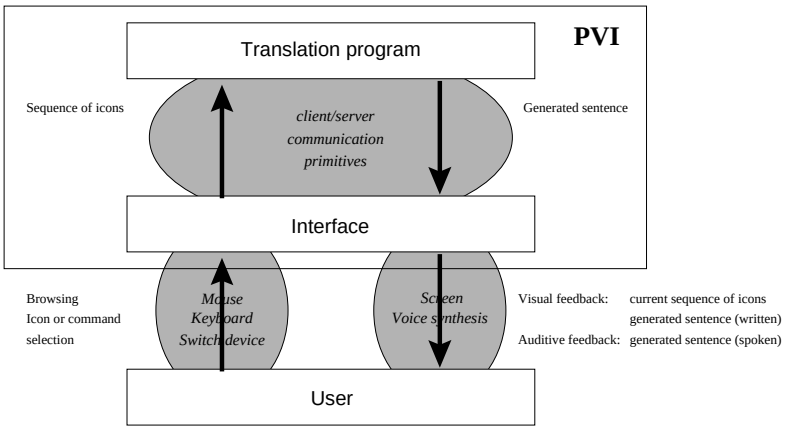

Fig. 4. The two processes in PVI

4.1 Interface

The graphical interface is a segmented window displaying icons (Figure 5). It deals with four data types: text, pictures, sounds and moving pictures. Internally, the icons may be of two types: symbols or actions. The symbols are transmitted to the translation program, whereas the actions directly command some parameters of the interface (e.g. change window, increase sound volume).

The windows are used to group homogeneous icons. Both the static aspects (position, size, colour of icons and windows, sound volume...) and the dynamic aspects (pointing method in use, editing commands...) of the interface are customizable.

Various selection methods have been implemented and are available for the design of each individual interface configuration. They may be selected to adapt to users' specific motor troubles.

Point and click: if the user is capable of controlled pointing movements, with a mouse or any other special input device (e.g. headpointer), classical cursor movement is used: clicking may be possibly realized by leaving the cursor for a given period of time.

Direct access: if the user is capable of using a keyboard with a large number of keys, keys on the keyboard can be mapped onto icons on the screen.

Scanning: if the user has reduced mobility and can only use a binary information device (such as a switch), the system has to make up for the missing information by using a motion automaton: since the user cannot move the cursor, the cursor is moved forward, scanning every square of the window in turn, and the user just has to validate the selected icon once the cursor is in front of it.

Scanning automata have varying degrees of sophistication: the cursor may move across rows or columns (every icon selection then requires 2 clicks), or even point to whole regions of the screen, zooming in with each selection until they come down to one square[1].

During the selection process, the user can see the current sequence of selected icons on a special window at the bottom of the screen. The icon currently under the cursor is displayed magnified in a zoom window located at the left bottom of the screen, thus allowing the user to make sure that he is about to select the right symbol. As for the system output, the character string yielded by the translation program is displayed in its written form on the screen, while it is sent in parallel to a voice synthesis device for audio output.

4.2 Translation Program

The translation program is itself composed of three serial components: a semantic analyser of the symbol sequences, yielding a conceptual graph that carries the message's meaning; a lexical choice component, which maps icons into French words; and a generation module, which builds syntactic trees out of the conceptual graph (Figure 6).

The components are presented in more detail in Sections 5 and 6 below.

5 Interpretation of Utterances' Meaning

The analysis module takes a sequence of icon identifiers as input, and translates it into a meaning representation, without making use of grammatical information.

As in any such analysis, the principle here is to build up the semantic relations between predicates and attributes, so as to arrive at a meaning representation where every agent is attributed its proper case role. In MAN/DRINK/WINE for example, the interpretation [DRINK: *agent*=MAN, *object*=WINE] must be obtained. What is specific to this system, compared to classical NLP analysers, is that no syntactic (word order) or morphological (inflectional) information is available to facilitate case attribution. Hence the only basis for interpretation has to be semantic information.

[1] For example, in a dichotomic scanning automaton over a 32-squares display, every icon selection requires 5 successive selections; but at each one of the 5 steps, the user is certain to make a quick selection.

5.1 Knowledge Representation

This information is collected in a semantic lexicon of concepts corresponding to the icons recognized by the system.

Following linguistic studies on relevant systemic description of semantic phenomena (Rastier 1987; Rastier *et al.* 1994), we have chosen to represent the semantic contents of the icons in the descriptive framework of differential semantics.

What we include in the lexicon is semantic primitives that represent the minimal differences relevant to the description of semantic interaction in a given corpus. Let us assume for example that we have a corpus composed of two sequences: CAT/EAT and DOG/EAT. There the description of CAT and DOG should be strictly equivalent (and void). If we want to add BARK to the lexicon, we are compelled to add a new semantic feature to distinguish a DOG from a CAT specifically in their interaction with the verb BARK. The name we give to the semantic feature, e.g. `may-bark` ($\in \{-1, +1\}$), is of no importance there: this approach has no ontological or lexicographical ambition; it is strictly operational and corpus-driven.

We have organized this lexicon on three levels (inspired by (Rastier 1987)):

1. The *domains* (e.g. //eating// or //playing//) factor general themes to avoid basic misinterpretations;
2. the *taxemes*, minimal categories of concepts able to play the same semantic role in standard contexts, gather semantic contents common to all their members (e.g. all vegetables may be eaten);
3. the *icons*, finally, have their specific contents (distinguishing them from other members of the same taxeme).

Semantic content is modelled by *semantic features*, meaning atoms which represent the relevant minimal differences. There are two types of semantic features:

1. *Intrinsic* semantic features are simple Attribute-Value structures. They carry information on the concept itself, considered individually.
2. *Extrinsic*, or *selectional*, semantic features are Attribute-Value structures attached to case roles (in a case grammar such as described by Fillmore (1968)). They carry information on the conditions that predicative concepts impose on their case-fillers.

The *extrinsic* features, despite their name, are part of the definition of a concept. They simply concern the concept considered in relation to its context—while *intrinsic* features concern the concept's kernel. As a good illustration, `<canine,+1>` is an intrinsic feature in the definition of 'dog', whereas `<agent,<canine,+1>>` is an extrinsic feature in the definition of 'bark' (still quite necessary to *bark*'s definition).

During the analysis process, all this information is available for every icon in the sequence: intrinsic as well as selectional features, belonging to the icon's specific contents or inherited from the taxeme level.

5.2 Analysis Algorithm

The basic analysis principle in PVI is to build the semantic structure of the message through adequate assignment of the predicates' agents. So, in an input sequence, the

first phase consists of spotting the predicative icons: those which have selectional features, i.e. an implicit case structure. Then the second phase consists of looking for the best candidates, in the sequence context, to fill the cases.

Hence the central question here is how to determine those "best" candidates. In other words, we need to define a notion of semantic compatibility, which will be the criterion used to score the value of case assignments.

The process then works by computing the best global assignment of icons to cases on the overall sequence.

The input to analyse is a sequence of iconic symbols s_1, s_2, ... s_n, each of which has a set of intrinsic features:

$$\mathcal{IF}(s_i) = f_i$$

(where f_i is a set of simple Attribute-Value semantic features, used to represent intrinsic features of the concept—like {<human,+1>,<male,+1>} for *Daddy*).

Some of the symbols also have selectional features, which if we group them by case type form a case structure:

$$\mathcal{CS}(s_i) = \{\langle c_1, f_{i1}\rangle, \langle c_2, f_{i2}\rangle, \ldots \langle c_N, f_{iN}\rangle\}$$

(where each of the N c_j is a case type such as *agent*, *object*, *instrument*..., and each f_{ij} a set of simple Attribute-Value semantic features, used to determine what features are *expected* from a given case-filler—e.g. <human,+1> is a feature that the *agent* of the verb *write* should possess).

Every couple $\langle c_j, f_{ij}\rangle$ present in the case structure means that f_{ij} is a set of Attribute-Value couples which are attached to s_i as selectional features for the case c_j:

$$\mathcal{SF}(s_i, c_j) = f_{ij} \iff \langle c_j, f_{ij}\rangle \in \mathcal{CS}(s_i)$$

For example, we can write:
$\mathcal{SF}$(*write*,*agent*)={<human,+1>}

The *semantic compatibility* is the value we seek to maximize to determine the best assignments. It is an asymmetric binary relation: it measures the degree of fit of an icon to a case place in the case structure of another icon.

Feature level: Atomic compatibility is measured at the feature level, between an intrinsic feature of the "candidate" icon and a selectional feature of the predicate. It gets a zero value when the two features are orthogonal (the attributes are different). If the attributes are the same, it is defined as the product of the values:

$$(1)\quad \begin{array}{lcrl} \mathcal{C}(\langle a_1, v_1\rangle, \langle a_2, v_2\rangle) & = & 0 & \text{if } a_1 \neq a_2 \\ \mathcal{C}(\langle a, v_1\rangle, \langle a, v_2\rangle) & = & +1 & \text{if } v_1 \text{ and } v_2 \text{ are equal integers} \\ & & -1 & \text{if } v_1 \text{ and } v_2 \text{ are distinct integers} \\ & & v_1.v_2 & \text{if one of the values is real} \end{array}$$

The values of selectional features are real coefficients. This modelling parameter allows the restricting power of these features to be "tuned" precisely. The values of

intrinsic features are signed integers ($\{-1,+1\}$ in the general case where the feature is a binary differential feature).

For example, the compatibility between `<human,+1>` and `<vehicle,+1>` is simply zero; but the compatibility between `<human,+1>` and `<human,+3.0>` is the real value 3, while the compatibility between `<human,+1>` and `<human,-1>` is -1.

Feature structure level: The semantic contents of the icons are represented by sets of features. Semantic compatibility is calculated between two homogeneous sets of Attribute-Value couples: on one side the selectional features attached to a given case slot of the predicate icon—stripped here of the case type—, on the other side the intrinsic features of the candidate icon.

The compatibility of the second set *with*[2] the first is defined as the sum of the semantic compatibilities of the features common to the two sets, divided by the total number of features in the first set:

$$\mathcal{C}(f_1, f_2) = \frac{\sum_{a|\langle a,v_1\rangle \in f_1, \langle a,v_2\rangle \in f_2} \mathcal{C}(\langle a, v_1\rangle, \langle a, v_2\rangle)}{\text{cardinal of } f_1} \tag{2}$$

Examples of calculation are given a few lines below, when the application to the case-slot level is explained.

Case slot level: At this level, we need to determine the degree of fit of a candidate icon s_j to a given case slot of a predicative icon s_i. This is measured by the semantic compatibility of the set of intrinsic features of s_j with the set of selectional features attached to s_i for the case c_k:

$$\mathcal{C}(\mathcal{SF}(s_i, c_k), \mathcal{IF}(s_j)) \tag{3}$$

This value represents the "conformity" of the intrinsic semantic contents of s_j with the restrictions that s_i imposes on its case-filler for case c_k. The unification of s_j into the case slot is dependent on it.

For example, if we want to match *Daddy* to the case *agent* of the verb *write*, the division will yield: $1/1 = 1$ ('human' matches; 'male' does not, but it does not count since it is not part of the features explicitly "expected" from the agent of *write*). Daddy is hence a fairly good candidate for the activity of writing. On the contrary, let's suppose we had a verb *give birth*. The list of features expected from the agent of this verb, $\mathcal{SF}$(*give birth*,*agent*), should be defined as {`<human,+1>`,`<male,-1>`}. Daddy would then be a rather poor candidate, since the result would be: $(1-1)/2 = 0$ ('animate' matches; 'male' matches too, but yields the negative result -1). Conversely Mummy, having the list of intrinsic features {`<human,+1>`,`<male,-1>`}, would be a far better candidate with a result of $(1+1)/2 = 1$.

For every predicative icon of the input sequence, we look for the best assignment of case-fillers. An *assignment* is a combination of possible slot fillings for predicate s_i, i.e. a mapping of the set of cases in the case structure of s_i:

$\{c_1, c_2, \ldots c_N\}$,

into a set of candidate icons from the input sequence:

[2] At this level the relation becomes asymmetric.

$\{s_{i1}, s_{i2}, \ldots s_{iM}\}$,
Let A be a mapping of this type:

(4) $$A = \{\langle c_x, s_{if(x)}\rangle\}, \text{ where } x \in [1, N] \text{ and } f(x) \in [1, M].$$

(For example, if we want to determine the case-fillers of the verb *write*, which has the cases *agent*, *recipient* and *instrument*, and we have the icons *Daddy* and *pencil* available in the input sequence, possible assignments can be: $\{\langle agent,Daddy\rangle, \langle recipient,pencil\rangle\}$, $\{\langle agent,Daddy\rangle, \langle instrument,pencil\rangle\}$, $\{\langle recipient,Daddy\rangle, \langle instrument,pencil\rangle\}$... and so on).

We seek to maximize a global value for this assignment. This global value is the sum of the values of all the single $\langle c_x, s_{if(x)}\rangle$ allocations that make up the assignment A:

(5) $$\begin{aligned} \mathcal{V}(A) &= \mathcal{V}(s_i, \{\langle c_1, s_{if(1)}\rangle, \langle c_2, s_{if(2)}\rangle, \ldots \langle c_N, s_{if(N)}\rangle\}) \\ &= \sum_{j \in [1,N]} \mathcal{V}(s_i, c_j, s_{if(j)}) \end{aligned}$$

The value of every single allocation, $\mathcal{V}(s_i, c_j, s_{if(j)})$, is not simply the value of the semantic compatibility of $s_{if(j)}$ with the case c_j of the predicate s_i (Expression 3). To evaluate the value of such an allocation within the context of an icon sequence, it is in fact necessary to take a factor into account: the *distance* of the candidate icon from the predicate. By *distance* here we mean the number of icons separating the candidate from the predicate in the input sequence (i.e. a distance on the expression plane, and in no way a kind of semantic relatedness).

As a matter of fact, the input sequences of PVI, even though agrammatic, take place in a linear continuum, and hence are subject to the most universal phenomena linked to this dimension of semiotic systems: the recency effect and its correlates. For example, when analyzing a long sequence such as DADDY/SEE/CAT/GOOD/EAT/MEAT, we need to know that CAT has to be preferred to DADDY as the subject of predicate GOOD because of its closeness—otherwise, outside any context, both CAT and DADDY have the same semantic compatibility with the adjective GOOD.

So, we define the value of the allocation of $s_{if(j)}$ to case c_j of predicate s_i as the semantic compatibility of $s_{if(j)}$ with case c_j of s_i multiplied by a coefficient which is a function[3] D of the distance between s_i and $s_{if(j)}$:

(6) $$\mathcal{V}(s_i, c_j, s_{if(j)}) = D(s_i, s_{if(j)}).\mathcal{C}(\mathcal{SF}(s_i, c_j), \mathcal{IF}(s_{if(j)}))$$

An assignment does not enforce systematic allocations for every possible case slot: some icon-to-slot allocations are tested and rejected if their values (in the sense of Equation 6) are lower than a given threshold T. A case slot may thus remain empty.

So, every assignment A computed by the PROLOG engine at this level, for every

[3] Decreasing, valued on $[0, 1]$.

predicative icon s_i in the input sequence, is a set of couples $\langle c_j, s_{if(j)} \rangle$ such that for every one of them:

$$\mathcal{V}(s_i, c_j, s_{if(j)}) = D(s_i, s_{if(j)}).\mathcal{C}(\mathcal{SF}(s_i, c_j), \mathcal{IF}(s_{if(j)})) > T$$

and the total value of A is:

$$\mathcal{V}(A) = \sum_{\langle c_j, s_{if(j)} \rangle \in A} \mathcal{V}(s_i, c_j, s_{if(j)}) \tag{7}$$

5.3 Semantic Harmony

An *interpretation* of the input sequence is a set of assignments corresponding to the predicative icons in the sequence. It is obtained by: (1) scanning the input sequence to find possible predicates, (2) computing the possible case assignments for each of them, and sorting these assignments by order of decreasing value (in the sense of Equation 7), (3) unifying the different assignments in a graph representing a possible global meaning of the message.

It can be considered that for a given interpretation of the sequence, the sum of the values of the assignments for every predicate constitute a sort of global "semantic harmony" of the interpretation: maximizing this value equates to having a good understanding of the message.

The PROLOG program computes the possible interpretations by combining the assignments in the order they come in. So if the possible assignments for the first predicate in the sequence are: A_{11}, A_{12}, $\ldots A_{1m_1}$; for the second predicate: A_{21}, A_{22}, $\ldots A_{2m_2}$; and so on until the nth predicate: A_{n1}, A_{n2}, $\ldots A_{nm_n}$; then the interpretations are yielded in this order:

$$\begin{array}{llll}
A_{11}, & A_{21}, & \ldots & A_{n1} \\
A_{11}, & A_{21}, & \ldots & A_{n2} \\
 & & & \vdots \\
A_{11}, & A_{21}, & \ldots & A_{nm_n} \\
 & \vdots & & \\
A_{1m_1}, & A_{2m_2}, & \ldots & A_{nm_n}
\end{array}$$

The order above, if it does not ensure a monotonic decrease in semantic harmony, ensures at least that the first yielded solution is an absolute maximum for this variable (being the sum of the local maxima).

6 French Sentence Generation Using TAGs

6.1 Lexical Choice

The output of the analysis component is an interpretation of the input sequence, i.e. a sequence of assignments of the $\langle s_i, \langle c_j, s_{if(j)} \rangle \rangle$ type. This format is topologically equivalent to a conceptual graph (Sowa 1984), with an extra piece of information

being carried by the order of the predicates in the linear representation, which reflects the topical order of the concepts in the source agrammatic speech. The interpretations [EAT: *agent*=CAT, *object*=BIRD]; [BEAUTIFUL: *qualifies*=BIRD] and [BEAUTIFUL: *qualifies*=BIRD]; [EAT: *agent*=CAT, *object*=BIRD], for example, which differ only by the order of the predicates and are topologically equivalent to the same conceptual graph, are respectively expressed by the messages: "*The cat eats the beautiful bird*" and "*The bird that the cat eats is beautiful*" (see Appendix for more examples).

The function of the lexical choice layer, which comes just before the generation module, is to convert the "iconic" conceptual graph into a "linguistic" conceptual graph. The distribution of meaning may in fact be different between the iconic symbol set and the natural language used for generation, due to differing implicit ontologies, or specialization in grammatical categories.

We have had fairly simple transfer mechanisms to implement, since the visual language we used for the iconic interface in the PVI system (see Figure 5), COMMUN-I-MAGE, was originally designed to map the meaning of French words. The lexical choice layer in the present version of the PVI system mainly does syntactic processing (it prepares the graph for the generation phase by adding uninstantiated morphosyntactic variables). However, the presence of this layer is an important element in ensuring the independence of the generation module from the analysis module. It could be redesigned to match the semantics of a new iconic language if need be. The alternative, keeping the analysis module as it is, but plugging a generation module for any other language than French—provided that it can take conceptual graphs as input—, would require no more than adaptation of the lexical choice layer.

6.2 Generation of Syntactic Trees

The "linguistic" conceptual graph is finally transformed into one or more French sentences. The first sentence is generated during a one-shot depth-first scan of the graph, following the case relations in a forward direction, and beginning from the first predicative node in the sequence (i.e. the first in the topical order).

Every node, as a lexical item, has an entry in a lexicalized grammar of a subsystem of the French language. For every entry in this lexical database, syntactic trees which represent canonical ways of putting the concept and its case-fillers into a French turn of phrase are stored. The syntactic trees are elementary trees of the TAG formalism (proposed by (Joshi *et al.* 1975); the principles of lexicalized TAGs may be found in (Schabes *et al.* 1988)). The lexical entry they are stored under appears as their anchor, and the possible case-fillers as places for substitution—an indexical structure, stored in parallel, gives the mapping between the places for substitution and the cases to which they correspond.

During the tree scanning process, at node n, the PROLOG engine will try in turn every tree lexicalizing n, and will try to apply recursively this process to the nodes under n, say, n_1^1 and n_2^1. The generation of the tree at the level of n then depends on the successful unification of the trees lexicalizing n_1^1 and n_2^1 on the right places

for substitution. If it fails, the next possible tree for n will be tested—and so until the list is exhausted. This process is illustrated in Figure 7.

Moreover, it may occur that the first scan of the graph has left parts of it not scanned, because of the impossibility of following the relational edges upstream. In other words, it may be impossible to find a connected spanning tree of the whole graph. In such cases, the remaining connected parts of the graph are scanned in turn, still following a topical order. Two strategies are tried out to express them; the first and preferred one is to find an auxiliary tree that could be adjoined to the first sentence, becoming for example an attributive adjective, or an auxiliary verb. The second one is to generate a new sentence. So, a conceptual graph like [SEE: *agent*=I, *object*=CAT]; [GOOD: *qualifies*=CAT] can be realized as "*I see the good cat.*", or, secondarily, "*I see the cat. The cat is good.*".

The agreement rules are represented by frozen PROLOG goals at the level of the syntactic structures within which they articulate. An agreement of the verb with its subject NP, for example, is a frozen goal, raised when first invoking the S node just above, but frozen until the morphosyntactic features of the NP are known. At this point, the morphosyntactic features of the verb (person, number) automatically follow.

The final sentence is obtained by concatenation of the terminal nodes of the tree, looking for the correct inflected forms in a morphological database.

7 Evaluation and Discussion

The first benchmark of the system was conducted on 200 sample input sequences, copied or adapted from real corpus sequences.

The original corpus sequences come from two production situations. The first one consists of spontaneous communication exercises performed by the users during speech therapy sessions. In these exercises, users select among about 60 icons or images displayed on a paper support (loose-leaf file), and the interpretation of the messages is left to the therapist (but the manually collected sequences simply contain the list of icons selected by the disabled user). The second situation consists of log files automatically stored during the use of another system, Mutavox. These files contain the list of lexical stems corresponding to the icons selected by the user.

The vocabulary in Mutavox (more than 300 icons) being larger than the one in the test version of this system (152 icons), some adaptations have been made, wherever the sequence pattern could be preserved, but the vocabulary had to be changed to fit in the lexicon of PVI (e.g. a sequence like "Mummy buy me album" has been changed to "Mummy give me book": since neither the word 'buy' nor the word 'album' were present in PVI's vocabulary, they have been replaced by equivalent concepts).

A file containing the 200 sample sequences was then fed to the system automatically. This benchmark was designed to test the translation program alone on formatted input.

The system output was classified into four categories: (I) Correct interpretation, correct generation; (II) Correct interpretation, clumsy (but comprehensible) genera-

tion; (III) Incomplete or not quite correct interpretation; (IV) Wrong or meaningless interpretation. Some sample results are presented in the appendix.

Benchmark results were the following: Category I: 147; Category II: 15; Category III: 15; Category IV: 18. These results mean that we may consider that the system score is 73.5% for correct processing, and 80.5% for acceptable answers. However, this score is not to be confused with a global acceptance rate of the system—which the user alone may determine.

On-site evaluation was led by therapists and carers at the Kerpape rehabilitation center (Brittany, France) from December 95 to May 96. A prototype, with a test version of the interface, containing 152 icons (among which 19 verbs), was tested by each of the four users during a one-month period. More qualitative feedback was gathered which indicated that more had to be done before the product could be acceptable to users.

First, it was stressed that a given error rate could seem less acceptable in real life than on a benchmark. Remembering the fact that a disabled person, accessing the symbols via a column/row scan mechanism, may take several minutes to compose a complete sequence, it is comprehensible that a rejection should be very frustrating to him. Ergonomic changes were suggested. For example, we have eventually, in a last version of the system, replaced the original error messages by simple linear transcriptions of the rough input sequences—which seem to be, although unsatisfactory, far better tolerated than the ("*Message not understood*")...

As for the translation program, a number of errors still appeared, mainly caused by the analysis module (3 cases on 34 were attributable to the generation component, and they were fairly easily corrected).

As a matter of fact, a remaining central problem in the PVI approach is the fragile balance between interpretation accuracy and expressive power. The constraints put on interpretation by the selection restrictions (modelled by the selectional features) may be too weak in some contexts, leading to inappropriate case assignments, but they may be too strong in other contexts, preventing the user from expressing anything other than what was prescribed in a lexical entry with a restrictive view of a concept's case structure. The misinterpretations are all consequences of an imbalance of this nature.

A crucial problem in many natural language systems is lexical ambiguity. This problem has also arisen in PVI, in the case of verbal concepts (as for the lexical categories representing more concrete vocabulary, the problem is marginal due to the restrictions of the context: iconic symbols stand for very precise and concrete objects of the real world). It has been solved by the implementation of homonymy: a single symbol on the screen may in fact point to two or three homonyms, and the system selects the concept which gives the highest semantic harmony in the best interpretation. This solution seemed to be sufficient for the restricted vocabulary used in the version described in this paper. It is not certain to prove satisfactory if the size of the verb lexicon increases. A possible solution to the multiplication of cases of verbal ambiguity could be the intervention of users, asked to chose between homonyms among different categories (taxemes)—but this has not been

implemented, and it is not sure that the extra cognitive load imposed on users would be easily accepted.

This takes us to a problem of prime concern: the growing complexity of the lexicon. 152 icons is not enough, even for a minimal vocabulary (it is generally admitted than a lexicon should have a size of about 300 or 400 items to allow conversation); hence upgrading this system to a real operational one would require augmenting the lexicon up to twice its size. Yet this is problematic: augmenting the lexicon in PVI is fairly simple as far as persons, pets, vegetables, or other members of well-defined categories are concerned. When it comes to defining new complex predicates (generally expressed by verbs in natural language), we are faced with defining the possible semantic interaction of this verb with icons already present in the lexicon: i.e., defining selectional features on the verb, but also sometimes defining additional intrinsic features on older icons, which may themselves interact with others, etc. To sum up, every new word added to the lexicon requires a new comprehensive test of a reference corpus to check for side effects, with a possible re-tuning at every cycle.

In a very rough approximation (if we consider that any new lexical item could be predicative), the size of the training corpus would then have to grow in proportion of the square of the size of the lexicon, which is a bit impractical. We believe that the problem could be solved by the design of an intelligent interface for lexicon edition, which would guide the user in the definition of new concepts but would only ask for relevant defining information, limiting it in function of the possibly interacting categories. Yet this would be a whole new research program.

8 Prospects for Future Expansion

We have described a system able to interpret sequences of pictograms with no grammatical (syntactical or morphological) clues, and to generate well-formed natural language sentences carrying their deduced meaning.

The aim of such a system is to help language-impaired persons suffering from agrammatism to communicate more easily with anybody: in their present condition, they devise communication codes with people who are close to them (parents, therapists), and from whom they receive very special understanding. Yet it is generally hard for them to get themselves understood by a new person. Natural language sentences coming out of a voice synthesis program represent a great step in the improvement of their communicative abilities, and improving this principle might be a new direction of research for AAC systems.

Possible extensions in this direction would first have to tackle the tricky problem of lexicon structure. In the PVI prototype, the vocabulary was too restricted to allow real life communication. A device adapted to these needs would have to include a great number of common concepts, especially verbs, which may prove to be difficult to define in interaction with the whole lexicon. Studies on big lexical databases could profitably be applied to this field, by helping to define a sophisticated lexicon access layer, controlling the coherence and completeness at every

augmentation cycle. A new project cycle is beginning this year, and special focus will be placed on verbal semantics.

Further possible extensions could also try to take into account contextual phenomena. In our view, this could well constitute a very fertile research path. Contextual modelling would help resolve a number of sequences that the system in its present state is unable to interpret: ellipses in particular, which are rather frequent in disabled users' utterances (probably due to the cost, in terms of time and effort, of selecting an icon that seems implicitly obvious).

Other applications might be imagined for such a program. It might help people to be understood in a foreign country: the grammatical differences between the origin language and the target language would be no obstacle here, since the system takes no account of word order or grammatical words. The reliability in this system is lower than in up-to-date machine translation programs, but it should be noted that (1) not all languages are included in machine-translation systems—especially minority languages, or languages from countries not involved in the development of such technology; and (2) not all people are literate and able to use alphabetic-entry systems. The possibility of changing the generation module without changing the analysis module would facilitate the adaptation of this principle to target languages other than French.

Acknowledgements

The PVI project was funded by Thomson-CSF and AGEFIPH (French National Fund for Professional Integration of the Disabled) in the framework of a research and development program for technological aids for the handicapped. It has involved constant help and cooperation from the Rehabilitation Centre of Kerpape (Brittany), and we would especially like to thank Jean-Paul Departe, who has worked on the project. The graphical interface was implemented in its entirety by Michaël Checler, and the author of this paper has done nothing but report his work. The present version of this paper owes much to the reviewers of the journal, who showed where information could be added, or where concepts had to be better explained. Lastly, we owe thanks to Ariane Halber, Jean-Michel Grandchamp and Pierre Saveant for reading and commenting on a previous version of this article; and to Huw Sanderson, who did his best to translate this paper from Frenglish into English.

References

A.P.A. (American Psychiatric Association). 1987. *Diagnostic and Statistical Manual of Mental Disorders*. Washington, DC. 3^{rd} ed.

Demasco, P. W. and McCoy, K. F. 1992. Generating Text from Compressed Input: An Intelligent Interface for People with Severe Motor Impairments. *Communications of the ACM*, **35**(5):68–78.

Eustache, F. and Lechevalier, B. (eds.). 1993. *Langage et aphasie : séminaire Jean-Louis Signoret*. Brussels: De Boeck–Wesmael.

Fillmore, C. J. 1968. The Case for Case. In Bach, E. W. and Harms, R. T.(eds.), *Universals in linguistic theory*, pp. 1–90. New York: Holt, Rinehart and Winston.

Gérard, C.-L., Dugas, M., and Lacert, P. 1987. Dysphasia and Early Focal Brain Injury. Hypothesis for Postlesional Cerebral Reorganization. In *First international symposium on specific speech and language disorders in children*, University of Reading (United Kingdom).

Gérard, C.-L. 1993. *L'enfant dysphasique.* Brussels: De Boeck–Wesmael.

Hehner, B. 1980. *Blissymbols for use.* Toronto: The Blissymbolics Communication Institute.

Hunnicutt, S. 1986. Bliss Symbol-to-Speech Conversion: BLISSTALK. *Journal of the American Voice I/O Society*, **3**:19–38.

Jones, M., Demasco, P., McCoy, K., and Pennington, C. 1991. Knowledge Representation Considerations for a Domain Independent Semantic Parser. In Presperin, J. J. (ed.), *Proceedings of the 14 th annual RESNA conference.* Washington, DC: RESNA Press.

Joshi, A. K. and Levy, L. S. and Takahashi, M. 1975. Tree Adjunct Grammars. *Journal of Computer and System Sciences.*

Lloyd, L. L., Quist, R. W., and Windsor, J. 1990. A proposed Augmentative and Alternative Communication Model. *AAC Augmentative and Alternative Communication*, pp. 172–183.

Pasero, R. and Sabatier, P. 1997. Concurrent Processing for Sentences Analysis, Synthesis and Guided Composition. To appear. Lecture Notes in Computer Science. Berlin: Springer Verlag.

Rastier, F. 1987. *Sémantique interprétative.* Formes sémiotiques. Paris: Presses Universitaires de France.

Rastier, F., Cavazza, M., and Abeillé, A. 1994. *Sémantique pour l'analyse.* Sciences Cognitives. Paris: Masson.

Saffran, E. M., Schwartz, M. F., and Marin, O. S. M. 1980. The Word Order Problem in Agrammatism. *Brain and language*, **10**(2):249–280.

Schabes, Y. and Abeillé, A. and Joshi, A. K. 1988. Parsing Strategies with Lexicalized Grammars: Application to TAG. In *Proceedings of the 12 th International Conference on Computational Linguistics (COLING)*, Budapest.

Sowa, J. F. 1984. *Conceptual structures: information processing in mind and machine.* New York: Addison Wesley.

Steele, R., Weinrich, M., Wertz, R., Kleczewska, M., and Carlson., G. 1989. Computer-Based Visual Communication in Aphasia. *Neuropsychologia*, **27**:409–426.

Van der Lely, H. K. J. 1994. Canonical Linking Rules: Forward versus Reverse Linking in Normally Developing and Specifically Language-Impaired Children. *Cognition*, (51):29–72.

Van Tatenhove, G. and Micher, J. 1997. *Frequently Asked Questions about Minspeak* [online]. Available at URL : `http://kaddath.mt.cs.cmu.edu:80/scs/faq.html`. Document accessed on 4 March 1997.

Appendix: sample I/O of the PVI system

Examples of correct results (category I):

```
Squence : moi.manger.poulet.viande.nil
Graphe  : manger(agent(moi).objet(poulet).objet(viande).nil).nil
Phrase  : Je mange le poulet et la viande. (1366 ms)
```

(I/EAT/CHICKEN/MEAT → *"I eat the chicken and the meat."*)

```
Squence : moi.donner.toi.ballon.nil
Graphe  : donner(agent(moi).recepteur(toi).objet(ballon).nil).nil
Phrase  : Je te donne le ballon. (616 ms)
```

(I/GIVE/YOU/BALL → *"I give you the ball."*)

```
Squence : moi.aller.plage.maman.papa.beatrice.nil
Graphe  : aller(agent(moi).destination(plage).agent(maman).agent(papa).\
agent(beatrice).nil).nil
Phrase  : Moi, Maman, Papa et Batrice allons  la plage. (7183 ms)
```

(I/GO/BEACH/MUMMY/DADDY/BEATRICE → *"I, Mummy, Daddy and Beatrice go to the beach."*)

```
Squence : docteur.gentil.non.nil
Graphe  : gentil(qualifie(docteur).nil).non(portee(gentil).nil).nil
Phrase  : Le docteur n'est pas gentil. (250 ms)
```

(DOCTOR/NICE/NO → *"The doctor is not nice."*)

```
Squence : chat.manger.oiseau.nil
Graphe  : manger(agent(chat).objet(oiseau).nil).nil
Phrase  : Le chat mange l'oiseau. (584 ms)
```

(CAT/EAT/BIRD → *"The cat eats the bird."*)

```
Squence : chat.oiseau.manger.nil
Graphe  : manger(agent(chat).objet(oiseau).nil).nil
Phrase  : Le chat mange l'oiseau. (517 ms)
```

(CAT/BIRD/EAT → *"The cat eats the bird."*)

```
Squence : oiseau.manger.chat.nil
Graphe  : manger(agent(chat).objet(oiseau).nil).nil
Phrase  : Le chat mange l'oiseau. (567 ms)
```

(BIRD/EAT/CAT → *"The cat eats the bird."*)

```
Squence : moi.manger.spaghetti.nil
Graphe  : manger(agent(moi).objet(spaghetti).nil).nil
Phrase  : Je mange les spaghetti. (450 ms)
```

(I/EAT/SPAGHETTI → *"I eat the spaghetti."*)

```
Squence : moi.manger.fourchette.nil
Graphe  : manger(agent(moi).instrument(fourchette).nil).nil
Phrase  : Je mange avec la fourchette. (950 ms)
```

(I/EAT/FORK → *"I eat with the fork."*)

```
Squence : moi.manger.viande.fourchette.nil
Graphe  : manger(agent(moi).objet(viande).instrument(fourchette).nil).nil
Phrase  : Je mange la viande avec la fourchette. (950 ms)
```

(I/EAT/MEAT/FORK → *"I eat the meat with the fork."*)

```
Squence : moi.donner.chat.viande.nil
Graphe  : donner(agent(moi).recepteur(chat).objet(viande).nil).nil
Phrase  : Je donne la viande au chat. (467 ms)
```

(I/GIVE/CAT/MEAT → *"I give the meat to the cat."*)

```
Squence : moi.donner.chat.papa.nil
Graphe  : donner(agent(moi).recepteur(papa).objet(chat).nil).nil
Phrase  : Je donne le chat  Papa. (650 ms)
```

(I/GIVE/CAT/DADDY → *"I give the cat to Daddy."*)

```
Squence : chat.manger.oiseau.gentil.nil
Graphe  : manger(agent(chat).objet(oiseau).nil).gentil(qualifie(oiseau)\
.nil).nil
Phrase  : Le chat mange le gentil oiseau. (1517 ms)
```

(CAT/EAT/BIRD/NICE → *"The cat eats the nice bird."*)

```
Squence : oiseau.gentil.manger.chat.nil
Graphe  : gentil(qualifie(oiseau).nil).manger(agent(chat).objet(oiseau)\
.nil).nil
Phrase  : L'oiseau que le chat mange est gentil. (1350 ms)
```

(BIRD/NICE/EAT/CAT → *"The bird which the cat eats is nice."*)

Examples of incorrect results:

Category II (correct interpretation, clumsy generation):

```
Squence : infirmiere.vouloir.moi.toilettes.nil
Graphe  : vouloir(agent(infirmiere).objet(toilettes).nil).toilettes(age\
nt(moi).nil).nil
Phrase  : L'infirmire veut que je vais aux toilettes. (2533 ms)
```

(NURSE/WANT/ME/TOILETS → *"The nurse wants that I go to the toilets."*)
(Here, in French, the subjunctive should have been used: "*L'infirmire veut que j'aille aux toilettes.*")

Category III (incomplete or incorrect interpretation):

```
Squence : moi.promenade.plage.nil
Graphe  : promenade(agent(moi).nil).plage(sur(moi).nil).nil
Phrase  : Je me promne. Je suis sur la plage. (250 ms)
```

(I/STROLL/BEACH → *"I am strolling. I am on the beach."*)
(Here, the user would expect the result "*I am strolling on the beach.*")

Category IV (wrong or meaningless interpretation):

```
Squence : posseder.docteur.toi.ballon.nil
Graphe  : posseder(agent(docteur).objet(toi).objet(ballon).nil).nil
Phrase  : Toi et le ballon tes au docteur. (483 ms)
```

(OWN/DOCTOR/YOU/BALL → *"You and the ball belong to the doctor."*)
(Here, the system simply fails to allocate YOU to the good case, and yields the meaningless result quoted above instead of the expected "*The ball belongs to the doctor and to you.*")

Fig. 5. Interface display in the test version of PVI. This version can display 152 icons; column/row scanning is in use.

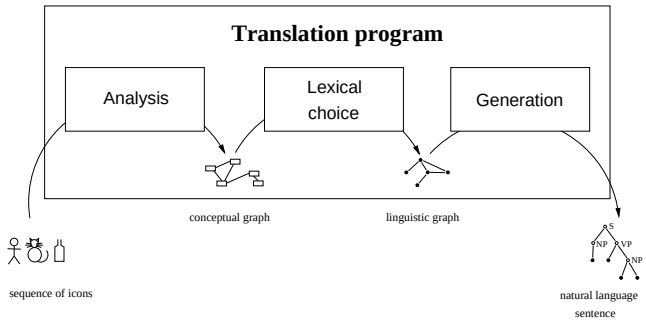

Fig. 6. The modules in the translation program

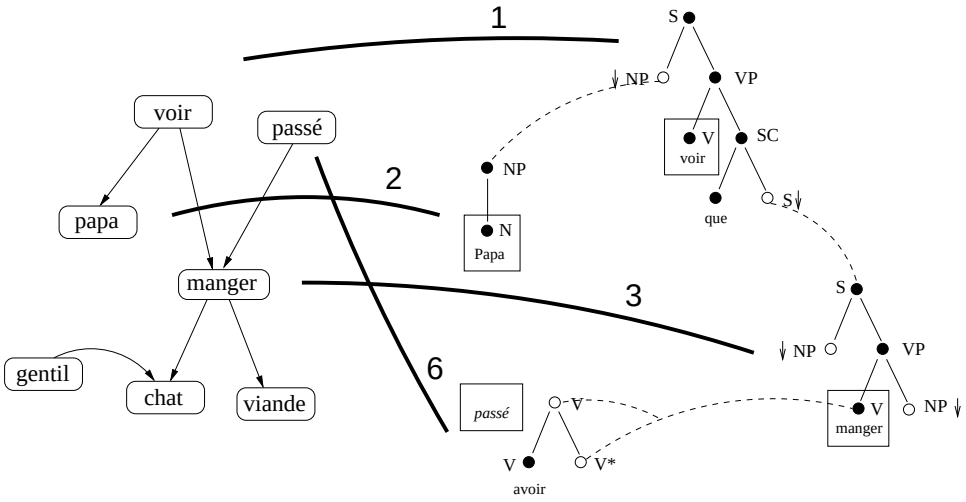

Fig. 7. The generation process for the graph [SEE: *agent*=DADDY, *object*=EAT]; [EAT: *agent*=CAT, *object*=MEAT]; [PAST: *when*=EAT]; [GOOD: *qualifies*=CAT] (*Daddy sees that the good cat has eaten the meat*). 1 is the first sketch of the main sentence. 2 and 3 are substitutions. 6 is an adjunction.