JCSCR
Journal of Computer Science & Research

Journal of Computer Science & Research (JCSCR) - ISSN 2227-328X
http://www.jcscr.com
Vol. 1, No. 3, Pages. 50-57, June 2012

Natural Language Generation: An Overview

Paul Semaan

LACSC – Lebanese Association for Computational Sciences
Registered under No. 957, 2011, Beirut, Lebanon

Abstract

In this paper, we are discussing the basic concepts and fundamentals of Natural Language Generation, a field in Natural Language Engineering that deals with the conversion of non-linguistic data into natural information. We will start our investigation by introducing the NLG system and its different types. We will also pin point the major differences between NLG and NLU also known as Natural Language Understanding. Afterwards, we will shed the light on the architecture of a basic NLG system, its advantages and disadvantages. Later, we will examine the different applications of NLG, showing a case study that illustrates how an NLG system operates from an algorithmic point of view. Finally, we will review some of the existing NLG systems together with their features, taken from the real world.

Keywords

Natural Language Generation, NLG System, Computational Linguistics

1. Natural Language Generation

NLG or Natural Language Generation is the process of constructing natural language outputs from non-linguistic inputs. One of the central goals of NLG is to investigate how computer programs can be made to produce high-quality, expressive, uncomplicated, and natural language text from computer-internal sophisticated representations of information [1].

2. NLG vs. NLU

NLG is the inverse of NLU (Natural Language Understanding) or NLI (Natural Language Interpretation), in that NLG maps from meaning to text; while, NLU maps from text to meaning [2]. NLG is easier then NLU because a NLU system cannot control the complexity of the language structure it receives as input while NLG links the complexity of the structure of its output. Table 1 delineates the differences between NLG and NLU.

Table 1 – NLG vs. NLU

NLG	NLU
Relatively Unambiguous	Ambiguity in input
Well-formed	ill-formed input
Well-specified	Under-specification

3. NLG System

- Goal: Computer software which produces understandable and appropriate texts in English or other human languages.
- Input: some underlying non-linguistic representation of information.
- Output: Documents, reports, explanations, help messages, and other kinds of texts.
- Knowledge sources required: knowledge of language and of the domain.

4. Types of NLG Systems

There exist different types of NLG systems starting with the simplest ones - the canned text and template filling systems, to end with sophisticated systems that adapt to realistic changes and variations in the information of a particular domain [3].

4.1. Canned Text

The process to generate text can be as simple as keeping a list of canned text that is copied and pasted, possibly linked or concatenated with some glue text. The results may be satisfactory in simple domains such as

JCSCR
Journal of Computer Science & Research

Journal of Computer Science & Research (JCSCR) - ISSN 2227-328X
http://www.jcscr.com
Vol. 1, No. 3, Pages. 50-57, June 2012

horoscope machines or generators of personalized business letters. Canned Text NLG type systems are easy to implement, but are unable to adapt to new situations without the intervention of a programmer [4].

4.2. Template Filling

In this approach, you fill a template by entering data into slots and fields, and a natural statement is generated. Junk mail is generated using template filling systems in which a mail is sent with addressee name in the right place. Template filling is easy to implement but not flexible enough to handle applications with any realistic variation in the information being expressed or in the context of its expression. Figure 1 depicts the architecture of a template filling type NLG system.

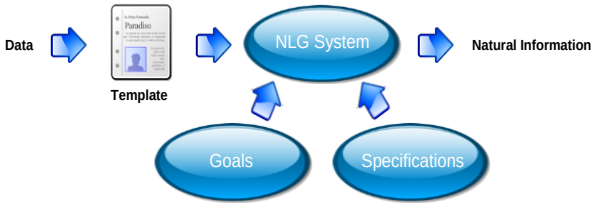

Figure 1 – Template Filling NLG System

4.3. Advanced NLG Systems

As stated previously, canned text and template filling systems are not that flexible to deal with emerging situations and real word problems. Therefore, new NLG systems were investigated in order to solve complex and advanced problems. Those new NLG systems must take the following choices [5]:

Content Selection: The system must choose the appropriate content to express and generate natural output based on a specific communicative goal.

Lexical Selection: The system must choose the lexical items most appropriate for expressing particular concepts.

Sentence Structure:

Aggregation: The system must generate phrases, clauses and sentence-sized chunks.

Referring Expressions: The system must determine how to refer to the objects being discussed

Discourse Structure: The system must deal with multi-sentence discourse which has a coherent structure.

5. NLG System Architecture

A modern architecture for NLG systems comprises a knowledge base, a discourse planner, and a surface realizer. The discourse planner selects from a knowledge pool which information to include in the output, and creates a text structure to ensure coherence. On a more local scale, the planner organizes the content of each sentence and orders its parts. The surface realizer is fed by the discourse specification in order to convert sentence-sized chunks of representation into grammatically correct sentences [6]. Figure 2 shows the basic architecture of an NLG system.

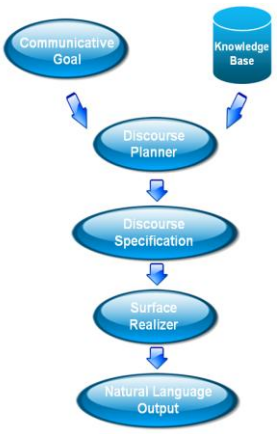

Figure 2 - NLG System Architecture

JCSCR
Journal of Computer Science & Research

Journal of Computer Science & Research (JCSCR) - ISSN 2227-328X
http://www.jcscr.com
Vol. 1, No. 3, Pages. 50-57, June 2012

5.1. Knowledge Base

It contains all information of a specific domain. It is a large general-purpose knowledge base that acts as support for domain-specific application which would help to speed up and enhance generator porting and testing on new applications.

5.2. Communicative Goal

It designates the intended audience who is going to use the system. The stylistic variations serve to express significant interpersonal and situational meanings (text can be formal or informal, slanted or objective, colorful or dry, etc.)

5.3. Discourse Planner

It selects the content from the knowledge base and then structures that content appropriately. The result is a specification for all choices made for the entire communication, potentially spanning multiple sentences and including other annotation. In other words the discourse planner takes a specified input and generates linear chunks of information. The two approaches used by discourse planners are Text Schemata and Rhetorical Relations [7].

5.3.1 Text Schemata

It is a mechanism based on expressing expressions as different high-level procedures similar to states in order to structure the output.

5.3.2 Rhetorical Relations

It is based on RTS (Rhetorical Structure Theory) which designates a central segment of text called nucleus and a more peripheral segment called the satellite. RST relations are defined in terms of the constraints they place on nucleus, on the satellite and on the combination of the nucleus and satellite [8].

5.4. Surface Realizer

It receives the fully specified discourse plan and generates individual sentences as contained by its lexical and grammatical resources. In other words the surface realizer converts text specifications into actual natural text. The different linguistic realizations involved in surface realization process are the following:

- Insert function words
- Choose correct inflection of content words
- Order words within a sentence
- Apply orthographic rules

The two approaches used by surface realizers are Systemic Grammar and Functional Unification Grammar.

5.4.1 Systemic Grammar

It represents sentences as collections of functions and maintains rules for mapping those functions onto explicit grammatical forms. In Table 2, the one who is doing the action is the subject I and the action (verb) or the process being committed by the actor is eat and finally the object acted upon is the sandwich [9].

Table 2 – Systemic Grammar Example

Sentence	**I**	**eat**	**sandwich**
Mood	Subject	Predictor	Object
Transitivity	Actor	Process	goal

5.4.2 Functional Unification Grammar

It is based on features grammar where the basic idea is to build the generation grammar as a feature structure with a list of all possible alternations and then unify this grammar with an input specification built using the same sort of feature structure. Below is a possible feature matrix.

JCSCR
Journal of Computer Science & Research

Journal of Computer Science & Research (JCSCR) - ISSN 2227-328X
http://www.jcscr.com
Vol. 1, No. 3, Pages. 50-57, June 2012

$$\begin{bmatrix} \text{CAT} & \text{NP} \\ \text{NUMBER} & \text{SG} \\ \text{PERSON} & 3 \end{bmatrix}$$

6. Applications of NLG Systems

- **Database Content Display:** The description of database contents in natural language is not a new problem, and some such generators already exist for specific databases. The general solution still poses problems, however, since even for relatively simple applications it still includes unsolved issues in sentence planning and text planning.
- **Expert System Explanation:** This is a related problem, often however requiring more interactive ability, since the user's queries may not only elicit more information from a (static, and hence well-structured) database, but may cause the expert system to perform further reasoning as well, and hence require the dynamic explanation of system behavior, expert system rules, etc. This application also includes issues in text planning, sentence planning, and lexical choice.
- **Speech Generation:** Simplistic text-to-speech synthesis systems have been available commercially for a number of years, but naturalistic speech generation involves unsolved issues in discourse and interpersonal pragmatics (for example, the intonation contour of an utterance can express dislike, questioning, etc.). Today, only the most advanced speech synthesizers compute syntactic form as well as intonation contour and pitch level.
- **Limited Report and Letter Writing:** As mentioned in the previous section, with increasingly general representations for text structure, generator systems will increasingly be able to produce standardized multi-paragraph texts such as business letters or monthly reports. The problems faced here include text plan libraries, sentence planning, adequate lexicons, and robust sentence generators.
- **Automated document production:** Such as weather forecasts, simulation reports, letters etc.
- **Presentation of information to people in an understandable fashion:** Such as medical records, expert system reasoning etc.

7. Case Study: Weather Forecast

In this case study, we will discuss the specifications of a specific NLG system for weather forecasting showing the different phases needed to transform specifications text into natural output text. Figure 3 depicts the weather forecast NLG system structure [10].

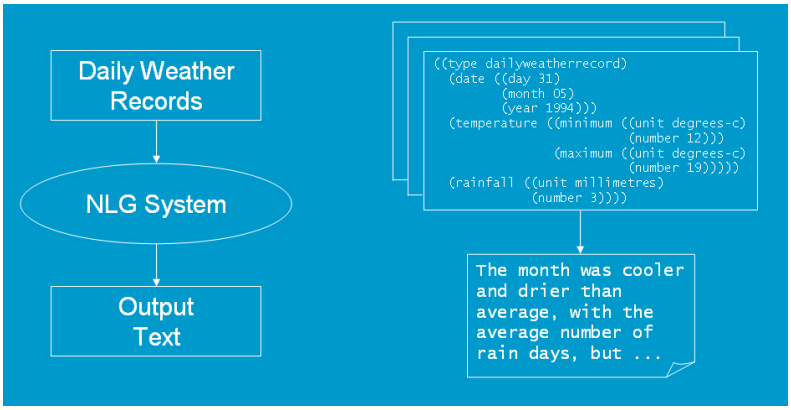

Figure 3 – Weather Forecast NLG System Structure

7.1. Specifications

- Goal: Produce understandable natural texts in English to indicate weather situations
- Input: Special commands or syntax representation of information.

JCSCR
Journal of Computer Science & Research

Journal of Computer Science & Research (JCSCR) - ISSN 2227-328X
http://www.jcscr.com
Vol. 1, No. 3, Pages. 50-57, June 2012

- Output: Report of natural English texts.
- Knowledge sources required: knowledge of the English language and of the domain of weather

7.2. Phases

The Discourse Planner takes as input the language commands and generates different chunks of information, classified in a tree-like structure which is depicts in Figure 4.

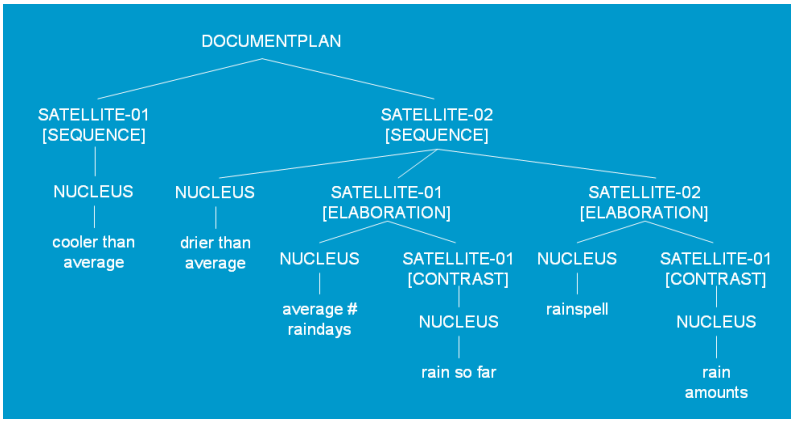

Figure 4 – Discourse Planner Results

The Surface Realizer takes as input the leaves of the tree produced previously and generates single grammatically correct natural sentences.

The month was cooler than average.
The month was drier than average.
There were the average numbers of rain days.
The total rain for the year so far is well below average.
There was rain on every day for 8 days from 11th to 18th.
Rainfall amounts were mostly small.

The Surface Realizer will process then the above sentences and produces a coherent English natural text paragraph.

The month was cooler and drier than average, with the average number of rain days, but the total rain for the year so far is well below average.
Although there was rain on every day for 8 days from 11th to 18th, rainfall amounts were mostly small.

8. Existing NLG Systems

In this section, we are presenting some of the existing NLG systems, taken from the real world.

8.1. FoG

- Function: Produces textual weather reports in English and French
- Input: Graphical/numerical weather depiction
- User: Environment Canada (Canadian Weather Service)
- Developer: CoGenTex
- Status: Fielded, in operational use since 1992

Figure 5 shows the input of FoG; while, Figure 6 shows its output.

JCSCR
Journal of Computer Science & Research

Journal of Computer Science & Research (JCSCR) - ISSN 2227-328X
http://www.jcscr.com
Vol. 1, No. 3, Pages. 50-57, June 2012

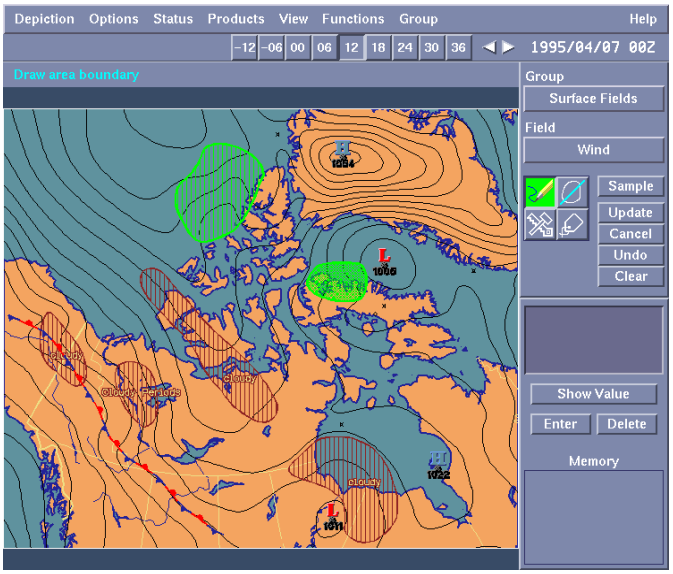

Figure 5 – FoG Non-Linguistic Input

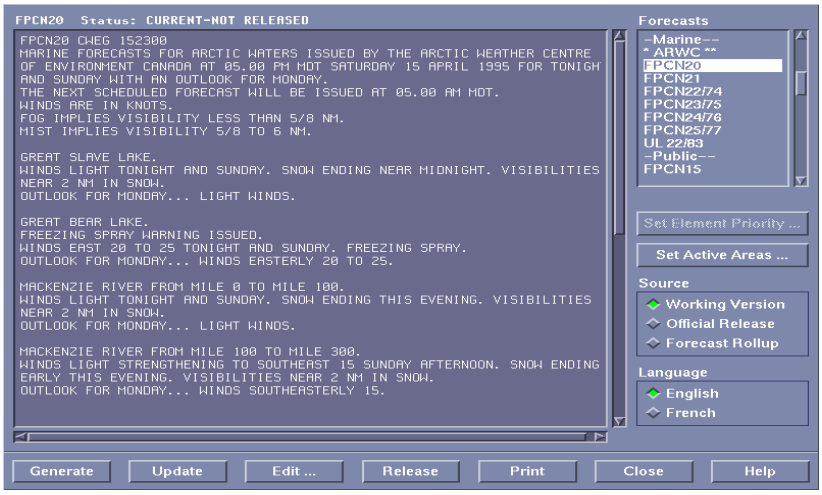

Figure 6 – FoG Natural Text Output

8.2. STOP System

- Function: Produces a personalized smoking-cessation leaflet
- Input: Questionnaire about smoking attitudes, beliefs, history
- User: NHS (British Health Service)
- Developer: University of Aberdeen

Figure 7 shows the input of STOP; while, Figure 8 shows its output.

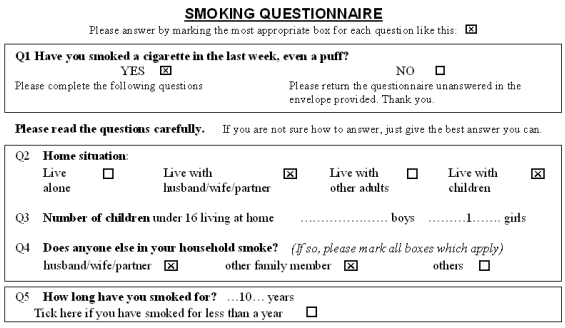

Figure 7 – STOP Questionnaire

JCSCR
Journal of Computer Science & Research

Journal of Computer Science & Research (JCSCR) - ISSN 2227-328X
http://www.jcscr.com
Vol. 1, No. 3, Pages. 50-57, June 2012

Dear Ms Cameron

Thank you for taking the trouble to return the smoking questionnaire that we sent you. It appears from your answers that although you're not planning to stop smoking in the near future, you would like to stop if it was easy. You think it would be difficult to stop because *smoking helps you cope with stress; it is something to do when you are bored, and smoking stops you putting on weight*. However, you have reasons to be confident of success if you did try to stop, and there are ways of coping with the difficulties.

Figure 8 – STOP Natural Output

8.3. Loughaty

- Function: Generator of natural programming instructions
- Input: Template wizards, you fill in to generate programming instructions
- Usage: Learning the basic concepts of programming

Figure 9 shows the input of Loughaty; while, Figure 10 shows its output.

Figure 9 – Loughaty's Fill-in Template

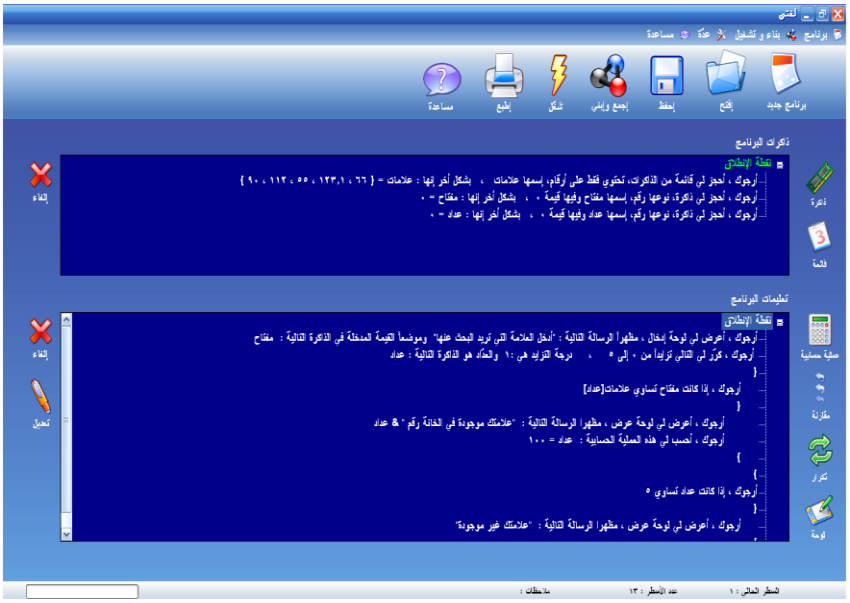

Figure 10 – Loughaty's Generated Natural Instructions

JCSCR
Journal of Computer Science & Research

Journal of Computer Science & Research (JCSCR) - ISSN 2227-328X
http://www.jcscr.com
Vol. 1, No. 3, Pages. 50-57, June 2012

Acknowledgment

This research was funded by the Lebanese Association for Computational Sciences (LACSC), Beirut, Lebanon, registered under Decree No. 957, 2011, Beirut, Lebanon.

References

[1] Ehud Reiter and Robert Dale, "Building Natural Language Generation Systems", Cambridge University Press, 1999.

[2] Daniel Jurafsky and James H. Martin, "Speech and Language Processing", 2nd ed., Pearson Prentice Hall, 2008.

[3] Harris MD, "Building a Large-Scale Commercial NLG System for an EMR", Proceedings of the Fifth International Natural Language Generation Conference, pp. 157–60, 2008.

[4] Christopher D. Manning and Hinrich Schütze, "Foundations of Statistical Natural Language Processing", The MIT Press, 1999.

[5] Warschauer, M., & Healey, D., "Computers and language learning: An overview", Language Teaching, vol. 31, pp. 57-71, 1998.

[6] Bates, M., "Models of natural language understanding", Proceedings of the National Academy of Sciences of the United States of America, vol. 92, no. 22, pp. 9977–9982, 1995.

[7] Iosias Jody, Natural Language Generation, Cred Press, 2012.

[8] Elke Teich, Systemic Functional Grammar & Natural Language Generation, Continuum Press, 1999.

[9] Laurence Danlos, The Linguistic Basis of Text Generation, Cambridge University Press, 2009.

[10] Goldberg E, Driedger N, Kittredge R, "Using Natural-Language Processing to Produce Weather Forecasts", IEEE Expert, vol. 9, no.2, pp. 45–53, 1994.