Functional Grammar Processing for English & Chinese Texts[1]

Jonathan J Webster

Principal Lecturer
Department of Applied Linguistics
City Polytechnic of Hong Kong

Introduction

The Functional Grammar Processor (FGP) is a software tool designed to assist with the analysis of texts following the approach of M.A.K. Halliday. At present, the FGP for English texts runs on any IBM PC AT compatible and is fully integrated with Sprint, a popular word processing package from Borland International. Under a strategic research grant from the City Polytechnic of Hong Kong, work is proceeding toward implementation of a Chinese version of the FGP. In this paper, I will explain the operation of the Functional Grammar Processor as it has been implemented so far for English texts. In addition, I will discuss the proposal for an FGP for Chinese texts.

The Functional Grammar Processor - a meaning processor

The FGP is a 'meaning processor'; it is not an automatic parser. I would like to draw an analogy between the FGP as a meaning processor and that kind of software we are all so familiar with - a word processor. A word processor assists the writer by making the task of writing easier to accomplish. Of course, the writer must still know what (s)he wants to communicate, the computer does not itself create the document. Similarly, the Functional Grammar Processor assists the user in analyzing text; it also facilitates the subsequent retrieval of information about the text by collecting all the clause analyses into a global database. However, the user must still determine the meaningful elements that make up the text. The computer does not interpret meaning - perhaps that will come later. For now, it only records the interpretation made by the user. The user must know how to analyze the text, the computer does not itself understand the text.

Halliday's approach to analyzing language, called functional grammar, identifies the structures at

clause level that contribute to the meaning of a text. Analysis along these lines shows 'how and why' a text means what it does. Halliday identifies three components of meaning or semantic functions: textual, interpersonal, and ideational. Halliday describes a clause as a complex realization of these three functional-semantic components. Each contributes in its own way to the form of the clause. Corresponding to each component there is some form of structural representation. Textual meaning is represented by theme-rheme and information structures; interpersonal meaning by mood-residue structure; and ideational meaning by transitivity structure.

The FGP includes four separate modules; one for each kind of structural analysis. The modules are independent of one another, each has its own unique terminology and organization. Theme-rheme structure includes elements not found in mood-residue, transitivity or information structures. The same holds true for mood-residue, transitivity and information structures when compared with one another. Each module assembles a partial solution to the larger question of what is the meaning which the text as a whole conveys.

The FGP interfaces between user and text taking one through the process of clause analysis. When the user completes and saves an analysis using the FGP, (s)he is also building a database of entries each of which identifies the elements of theme-rheme, mood-residue, transitivity and information structures that provide the wherewithal to create meaning.

The look and feel of the FGP

The text to be analyzed is first loaded like any other document into Sprint, the word processor. The example document, TEXT1.SPR, as it appears in Sprint, is shown in Slide 1. The user then highlights the clause to be analyzed using either the keyboard or mouse and chooses a type of analysis from the FGP pop-up menu. In Slide 2, the first clause "Stitching together the ideal computer system for your business can often be a difficult operation" is highlighted. From the Mouse menu, the user selects the **Functional Grammar Processor.** *Up pops the FGP menu;* **Theme** is selected. Next, from the Operation menu, **Analyze clause** is chosen. With the Functional Grammar Processor activated, the data entry form for Theme-Rheme Structure [Slide 3] now appears on the screen. The highlighted clause is already entered for the user in

the *Clause (Complex)* field at the top of the form. The topical theme for this clause is "Stitching together the ideal computer system for your business". First we move the cursor to the *Ideational* field using the tab key, then with every press of the F5 key each successive word is cut from the *Clause (Complex)* field and pasted to the *Ideational* field [Slide 4]. Next, the user presses the Return/Enter key and the topical theme from the *Ideational* field is entered in the *Theme* field [Slide 5]. The contents of the *Rheme* field are automatically adjusted leaving only the remainder of the clause "can often be a difficult operation." As indicated by the status line at the bottom of the screen, pressing F10 saves the analyzed clause to disk and returns the analyzed clause in a Sprint document. Notice, however, in the lower left hand corner of the screen displayed in Slide 6, that this document is now TEXT1.THM, no longer TEXT1.SPR. Originally there existed only the one Sprint document file, TEXT1.SPR. When the user highlighted the clause to be analyzed and selected Theme-Rheme analysis from the FGP pop-up menu, a new file "TEXT1.THM" was immediately created to receive back the analyzed clause. TEXT1.THM looks exactly like TEXT1.SPR except for that previously highlighted clause which now appears in analyzed form. Once TEXT1.THM has been created, further theme-rheme analysis must be carried out within it and not the original document. In fact, once TEXT1.THM has already been created and you attempt to do theme-rheme analysis on a highlighted clause in the original document, the program will automatically replace the original document, TEXT1.SPR, on the screen with the existing TEXT1.THM file (containing the text view of the theme-rheme structure). Similarly, new text files are created to provide text-views of clause analyses for mood-residue, transitivity and information structures.

On-line HELP

In order to do transitivity analysis on the same text, the user re-opens the original document, TEXT1.SPR [Slide 7], and this time selects **Process** in the FGP menu. Transitivity analysis involves identifying the process, participant(s) or role(s), and circumstance(s). The possible roles played by participants will depend on the process: actor in a material process, behaver in a behavioural process, carrier in a relational process, etc. The first participant in our highlighted clause is "Stitching together the ideal computer system for your business".

Once the user has entered this participant into the *Participant* field of the entry form, up pops the Process menu waiting for the user to identify the type of process in which this participant is engaged [Slide 8]. If, however, one is not quite sure what is meant by material process, they can then press F1 and a HELP window appears listing information relevant to the current menu or field. Slide 9 shows the HELP window for Process which lists each process type and its *corresponding category meaning*, e.g. a material process involves doing, a mental process involves sensing, etc. Press ESCape and the HELP window disappears. The participant is participating in a relational process so we select **Relational** in the Process menu. Next pops up the Type submenu indicating that there are three types of relational process: intensive, circumstantial and possessive. Again, suppose the user is not sure about the type of relational process to assign to 'can often be' - is it intensive, circumstantial, or possessive? While still in the Type submenu, pressing the HELP key (F1) will bring up the HELP information about the three types of relational process in English [Slide 10]. Because the relationship between the two participants is one of sameness, the user selects **Intensive**. From the Mode submenu [Slide 11], the user picks **Attributive**. In attributive mode, the participant can be either the **Carrier** or an **Attribute**. Slide 12 shows the user has identified the participant as the carrier. Next the user enters the process into the *Process* field [Slide 13]. Having already identified the first participant as carrier in a relational process, the selection bar in the **Process** menu is now drawn on **Relational**. The user presses the Enter key and the Type submenu appears [Slide 14]. The user selects **Intensive** with the result shown in the *Analysis* field [Slide 15]. The user next moves the cursor back to the *Participant* field, presses F6 to clear the field, enters the other nominal group in this clause -- "a difficult operation" [Slide 16], and repeats the process for entering information. The relational process is **Intensive** [Slide 17], **Attributive** mode [Slide 18], and the element in question is an **Attribute** [Slide 19]. With the analysis of process and participant(s) now completed [Slide 20] the transitivity structure for this clause is saved to disk and returned to the Sprint document, TEXT1.PPC [Slide 21].

Viewing an analysis of another structural type

While analyzing a clause for one structure, say mood-