Based on E-MELD Project Consultant's Report 2004-10-01

What a Linguist Needs to Know about Word Processing

or: Don't hack it - design it!

Dafydd Gibbon

Universität Bielefeld, 2007-04-18 (Version 09, comments welcome)

Table of Contents

1 Objective and motivation

This report explains why and how to use stylesheets in word processing, from a linguistic perspective: documents are - like words, sentences, traffic signals - *signs*, though very complex ones, with *content*, *structure*, and a *rendering* in some medium. The present document is intended to assist both in paper and hypertext production, and is also used to illustrate these principles.

Many users of word processors are unfamiliar with the advantages of using well-defined styles for document objects like titles, headings, paragraphs of specific types, which have the same consistently defined rendering properties (font, spacing, centring, indentation etc.) wherever they occur in a document.

A common practice is to assign rendering properties of this kind to each local occurrence of a document object separately: a stretch of text is marked, and instantly formatted using toolbar buttons. For short documents this is a handy, "quick and dirty" procedure. For long and complex documents it leads to waste of time, patience and energy - and of course the personal satisfaction of having produced a professionally formatted document is missed. The advantages of using styles instead of the quick and dirty procedure are:

1. Time saving: if the rendering properties have to be changed for each individual occurrence of a document object such as a heading, this is unnecessary time-wasting - change a style once to change all occurrences of this style at the same time.
2. Automatic creation of *Table of Contents*, and automatic re-numbering of tables and graphics if they have to be moved.

3. Consistency preserving: if the rendering properties are changed separately by hand, inevitably errors occur, leading to inconsistency of formatting.
4. Consistent cross-media conversion: well-defined styles in one formatting system, such as a word-processor, can be converted consistently into well-defined styles in another, such as the world-wide web formatting.
5. Consistent cross system conversion: with well-defined styles, there is a much greater chance of comparable results when different word processors or different versions of a word processor are used.
6. Support of barrier free access: screen readers for the visually impaired can process styles intelligently and produce well-structured output.

Some examples:

- Wherever headings of the same level occur, they are expected to look the same.
- Wherever text paragraphs occur, they are expected to look the same as the others of the same type.
- Wherever bibliographical entry paragraphs occur, they are expected to look the same as the others of the same type, and so on.
- Wherever indentations occur, they should be of the same size.
- Wherever spaces before and after paragraphs occur, they should be consistent, and easily adjustable to any arbitrary spacing.

In each of these cases, why not assign these properties to all objects of a particular type at the same time, rather than separately for each of the many times that they occur in a document? If well-defined styles are not used, none of these advantages can be used.

2 Word processing as document production

2.1 A linguistic perspective

Ordinary word processing - as used in linguistic articles and reports (including this one) but not only in these - is seen here from the applied linguistic perspective of *systematically planned text generation*, not from the naive vantage point of *writing with a computer*. The idea is that linguists should not march from characters through words to sentences and stop thinking at that point, but should regard texts as meaningful and systematic language objects which are worthy of their full attention, and should apply this knowledge professionally in their own writing. Documents, and the language objects they consist of, are *document objects* with *semantic properties* and *media properties*, just like other language objects such as words and sentences. The hierarchy of characters, words, lines, paragraphs, pages etc. is the graphical layout analogue of the prosodic hierarchy in speech. Objects at each of these hierarchical levels

1. have semantic properties and (visual and sometimes acoustic) media properties,
2. can be described with a document grammar,
3. can receive text-structural descriptions, for example as tree-diagrams.

The moral of this story is that if linguists and language teachers (but particularly linguists concerned with language documentation, corpus tagging, XML markup and the like) do not use these objects and their properties intelligently in generating their own texts with a word processor ...[1]

The present document was composed using these techniques with OpenOffice (version 2.0.2), and will be distributed in the OpenOffice writer format (odt; compressed XML) for editing purposes, and in OpenOffice-generated PDF format, for general distribution as a model format.

2.2 Applications of linguistics in word processing

Evidently, *words* are within the purview of linguistics, and it is hardly surprising that linguists are heavily involved in the development of *word* processors like OpenOffice or MS-*Word*. A moment's

1 Well, maybe the moral should not be made too explicit. We don’t want to be too impolite to colleagues, after all.

reflection will show that the properties of *words* dealt with by modern *word* processors, are actually in the traditional domain of linguists and language teachers, which is why linguists are employed by software companies to solve problems in areas such as the following:

- spelling (cf. spell checkers),
- correct inflection (cf. grammar checkers),
- thesaurus as a writer's help,
- word completion facility,
- capitalisation facility,
- use of correct quotation marks,
- translation of terms for localisation to other languages.

There are also many add-ons to *word* processors concerned with functionalities such as the reading aloud of selected text portions or the translation of selected text portions, which involve the results of many years of research in linguistics, phonetics, speech technology and text technology. There are, of course, imperfections in each of these applications: nobody is perfect, no theory is complete, and neither are encyclopaedias, dictionaries and grammars, whether on paper or on the internet and, of course, word processing software.

3 Beyond word processing: documents and document objects

3.1 Document processing, not word processing

But why *word* processing? The term "*word* processing" is something of a misnomer. Presumably no user of a word processor just wants to write words. The goal is usually to write a *document*, consisting of *sections*, consisting of different *paragraph* types such as *headings*, and *text*, each paragraph consisting of *lines*, and each line composed of a stream of *characters*. Note that this account has nothing to say about linguistic units in the narrower sense, such as words and sentences.

Word processing is a form of *Applied Text Linguistics*, in which *texts* are dealt with in much the same way as *sentences* are dealt with in more conventional linguistics. The objects dealt with in Applied Text Linguistics are, among others, those just introduced. For convenience of reference they will be called *Document Objects*. There are many kinds of Document Object, but the main Document Objects handled by word processors are:

1. document,
2. page,
3. paragraph,
4. character,
5. list (enumerating list or bullet list),
6. table.

The basic Document Objects are document, page, paragraph and character; the other two types are more specialised.

3.2 The CSR model of documents as signs

Objects of all kinds have specific properties, and Document Objects are no exception, except that they have the special characteristic of being *signs*, not just objects. The properties which Document Objects have are of two main types:

1. *structural* properties, which define
    1. the components of the Document Object, e.g. paragraphs, lists and sublists,
    2. the contexts into which it fits (e.g. for paragraph objects: what the next paragraph object should be);
2. *interpretative* properties, which define
    1. the *Content* (i.e. the *meaning*) of a Document Object (this is entirely up to the author and the readers, of course, and will not be discussed further in this document),
    2. the *Rendering* or *style* (i.e. the appearance) of a Document Object; the conventional

linguistic analogue in the acoustic modality phonetic interpretation, while the media interpretation of a Document Object may be in the visual (text, figures, animations, film) or the acoustic (audio) modality.

The relation between document structure on the one hand, and its two interpretations as *content* and *rendering* on the other, are shown in Figure 1.

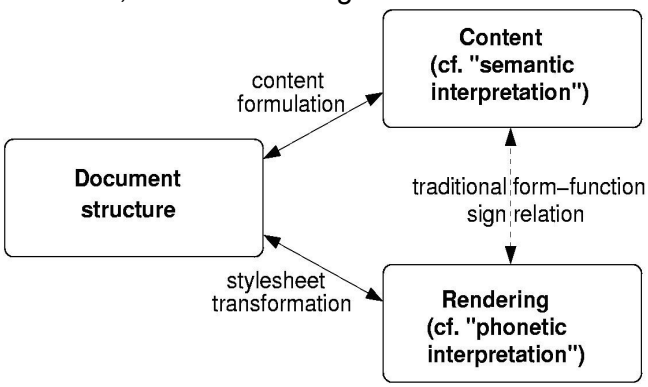


Figure 1: Semiotic text linguistic view of the Content-Structure-Rendering (CSR) sign model.

3.3 Document objects and their properties

The Document Structure component defines the deep structure of the document, consisting of Document Objects. The Content properties of a Document Object will not be dealt with. The Rendering properties of a Document Object are typically organised using *attributes*, i.e. collections of mutually exclusive properties, which are referred to as the *values* of the attributes. For example, a paragraph Document Object cannot be both centred and left-aligned at the same time, and a character Document Object cannot be both 12pt and 10pt, or italic and plain at the same time. Examples of some Document Objects and some of their attributes and values are given in Table 1.

Table 1: Document Objects and some of their properties.

Document Objects	***Attribute***	***Typical values***
Document	Name	relevant metadata
	Author	relevant metadata
	Version	relevant metadata
Page	Size	sub-attributes: size, top, bottom, left, right margins, header, footer, frame, ...
	Margin	sub-attributes: top, bottom, left, right margin
	Header	special page-linked paragraph type
	Footer	special page-linked paragraph type
Paragraph	Alignment	left, right, centred, justified
	Numbering	enumerated, bulleted
	Tabulator	horizontal tab settings
	Indentation	left & right margin, first line indentation
	Spacing	gap above and below
	Line	1, 1.5, 2, ...
	Frame	sub-attributes: line type, thickness, colour, ...
Character	Font	Arial, Helvetica, Times New Roman, Courier, ...
	Size	10pt, 12pt, ...
	Bold	bold, non-bold
	Italic	italit, non-italic
	Underline	underline, non-underline
	Colour	red, orange, ... white, black

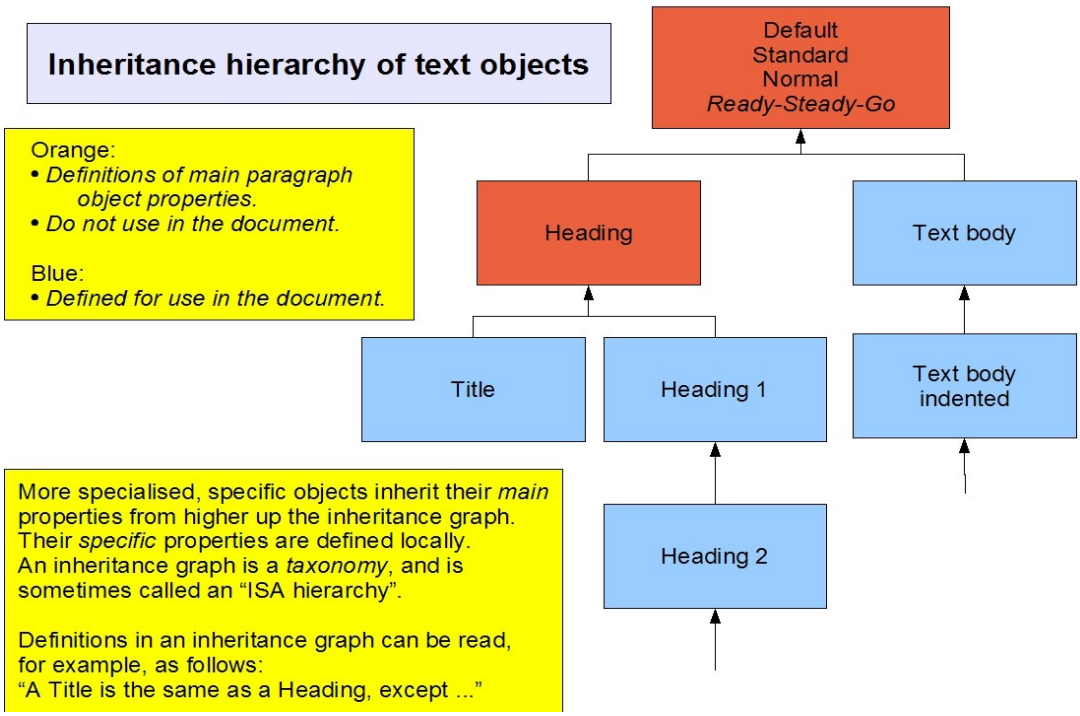


Figure 2: The paragraph type and subtype hierarchy.

Note that the four Document Objects in Table 1 are not in a simple hierarchy, but in two orthogonal hierarchies:

1. The *paragraph* is a part of *Document Structure*, which formulates (i.e. organises and structures) the meanings intended by the author. Further, the paragraph has its own *rendering properties*, which reflect the intentions of the author with respect to the paragraph's status as title, heading, subheading, etc., and of course *content* properties.
2. The *page* has no significance with respect to the formulations of the author (in general; bibliophile books are the exception which proves the rule), but reflects Rendering issues concerning layout production contraints determined by the publisher. A similar point applies to the line Rendering properties of paragraphs.

The definitions of rendering properties for Document Objects are known as *styles*, and a set of such styles for a document is known as a *stylesheet*. Stylesheets are generally provided by publishers for books, scientific articles, and so on, and traditionally are printed on paper, with examples. However, in electronically based publishing the stylesheets are simply the definitions of Document Objects in a word processor. Using the formatting facility of the word processor, a set of styles can be defined and assigned to an entire document stylesheet. For OpenOffice Writer, this is can be stored as a template file with the conventional ".ott" (Open Document Text Template) ending (rather than ".odt", Open Document Text), and in MS-Word as a template file with the ".dot" ending (rather than ".doc").

4 Constructing a document

4.1 Special paragraph object types

The most important object type in standard word processors is the paragraph. The concept of paragraph object used in word processors is very general, and many different subtypes of paragraph object can be defined. The subtypes inherit the basic rendering properties defined for the standard paragraph type; these properties may be overridden by the specific properties needed for the paragraph subtype.

The paragraph subtypes used in the present document are:

1. Pre-Title (user-defined)
2. Title (predefined but modified),

3. Subtitle (predefined but modified),
4. Author (user-defined),
5. Version (user-defined),
6. Heading 1 (i.e. level 1, not the first heading; predefined but modified),
7. Text body (predefined but modified),
8. Footer (predefined but modified).

An interesting paragraph subtype is the Heading (note the difference between a Title and a Heading), for which different types (usually interpreted as hierarchical levels) are pre-defined (Heading 1 being a main heading, Heading 2 being a sub-heading, and so on). First, headings are used to define the Outline, i.e. the main Document Structure, and therefore have their own outline rendering properties such as numbering. Second, headings are used to create the *Table of Contents* automatically. Third, headings are used to create corresponding hierarchical outlines in other media such as the World Wide Web.

Clearly, in all but plain typewriter rendering style such as those used in the body of plain emails, no distinction in terms of rendering properties occurs. However, in the context of a word processor, rendering differences can easily be defined. Some paragraph subtypes are predefined by the word processor, as indicated in the list. Many other paragraph subtypes are typically required in a long and complex document and can be defined by the author. Very useful examples of predefined styles are the paragraphs in a table of contents, which point to page numbers of sections, bibliographical references, or the paragraphs in a bibliography which contain the individual bibliographical entries.

Note that if the correct heading levels are used, a table of contents can be constructed automatically, and satisfactory renderings in other media, e.g. the web, can be achieved.

The styles are assigned to segments of text by means of *markup codes*; the most familiar markup code system is HTML, the markup language used for web documents.

4.2 Are you a word processor hacker?

So let's get more direct... If you do any of the following, as a linguist you should be ashamed of yourself. Consider the following issues:

1. Do you leave the *Default* style type in the style box? DON'T use the *Default* style in the text. WHY? This is the general style definition from which all other styles should inherit their properties by default. Instead, use *Text body* as your basic text style.
2. Do you click on the **B**, *I*, U, etc. buttons to render your text? DON'T do this. WHY? Styles are superior in so many ways, unless you are simply highlighting individual words or phrases.
3. To create a space between paragraphs, do you insert an empty line (actually an empty paragraph object consisting of one line)? DON'T DO THIS: define the space before and after the paragraph type which suits your needs. WHY? Preceding and trailing spaces are properties of paragraphs and should be defined in the style for the relevant paragraph type. If you just use an empty paragraph (i.e. type the return key) you are committed to a fixed spacing; if you define the spacing for each paragraph separately, you run the risk of inconsistencies and waste time. On the other hand, if you define a style, you will avoid these problems.
4. To create a table, do you use spaces and tab separators, and draw the table using separate graphical lines over the text? DON'T DO THIS: create a table object, and assign border properties to it. WHY? Spaces and tab separators may be defined differently on different machines even with the same word processor, and the table will consequently look a mess. A table rendered in this way is extremely hard to edit, and will not export consistently into another medium. A table object, with properly defined cells, rows, columns and framing, as well as *Table Heading* and *Table Contents* format is quick to construct, easy to edit, and exports consistently to other media.
   - Work out how to assign a *caption* above a table object (hint: click on the table with the right mouse button to obtain a context menu), and how to assign a *cross-reference* to the table in the text (hint: click on the Insert menu in the toobar).

5. Now work out how to insert a Figure, including a caption below the figure and cross-reference in the text.
6. To create a list, do you write numbers at the left of a new paragraph? DON'T DO THIS: use a list object, either with enumeration (pick the numbering style) or with bullets (pick the bullet style). WHY? If the list style does not suit you (e.g. the wrong kind of indentation is used), you can edit the style once for all and stop worrying. If the bullet points do not suit you, replace them in the style with other symbols, and the same applies.
7. To create a heading, do you mark the line (actually the paragraph) and directly assign centring, font size, etc. to the marked section? DON'T DO THIS: define the rendering style properties for the Document Object, a paragraph type. WHY? This is a common, but not so clever practice, which is a real time-waster and inconsistency-causer: imagine that an editor requires the headings to be changed to a different font, font size, or highlight convention, and imagine how long this would take for an article or a book if each heading were changed separately. Not only that, imagine the potential for errors and inconsistencies. If you change the style, you change it once, and all headings of the same type change automatically.
8. To change the numbering of a heading or subheading, do you add the number by hand to each heading individually? DON'T DO THIS. Add a numbering property to the heading style once using the *Outline* tool, and all headings will change automatically. WHY? This ensures automatic numbering which is both consistently counted and consistently formatted.

4.3 Document structure

The structure of a document can be described using a tree-graph derived from a document grammar, just like the structure of a sentence. The structure of the present document, for example, is illustrated in Figure 3.

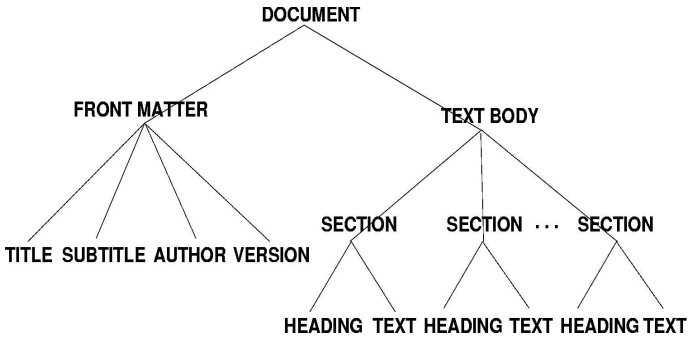


Figure 3: Document structure for the present document (simplified).

A more complex document may consist of front matter, a main text, and back matter. The front matter will include title, author, version (e.g. date, edition, table of contents, etc.), and in the case of a book can be very complex. The main text may consist of chapters, sections, subsections, subsubsections and so on, which in turn consist of Text Objects such as paragraphs, lists (in turn consisting of list elements), tables and so on. The back matter may consist of endnotes, bibliography and index. A linguist will not find it hard to write a grammar of this kind on the lines of the grammars used for the description of sentences of a language. Further details will be given below.

5 How to ...

The following instructions use OpenOffice terminology, but can be transferred easily to the corresonding menu functions in commercial word processors. Only the basic features of style definition are dealt with.

5.1 Work out the content

The content of a document may be, literally, anything, and is consqently outside the scope of this document. How the content is gathered and planned will vary between scientific documents (for which there are many advisory publications) to literary documents (for which there are probably even more advisory publications), personal documents, or commercial documents (for which there are no doubt the most advisory publications). Techniques vary from scraps of paper through notebooks to mindmaps and beyond, for all of which there are computer analogues.

5.2 Design the text structure and the rendering

When the content is clear, it needs to be formulated, i.e. structured in terms of Document Objects according to the requirements of the medium which is to be used:

1. Draw up a list of the paragraph Document Object types (such as Title, Heading 1, Heading 2, etc.) which you need for the various content elements and their relations (see the list given above for the present document).
2. Construct a table with 3 columns (for Document Objects, attributes and values).
3. Enter each of these paragraph Document Objects in the leftmost column (cf. the table given above).
4. Enter the attributes required for each object, and modify the default paragraph level values which these attributes should receive (e.g. spaces, justification, numbering, line centring).
5. Also enter the attributes which all character objects in this paragraph type should receive by default, and modify the default values accordingly (e.g. font, size, italic, colour).

The table is your *stylesheet* (cf. Table 1 and Table 3).

5.3 Implement the structure and the rendering

Each of the objects in the document has to be assigned its rendering properties. Assuming that an entirely new document is to be produced in OpenOffice, the following procedure can be followed.

1. Open a new document.
2. In one of the text fields in the function bar, the style field should appear. In a new document, this style entry is likely to be called *Standard*, *Default*, *Normal*, etc. Click on the select button attached to this field, and very likely you will see this style listed. Do not use this style in your document: it is simply a standard source of paragraph definitions, and many other paragraph styles ‘inherit’ their properties from it, so changing it is likely to change other paragraph styles.
3. Navigate the menu space via the *Styles and Formatting* menu point (or click on the Styles icon in the toolbar). A list of pre-defined styles will appear.
4. In the lower drop down menu select *All Styles* instead of *Automatic*, and more pre-defined styles will appear.
5. Check this list for styles in your list; stick to the names for standard predefined styles where possible.
6. Right-click on a style name, select *Modify*, and a dialogue box with 15 tabs at the top will appear. Each tab groups related attributes together.

The default values of the attributes can then be modified according to your stylesheet. If no suitable style is defined in your list, define a "New" style: the dialogue box with the 12 or so tabs will appear, first of all requesting a name for the file, the name of another style to switch to when the "Return/Enter" key is pressed, and a basic style from which the new style inherits its defult properties.

Note that no explicit distinction is made in this dialogue box between tabs for *paragraph*, *line* and *character* properties. The distinction is rather obvious, but it will be useful to list the tabs pertaining to each type of Document Object.

Commercial word processors use similar techniques for style definition; the appearance of the menu points and dialogue boxes can be very different, of course.

5.4 Test the style definitions

Write a short document containing the styles you have modified or defined, such as

- Title
- Heading 1 (i.e. top level heading)
- Heading 2 (i.e. next level subheading)
- Text body

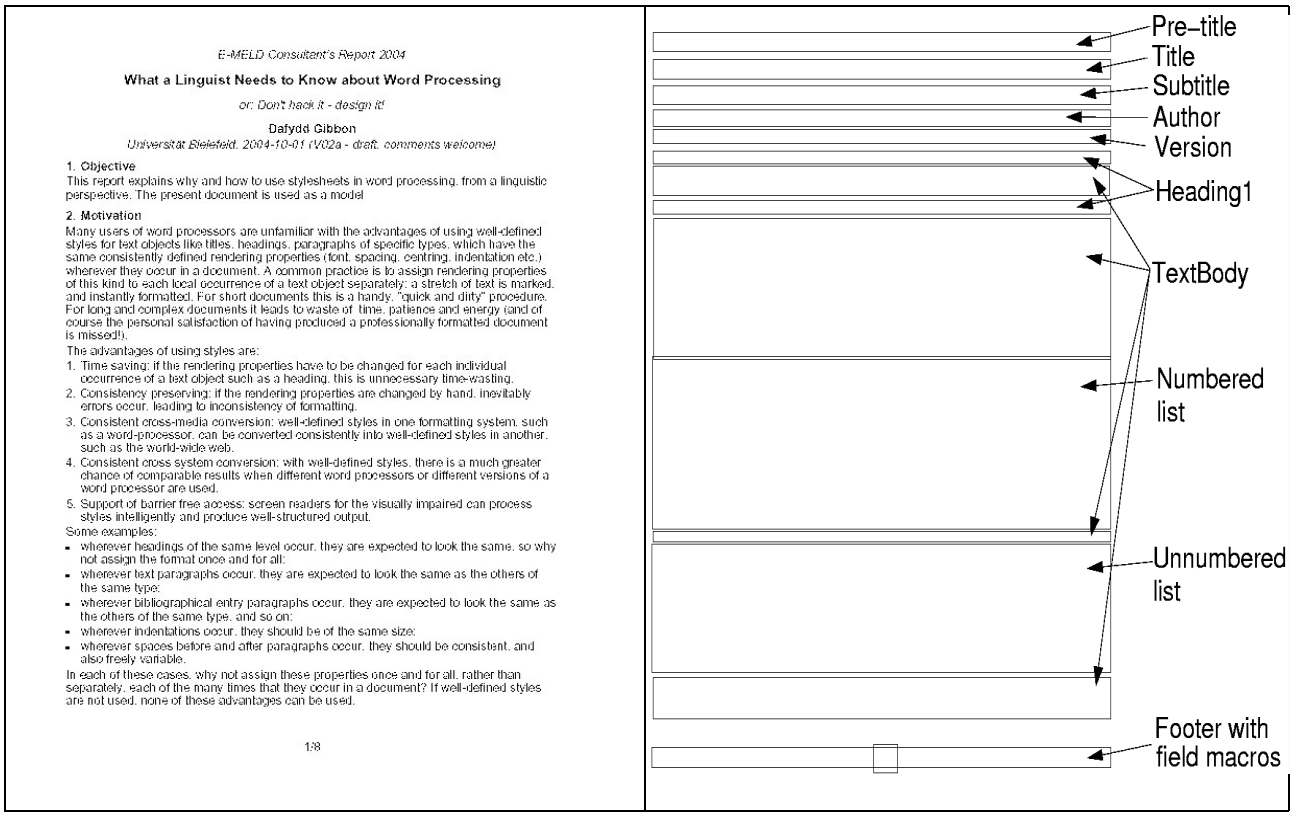
Add additional Document Objects such as lists and tables, with their styles defined appropriately.

Check the styles used in this report.

When you have done this, you will have

1. designed a text document on linguistic principles and, in particular,
2. defined a stylesheet in the form of an object-attribute-value table containing a set of styles relevant to your document,
3. implemented the stylesheet using a word processor,
4. tested the implementation in the production of a document.

Table 2: Document Objects on p. 1 of a previous version of the present document.



5.5 Check a practical example: the present document

Table 2 shows a reproduction of the first page of an earlier version of the present document as a sequence of document objects, along with a stylised version of the Document Objects and the names of the rendering styles assigned to them.

The page object itself is not labelled; it has upper, lower, left and right margin attributes, as well as a footer attribute. The stylised version of the document object shows the *bounding boxes* which show the rendered size of the document objects, determined by:

1. the size of the objects it contains (e.g. for a paragraph object, the number of lines it contains and their spacing),
2. its own size attributes (e.g. for a paragraph object, its left, right, upper and lower margins).

The rendering attributes assigned to each style obtain their values by inheriting them from a

default style, called *Default* or *Standard*, unless they are specifically modified to suit the special requirements of the document object concerned. For instance, the "Title" object contains one or more paragraphs with modified upper and lower spacing, which in turn contain characters with modified values for the size (16pt) and highlighting (bold) attributes. The *Default* style should NOT be used in the document, but kept simply as a source of standard, default definitions for the styles which are actually used.

The main attribute-value assignments for the paragraph and character objects in the styles of the present document are shown in Table 3.

The following phrase-structure grammar describes the arrangement of document objects on the first page of the current document; in order to complete the grammar for the whole document, table and figure objects need to be added:

DOCUMENT	→	FRONT-MATTER DOCUMENT-BODY
FRONT-MATTER	→	Pre-Title Title Subtitle Author Version
TEXT-BODY	→	SECTION SECTION*
SECTION	→	Heading1 TEXT
TEXT	→	TextBody | NumberedList | UnnumberedList

There is a clear analogy to standard phrase structure (constituent structure, context-free) grammars for sentence syntax of the "S → NP VP" type. In the notation used here, the arrow "→" can be read as "consists of", the space between the categories on the right-hand side of the arrow as "followed by", the asterisk "*" as "none or arbitrarily many", and the vertical bar "|" can be read as "or" (here the "exclusive or").

The notation used in different document description contexts varies in detail, but all notations employ some means of indicating hierarchical or *immediate dominance* structures (i.e. meronomies or part-whole networks) and sequences whose parts are in a relation of *linear precedence*.

Table 3: Document Objects and their Rendering Properties.

Style	***Inherited from***	***Paragraph attributes***	***Character attributes***
Default:		Left indentation: 0.00cm Right indentation: 0.00cm First line indentation: 0.00cm Upper spacing: 0.00cm Lower spacing: 0.00cm Left justified	Arial standard 12pt
Pre-title:	Default	Upper spacing: 0.40cm Lower spacing: 0.40cm Centred	Italic
Title:	Default	Upper spacing: 0.40cm Lower spacing: 0.40cm Centred	Bold 14pt
Subtitle:	Default	Upper spacing: 0.40cm Lower spacing: 0.40cm Centred	Italic
Author:	Default	Upper spacing: 0.20cm Lower spacing: 0.20cm Centred	
Version:	Default	Centred	
Heading1:	Default	Outline numbered Upper spacing: 0.40cm Lower spacing: 0.20cm	
TextBody:	Default	Upper spacing: 0.07cm Lower spacing: 0.07cm	
Numbered list:	Default	Numbering	
Unnumbered list:	Default	Bulleted	
Footer:	Default	Centred	10pt
Fields:	Footer		

5.6 Do a document production project

Based on the pricniples outlined in this document, a number of practical projects can be suggested. The following are just a small selection of possible projects:

1. Term paper about document production.
2. Short book:
    1. short story, novelette, poetry collection (try formatting e.e.cummins' poems),
    2. traveller's guide, cookery book, instruction manual,
    3. newspaper or magazine,
    4. internet hypertext: cyberpoetry, dictionary, ...
    5. ...
3. Table of Contents for your book.
4. HTML formatted version of your book.
5. Slide presentation about the book.
6. Advertising poster for the book.
7. Advertising flyer for the book.
8. CD accompanying the book (data, audio, video).
9. Internet site about the book.

6 Summary and conclusion

The main point to be made is that a document is a complex sign which has

1. a *meaning*, or *content*, consisting of objects and events and their properties and relations, propositions about these, speech acts in which these propositions are used, and discourse strategies such as narration, explanation, argumentation;
2. a *text linguistic structure* which is comparable with *sentence structure* in conventional linguistics, and may take on many forms: a plain list, tree-like, network-like;
3. a *rendering interpretation* which is comparable with *phonetic interpretation* in conventional linguistics, but more complex in that the layout of the text is involved, the realisation of elemements as graphics and, in hypertexts, as audio and video elements.

The structural levels of text and sentence have much in common, from a linguistic point of view:

- in semiotic terms, both documents and sentences are complex signs, which can be thought of as abstract or mental units, which are interpreted in terms of reality at two levels, i.e. as *meaning* and *form*,
- paradigmatically, the constituent objects and their properties are organised into an inheritance hierarchy, the most common kind in conventional linguistics being a type hierarchy, the most common kind in word processor text linguistics being a default hierarchy,
- syntagmatically, the constituent objects are organised into a compositional hierarchy, exactly like the constituents of a word or a sentence,
- the same formalisms can be used for describing the structure, the meaning and the form properties of constituent objects, for example context-free grammars, or as attribute-value structures and inheritance hierarchies.

The constraints on text structure are of course different in detail from constraints on sentence structure, just as constraints on sentence structure are different from constraints on word structure. For the syntagmastic hierarchy, typically *document grammars* are defined in order to permit electronic analysis and display of documents; in the XML document markup framework, grammars are expressed as Document Type Descriptions (DTSs) or XML Schemata.

Topics of this kind are dealt with in the discipline of Text Technology and a number of related disciplines, including Computational Linguistics and Software Engineering. The most widespread kind of approach goes under the heading of XML technologies, which comprise conventions for

document grammars and tools for creating, checking and presenting documents systematically. These technologies are gradually being adopted as a standard for the creation and rendering of internet documents. Of particular interest for the humanities is the use of these techniques to classify, archive and search documents of all kinds systematically, and in this area text technology is closely related to library and archive sciences.

There are many more dimensions to document production than those described here. The focus of attention in the present document was on the classic written paper. Many more rendering techniques are available, of course, including rendering as a hypertext (in graphical or text browsers), rendering by reading aloud with a speech synthesis application for blind users or users in visually adverse environments, and animated renderings. For any of these methods, a language specialist should be able to design documents professionally, thereby saving time, being consistent, ensuring the same appearance in different word processors and on different platforms, supporting document coversion for barrier-free document access in assistive technologies for the disabled.