Structures et modèles de documents

Jacques André, Vincent Quint

Structures et modèles de documents

Jacques André

IRISA/INRIA Rennes
Campus de Beaulieu
35041 Rennes Cedex

Vincent Quint

INRIA/IMAG Grenoble
2 rue de Vignate
38610 Gières

Structures et modèles de documents

Jacques André & Vincent Quint

Ce texte a pour objet d'introduire la notion de structure de document[1]. Il présente également les principales structures qui permettent la représentation des documents dans différentes normes et dans quelques systèmes informatiques.

1. Introduction aux structures de documents

Tout d'abord, qu'est-ce qu'un document? Nous ne donnerons aucune définition, mais voici quelques exemples :

— une petite annonce dans une boulangerie,
— une lettre, une facture,
— un quotidien, un hebdo,
— un roman,
— un exemplaire d'une revue de Sciences Humaines, ce livre,
— le rapport d'activité de l'INRIA,
— l'édition commentée du Cartulaire de Redon,
— le catalogue de la CAMIF.

Mais, par document, on entend aussi des choses moins « visibles » telles que :

— le dictionnaire *Le Robert* sur CD-ROM (disque optique numérique),
— les horaires des chemins de fer consultables par « 3615 code SNCF »,
— la table de césure des mots utilisée par le système Word3,
— un message dans un système de courrier électronique,
— etc.

En gros, on peut appeler « document » le contenu de ce que l'on produit, distribue, utilise ou garde lors d'un processus de communication écrite ou électronique. Nous limiterons toutefois ici ce terme aux objets ainsi manipulés lorsque la partie textuelle est prépondérante.

1.1 Notion de structure physique

Considérons d'abord un document avec l'œil d'un lecteur ou du typographe (ou d'une secrétaire munie d'un système de PAO[2]) qui le compose.

[1] L'ouvrage de base sur ce sujet est un livre (en anglais) basé sur les cours donnés lors d'une école de l'INRIA sur ce thème (Aussois, 1987) : J. André, R. Furuta et V. Quint (eds.), *Structured documents*, Cambridge University Press, 1989. Une grande partie de ce texte est, par ailleurs, tirée de V. Quint, *Une approche de l'édition structurée des documents*, Thèse d'Etat, Université Scientifique, Technologique et Médicale de Grenoble, Mai, 1987.

[2] On suppose que le lecteur de ce texte a un minimum de connaissances sur la PAO (« Publication Assistée par Ordinateur » ou « édition électronique »). Sinon, il pourra se reporter à des livres comme F. Richaudeau, *Typographie et mise en page*, Éditions Retz, Paris, 1989, ou Bernard Girard, *Le guide de l'édition d'entreprise*, Éditions AFNOR, Paris, 1988.

1.1.1 *Premier exemple*

Considérons un roman (figure 1) :

A part celles de début et de fin de livre[3], les pages se ressemblent beaucoup. Elles ont toutes un format de 10,5 cm × 17,5 cm. Le blanc de couture mesure 1 cm, le blanc de tête 1,7 cm, le blanc de grand fond 1,5 cm et le blanc de pied 2,8 cm, ce qui donne une justification de 8 cm et une hauteur du rectangle d'empagement de 13 cm. Le foliotage est dans le blanc de pied, à 0,8 cm du rectangle. Les pages sont formées de pavés. Chaque pavé est lui même formé de lignes de texte (composées en Times). On distingue deux types de ligne : des lignes composées en capitales, corps 14, centrées, précédées et suivies de grand interlignes (par exemple le titre de la figure 1) et des lignes en Times romain (avec quelques mots en italique) corps 11 justifiées (les premières lignes de paragraphes commencent par un cadratin que l'on peut considérer faire partie du texte et les dernières peuvent être creuses).

Cette mise en page peut être décrite de façon un peu plus sèche. Oublions les premières et dernières pages. On peut dire que :

— ce roman est formé de pages (de 10,5 cm × 17,5 cm),
— une page comprend plusieurs pavés et un numéro de page,
— un pavé (de 8 cm de large) est formé
  – soit d'un gros titre,
  – soit d'une ligne ;
— une ligne est formée de signes (justifiée, en Times, romain ou italique, en corps 11) ;
— un gros titre est formé de capitales (centré, en Times, corps 14, devant : un espace de 170 points, après : un espace de 56 points) ;
— un numéro de pages est formé de chiffres (en Times romain corps 11, appuyé à droite ou à gauche selon la parité de la page).

Les parties entre parenthèses donnent en quelque sorte la métrique de la mise en page. Si on les ignore pour le moment, on peut décrire cette mise en page comme en figure 2-a, où

= signifie « est »,

| signifie « ou »,

* signifie « 1 ou plusieurs fois »

`A B` signifie « A suivi de B ».

Les linguistes et les informaticiens appellent cette description une *grammaire* et disent qu'elle définit une certaine *structure*[4]. A cette grammaire, on associe souvent une repésentation graphique, ici l'arbre de la figure 2-b. On dit donc, en manipulation de document, qu'une mise en page est définie par une structure. De plus, puisque cette structure touche à l'aspect matériel, physique, du document, on dit que c'est une *structure physique*. On remarquera que la notion de paragraphe ne peut pas entrer dans cette description puisqu'un paragraphe peut être à cheval sur deux pages (voir ci-dessous les sections 1.5 et 1.7.3).

[3] Elles correspondent à la notion « logique » de péritexte. Voir à ce sujet, la section 1.7.1.

[4] Ces termes ont été répandus depuis l'ouvrage célèbre *Syntactic Structures* de Noam Chomsky.

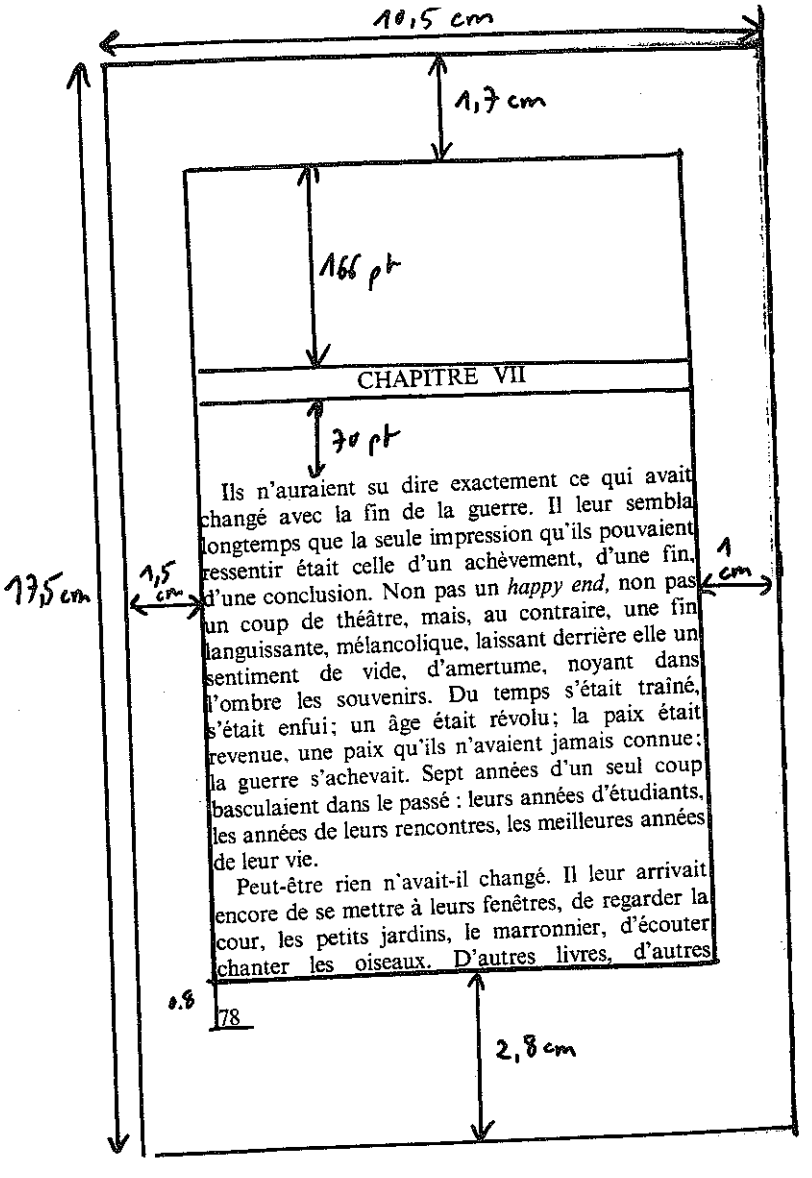
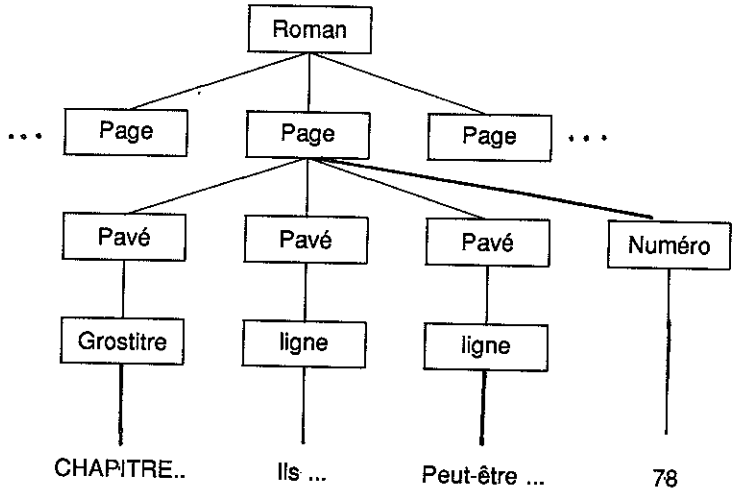


Figure 1: Un extrait de document : page 78 du roman *Les choses* de Georges Perec, dans l'édition de la collection 10/18 achevée d'imprimer en août 1988 sur les presses de l'imprimerie Bussière à Saint-Amand (Cher).

```
roman = page*
page=pave* numero
pave=ligne* | grostitre
ligne=signe*
grostitre=capitale*
numero= chiffre*
signe = a | b | ... A | B ... | . | , | ; | ...
chiffre= 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9
capitale= A | B | ... | Z
```

a – Grammaire associée à la mise en page du roman de la figure 1



b–Arbre associé à la grammaire de a)

Figure 2: Description de la structure physique d'une page de roman

1.1.2 Second exemple

Il y a des mises en page beaucoup plus complexes que celle du roman précédent. *Le Monde*, par exemple, est composé sur des feuilles de 29 cm x 44 cm. La une comprend en haut le logo du journal dans ses caractères spéciaux en gros corps puis, du haut vers le bas, l'adresse centrée en corps 8, un gros filet bleu, une ligne comprenant l'année, le numéro et le prix (tout celà en capitales maigres corps 8 sans sérifs, appuyés à gauche), la date (capitales avec empâttements italiques, corps 12, centrées) et les noms du fondateur et du directeur (capitales maigres corps 8 sans sérif appuyées à droite).
En bas de la page, se trouve, à gauche, un rectangle de 2,5 cm x 4,5 cm contenant le code barres. En bas à droite, sur 23,8 cm de justification, un filet et deux lignes (en bas de casse corps 8 sans sérif) avec le prix du journal dans divers pays étrangers.
La partie centrale est formee de pavés séparés par des filets verticaux ou horizontaux. Chaque pavé comprend un dessin humoristique, une publicité ou, le plus fréquemment dans le cas du *Monde*, une ou plusieurs lignes de titres (centrées, en romain gras corps 16 avec empâttements) suivies d'une ou plusieurs lignes (italiques corps 14 avec empâttements centrées) de sous-titres. Du texte (en romain sans serif, corps 10 justifié sur 4,4 cm) sur 1 à 4 colonnes séparées par des gouttières de 0,5 cm. En bas de la dernière colonne se trouve un nom (en capitales corps 12 sans serif, appuyées à gauche) et (en italique) la mention « Lire la suite page n ». etc.
On le voit, la structure physique d'une page du *Monde* est complexe et il faudra beaucoup de lignes comme celles de la figure 2 pour la décrire.

1.2 Notion de type

La majorité des systèmes de PAO proposent aujourd'hui la possibilité de pré-définir les éléments constituant une mise en page. Ceci correspond à la notion de plan ou de style de Word, de feuille de style d'Interleaf, etc. En Word3, par exemple, le titre de la figure 1 peut être défini comme suit :

```
Standard + Jeu: Times 14 Points, Centré,
    Espace Avant : 170 pt Après 56 pt
```

L'avantage de ceci est que si l'on veut changer de mise en page, par exemple appuyer le titre du chapitre à droite, il suffit de faire la modification dans la définition de style, une fois pour toutes : toutes les têtes de chapitre seront automatiquement mises en place comme désiré.
On peut ainsi « typer[5] » chaque élément de la structure physique.

1.3 Structures hiérarchiques, séquentielles et collatérales

Considérons maintenant le présent texte. On peut aussi en décrire la structure physique avec des règles comme celles de la figure 2, par exemple :

```
lignes=ligne | titre_de_section | titre_de_sous-section
titre_de_section = numero titre (en gras corps 14)
titre_de_ sous-section = numero titre (en gras corps 12)
titre = signes*
```

Ceci permet effectivement de produire de nombreux documents. C'est ainsi que l'on travaille très souvent, en PAO, avec des feuilles de style. Mais ceci permet aussi de produire des documents erronés, ne répondant pas à ce que le lecteur

[5]Ce mot est inspiré de la programmation où l'ont dit, par exemple, que tel nombre est de type entier ou réel.

ou l'auteur attendent. Rien n'interdit, par exemple, de commencer un roman par un pavé de texte, de le faire suivre par un gros titre, puis par un nouveau pavé de texte. De même rien n'interdirait dans le présent article de commencer par un titre de sous-section et de le faire suivre par un titre de section.
Pour affiner notre structure, et donner plus de contraintes aux éléments les uns par rapport aux autres, il faut utiliser trois notions :

séquentialité : il faut pouvoir exprimer que deux éléments de structure peuvent se suivre (par exemple qu'un pavé de texte peut suivre un gros titre ou un autre pavé de texte) ;

hiérarchie : il faut exprimer qu'un élément de structure ne peut exister qu'à l'intérieur d'un autre : le logo *Le Monde* est à l'intérieur de l'en-tête du journal ; un titre de sous-section ne peut être composé que si l'on a déjà composé un titre de section ;

collatéralité enfin : il faut, au contraire, pouvoir exprimer que des choses sont indépendantes. Par exemple que la position relative de deux articles du *Monde* sont (pour le typographe) indépendantes.

1.4 Structure logique

En fait, dans la sous-section précédente, nous confondons deux aspects d'un même document :

1. Dire qu'un livre est une succession de pages, que les titres de sections sont en Helvetica romain corps 14, etc. relève bien de ce que nous avons appelé la structure physique.

2. Dire qu'une section est formée de sous-sections relève d'autre chose, que nous appellerons la *structure logique* car elle correspond à la logique du document.

La structure logique est donc concernée par la relation entre les éléments d'un texte et la structure physique par leur représentation.
A un document, on peut donc associer une grammaire autre que celle de la structure physique ; on peut, par exemple, associer à un roman les grammaire et arbre de la figure 3.

1.5 Structure physique et structure logique

Plusieurs points de vue peuvent donc être envisagés pour décrire un document. Si on se réfère à la chaîne traditionnelle de production des ouvrages imprimés, on peut considérer la position de l'auteur et celle du typographe, qui sont chacun à une extrêmité de la chaîne :

— l'auteur s'intéresse d'abord au contenu du document et à son organisation logique, son découpage en parties, chapitres ou sections ;
— au contraire, le travail du typographe concerne plutôt la forme graphique.

On classe généralement les modèles de documents en deux grandes catégories : les modèles qui se réfèrent à la structure logique et ceux qui se réfèrent à la structure physique. Bien sûr cette classification est un peu sommaire, et on verra des modèles qui combinent l'aspect logique et l'aspect physique, mais elle permet, dans une première approche, de clarifier le propos.

```
roman = chapitre*
chapitre = paragraphe*
paragraphe= texte
```

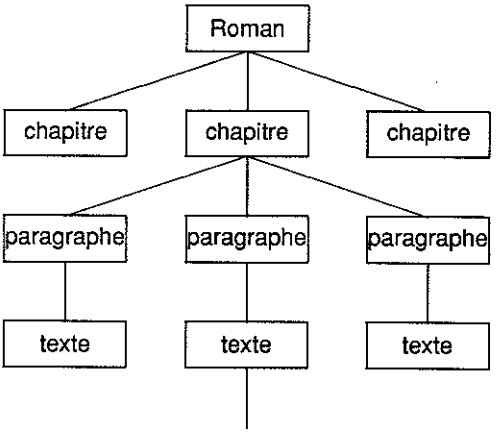


Figure 3: Structure logique et arbre d'un roman.

— Par *structure logique*, on entend donc l'organisation du document en entités telles que chapitres, sections, titres, paragraphes, notes, citations, etc... Ces entités délimitent des parties d'un document qui jouent chacune un rôle particulier dans l'organisation générale du texte. Elles ajoutent au contenu même un niveau de signification supplémentaire, qui permet au lecteur de se repérer dans le parcours du document. Passant d'un paragraphe à l'autre, le lecteur sait qu'il aborde une nouvelle idée ; lisant le titre d'un chapitre, il sait de quoi il va être question dans la suite ; une note lui indique une information liée au paragraphe ou à la phrase qu'il est en train de lire. Ces éléments structuraux ont trait à l'organisation logique du document et s'adressent à l'esprit du lecteur.
— Au contraire, la *structure physique* s'adresse d'abord à l'œil. C'est elle qui découpe le document en pages, qui détermine les espaces et les zones de texte. C'est la structure physique qui joue sur les polices, les corps et les attributs typographiques du texte. Même si, dans certains cas, comme dans la publicité par exemple, la structure physique a pour principal rôle d'attirer l'œil et de faire passer une information ou un sentiment à travers le seul aspect graphique, le plus souvent elle traduit seulement la structure logique.

Même dans les documents qui ne mettent pas l'accent sur la structure graphique, celle-ci n'est pas entièrement liée à la structure logique. Le découpage en pages d'un document illustre bien cet aspect.

— Certaines pages ne traduisent rien d'autre que les limites physiques du support : la feuille de papier a des dimensions finies et il faut changer de page lorsqu'une feuille est pleine. Il n'y a là aucune information sur l'organisation logique du document, et le saut de page peut avoir lieu aussi bien au milieu d'un paragraphe qu'entre deux sections.
— En revanche, le saut de page qui se produit au début de chaque chapitre ou au début de certaines parties du document (couverture, préface, annexe, bibliographie, table des matières...) est, lui, révélateur de la structure logique. Il marque la traduction en termes graphiques d'une rupture dans la séquence du document, l'introduction d'un nouvel élément logique.
— Comme pour les pages, les coupures de lignes peuvent, dans certains cas, traduire la structure logique, sans toutefois refléter toujours une information logique. Le plus souvent le saut de ligne n'est dû qu'à la largeur limitée de la page, mais lorsqu'il ne se produit pas après une ligne pleine, il indique un changement d'élément structural : un nouveau paragraphe, par exemple, ou s'il est accompagné d'un espace vertical, l'introduction d'un élément d'une plus grande importance.

Au contraire des pages et des lignes, d'autres composants de la structure graphique indiquent d'une façon systématique une information de nature logique.

— Si les pages, les lignes ou les colonnes sont en grande partie découpées selon des contraintes purement physiques, tout ce qui touche aux espaces et au choix des caractères traduit toujours une information logique. Les changements de marge, les espaces verticaux, les changements de corps, de style ou de police de caractères indiquent de façon générale un changement d'entité logique. Là encore, il faut admettre les exceptions des documents qui sont plutôt destinés à être vus qu'à être lus, comme certaines affiches ou annonces publicitaires, mais les documents les plus courants suivent cette règle.

— Un nouveau chapitre n'est pas seulement indiqué par un saut de page, mais par plusieurs autres paramètres graphiques : le titre est imprimé dans un corps important, il peut être en gras et les lignes ne sont pas ajustées de la même façon que le reste du texte. Il est de plus suivi d'un espace important. Pour une nouvelle section, les effets graphiques sont moins marqués : il n'y a pas de saut de page, le corps du titre est moins fort et les espaces verticaux sont plus faibles. L'importance des effets graphiques reflète l'importance logique de l'entité à laquelle ils sont liés.

Ainsi, peut-on, au premier abord, distinguer deux types de structure dans un document. La structure logique traduit l'organisation du document telle que l'a voulue l'auteur, la structure physique est le résultat des contraintes dues au support, mais traduit aussi, d'une façon visuelle, la structure logique, la rendant ainsi plus facilement perceptible au lecteur.
Cette division correspond également à la division du travail. L'auteur n'a, *a priori*, pas à faire le travail du typographe.

1.6 Structures génériques et structures spécifiques

Considérons deux rapports, A et B définis comme suit :

```
Rapport A                    Rapport B
  Introduction                 Introduction
  Chapitre 1                   Chapitre 1
       Section 1.1                  Section 1.1
       Section 1.2                  Section 1.2
  Chapitre 2                        Section 1.3
       Section 2.1                  Chapitre 2
       Section 2.2                       Section 2.1
       Section 2.3                       Section 2.2
  Chapitre 3                             Section 2.3
  Conclusion                             Section 2.4
                                  Conclusion
```

Le rapport A comprend une introduction suivie de trois chapitres et d'une conclusion. Le premier chapitre comprend deux sections, le deuxième trois sections. Cela constitue la structure *spécifique* du rapport A. On peut présenter de même la structure spécifique du rapport B : une introduction, deux chapitres, une conclusion. Le chapitre 1 est composé de trois sections et le chapitre 2 de quatre. Les structures spécifiques de ces deux documents sont donc différentes.
La *structure générique* définit le mode de construction des structures spécifiques. Les documents A et B, bien que différents, sont construits selon la même structure générique, qui spécifie qu'un rapport comporte une introduction suivie d'un nombre variable de chapitres et d'une conclusion, chaque chapitre comportant un nombre variable de sections.
La structure générique est, par exemple, la structure globale de tous les livres de la collection *Que Sais-je?*, tandis que chaque numéro de la collection, chaque instanciation, a sa propre structure spécifique.

1.7 Structures textuelles

Jusqu'à présent, nous n'avons donné que quelques exemples naïfs de structures logiques textuelles. En voici quelques-unes qui concernent le livre ou les revues

scientifiques[6].

1.7.1 Le péritexte

Un livre est formé de son texte et du *paratexte*[7], lequel comprend, en première approximation, l'*épitexte* (tout ce qui est lié à un texte sans en faire typographiquement partie : interviews, entretiens, correspondance, etc.) et le *péritexte*, le début et la fin du livre, qui décrit les caractéristiques spatiales (où ce livre a-t-il été imprimé, édité?), temporelles (quand?), fonctionnelles (à qui s'adresse ce livre?), etc. En voici les principaux éléments :

— le péritexte éditorial[8],
— les dédicaces,
— les épigraphes,
— l'instance préfacielle[9],
— l'instance postfacielle,
— le prière d'insérer,
— etc.

L'arbre de cette structure logique ressemble souvent à un rateau ou à un peigne car elle est essentiellement une juxtaposition de nombreux éléments au même niveau. Les structures physiques correspondantes sont, elles, très variées[10] et comprennent les notions de titre courant, haut ou bas de page, foliotage, etc.

1.7.2 Structures textuelles microscopiques

Au niveau le plus bas, on peut s'intéresser à la structure syntaxique du texte et reconnaître des phrases contenant des groupes nominaux, des groupes verbaux, etc., chaque groupe étant découpé en articles, noms, adjectifs ou verbes. Sauf cas extrêmes (prototypes pour la détection de fautes d'orthographe par exemple), les systèmes de manipulation de document n'ont pas besoin de cette structuration.

---

[6] Ces éléments, bien sûr non exhaustifs, correspondent néanmoins aux principaux éléments utilisés en édition. La structure physique, elle, est plus riche mais il suffit de regarder une page d'un quotidien ou d'un hebdomadaire, par exemple, pour se rendre compte que sous une structure physique assez riche se cache une structure logique très pauvre : on n'y trouve guère que les notions de titre, sous-titre, paragraphe et de signature...

[7] Cette expression semble due à Gérard Genette, *Seuils*, éditions du Seuil, 1987. Les éléments textuels décrits dans cette section en sont très inspirés.

[8] Il correspond à l'aspect « édition » et comprend des choses très variées telles que la une de couverture (où l'on trouve, de façon optionnelle, des éléments comme : le nom ou pseudonyme de l'auteur ou des auteurs, le titre de l'auteur, le titre de l'ouvrage, le nom du ou des traducteurs, du ou des préfaciers, du ou des responsables d'établissement du texte et de l'apparat critique, le portrait de l'auteur, le fac-similé de la signature de l'auteur, l'illustration spécifique, le titre ou l'emblème de la collection, le nom du responsable de cette collection, les nom, raison sociale, emblème ou sigle de l'éditeur, l'adresse de l'éditeur, le numéro de tirage ou collection ou mille, la date, le prix de vente, etc.), les pages 1 et 2 dites *pages de garde*, la page de « faux-titre » , les pages 4 à 6 (titre de la collection, éditions de luxe, liste des œuvres du même auteur, *copyright*, etc.), la page 5 (page de titre) et après l'œuvre elle même : le colophon, la liste des ouvrages dans la même collection, etc.

[9] Au choix : introduction, avant-propos, prologue, note, notice, avis, présentation,examen, préambule, avertissement, prélude, discours préliminaire, exorde, avant-dire, etc.

[10] C'est pourquoi on néglige souvent cette partie dans les systèmes structurés.

L'élément de base d'un texte sera donc, en général, le *paragraphe* (voir ci-dessous, section 1.7.3).
Par contre, dans un paragraphe, la structure logique peut s'intéresser à des notions plus fines, comme les suivantes :

— l'emphase,
— les mots en langue étrangère,
— les définitions,
— les noms d'œuvres,
— les noms d'auteurs,
— etc.

Cette structuration fine est notamment utile pour lui faire correspondre une structure physique particulière. Ainsi, à la structuration précédente, pourra-t-on faire la correspondance suivante :

— l'emphase sera en italique,
— les mots en langue étrangère seront en italique,
— les définitions seront en gras,
— les noms d'œuvres en italique,
— les noms d'auteurs en petites capitales,
— etc.

Le passage du logique au physique pour ces règles « linéaires » (en opposition aux règles de mise en page car les premières ne relèvent que de la ligne) sont codifiées dans *Le Code Typographique*. Mais, s'adressant au typographe et non à l'auteur, ces règles sont classées par concepts physiques (« Où mettre l'italique? ») et non logiques (« Comment écrire un nom d'auteur? »), ce qui rend cet ouvrage d'emploi difficile ...

1.7.3 Structures macroscopiques

Elles permettent de traduire l'organisation d'un document à un niveau plus ou moins élevé : on peut découper un livre, par exemple, en préface, chapitres et sections. Voici quelques divisions rencontrées dans divers textes :

— livre, revue, magazine,
— article dans une revue,
— chapitre de livre,
— section, sous-section, sous-sous-section, etc.
— paragraphe : il s'agit en général de la plus petite entité structurelle des textes. Un paragraphe est souvent décrit comme la division d'un écrit offrant une unité de pensée[11]. Au paragraphe correspond une entité physique, le pavé de texte, séparé du précédant ou du suivant par un *alinea*[12].
— les listes : C'est probablement la plus ancienne structure textuelle de l'humanité puisqu'elle correspond aux listes d'inventaires des premières tablettes de l'antiquité[13]. On peut, en première approximation, donner deux types de listes :

[11] Les linguistes ne savent pas bien où se situe cette unité par rapport aux structures styntaxiques de la phrase. Voir par exemple *La notion de paragraphe*, éditions du CNRS, 1985.

[12] On verra dans Roger Laufer, « L'espace visuel du livre ancien » et « Les espaces du livre », *Histoire de l'édition*, Promodis 1983, tome 1, 479–497 et tome 2, 134, comment cette notion a évolué, typographiquement, depuis le XVI[e] siècle.

[13] Jacques Goudy, *La raison graphique*, Les éditions de minuit, 1977. Voir en particulier le chapitre 5 : « Que contient une liste? », p. 145–196.

« La province de Namur c'est 6 régions, ses villes d'arts, à chaque virage un nouveau paysage, la Meuse, ses affluents et ses plans d'eau, son hébergement, sa gastronomie et, enfin, les centres provinciaux à vocation touristique. »

« La province de Namur c'est ...
C'est d'abord ... 6 régions ...
C'est ensuite ... ses villes d'art ...
C'est aussi ... à chaque virage un nouveau paysage ...
C'est surtout ... la Meuse, ses affluents et ses plans d'eau.
C'est encore ... son hébergement, sa gastronomie.
C'est enfin ... les centres provinciaux à vocation touristique ».

« La province de Namur c'est :

- 6 régions,
- ses villes d'arts,
- à chaque virage un nouveau paysage,
- la Meuse, ses affluents et ses plans d'eau,
- son hébergement, sa gastronomie,
- les centres provinciaux à vocation touristique. »

Figure 4: Diverses représentations physiques d'une même liste (texte inspiré d'une brochure de l'office du tourisme de la Province de Namur, citée par Gilbert Turco et Danielle Coltier, « Les agents doubles de l'organisation textuelle : les marqueurs d'intégration linéaire », *Pratiques*, n° 57, mars 1988, 57–79.).

1. les listes pour lesquelles il existe une relation d'ordre, cette relation étant en général « plus grand que ». C'est par exemple l'ordre de classement des reçus à un examen (premier, second, troisième, ...), l'ordre dans lequel il faut faire des actions (premièrement, deuxièmement, etc., ou *a), b) c) ...*), etc.
   Les structures physiques correspondantes mettent souvent en valeur cet ordre (c'est par exemple le cas de la présente énumération où les chiffres 1 et 2 sont suivis d'un renfoncement).
2. les listes pour lesquelles il n'y a pas d'ordre net, comme « Au printemps, poussent dans les jardins des jonquilles, des crocus, des narcisses, des primevères, des jacinthes, etc. ».
   Les structures physiques correspondantes sont très nombreuses ; souvent on marque les éléments d'une liste par un tiret (–) ou, depuis quelques années et sans doute par influence américaine, par un boulet (•). La figure 4 donne la même liste sous diverses formes physiques.

De telles listes sont récursives : comme dans le cas de la présente liste, un élément de liste peut lui-même être une sous-liste.

— des vers,
— des dialogues[14],
— des citations, des gloses, des commentaires dans les marges, etc.
— des résumés,
— des références bibliographiques[15],
— des illustrations, des formules de mathématiques ou de chimie, des tableaux etc. sur lesquels nous reviendrons en section 1.10,
— des renvois (« voir page ... »), des bibliographies et notes en bas de page ou *in fine*, etc. sur lesquels nous reviendrons en section 1.12,
— etc.

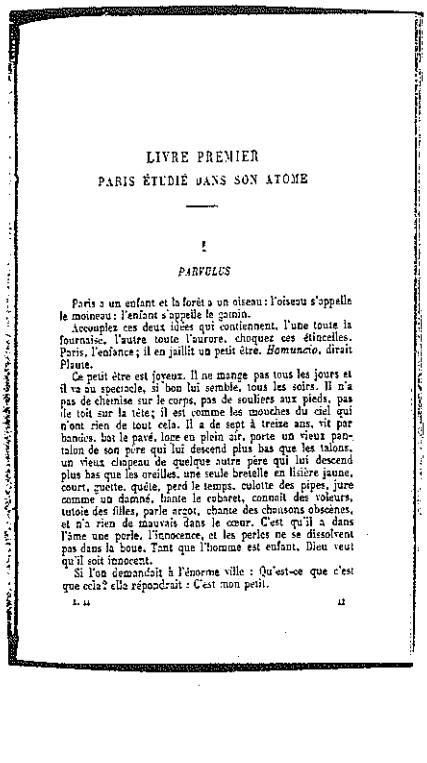
1.8 Correspondances structure logique – structure physique

On peut éditer un roman de plusieurs façons. La figure 5 montre le même début de chapitre dans deux éditions différentes. A une même structure physique peut, en effet, correspondre plusieurs structures physiques.
On trouve de telles différences aussi dans un même ouvrage : certaines œuvres offrent des listes, par exemple, qui, selon l'humeur de l'auteur[16] (et ici il s'agit bien de l'auteur, non du typographe), sont composées d'une façon ou d'une autre. Mais on ne peut quand même pas dire qu'il y ait quelque connotation sémantique ; ce ne sont pas deux entités logiques différentes.
Réciproquement, à une même structure physique peuvent correspondre plusieurs structures logiques. L'italique, structure physique, peut ainsi correspondre dans un même texte soit à un mot étranger, soit à de l'emphase, soit à un titre de revue, etc. Remplacer partout dans un texte l'italique par une mise entre guillemets est

[14] Il semble que ni la signification logique ni la structure physique des dialogues ne soit vraiment connue aujourd'hui. Voir, par exemple, Sylvie Durrer, « Le dialogue romanesque : essais de typologie », *Pratiques*, n° 65, mars 1990, 37–62.

[15] Typiquement on a là un exemple de structure logique qui se décompose en une très grande classe de sous-structures selon que l'on cite, par exemple, un article dans une revue (nom d'auteur, titre de l'article, nom de la revue, date, numéro, pages, ...), un livre (nom d'auteur, titre du livre, éditeur, ville d'édition, année de parution, éventuellement tomaison, etc.), une thèse (auteur, titre, nature de la thèse, université, année), etc. A chacun de ces élément correspond une structure physique conventionelle (les noms d'œuvre – le titre dans le cas d'un livre ou le nom de la revue – en italique, les noms d'auteurs en petites capitales, etc.)

[16] Georges Pérec, particulièrement dans *La vie mode d'emploi*, est coutumier de ce fait.



LIVRE PREMIER

PARIS ÉTUDIÉ DANS SON ATOME

I

PARVULUS

Paris a un enfant et la forêt a un oiseau : l'oiseau s'appelle le moineau ; l'enfant s'appelle le gamin.

Accouplez ces deux idées qui contiennent, l'une toute la fournaise, l'autre toute l'aurore, choquez ces étincelles, Paris, l'enfance ; il en jaillit un petit être. *Homuncio*, dirait Plaute.

Ce petit être est joyeux. Il ne mange pas tous les jours et il va au spectacle, si bon lui semble, tous les soirs. Il n'a pas de chemise sur le corps, pas de souliers aux pieds, pas le toit sur la tête ; il est comme les mouches du ciel qui n'ont rien de tout cela. Il a de sept à treize ans, vit par bandes, bat le pavé, loge en plein air, porte un vieux pantalon de son père qui lui descend plus bas que les talons, un vieux chapeau de quelque autre père qui lui descend plus bas que les oreilles, une seule bretelle, une seule brelle en lisière jaune, court, guette, quête, perd le temps, culotte des pipes, jure comme un damné, hante le cabaret, connaît des voleurs, tutoie des filles, parle argot, chante des chansons obscènes, et n'a rien de mauvais dans le cœur. C'est qu'il a dans l'âme une perle, l'innocence, et les perles ne se dissolvent pas dans la boue. Tant que l'homme est enfant, Dieu veut qu'il soit innocent.

Si l'on demandait à l'énorme ville : Qu'est-ce que c'est que cela? elle répondrait : C'est mon petit.

III. MARIUS

LIVRE PREMIER : PARIS ÉTUDIÉ DANS SON ATOME

I. PARVULUS[46]

Paris a un enfant et la forêt a un oiseau ; l'oiseau s'appelle le moineau ; l'enfant s'appelle le gamin.

Accouplez ces deux idées qui contiennent, l'une toute la fournaise, l'autre toute l'aurore, choquez ces étincelles, Paris, l'enfance ; il en jaillit un petit être. *Homuncio*, dirait Plaute.

Ce petit être est joyeux. Il ne mange pas tous les jours et il va au spectacle, si bon lui semble, tous les soirs. Il n'a pas de chemise sur le corps, pas de souliers aux pieds, pas de toit sur la tête ; il est comme les mouches du ciel qui n'ont rien de tout cela. Il a de sept à treize ans, vit par bandes, bat le pavé, loge en plein air, porte un vieux pantalon de son père qui lui descend plus bas que les talons, un vieux chapeau de quelque autre père qui lui descend plus bas que les oreilles, une seule bretelle en lisière jaune, court, guette, quête, perd le temps, culotte des pipes, jure comme un damné, hante le cabaret, connaît des voleurs, tutoie des filles, parle argot, chante des chansons obscènes, et n'a rien de mauvais dans le cœur. C'est qu'il a dans l'âme une perle, l'innocence, et les perles ne se dissolvent pas dans la boue. Tant que l'homme est enfant, Dieu veut qu'il soit innocent.

Si l'on demandait à l'énorme ville : Qu'est-ce que c'est que cela? elle répondrait : C'est mon petit.

2. QUELQUES-UNS DE SES SIGNES PARTICULIERS

Le gamin de Paris, c'est le nain de la géante.

N'exagérons point, ce chérubin du ruisseau a quelquefois une chemise, mais alors il n'en a qu'une ; il a quelquefois des souliers, mais alors ils n'ont point de semelles ; il a quelquefois un logis, et il l'aime, car il y trouve sa mère ; mais il préfère la rue, parce qu'il y trouve la liberté. Il a ses jeux à lui, ses malices à lui dont la haine des bourgeois fait le fond ; ses métaphores à lui : être mort, cela s'appelle *manger des pissenlits par la racine* ; ses mots à lui, amener des huîtres, baisser les marchepieds des voitures, établir des péages d'un côté de la rue à l'autre dans les grosses pluies, ce qu'il appelle faire *des ponts des arts*, crier les discours prononcés par l'autorité en faveur du peuple français, gratter l'entre-deux des pavés ; il a sa monnaie à lui, qui se compose de tous les petits morceaux de cuivre façonné qu'on peut trouver sur la voie publique. Cette curieuse monnaie, qui prend le nom de *loques*, a un cours invariable et fort bien réglé dans cette petite bohême d'enfants.

Enfin il a sa faune à lui, qu'il observe studieusement dans des coins ; la bête à bon Dieu, le puceron tête-de-mort, le faucheux, « le diable », insecte noir qui menace en tordant sa queue armée de deux cornes. Il a son monstre fabuleux qui a des écailles sous le ventre et qui n'est pas un lézard, qui a des pustules sur le dos et qui n'est pas un crapaud, qui habite les trous des vieux fours à chaux et des puisards desséchés, noir, velu, visqueux, rampant, tantôt lent, tantôt rapide, qui ne crie pas, mais qui regarde, et qui est si terrible que personne ne l'a jamais vu ; il nomme ce monstre « le sourd ». Chercher des sourds dans les pierres, c'est un plaisir d'un genre redoutable. Autre plaisir, lever brusquement un pavé, et voir des cloportes. Chaque région de Paris est célèbre par les trouvailles intéressantes qu'on peut y faire. Il y a des perce-oreilles dans les chantiers des Ursulines, il y a des mille-pieds au Panthéon, il y a des vers rétards dans les fosses du Champ de Mars.

Quant à des mots, cet enfant en a comme Talleyrand. Il n'est pas moins cynique, mais il est plus honnête. Il est doué d'on ne sait quelle jovialité imprévue ; il ahurit le boutiquier de son *fou rire*. Sa gamme va gaillardement de la haute comédie à la farce.

Un enterrement passe. Parmi ceux qui accompagnent le mort, il y a un médecin. — Tiens, s'écrie un gamin, depuis quand les médecins reportent-ils leur ouvrage?

Un autre est dans une foule. Un homme grave, orné de lunettes et de breloques, se retourne indigné : — Vaurien, tu viens de prendre « la taille » à ma femme. — Moi, monsieur! fouillez-moi.

3. IL EST AGRÉABLE

Le soir, grâce à quelques sous qu'il trouve toujours moyen de se procurer, l'*homuncio*[47] entre à un théâtre. En franchissant ce seuil magique, il se transfigure ; il était

46. Le tout petit. 47. Le petit homme.

Figure 5: A une même structure logique (le début d'un chapitre des *Misérables* peuvent correspondre deux structures physiques différentes : mises en page différentes de (à gauche) Les Classiques Flammarion et (à droite) de la collection l'Intégrale au Seuil.

une opération sur la structure physique qui n'a pas de sens logique. C'est pourtant ce que proposent la plupart des systèmes WYSIWYG[17] non structurés.
Nous l'avons dit plusieurs fois, un auteur pense en terme de structure logique et un typographe en réalise la forme physique. Mais il y a des cas où la structure physique influence la structure logique.
Exemple : La presse et certaines collections ont des impératifs très stricts de pagination (par exemple un « Que Sais-je? » fait 128 pages[18]). On joue alors sur la structure logique pour coller avec la structure physique, par exemple en mettant en note de bas de page du texte (qui sinon serait plus long), en créant deux types de paragraphes (les uns composés en corps normal, les autres en corps plus petit), en regroupant des sections (pour gagner la place du titre), voire en supprimant du texte[19] !

1.9 Parenthésage et structures

Pour piloter une photocomposeuse de la seconde génération, on mettait dans le texte des *balises*. Ainsi, pour composer la ligne

Il faut, *a priori*, tout coder.

devait-on fournir quelque-chose comme

Il faut, <CF2>a priori<CF1>, tout coder.

où `CF1` et `CF2` signifient prendre la police numéro 1 (on suppose que c'est du romain) et 2 (italique), respectivement. Une autre façon de faire est d'indiquer le début et la fin de ce que l'on veut mettre en italique, c'est-à-dire d'écrire quelque chose comme :

Il faut, <ITAL>a priori<FIN ITAL>, tout coder.

On a « parenthèsé » la commande. Ceci simplifie le codage (on n'a pas à se soucier de savoir, en fin d'italique, dans quelle police on était : on y revient automatiquement). Mais c'est aussi un moyen de contrôle car toute parenthèse ouverte doit être fermée.
Les structures se prêtent très bien à cette notion du fait que l'on peut les emboîter. Par exemple, une section formée de deux sous-sections sera codée comme suit :

```
\debut{section}
      \debut{sous_section}
            texte de la sous_section 1
      \fin{sous_section}
      \debut{sous_section}
            texte de la sous_section 2
      \fin{sous_section}
\fin{section}
```

Ici, les commandes `\{ ...}` sont donc des parenthèses de nature différente (une `fin section` ne peut que fermer un `debut section`).

[17] *What You See Is What You Get*, nom plus ou moins générique des systèmes interactifs de manipulation de documents. Voir section 3.3 à ce sujet.

[18] Limitation de la structure physique : un livre est formé de cahiers qui font en général 32 pages. Un Que Sais-je? est formé de 4 cahiers et, vu le bas prix des ouvrages de la collection, l'éditeur ne veut pas supporter les coûts qu'imposerait la fabrication ou l'encartage d'un cahier non complet.

[19] Ceci se pratique surtout dans la presse.

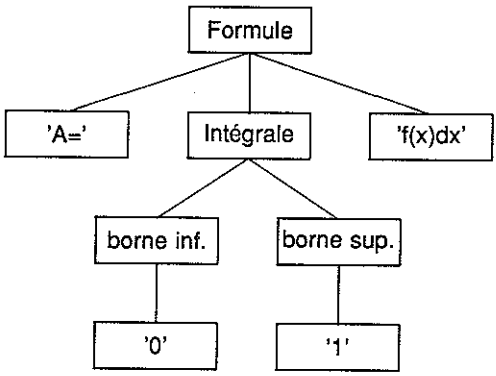


Figure 6: Une structure possible pour la formule $A = \int_0^1 f(x)dx$

1.10 Structures d'objets

Nous avons jusqu'ici principalement abordé la structure globale des documents, sans détailler le contenu des éléments terminaux de cette structure. Certains documents se limitent à un contenu purement textuel ; les éléments de structure, titres ou paragraphes par exemple, contiennent un simple texte linéaire. Mais d'autres documents contiennent également d'autres *objets* :

— des tableaux,
— des schémas,
— des photographies,
— des formules mathématiques,
— des fragments de programme,
— etc.

Un modèle de portée générale doit donc permettre leur représentation.
De la même façon que pour l'ensemble d'un document, on peut considérer les objets de ce genre sous l'angle de leur structure logique. Certains sont clairement structurés, d'autres le sont moins. On peut reconnaître une structure logique dans les formules mathématiques, dans les tableaux, ou dans certains types de schémas. En revanche, il est plus difficile de définir la structure logique d'une photographie ou de certains dessins.
A titre d'exemple, examinons les formules mathématiques pour lesquelles il y a plusieurs façons de considérer la structure :

— On peut considérer leur structure géométrique, comme le faisaient les premiers systèmes de traitement de texte dotés de possibilités de traitement des mathématiques. Pour eux, une formule est essentiellement une mosaïque de caractères et de symboles juxtaposés dans le plan, qui constituent l'aspect graphique de la formule. Ce n'est pas réellement une structure logique.
— A l'opposé, les compilateurs considèrent la structure mathématique des formules dans leurs moindres détails, de façon à pouvoir les calculer ; mais la représentation linéaire qu'ils demandent rend les formules difficilement lisibles.

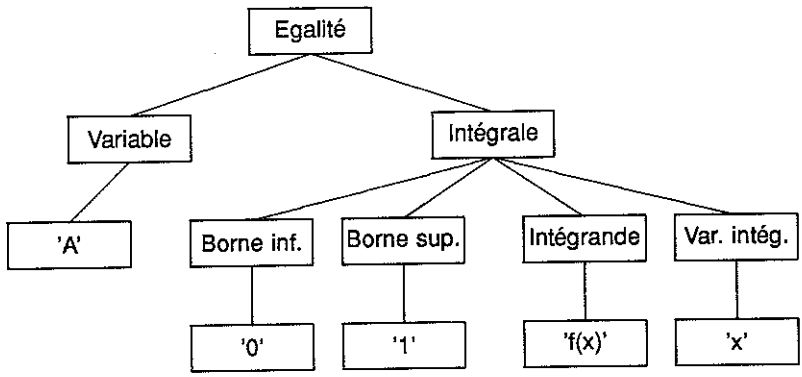


Figure 7: Une structure fine pour la formule $A = \int_0^1 f(x)dx$

La structure retenue doit permettre une saisie et des modifications aisées ainsi que la restitution des formules sur différents supports (écrans et imprimantes aux caractéristiques variées). L'aspect graphique d'une formule est fait pour que la structure mathématique apparaisse clairement au lecteur.
Comme cette structure représente la base sur laquelle reposent les traitements du système, elle doit être connue de l'utilisateur. Les structures de formules dans les systèmes de production de documents sont donc le résultat d'un compromis entre la richesse de la structure et la simplicité de sa manipulation pour l'utilisateur.
La plupart des systèmes utilisent une structure arborescente, mais avec des niveaux de finesse variables.

— La structure minimum ne prend en compte que les éléments structuraux qui affectent le positionnement des différentes expressions constituant une formule. Toute structure mathématique qui s'écrit sous la forme de caractères alignés les uns à la suite des autres est ignorée. La structure arborescente qui représente une formule est constituée d'éléments tels que des fractions, des intégrales, des sommations, des indices et exposants, etc. (voir figure 6). Selon ce modèle, l'équation $y = ax + b$ n'a pas d'autre structure que celle d'une chaîne de caractères, même si on peut lui reconnaître une structure mathématique faisant intervenir une égalité, une somme, un produit, des variables et des paramètres.
— Au contraire, d'autres modèles essaient de considérer ces entités mathématiques et représentent une expression aussi simple que $y = ax + b$ par un arbre où apparaissent tous les détails mathématiques de la formule (voir figure 7). Ces détails peuvent servir à produire une image de la formule qui respecte les règles de la typographie mathématique. Grâce aux informations contenue dans la structure, il est possible, par exemple, d'augmenter les espaces de part et d'autre du signe = ou de l'opérateur +. On peut aussi mettre en italique les variables et les paramètres.

Les tableaux constituent une autre classe d'objets intéressante. En effet, tout le monde s'accorde à dire que c'est ce qu'il y a de plus difficile en typographie. Aussi de nombreux produits ont-ils été proposés pour en faciliter la saisie. Malheureusement, ils sont tous basés sur la structure physique et il est alors pratiquement impossible de transposer un tableau (inverser lignes et colonnes), voire de per-

muter deux colonnes. Il faut dire que la structure logique des tableaux n'est pas encore bien maitrisée[20]

1.11 Structure de forêt

Les structures textuelles que nous avons vues induisent un ordre total (séquentiallité ou hiérarchie) entre tous les éléments du document. Mais un document ne comporte pas que des éléments aussi clairement reliés entre eux. Il existe des éléments dont la position par rapport à la structure du document n'est pas déterminée. C'est notamment le cas des figures ou des notes. Une figure peut être citée depuis plusieurs points du même document et sa place dans le document physique peut varier au cours de la vie du document, sans rien changer au sens ou même à la clarté du document. Elle peut être placée une fois en fin de document, avec toutes les autres figures, une autre fois en haut de la page où se trouve la première citation. Les figures peuvent être dispersées à travers le document ou groupées ensemble. Il en est de même pour les notes, qui peuvent être imprimées en bas de la page où elles sont désignées, groupées en fin de chapitre ou placées en fin d'ouvrage. Bien sûr, il s'agit là de la position physique des éléments dans les documents, mais cela reflète l'instabilité structurale de ces éléments. On ne peut pas les traiter de la même façon que des éléments (comme les paragraphes ou les sections) dont les positions dans la structure sont directement liées à la sémantique du document.
Pour désigner ces éléments dont la position n'est pas fixée de façon unique dans la structure du document, mais qui en font toutefois partie, nous utilisons le terme d'*élément associé*. Ces éléments sont eux-même structurés, c'est-à-dire que leur contenu est organisé logiquement, sous forme de liste ou d'arbre. La figure 8 donne un exemple de document représenté par un arbre principal et un ensemble d'arbres traduisant chacun la structure d'un élément associé. Les trois arbres correspondent aux trois rectangles extérieurs. L'ensemble constitue une *forêt*.

1.12 Structure de graphe et hypertextes

Il existe encore d'autres relations logiques à l'intérieur d'un document qu'aucune des structures précédentes ne peut traduire ; ce sont les renvois et les références croisées. Un passage d'un texte peut renvoyer le lecteur à un autre passage ou à un élément de la structure du document, une section ou un chapitre par exemple. Il y a alors un lien logique entre deux parties du document et ce lien n'est pas traduit par les autres relations (d'ordre ou hiérarchique) apparaissant dans le document. Ce type de lien peut être établi entre des points quelconques. Les références peuvent renvoyer aussi bien en avant (« comme on le verra dans la section *n* ») qu'en arrière (« comme on a vu au chapitre *X* »). Elles peuvent se trouver dans les niveaux les plus profonds d'une structure hiérarchique et renvoyer à un niveau supérieur, ou au contraire pointer depuis les niveaux les plus élevés vers les plus bas.
Un même élément, qu'il fasse partie de la structure principale ou d'un élément associé, peut être désigné en plusieurs points du document et cette relation est bidirectionnelle : un passage du texte renvoie à une figure, par exemple, et cette figure est désignée dans ce passage. Ces relations n'imposent pas nécessairement un nouveau type de structure pour représenter l'organisation logique des documents.

[20] On s'oriente actuellement vers une structure croisée permettant de considérer un tableau comme formé à la fois de lignes de colonnes ou de colonnes de lignes.

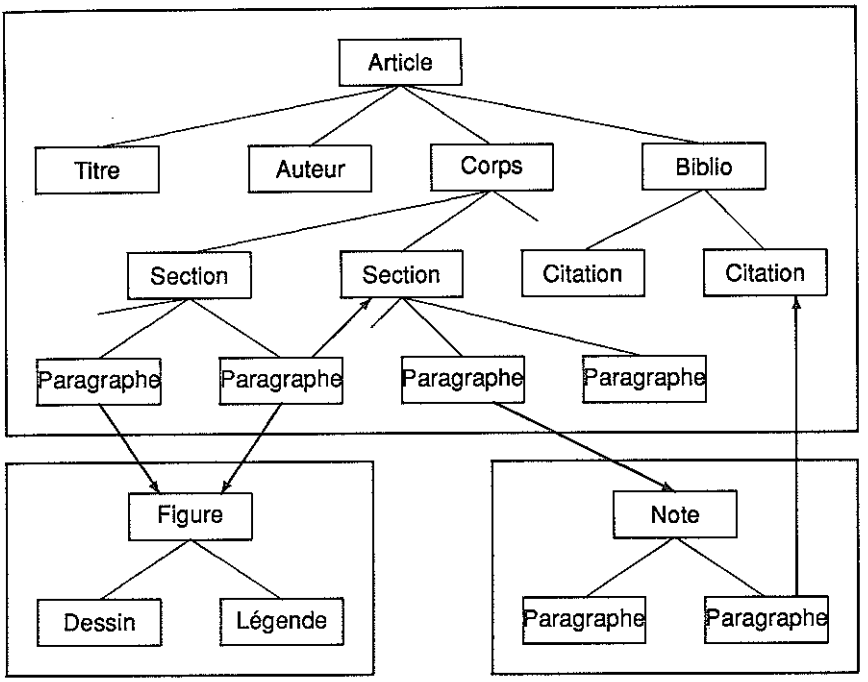


Figure 8: Une structure de forêt avec des références

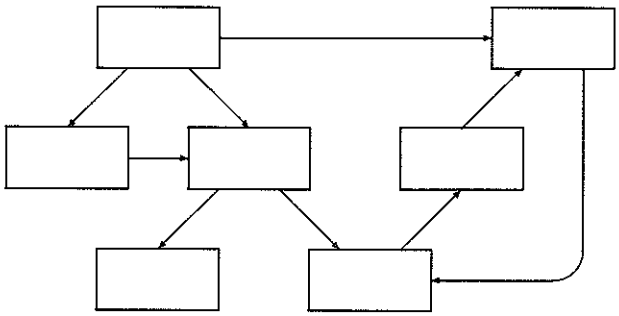


Figure 9: Une structure de graphe

Des liens bi-directionnels peuvent être plaqués par dessus une structure d'arbre ou de forêt, comme on le voit sur la figure 8.
Si certains documents utilisent peu les références, d'autres au contraire en font un usage fréquent[21] voire intensif ou prépondérant. Dans ce cas, une structure hiérarchique, même étendue avec des références, ne rend pas correctement compte de l'organisation du document et un autre type de structure est nécessaire. C'est notamment le cas pour certains documents techniques qui ne sont pas faits pour être lus séquentiellement, mais plutôt pour être consultés à partir d'index et de tables (table des matières, tables des illustrations, table des tableaux...) Le lecteur de tels documents s'intéresse davantage aux références et renvois qu'à la séquence des pages et voit plutôt le document comme un ensemble d'unités d'information reliées entre elles par de multiples relations qu'il utilise selon le type de recherche qu'il effectue. Il n'y a plus vraiment de relations d'ordre ou de relations hiérarchiques dominantes. Pour rendre compte de ce type d'organisation, la structure la mieux adaptée est celle des *graphes* (voir figure 9).
Avec ce type de structure, la notion de document perd un peu de son sens. Les références et les renvois ne relient pas seulement des composants d'un même document, mais également des composants appartenant à plusieurs documents. Et si le découpage en documents différents (volumes, manuels) subsiste, c'est souvent pour des raisons physiques plus que logiques. Des ouvrages de plusieurs milliers de pages poseraient des problèmes de manipulation (poids, encombrement...) mais l'utilisateur est souvent obligé de passer d'un volume à l'autre.
Si l'on fait abstraction du support physique, une encyclopédie ou une grosse documentation technique peut être considérée comme un seul ensemble d'informations dont les éléments ont de multiples relations entre eux.
On en arrive ainsi à la notion de textes que l'on peut parcourir de plusieurs façons. L'une, la plus classique, est de parcourir le texte linéairement, dans l'ordre séquentiel de sa structure logique : chapitre par chapitre et dans chaque chapitre, paragraphe par paragraphe. Une autre façon est de procéder comme à la lecture d'un dictionnaire : partant de l'entrée correspondant à un certain mot, on passe

[21] Signalons ici la mode différente qui existe en Sciences Humaines où l'on utilise beaucoup les notes en bas de page (comme dans ce texte) et en Sciences physiques où ces notes sont très rares. Ceci implique d'ailleurs des structures logiques différentes, lesquelles sont parfois induites par la structure physique selon la possibilité que l'on a, ou du moins « avait » du temps du plomb, de mettre des notes en bas de page ou pas.

à une autre entrée en suivant le lien apparaissant dans une définition, qu'il soit implicite (« **table** : meuble formé ... » : on va regarder à « meuble ») ou explicite (« voir ce mot »). Ces liens peuvent être de nature très variée dans un même texte. De tels textes font partie des *hypertextes*[22]

1.13 Les redondances

Les références permettent de partager certaines parties de document entre plusieurs points du texte et ainsi d'éviter des redondances. Au contraire, certains documents comportent des répétitions voulues, dans le but d'assister le lecteur dans le parcours du document. La table des matières est un bon exemple de ce phénomène. Elle ne contient aucune information qui ne se trouve déjà dans le livre ; elle ne fait que répéter, sous une forme condensée, des informations disséminées à travers tout le document : le numéro et le titre de chaque chapitre et de chaque section ainsi que le numéro de page où débute chaque élément. On peut citer également la table des figures ou des illustrations que l'on trouve à la fin de certains documents.

Un autre cas de redondance est celui des index. Un index regroupe, à la fin ou en tête d'un ouvrage, un ensemble de termes importants utilisés une ou plusieurs fois dans le corps de l'ouvrage. Ces termes sont généralement classés par ordre alphabétique et sont accompagnés des numéros de page ou de section où ils apparaissent.

Bien qu'ils représentent un aspect structural important, les index ou les tables ne constituent pas une structure supplémentaire, s'ajoutant aux précédentes. Il s'agit simplement d'une autre transposition graphique de la même structure logique. Si on se réfère au modèle arborescent (voir figure 8), l'arbre représente *à la fois* le corps du document et sa table des matières. Il suffit, pour voir la table des matières, de ne retenir dans l'arbre du document que les nœuds qui représentent les titres de chapitre et les titres de section. La structure de la table des matières est contenue dans la structure globale du document. Il en est de même pour une table des figures.

1.14 Pourquoi structurer?

Posons nous, maintenant que nous savons ce qu'est une structure, la question de l'intérêt des structures. Pourquoi structurer?

— parce que c'est naturel. Les auteurs pensent naturellement (ou du fait de leur éducation?) en termes de chapitres, sections, paragraphes, etc.

— pour séparer des processus différents. Notamment, séparer la mise en relation des éléments d'un texte de leur représentation revient à séparer le travail de l'auteur de celui du typographe ;

— pour vérifier la cohérence d'un texte : sans structures, on ne peut pas, sauf en lisant un texte (et encore !), découvrir des incohérences comme l'inversion de deux éléments structurels ;

— numéroter les sections, sous-sections etc.

[22] L'objet de ce cours est, notamment, de présenter les hypertextes. Ce sera l'objet des exposés de R. Dachelet et de V. Quint ; on lira donc les articles correspondants dans cet ouvrage, *passim*.

— pour des possibilités de « sécurité » : la structuration logique d'un texte permet d'associer une certaine redondance nécessaire à la détection d'erreurs[23] ;
— pour des économies de place ou d'occupation de lignes : un document prend moins de place lorsqu'il est structuré logiquement que si l'on transmet sa structure physique ;
— pour transmettre des styles de documents ou des consignes de saisie, tout comme on transmet des recettes de cuisine ;
— etc.

1.15 Un exemple d'application : la reconnaissance optique des textes

L'intérêt de la reconnaissance optique des textes n'est plus à démontrer. Mais, même si d'énormes progrès sont en cours[24], il n'en reste pas moins que des documents tant soit peu complexes, un recueil de textes de lois par exemple, sont encore difficiles à lire automatiquement.
Une approche très intéressante a été proposée par Rolf Ingold[25] : L'idée est que la structure d'un document (si on la connait *a priori*, ce qui est le cas des documents d'archive) peut aider à la reconnaissance des parties de texte (titres, résumé, articles d'une loi, etc.) et, dans celles-ci, à la reconnaissance des caractères (dans une date par exemple, l'ambiguïté classique entre la lettre l et le chiffre 1 peut être levée : on interprétera *19 juillet 1881* et non *l9 juillet l88l*).

2. Normes pour les documents structurés

Lors de l'apparition de la micro-informatique, on a cru que le fait d'avoir un texte sur une disquette allait permettre d'envoyer ce « compuscrit » à un éditeur qui n'aurait plus qu'à le faire flasher par un photocompositeur pour en sortir un livre à bas prix. On a vite déchanté : d'une part peu d'auteurs utilisaient des systèmes de structuration logique[26], mais surtout on a vu que ces systèmes n'étaient pas compatibles. Cette section va montrer où en sont les travaux de normalisation.
La multiplication des systèmes de production de documents ainsi que l'évolution des imprimantes et autres appareils d'affichage a donc rendu très difficile l'échange de documents sous leur forme électronique. Chaque système a son propre format de représentation, chaque imprimante a son propre jeu de commandes.
Pour résoudre ce problème, les organismes de normalisation ont défini des modes de représentation des documents qui doivent faciliter les échanges entre systèmes différents.

---

[23] L'une des nombreuses causes de l'accident de train qui, Gare de Lyon à Paris en Juin 1988, causa la mort d'une soixantaine de personnes est due à la mauvaise mise en page d'un manuel de réparation de freins : l'emploi d'un formateur structuré aurait détecté cette erreur de parenthésage (voir section 1.9) ; voir Jacques André, « LaTeX pouvait-il faire éviter l'accident de la gare de Lyon? », *Cahiers Gutenberg*, numéro 1, avril 1989, 21–25.

[24] On consultera sur ce sujet les actes du colloque *Reconnaissance automatique de l'écrit*, Le Havre 18 mai 1990, dans *Bigre* n° 68, mai 1990.

[25] Rolf Ingold, *Structures de documents et lecture optique : une nouvelle approche*, Presses Polytechniques Romandes, collection Meta, Lausanne, 1989.

[26] Il fallait alors débarasser les textes des attributs typographiques amateurs pour les remplacer par ceux de professionnels.

Par ailleurs, un certain nombre de produits ont atteint une diffusion suffisante pour devenir des normes de fait, des « standards » du marché. PostScript en est un bon exemple[27].

Les normes dont il est question ici sont d'une autre nature. Elles sont, au contraire, le résultat d'un travail impliquant un grand nombre de spécialistes provenant de nombreux pays et organisations. Elles ont été élaborées sous l'égide des organismes internationaux tels que l'ISO *(International Standards Organization)* ou le CCITT (Comité Consultatif International du Télégraphe et du Téléphone) et ont de ce fait un label officiel[28].

Il existe deux catégories de normes de représentation des documents.

— Les unes concernent les documents formatés (elles décrivent, par la structure physique, des pages que l'on ne peut qu'imprimer ou afficher sur un écran),
— d'autres s'intéressent à des documents révisables (dont il est possible de modifier le contenu et l'organisation, c'est-à-dire de jouer sur la structure logique et le contenu[29] ;
— il faut enfin mentionner les normes touchant au domaine du graphique[30] qui peuvent décrire certaines parties des documents.

L'ISO propose deux normes concurrentes pour la représentation des documents révisables, SGML et ODA, dont nous montrons maintenant les principales caractéristiques[31]

2.1 SGML

SGML *(Standard Generalized Markup Language)* est une norme d'origine américaine. Elle est dérivée de GML *(Generalized Markup Language)* d'IBM[32]. Le projet de norme a d'abord été soumis à l'ANSI *(American National Standards Institute)* avant d'être proposé à l'ISO et de devenir ainsi une norme internationale.

---

[27] Il s'agit d'un langage de « description de page » permettant de piloter une imprimante à laser ou une photocomposeuse. Il a été conçu par une petite équipe, les fondateurs de la société Adobe, indépendemment de tout organisme officiel de normalisation, puis il a été mis sur le marché. Plusieurs constructeurs d'imprimantes l'ont adopté ainsi qu'un nombre croissant de développeurs de logiciels, jusqu'à en faire un passage presque obligé entre un système de production de documents et une imprimante.

[28] Ces organismes ont des représentants nationaux, comme l'AFNOR en France, l'ASA *(American Standard Association)* ou le DIN *(Deutsch Institut für Normalization)*, ces deux derniers étant bien connus pour leurs normes en matière de films photographiques !

[29] Voir V. Joloboff, « Trends and standards in document representation ». In J. C. van Vliet, editor, *Text Processing and Document Manipulation, Proceedings of the International Conference*, pages 107–124, Cambridge University Press, 1986.

[30] On trouvera une synthèse sur ces normes dans P. R. Bono. « A survey of graphics standards and their role in information interchange », *IEEE Computer*, 18(10):63–75, October 1985, et dans M. Lucas, « Les langages de programmation graphique », *Traité d'informatique*, Techniques de l'ingénieur, Paris 1987. Nous n'en parlerons pas plus ici.

[31] Ces deux documents sont officiellement définis dans les documents ISO suivants : *Information Processing—Text and Office Systems—Office Document Architecture (ODA) and Interchange Format, ISO/DIS 8613*, 1986 et *Information processing—Text and office systems—Standard Generalized Markup Language (SGML), ISO 8879*, 1986.

[32] GML est le langage du produit DCF *(Document Composition Facility)*. C'est Charles Goldfarb, du laboratoire de recherche d'IBM à San Jose, Californie, qui en est à l'origine. Cf. C. F. Goldfarb, « A generalized approach to document markup », *ACM SIGPLAN Notices*, 16(6):68–73, 1981.

2.1.1 Les principes

SGML est un langage de marquage. Il permet de définir et d'utiliser des *marques* qui délimitent, dans le texte d'un document, les différents constituants logiques. Dans son esprit il est proche des langages des formateurs comme Scribe ou LaTeX. Il considère un document comme une structure abstraite formée d'éléments de types divers : chapitres, sections, paragraphes, figures, légendes, notes, etc.
C'est aussi un langage générique, puisqu'il permet de spécifier les types d'éléments à utiliser pour décrire les documents d'une classe donnée, ainsi que les relations que ces éléments peuvent avoir entre eux dans la structure logique du document. La structure logique est essentiellement arborescente, mais il est possible, par le biais des attributs (voir plus loin), d'introduire des relations non-hiérarchiques entre éléments.
Le marquage est de type déclaratif. Les marques (l'information ajoutée au texte même du document) délimitent et identifient les éléments de la structure et leur associe des attributs. Elles ne portent aucune indication de traitement. Les traitements ne sont pas définis dans le document lui-même, mais par les applications qui les associent aux marques fournies par SGML. C'est ce qui permet à plusieurs applications, par exemple des formateurs ou des systèmes documentaires, de travailler sur la même description du document. Il est toutefois possible d'insérer dans le document certaines procédures de traitement destinées à un système particulier, mais elles sont alors clairement isolées du reste du document et peuvent être ignorées par les autres systèmes.
Le marquage utilisé est défini formellement pour chaque classe de documents. Des définitions de types indiquent quels éléments et quels attributs peuvent figurer dans un document et dans quel ordre ils peuvent apparaître. Ces définitions permettent de spécifier des classes de documents et de vérifier si un document donné est bien conforme au modèle de sa classe. Elles permettent aussi, comme on le verra, un marquage incomplet des documents, puisque les marques absentes peuvent être retrouvées de façon non ambiguë à partir des définitions.

2.1.2 Le langage

La norme définit un méta-langage pour la déclaration des structures logiques des documents et des différents types d'éléments qui les composent. Ces types d'éléments sont appelés *identificateurs génériques*. L'exemple suivant montre, d'une façon très simplifiée, comment on peut déclarer les éléments qui composent un article et leurs relations structurales.

```
<!ELEMENT article      - -  (titré, auteur+, section+)   >
<!ELEMENT titre        O O  (#PCDATA)                    >
<!ELEMENT auteur       - O  (#PCDATA)                    >
<!ELEMENT section      - O  (titrechap, paragraphe+)     >
<!ELEMENT titrechap    O O  (#PCDATA)                    >
<!ELEMENT paragraphe   - O  (#PCDATA)                    >
```

D'après cette déclaration – en ignorant pour l'instant les deux colonnes de signes - et O – un article est formé d'un titre, suivi d'un ou plusieurs auteurs, eux-mêmes

[33] Des *balises* dans le langage des typographes ; toutefois, ces balises sont souvent axées sur la structure physique (« prendre telle fonte », « faire un renfoncement de 12 points » etc.) ou sur les types au sens de la section 1.2 (« début de section » donc – grâce à des macros – « interligner de 12 points, passer en gras corps 14, etc. ») mais très rarement sur la structure logique hiérarchique. Voir section 1.9.

suivis d'une suite de sections. Le signe + après `auteur` indique qu'il peut y avoir plusieurs auteurs, mais qu'il en faut au moins un. De même, il y a au moins une section dans un article. Le titre est obligatoire. Le titre et un auteur sont chacun constitués d'une simple chaîne de caractères (`#PCDATA`). Chaque section contient un titre, obligatoire, suivi d'une suite de paragraphes (au moins un paragraphe). Le titre de section et le paragraphe sont de simples suites de caractères.
La syntaxe de SGML définit un type d'élément par son identificateur générique (ici, `article, titre, auteur, section...`) et, entre parenthèses, son contenu. Dans la définition du contenu, on trouve des identificateurs génériques reliés par des *connecteurs* et éventuellement suivis d'un *indicateur d'occurrence.*
Les connecteurs sont au nombre de trois :

— La virgule, entre deux éléments, indique que les éléments doivent figurer dans l'ordre indiqué.
— Le signe `&`, à la place des virgules, indique que l'ordre des éléments peut être quelconque.
— La barre verticale | indique qu'un seul élément doit figurer, choisi parmi ceux qui sont séparés par des barres.

Les indicateurs d'occurrence sont également au nombre de trois :

— Le point d'interrogation ? indique que l'élément qu'il suit est facultatif.
— L'étoile * indique que l'élément qu'elle suit est facultatif et répétable (0 ou n fois)
— Le signe + indique que l'élément qu'il suit est obligatoire et répétable (1 ou n fois).

La définition du contenu d'un élément peut être suivie de la définition des *attributs* applicables à cet élément. La définition d'un attribut spécifie le nom de l'attribut, ses valeurs possibles et une valeur par défaut qui sera affectée à l'élément si aucun attribut ne figure dans la description d'un document.
L'ensemble de ces définitions permet de spécifier la structure logique générique d'une classe de documents. Pour que le récepteur d'un document puisse interpréter correctement ce qu'il reçoit, ces définitions accompagnent tout document et précèdent la description du document lui-même, c'est-à-dire sa structure logique spécifique et son contenu. A titre d'exemple, voici comment serait représenté le présent texte en fonction de la déclaration donnée ci-dessus :

```
<article> Structures et modeles de documents
<auteur> Jacques Andre ...
<auteur> Vincent Quint ...
<paragraphe>
Ce texte ...
<section>
Introduction aux structures de documents
<paragraphe>
Tout d'abord, qu'est-ce qu'un document ? Nous...
<paragraphe>
En gros, on peut appeler...
 ...
<section>
Normes pour les documents structures
<paragraphe>
On a cru ...
</article>
```

Sur cet exemple très simplifié, on peut remarquer que certaines marques n'apparaissent pas dans le document. Ce sont les marques implicites indiquées dans la définition de type. En effet, les symboles - et o qui apparaissent dans la définition des éléments dénotent les marques qui peuvent être omises (o) et celles qui doivent nécessairement figurer dans le document (-). La première colonne concerne la marque de début de l'élément, la deuxième colonne concerne la marque de fin. La marque de fin se distingue de la marque de début par le caractère '/' après '<'. Comme une section commence toujours par un titre de section, il n'y a pas de marque explicite de début de titre de section. Pour la même raison, il n'y a pas non plus de marque de début pour le titre de l'article. La plupart des marques de fin sont omises, puisque la marque de début d'un élément indique sans ambiguïté la fin du précédent. Seul l'article lui-même nécessite une marque de fin explicite.

2.1.3 Utilisation de SGML

Comme on peut le voir sur cet exemple simple, les déclarations et les descriptions de documents peuvent être produites « à la main », c'est-à-dire avec un simple éditeur de texte, et sont donc lisibles pour l'œil humain. Bien que des logiciels spécifiques soient utiles pour produire des documents selon la norme, SGML n'impose pas l'utilisation de tels logiciels et peut même être considéré comme un moyen d'échange de documents entre êtres humains, par exemple un auteur et un éditeur. Il faut néanmoins noter que des environnements spécifiquement adaptés à l'édition de documents SGML sont d'une grande utilité pour les auteurs[34]
SGML ne permet pas seulement de décrire la partie textuelle des documents. Il permet aussi d'inclure des éléments non textuels dans les feuilles de la structure, notamment des images et des graphiques. Il permet aussi de structurer des éléments comme des formules mathématiques ou des tableaux.
SGML est un langage général qui peut s'adapter à de nombreux types de documents et d'objets et à de nombreuses applications. Sa généralité même a conduit ses utilisateurs à définir un cadre dans lequel la norme doit être utilisée pour certaines catégories d'applications.

— La première grosse expérience vient de l'Association Américaine des Editeurs (AAP : *American Association of Publishers*). Son projet Manuscrit Electronique a pour objectif de fournir aux auteurs et aux éditeurs un format unique de description des documents[35]. En choisissant une représentation abstraite des documents, les éditeurs accroissent les possibilités d'utilisation des manuscrits. Si tous les manuscrits respectent le même format, les éditeurs pourront les utiliser efficacement grâce à des bases de données adaptées. Ils pourront aussi obtenir automatiquement plusieurs versions d'un même travail : un article dans un journal, un fichier dans une base de documents ou une entrée dans une base bibliographique. Ce projet vise directement les publications scientifiques : livres, monographies, thèses, articles, rapports techniques, actes de conférences, etc. Il comprend également une partie consacrée aux formules mathématiques. Le ministère américain de la défense (DOD) fait actuellement un gros *forcing* sur SGML, ce qui risque de rendre cette norme très utilisée.

---

[34] Voir Le van Huu. « An environment for SGML documents preparation », *Proceedings of the Summer 1987 USENIX Conference*, pages 43–52, USENIX Association, Phoenix, Arizona, June 1987.

[35] Association of American Publishers, *Standard for Electronic Manuscript Preparation and Markup*, A.A.P., Washington, February 1986.

— En France, le Syndicat National de l'Edition a mené une opération comparable, mais dont les résultats ne sont pas très rapides[36].

2.2 ODA

La norme ODA *(Office Document Architecture* ou *Open Document Architecture)* est, elle, d'origine européenne[37].
A la différence de SGML, ODA prend en compte le formatage (la structure physique) des documents en plus de leur structure logique et de leur contenu. Plus précisément, ODA permet de décrire

— des documents formatés, qui peuvent seulement être affichés ou imprimés selon les intentions exprimés par l'émetteur,
— des documents révisables non formatés, qui sont destinés à être édités et formatés par le récepteur,
— des documents révisables et formatés qui peuvent être soit imprimés tels qu'ils sont, soit édités et reformatés par le récepteur.

ODA distingue une *structure logique* et une *structure physique* dans les documents. Un document formaté n'a qu'une structure physique, un document révisable non formaté n'a qu'une structure logique et un document révisable et formaté possède les deux structures à la fois.

2.2.1 Structure logique et structure physique

Dans cette section, on décrit les deux structures, mais précisons bien que, selon l'usage que l'on veut faire du document, une seule des deux structures peut être présente.

— La structure logique est une structure arborescente où chaque nœud de l'arbre représente un objet logique du document. Tout nœud de la structure est accompagné d'*attributs* qui fournissent un complément d'information sur l'objet. Contrairement à SGML, les attributs disponibles sont définis dans la norme elle-même, et non pas dans la classe de documents. Ils n'ont d'ailleurs pas le même sens. Les attributs d'ODA peuvent en particulier exprimer des relations entre objets et notamment des relations non hiérarchiques. La structure ne décrit que l'organisation logique du document, et tout son contenu se trouve exclusivement dans les feuilles de l'arbre.
— La structure physique est également arborescente et représente un découpage du document en *ensembles de pages*, en *pages*, en *cadres* et en *blocs*. Les cadres et les blocs sont des zones rectangulaires à l'intérieur des pages, dont les côtés sont parallèles aux bords de la page qui les contient. Les cadres peuvent être décomposés à leur tour en cadres et en blocs, les blocs représentant

[36] Certains résultats sont cités dans D. Vignaud. *L'édition structurée des documents, SGML, application à l'édition française*, Éditions du Cercle de la librairie, 1989. Cet ouvrage est surtout une bonne introduction, en Français, à SGML. Deux autres ouvrages utiles sur SGML, en anglais, sont ceux de M. Bryan, *SGML, an Author's Guide to the Standard Generalized Markup Language*, Addison-Wesley, 1988 et de E. van Herwijnen, *Practical SGML*, Kluwer Academic Publishers, 1990.

[37] Avant d'être proposée à l'ISO et au CCITT, elle a d'abord été élaborée au sein de l'ECMA *(European Computer Manufacturers Association)* : *Standard ECMA-101 Office Document Architecture*, E.C.M.A., Genève, Septembre 1985.

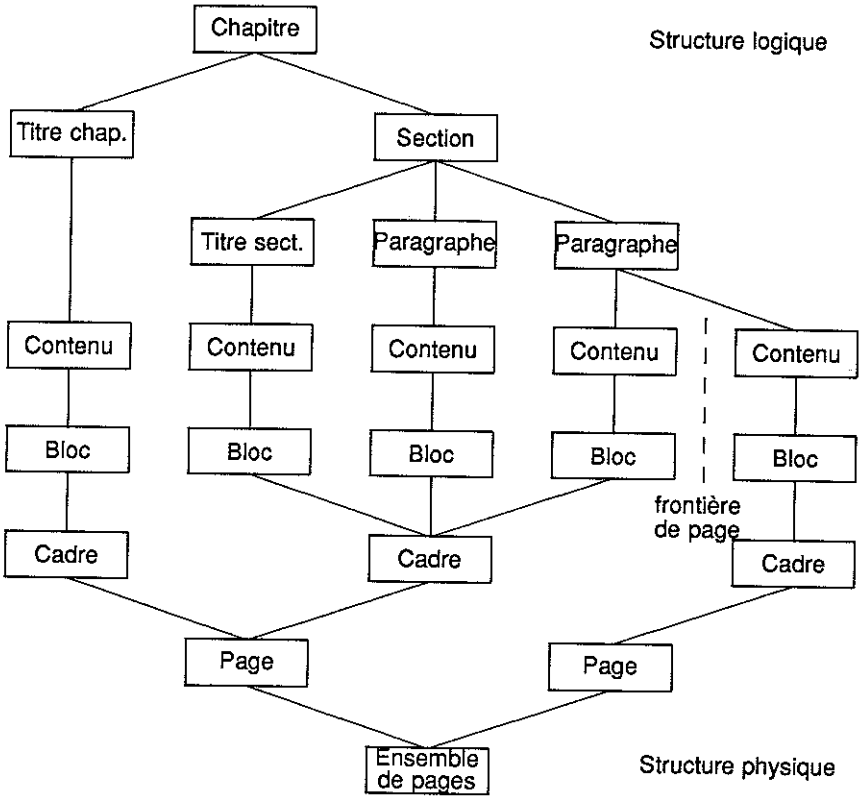


Figure 10: les structures logique et physique de ODA

les atomes de cette structure. Un ensemble de pages peut correspondre à un chapitre, ou à une annexe d'un rapport, en général, à une grande division du document. Les cadres peuvent être utilisés pour représenter les colonnes d'une page, chaque colonne contenant des blocs qui représentent chacun un paragraphe. A la place d'un de ces blocs, on peut trouver, pour une figure, un autre cadre qui regroupe deux blocs, l'un représentant un dessin et l'autre la légende de la figure. Tout comme la structure logique, la structure physique n'autorise un contenu que pour ses atomes, les blocs.

Le contenu est partagé entre les deux structures, comme le montre la figure 10. Sur cet exemple, on voit un extrait d'un livre commençant au début d'un chapitre. La partie supérieure de la figure montre la *structure logique spécifique* de cette partie de document, alors que la partie inférieure montre sa *structure physique spécifique*. Entre les deux, se trouve le contenu. On peut remarquer que, dans la structure logique, le deuxième paragraphe possède deux portions de contenu, non pour des raisons d'organisation logique du document, mais pour des raisons de mise en page. Lors du formatage, le paragraphe ne pouvant tenir intégralement dans la page, il a été coupé en deux blocs, l'un en bas de la première page du chapitre, l'autre en haut de la deuxième page. Ce découpage en deux blocs a entraîné le découpage du contenu en deux portions.
Les feuilles des structures logique et physique sont des portions de contenu, qui peuvent être de natures différentes. Ces natures sont appelées dans ODA des *architectures de contenu*. Les trois premières architectures de contenu définies sont les caractères, le graphique photographique et le graphique géométrique. Chacune de ces architectures définit la forme d'un type de contenu et les règles de traitement qui s'y appliquent. Il y a plusieurs niveaux de représentation des différents contenus. Ainsi, l'architecture du contenu photographique correspond, au niveau le plus bas, aux normes CCITT de la télécopie, alors que les niveaux les plus élevés autorisent des traitements plus élaborés, comme des agrandissements, des réductions ou du découpage.
Le contenu caractère, au plus bas niveau, est compatible avec les normes CCITT des services Télex et Télétex. L'architecture de contenu définit le positionnement des caractères, la direction de l'écriture, les attributs graphiques des caractères (graisse, style, souligné...), le choix des polices, le positionnement des lignes et des mots dans les lignes, les retraits, les tabulations, etc.

2.2.2 Structures génériques

ODA utilise une approche objet où les *classes* jouent le même rôle que les types dans SGML. Elles représentent un ensemble d'*objets* qui ont les mêmes caractéristiques. Les objets qui composent un document particulier sont des exemplaires de ces classes. Le document lui-même est un objet et il existe donc des classes de documents. Tout comme les types de SGML, les classes de ODA ne sont pas définies par la norme, qui offre seulement les moyens de les spécifier. Cette spécification se fait à travers des *structures génériques* logiques et physiques qui décrivent les classes d'objets logiques et physiques. Un document particulier a une structure spécifique logique et/ou une structure spécifique physique qui sont conformes à ces structures génériques. Les structures génériques assurent la cohérence des structures spécifiques et servent à guider la création et le formatage des documents.

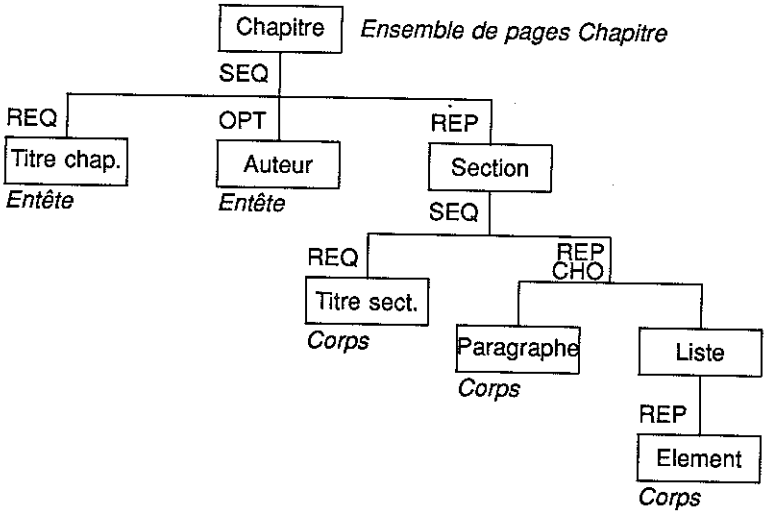


Figure 11: ODA, structure logique générique

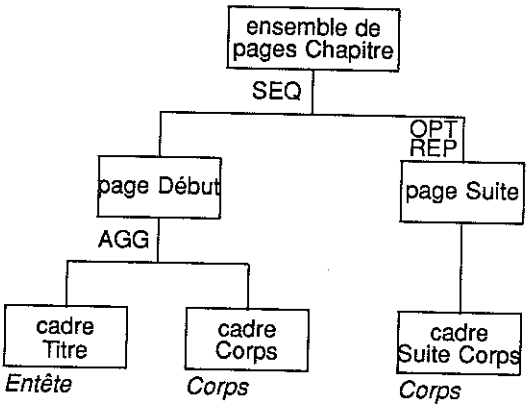


Figure 12: ODA, structure physique générique

Une structure générique est un ensemble de définitions de classes d'objets. Dans chaque définition, un attribut« 'générateur de subordonnés » spécifie les composants possibles de l'objet et leur mode d'assemblage. Chaque composant peut être :

— obligatoire : il doit apparaître une fois et une seule (noté REQ),
— facultatif : il peut apparaître une fois ou pas du tout (noté OPT),
— répétable : il peut apparaître une ou plusieurs fois (noté REP).

Il est à noter qu'un même composant peut être à la fois facultatif et répétable (noté OPT REP), c'est à dire qu'il peut ne pas apparaître ou apparaître une ou plusieurs fois. L'attribut indique également la façon d'assembler les composants :

— en séquence, dans l'ordre indiqué (noté SEQ),
— par agrégation, sans ordre déterminé (noté AGG),
— au choix, un parmi les composants possibles (noté CHO).

Les structures génériques sont également décrites sous forme d'arbre. La figure 11 montre la structure logique générique qui a permis de construire la structure logique spécifique de la figure 10. Elle indique qu'un chapitre est formé, dans l'ordre (SEQ), d'un titre de chapitre obligatoire (REQ), d'un nom d'auteur facultatif (OPT), et d'une ou plusieurs (REP) sections. Chaque section possède, dans l'ordre (SEQ), un titre obligatoire (REQ) et une suite (REP) de paragraphes ou (CHO) de listes. Une liste est formée d'une suite (REP) d'éléments de liste.
Pour obtenir la structure physique spécifique de la figure 10, on a utilisé la structure physique générique décrite par la figure 12. Celle-ci définit notamment deux types de page pour un chapitre, un type pour le début de chaque chapitre et un autre type pour la suite du chapitre. La première page d'un chapitre comporte un cadre pour l'en-tête, qui contiendra le titre du chapitre et le nom de l'auteur s'il est présent, et un cadre pour le début du corps du chapitre. Si le chapitre fait plus d'une page, des pages de suite de chapitre seront créées, chacune contenant un cadre pour la suite du corps. Cet exemple est très simplifié, et dans la réalité d'autre cadres sont nécessaires pour les titres courants, les numéros de page, les colonnes, etc.

2.2.3 Mise en page

La mise en page s'appuie sur les *styles de mise en page* et les *styles de présentation*. Les styles sont associés aux objets logiques. Les styles de mise en page indiquent comment créer des pages et comment diviser ces pages en cadres et en blocs, alors que les styles de présentation, qui ne peuvent s'attacher qu'aux objets de base, aux feuilles des structures, indiquent comment formater et visualiser les portions de contenu.
La mise en page construit la structure physique spécifique d'un document à partir de sa structure logique spécifique, et en utilisant les structures génériques, les architectures de contenu et les styles pour placer le contenu du document dans des blocs, des cadres et des pages. Plus précisément, on utilise les architectures de contenu et les styles de présentation pour mettre les portions de contenu dans des blocs. Ces blocs sont ensuite placés dans des cadres et des pages selon le style de mise en page et la structure physique générique. Notons que cette façon de décrire le processus de mise en page est purement formelle ; il s'agit d'un modèle et non pas d'une spécification de réalisation.
Les styles de mise en page établissent une correspondance entre les objets de la structure logique et ceux de la structure physique. Les principaux attributs qui

jouent ce rôle dans les styles de mise en page sont les *classes d'objets physiques*, les *catégories physiques* et les *catégories permises*.
Une *classe d'objets physiques* est utilisée pour indiquer qu'un objet entier de la structure logique doit être mis dans un seul objet, d'un type déterminé, de la structure physique. Les seuls objets physiques utilisables avec cet attribut sont les pages et les ensembles de pages, ce qui permet d'opérer les grandes divisions d'un document. Dans notre exemple, la classe d'objet physique de l'objet logique Chapitre est l'ensemble de pages Chapitre. Ainsi chaque chapitre a son propre ensemble de pages, et commence par une page particulière, la page Début (voir figure 12). Sur la figure 11, le texte en italique à côté de l'objet Chapitre représente l'attribut classe d'objet physique de cet objet logique.
Dans ces grandes divisions d'un document, les deux attributs *catégorie physique* et *catégorie permise* servent à diriger les objets logiques de base vers les cadres de la structure physique. La catégorie physique s'applique aux objets de la structure logique alors que la catégorie permise s'applique aux cadres. Un objet logique qui a un nom de catégorie physique ne peut aller que dans un cadre qui a le même nom parmi ses catégories permises.
Dans notre exemple, les catégories physiques sont indiquées en italique en dessous de chaque objet de base sur la figure 11. Il n'y a que deux catégories physiques utilisées : Entête et Corps. Les catégories autorisées apparaissent de la même façon sur la figure 12. Dans le processus de mise en page, lorsque le titre de chapitre est traité, on cherche un cadre qui comporte la catégorie permise Entête. On trouve ainsi le cadre Titre de la page Début et on met le contenu du titre dans ce cadre. De la même façon, on met dans ce cadre le nom de l'auteur s'il est présent. Les autres composants d'un chapitre ont Corps pour catégorie physique ; ils sont donc mis dans le cadre Corps de la page Début, jusqu'à ce que ce cadre soit plein. Les composants suivants seront ensuite mis dans des cadres Suite Corps de pages Suite.
Grâce à ces attributs, le contenu de la structure logique est dirigé dans des pages et des cadres. Il reste encore à construire les blocs et à les placer dans les cadres. Pour cela, les blocs ont un attribut dimension qui détermine leur taille maximale. Ainsi un bloc paragraphe prendra la largeur de son cadre, et s'étendra verticalement jusqu'à contenir tout son texte. Un attribut position peut déterminer la position d'un bloc par rapport au cadre ou à la page qui le contient, mais le plus souvent les positions sont définies relativement aux autres objets par un ensemble d'attributs : *ordre de remplissage*, *décalage* et *séparation*, pour les objets de base, et *object path* pour les pages et les cadres.
L'attribut *object path* indique dans quelle direction une page ou un cadre doit être rempli. C'est habituellement de haut en bas, chaque bloc venant se placer en dessous du précédent. Cet ordre peut être changé pour certains objets de base qui portent un attribut *ordre de remplissage*. La position précise de chaque bloc est contrôlée par les attributs *décalage* et *séparation* qui indiquent la distance minimum entre deux blocs consécutifs et la distance entre un bloc et le cadre qui le contient.

2.3 Conclusion

Bien que le vocabulaire ne soit pas le même, il n'y a pas de différence fondamentale entre SGML et la partie logique de ODA. Les différences se situent évidemment au niveau physique, puisque SGML ne prend pas cette représentation en compte, du moins dans la version actuelle. Il faut toutefois noter que des extensions sont

à l'étude pour combler cette lacune ; c'est notamment le but du projet de norme DSSSL *(Document Style Semantics and Specification Language)*.

Alors que les objets des structures logiques d'ODA présentent une certaine uniformité (ils sont divisés en objets de base, décrivant le contenu, et en objets composés, qui sont tous les autres objets), les objets des structures physiques sont très variés ; ils se répartissent en cinq catégories : les ensembles de pages, les pages de base, les pages composées, les cadres et les blocs.

Qu'il s'agisse de la structure logique ou physique, dans ODA, les feuilles des arbres sont des portions de contenu, qui peuvent être de nature textuelle, géométrique ou photographique. D'autres natures (ou « architectures ») de contenu seront ajoutées par la suite. Chaque architecture de contenu a sa propre structure, qui est organisée différemment de celle du document.

Alors que SGML permet de décrire les structures logiques des formules mathématiques ou des tableaux, ODA ne considère pas ce genre d'éléments. L'approche retenue consiste à les représenter à l'avenir avec des architectures de contenu spéciales, bien que les outils existant au niveau logique soient utilisables. Il est vrai qu'au niveau physique, ODA se prête mal à la description de ce genre d'objets.

On a vu que la description SGML d'un document pouvait être produite avec des outils très simples et d'usage courant. ODA, au contraire, nécessite des outils spécialisés et il est clair que seuls des éditeurs structurés spécialement conçus pour la norme pourront tirer parti de toutes ses possiblités. La construction de tels systèmes est une tâche importante et de longue haleine. Bien que des projets allant dans ce sens, comme Hérode ou PODA[38] aient été lancés relativement tôt, toute la puissance de ODA ne sera pas disponible avant quelque temps.

Malgré certaines différences entre ODA et SGML, il peut sembler étrange que les organismes de normalisation proposent simultanément deux normes pour la même fonction, la représentation des documents. A cause de leurs différences, les domaines d'application ne sont pas les mêmes. SGML est déjà utilisé dans le domaine de l'édition et de la documentation technique, alors que ODA s'orientera plutôt vers les applications de bureau. Mais ces deux domaines ne sont pas complètement disjoints et il arrive qu'un document saisi d'abord avec des outils bureautiques soit ensuite destiné à une publication plus large et doive être traité par des typographes. Il faudra donc des passerelles entre les deux normes...

3. Logiciels de préparation de documents

Dans cette section, après un bref rappel historique sur l'évolution des systèmes de préparation de documents, nous présentons deux grandes familles de systèmes qui sont aujourd'hui largement utilisés ou sur le point de l'être. Il existe bien d'autres types de systèmes, mais nous avons retenus ceux qui tirent le meilleur avantage des notions de structure développées au début de cet article. Il s'agit des formateurs et des éditeurs structurés.

---

[38] Voir respectivement : G. Krönert, G. Lauber, J. Lörcher, E. Meynieux, W. Postl, U. Schneider, S. Seisen, and K. Tombre, « Document editing and entry based on the standardised office document architecture », In Directorate General XIII, editor, *ESPRIT'87 Conference*, North-Holland, 1987 et P. J. Robinson, « The Esprit PODA demonstration of ODA at Cebit 87 », In Commission of the European Communities, editor, *Esprit'87, Achievements and Impact, Proceedings of the 4th Annual Esprit Conference, Part 2*, pages 1378–1388, North-Holland, 1987.

3.1 Historique

Les systèmes de production de documents ont démarré timidement dans les années 60, d'abord comme de simples prolongements des outils d'édition de programmes. Ils ont depuis connu un développement important et ils constituent maintenant toute une classe d'outils très variés[39].

Les premiers systèmes étaient des formateurs de bas niveau, qui traitaient un texte saisi à l'aide du même éditeur que celui utilisé pour l'écriture des programmes. Le rôle de ces formateurs était très simple. Il consistait essentiellement à construire des lignes d'égale longueur et à former des pages, à partir d'un texte « au kilomètre ». Quelques commandes pouvaient être glissées dans le texte pour contrôler cette mise en page. Il s'agissait de commandes élémentaires, très proches de celle traitées par l'imprimante (espaces, sauts de lignes, sauts de page...). Les possibilités de formatage étaient limitées par les capacités des imprimantes : peu de caractères disponibles, chasse fixe et positionnement grossier des caractères.

Le langage dans lequel s'exprimaient les commandes était peu formalisé et de bas niveau, comparable à un langage d'assemblage. Comme les langages de programmation, ces langages ont évolué. Ils ont d'abord été dotés de macro-instructions, permettant ainsi d'adapter le formateur et d'offrir à l'utilisateur des commandes d'un niveau plus élevé. Ces commandes se sont également enrichies pour permettre de tirer parti des possibilités de nouvelles imprimantes, voire même de photocomposeuses, comme dans Troff.

Les formateurs ont également été enrichis pour traiter des éléments non-textuels. L'ensemble des outils de formatage disponibles sous Unix[40] constitue un bon exemple de cette évolution. Un ensemble de préprocesseurs traite différents types d'objets dans les documents destinés à Troff : Tbl traite les tableaux, Eqn traite les formules, Pic traite les schémas. D'autres outils encore assistent l'écrivain en prenant en charge les références bibliographiques ou en détectant les fautes d'orthographe ou même de style.

Une autre approche a été développée à la fin des années 70, notamment par B. Reid (on reviendra sur son langage, Scribe, en section 3.2.1), en considérant le langage des formateurs comme un langage de haut niveau qui décrit le document à produire en termes logiques et non plus en fonction de la présentation souhaitée. Le rôle du formateur est devenu plus important, puisqu'il doit, à partir de la description logique d'un document, décider de sa mise en page avant de produire son image sur une imprimante. A la même époque, l'arrivée des imprimantes à laser a également poussé les formateurs vers une plus grande qualité typographique. C'est le cas

---

[39] Pour une bibliographie sur le sujet, voir : J. André, « Bibliographie analytique sur les manipulations de documents », *TSI*, vol.1, n° 5, 1982, 445–455 ; J. C. van Vliet and J. B. Warmer, « An Annoted Bibliography on Document Processing », *Text Processing and Document Manipulation, Proceedings of the International Conference on Electronic Processing*, Cambridge University, 1986, 260–276 ; J. André, « Langages de publication assistée par ordinateur », *Traité d'informatique* art. n° H 2440, Les techniques de l'ingénieur, Paris 1989.

[40] Cette chaîne et ses divers produits sont décrits dans B. W. Kernighan and L. L. Cherry, « A System For Typesetting Mathematics », *Communications of the ACM*, 1975, n° 3 ; B. W. Kernighan and M. E. Lesk and J. F. Ossana, « UNIX Time-Sharing System: Document Preparation », *The Bell System Technical Journal*, vol.57, n° 6, 1978, 2115–2135 ; M. E. Lesk, « TBL A Program to Format Tables », *UNIX Programmer's Manual*, 7th edition volume 2A Bell Telephone Laboratories, 1979 ; D. M. Ritchie, « UNIX Time Sharing System: A retrospective », *The Bell System Technical Journal*, 57, 6, July-August, 1978 ; L. Cherry, « Computer Aids for Writers », *ACM SIGPLAN Notices* 16, 6, 1981, 61-67.

notamment de TEX[41].
Le complément indispensable d'un formateur est un éditeur qui assure la saisie et la mise à jour de la description du document qui doit être soumise au formateur. D'abord cantonnés dans ce rôle, certains éditeurs ont évolué en intégrant des fonctions de formatage, pour devenir des éditeurs-formateurs, qui produisent un document imprimable directement, puisqu'il est déjà mis en page. Mais si ces systèmes n'ont pas supplanté les formateurs, c'est que les fonctions de formatage correspondent à celles qu'on trouvait sur les tout premiers formateurs, c'est à dire des fonctions de bas niveau. Néanmoins, ces éditeurs-formateurs ont connu un succès important, d'abord sur des machines spécialisées de traitement de texte, puis comme logiciel d'application sur les micro-ordinateurs.
La dernière évolution de ce type de système est celle qui utilise des écrans graphiques pour afficher une image aussi conforme que possible à celle qui sera produite sur une imprimante à laser. C'est l'approche WYSIWYG[42].

3.2 Les formateurs

Les derniers systèmes cités sont interactifs : l'utilisateur voit sur l'écran une image du document qu'il traite et chaque modification qu'il apporte au document se reflète immédiatement sur l'écran. Il peut ainsi réagir rapidement s'il n'obtient pas le résultat attendu. Les formateurs, eux, sont des outils non-interactifs. On leur soumet un travail décrit dans un fichier, ils effectuent ce travail et délivrent le résultat global après un certain délai. L'utilisateur n'a aucun moyen d'intervenir sur le déroulement du traitement. Si le résultat obtenu en fin de traitement ne le satisfait pas, il doit modifier, à l'aide d'un éditeur de texte, le fichier décrivant ce qu'il veut obtenir, puis relancer l'exécution de l'ensemble.
Bien que ce mode de travail puisse paraître lourd, cette lourdeur est compensée par une grande puissance de traitement. La plupart des formateurs effectuent des opérations inconnues dans les systèmes de traitement de texte dont, notamment, les numérotations, les constitutions de tables ou d'index, ou la gestion des renvois. Ils se distinguent également par une plus grande qualité de la typographie, particulièrement en ce qui concerne la coupure des mots, les crénages, les ligatures, la construction des lignes et des paragraphes. Enfin ils sont capables de traiter de gros documents, ce qui est souvent difficile avec les systèmes interactifs.
On peut distinguer deux classes parmi les formateurs récents : les formateurs de haut niveau et les formateurs de bas niveau offrant un mécanisme de macro-instructions. Les premiers sont notamment représentés par Scribe. Les seconds regroupent des systèmes comme TEX ou Troff.
Malgré les traitements qu'ils effectuent et la qualités de leur typographie, les formateurs se voient souvent reprocher leur manque d'interactivité. Aussi des outils complémentaires ont-ils été développés pour combler, au moins partiellement, cette lacune. Ce sont les systèmes d'« épreuvage », qui affichent sur un écran graphique une représentation exacte des pages qui seront imprimées. Ce genre d'outil permet de gagner un peu de temps et d'économiser du papier. L'utilisateur peut détecter les erreurs à l'écran mais il ne peut pas les corriger sur l'épreuve qui lui est présentée. Il doit se replonger dans la description du document et retrouver la commande

[41] Prononcez TEK. Ce formateur, qui date d'environ 1978, a d'abord utilisé des imprimantes à points comme les Versatec, mais il s'est surtout répandu depuis qu'il existe des imprimantes à laser. Sur TEX, voir ci-dessous 3.2.2.

[42] Voir section 3.3

ou la partie de texte à changer. Cela limite l'intérêt de l'outil.

Le système Janus[43] va plus loin dans ce sens. Il a été conçu dès le début comme un système intégrant un formateur de haut niveau et des moyens interactifs. Il utilise deux écrans (d'où son nom), l'un, alphanumérique, affichant la description du document dans le langage du formateur, l'autre, graphique, présentant l'image formatée. Les deux écrans affichent simultanément la même partie du document sous ces deux formes. Ainsi l'utilisateur retrouve rapidement la partie de la description du document qui engendre une partie donnée de l'image formatée. Les modifications se font exclusivement sur la description du document et une commande particulière permet de les répercuter sur l'image formatée, au moment choisi par l'utilisateur. Il ne s'agit donc pas d'un vrai système interactif puisque d'une part l'effet des commandes est différé et que d'autre part l'image graphique ne peut pas être utilisée directement pour les modifications. Notons cependant que l'utilisateur peut apporter quelques changements au formatage effectué par le système, en agissant sur l'image formatée, mais c'est la seule possibilité de réelle interaction. On peut néanmoins considérer Janus comme un système de transition entre les formateurs et les éditeurs interactifs de documents structurés qui seront présentés plus loin.

Comme on l'a vu précédemment, on distingue schématiquement deux grandes classes de formateurs : les systèmes de haut niveau et les systèmes de bas niveau. Les premiers travaillent à partir d'une description logique des documents. L'utilisateur n'est pas contraint de spécifier les détails de la présentation qu'il souhaite obtenir. Il indique seulement le type des différentes parties du document (titre, résumé, section,...) et le formateur se charge de la présentation, ce qui permet notamment d'obtenir une grande homogénéité dans l'aspect des documents imprimés, puisque les éléments de même type sont présentés de la même façon.

Les formateurs de bas niveau permettent d'inclure dans la description du document des commandes qui produisent des changements de police ou des espacements, des modifications des marges, des justifications différentes, etc., en fait les mêmes commandes que celles qui sont proposées dans les systèmes de traitement de texte. Mais ces formateurs offrent en plus un moyen de spécifier des commandes de plus haut niveau, analogues à celles de la catégorie précédente. Cela se fait par des macro-commandes qui peuvent être définies dans le fichier contenant le document lui-même ou dans des bibliothèques. Avec les bibliothèques, les macro-commandes sont utilisables dans tous les documents et le formateur se rapproche alors de la catégorie précédente. LaTeX (voir section 3.2.2) est une illustration de cette démarche, puisque son ambition est d'offrir les fonctions de Scribe en s'appuyant sur TeX. De même la bibliothèque -ms de Troff permet de décrire les documents à un plus haut niveau que ne l'autorise Troff seul.

3.2.1 Scribe

Le but du projet Scribe[44] développé par B. Reid à l'université Carnegie-Mellon, était de fournir aux informaticiens un outil simple et relativement spécialisé, pour la

[43] D. D. Chamberlin, J. C. King, D. R. Slutz, S. J. P. Todd and B. W. Wade, « JANUS: An Interactive System for Document Composition », *ACM Sigplan Notices*, vol. 16, n° 6, 1981, 82–91.

[44] Brian Reid, « Scribe : Histoire et évaluation », *Actes des Journées sur la manipulation de documents* (J. André éd.), Rennes, mai 1983, 28–39. Voir aussi : B. K. Reid, « A high-level approach to computer document production », *Proceedings of the 7th Annual ACM Symposium on Principles of Programming Languages, ACM SIGPLAN-SIGACT*, January 1980.

production de documents attrayants et lisibles. Reid cherchait à tirer parti des imprimantes rapides à points tout en demandant un travail minimum à ses utilisateurs. Il voulait surtout leur éviter d'avoir à apprendre la typographie ou un nouveau langage de programmation. Un autre objectif du projet consistait à fournir un mécanisme de communication de documents abstraits entre laboratoires de recherche.

Les principes de Scribe suivent les méthodes de préparation et de production des documents dans la chaîne éditoriale traditionnelle, qui met en jeu plusieurs professionnels, chacun jouant un rôle particulier : l'auteur (ou les auteurs), le comité de lecture, l'éditeur, le directeur d'une collection ou le rédacteur en chef d'un journal, le maquettiste et le typographe. Scribe joue le rôle de plusieurs de ces professionnels. Il prend en charge la forme du document et laisse à l'utilisateur la seule responsabilité du contenu.

L'utilisateur travaille au niveau logique. Il décrit l'organisation du document, le document abstrait, et il exprime ses intentions, mais il ne spécifie pas les détails de la mise en page ou de la typographie. Toutes les informations liées à la présentation des documents sont contenues dans la base de données sur laquelle Scribe s'appuie pour produire l'image imprimée. Ainsi, en changeant d'imprimante, plusieurs résultats différents peuvent être obtenus à partir du même document : une imprimante à rosace n'offre pas les mêmes possibilités typographiques qu'une imprimante à laser. D'ailleurs B. Reid présente son système comme un compilateur acceptant un langage de haut niveau, le langage dans lequel sont décrits les documents abstraits. Scribe peut produire différents documents concrets sur différentes imprimantes, de la même façon qu'un programme en Pascal peut donner lieu à différents codes objets pour s'exécuter sur différentes machines.

Le langage de description des documents est un langage déclaratif, donc non-procédural. L'utilisateur décrit en effet les éléments qui constituent son document, mais il n'indique pas la façon de les traiter, de les imprimer. C'est aussi un langage abstrait, puisqu'il ne se réfère à aucune machine ou imprimante particulière, mais seulement à la structure logique du document.

Ce langage, dont la syntaxe est simple, sert essentiellement à introduire des « marques » dans le texte d'un document. Ces marques peuvent délimiter des régions ou indiquer des commandes adressées au compilateur, mais la syntaxe est unique pour ces différentes fonctions. Les régions définies par des marques de début et de fin sont typées, et déterminent des « environnements ». Un environnement peut porter sur un simple caractère, sur un mot, une expression, une phrase ou sur des éléments plus importants comme un paragraphe ou une page. A titre d'exemple, on peut citer les environnements italique, citation, exemple, énumération, description.

Les environnements utilisables sont définis dans la base de données. Celle-ci contient également la description des différents types de documents disponibles, qui représentent chacun l'environnement global qui peut être utilisé pour un document. La base de données Scribe propose de nombreux types de documents prédéfinis, tels que article, brochure, guide, lettre, manuel, rapport, thèse, etc. D'autres types de documents peuvent être ajoutés, tout comme de nouveaux environnements, mais il s'agit là d'un travail particulier qui n'est pas du ressort des utilisateurs ordinaires du système, et qui requiert une bonne connaissance de la typographie.

Dans la base de données, chaque type de document et chaque environnement est décrit en termes d'attributs, sous la forme d'une suite de paires attribut-valeur (voir figure 13). Scribe connaît une centaine d'attributs qui définissent la police à utiliser, le corps des caractères, les marges, la justification, le mode de construction des lignes, les espacements verticaux, l'interligne, etc. Lors du traitement d'un docu-

a) Définition de l'environnement Itemize dans la base de données :

```
@Define(Itemize, Break, Continue, Fill,
        LeftMargin +5, Indent -5,
        RightMargin -5, BlankLines break,
        Numbered <@@y[B]@, @@y[b]>, NumberLocation lfr,
        Spacing 1, Above 0.5lines, Below 0.5lines,
        Spread 0.5lines, Spaces compact)
```

b) Utilisation de l'environnement dans un document :

```
Cette figure contient trois parties:
@Begin(Itemize)
la definition de l'environnement dans la base de donnees,

l'utilisation de l'environnement dans un document,

le document formate.
@End(Itemize)
```

c) Le document formaté :
Cette figure contient trois parties :

- la définition de l'environnement dans la base de données,
- l'utilisation de l'environnement dans un document,
- le document formaté.

Figure 13: Un exemple, l'environnement *Itemize* de Scribe.

ment, le formatage est déterminé par les valeurs de ces attributs. Ces valeurs sont d'abord définies par le type du document, puis modifiées au cours du traitement en fonction des environnements rencontrés. Chaque environnement ne définit que quelques valeurs d'attribut, les valeurs des autres attributs sont héritées des environnements englobants ou du type du document. En effet, les environnements peuvent être imbriqués les uns dans les autres : on peut par exemple avoir un environnement italique dans un environnement énumération.
Le langage fournit également des commandes qui permettent de modifier certaines valeurs d'attribut soit pour l'ensemble d'un document (les commandes de modification sont alors en tête du document), soit pour une petite partie. Cela permet d'introduire des variations locales par rapport aux définitions de la base de données.
Scribe ne fournit pas seulement un moyen simple et puissant pour obtenir une typographie homogène et de bonne qualité, il effectue aussi certains traitements sur le document. En particulier, il gère les références croisées, les numérotations (chapitres, sections, pages, notes, etc.), il constitue les tables des matières et les index. Il est particulièrement bien adapté au traitement des documents scientifiques, et permet à chaque utilisateur de gérer sa propre base bibliographique. Il fournit de nombreux formats pour la présentation des références bibliographiques, conformes aux standards de la plupart des publications scientifiques.
Comme les compilateurs des langages de programmation, Scribe offre la possibilité de compilations séparées. Ainsi un gros document, ou un document rédigé par plusieurs auteurs, peut être formaté par parties, par exemple chaque chapitre in-

```
\documentstyle[Helvetica]{article}
\begin{document}
\bibliographystyle{plain}
\title{Structures et modeles de documents}
\author{Jacques Andre} \and {Vincent Quint}
\maketitle
\tableofcontents
Ce texte ...

\section{Introduction aux structures de documents}
Tout d'abord, qu'est-ce qu'un document ? Nous...
...
\bibliography{biblio}
\listoffigures
\end{document}
```

Figure 14: Un document décrit dans le langage LaTeX

dépendemment. Même dans ces conditions Scribe prend en charge l'ensemble du document et assure notamment que les références croisées et les numérotations sont cohérentes entre les différentes parties.
Les concepts mis en jeu dans Scribe ont inspiré le développement de nombreux formateurs, mais aussi de quelques systèmes interactifs, comme on le verra plus loin.

3.2.2 LaTeX

TeX[45] est reconnu comme étant l'un des meilleurs formateurs du point de vue de la qualité typographique des documents qu'il produit. Mais il se voit également souvent reprocher la complexité de son langage de description des documents. Ce langage est très riche et il permet de régler très finement les détails de présentation des documents, mais cette richesse le rend en même temps difficile à maitriser par certains utilisateurs, notamment ceux qui n'ont pas de connaissances en typographie ni en programmation. Ce défaut est compensé par un mécanisme de macro-instructions qui permet de définir de nouvelles commandes d'un niveau plus élevé.
C'est cette possibilité qui a été mise à profit par L. Lamport en développant LaTeX[46]. LaTeX peut être vu comme un formateur de haut niveau, très proche de Scribe, mais implanté sur TeX. Il bénéficie ainsi de la puissance et de la simplicité du langage de Scribe et de la qualité typographique de TeX. Comme TeX, il permet de décrire des formules mathématiques et des tableaux. Un environnement « picture » autorise en plus la description de graphiques composés de lignes droites, de flèches, de cercles et de texte. Comme Scribe, il traite les bibliographies, les numérotations,

[45] On consultera sur TeX l'ouvrage de base : Donald Knuth, *The TeXbook*, Addison-Wesley, 1985. En français, on consultera : R. Seroul, *Le petit livre de TeX*, InterEditions, 1989, ainsi que *Les Cahiers Gutenberg.*

[46] L. Lamport, LaTeX: *A Document Preparation System*, Addison-Wesley, 1986. Voir aussi *Les Cahiers Gutenberg* et Francis Borceux, LaTeX, *la perfection dans le traitement de texte*, ARTEL, Bruxelles, 1989.

les tables des matières, les index, les glossaires, etc.
Si LaTeX se distingue fortement de TeX dans la description des parties purement textuelles (voir figure 14), les tables et les figures sont traitées pratiquement comme dans TeX, où ces objets sont déjà décrits d'une façon relativement structurée. Par exemple, la formule (1) s'écrit en TeX :

$$\frac{1}{2n}\int_0^{\sqrt{y}}\left(\sum_{k=1}^{n} sin^2 x_k(t)\right) f(t)dt \qquad (1)$$

```
{1\over 2n}\int_0^{\sqrt y}\bigglp\sum_{k=1}^n
\sin^2x_k(t)\biggrp f(t)\,dt
```

Sur cet exemple, la seule différence notable entre TeX et LaTeX se trouve dans la description de la fraction, qui en LaTeX s'écrit :

```
\frac{1}{2n}
```

3.3 Les systèmes interactifs

Il y a de nombreux types de systèmes interactifs. Depuis les traitements de texte les plus simples jusqu'aux systèmes fortement structurés.
Les plus connus actuellemet sont souvent des systèmes WYSIWYG[47]. Leur approche permet de tirer parti de toutes les possibilités du matériel (nombreux styles et tailles de caractères, graphique intégré) et offre une interface utilisateur évoluée. Mais, dans cette partie, nous ne nous intéressons qu'aux systèmes qui mettent en œuvre un modèle de document présentant au moins un début de structure logique. Ainsi, les logiciels de mise en page comme PageMaker, par exemple, ne sont pas considérés.

3.3.1 Éditeurs-formateurs

Une première catégorie importante de systèmes interactifs, est constituée des éditeurs-formateurs, dont l'emploi se situe surtout dans le domaine de la documentation technique. Bien qu'ils ne couvrent pas la totalité du domaine complexe de la documentation technique, ces systèmes en abordent plusieurs aspects, et notamment l'insertion et le traitement d'images et de schémas dans les documents.

Modèle de document Ce type de système considère un document comme un ensemble ordonné, une liste, de *composants* typés. Les *types* peuvent être choisis et créés par l'utilisateur. Ainsi, pour une lettre, on utilisera les types date, adresse de destination, salutation, paragraphe, et signature. Pour un rapport, on utilisera les types titre, auteur, résumé, titre de section, titre de sous-section et paragraphe. Il n'y a pas de structure très forte : un document est une simple succession de composants, chaque composant ayant un type.

---

[47] *What You See Is What You Get*, cette approche a d'abord été développée au centre de recherche de Xerox, à Palo Alto, sur la machine Alto avec l'éditeur Bravo (B. W. Lampson, « Bravo Manual », *Alto User's Handbook*, (B. W. Lampson and E. A. Taft eds.), Xerox Palo Alto Research Center, November 1978) puis reprise dans des produits comme le Star (E. Harslem and L. E. Nelson, « A Retrospective on the Development of Star », *Proceedings of the 6th International Conference on Sotware Engineering*, Tokyo, september 1982, 377–383) ou le Macintosh (G. Williams, « The Apple Macintosh Computer », *Byte*, February 1984, 38–54).

L'utilisateur a toute liberté de créer selon ses besoins de nouveaux types et d'insérer dans le document, en tout endroit, un composant de n'importe quel type. Les types ne sont pas là pour structurer réellement les documents, mais plutôt pour assurer une certaine homogénéité de leur présentation. En effet, à chaque type est associé un ensemble de *propriétés*, qui sont des attributs de présentation, spécifiant comment les composants de ce type doivent être affichés et imprimés. Il est cependant possible de modifier les propriétés d'un composant particulier, de façon qu'il se présente différemment des autres composants du même type.

Formatage des documents L'essentiel du travail de formatage est effectué automatiquement à partir des propriétés. Celles-ci ne sont pas seulement attachées aux types de composants, mais également à l'ensemble du document et aux pages.
Les propriétés offertes peuvent être classées en trois groupes. Le premier groupe indique comment le texte d'un composant doit être découpé en lignes : espace au-dessus de la première ligne du composant, en dessous de sa dernière ligne, marge de droite et marge de gauche, retrait de la première ligne, interligne, mode d'ajustement des lignes, attributs typographiques des caractères : style, corps, graisse, italique, etc. Un deuxième groupe de propriétés concerne les tabulations, leurs positions et leur type. Un dernier groupe de propriétés des composants concerne leur position dans la page : saut de page imposé avant ou après le composant, saut de page autorisé ou non à l'intérieur du composant, contrôle de veuve et d'orphelin.
Toutes ces propriétés associées à chaque élément sont automatiquement appliquées par l'éditeur au cours de la manipulation du document et l'image affichée correspond en permanence à ces propriétés. On a donc sur l'écran à tout moment l'image fidèle de ce qui sera obtenu sur l'imprimante, y compris le découpage du document en pages.

Edition de graphique Les éditeurs-formateurs comportent en général un éditeur graphique qui permet de manipuler des dessins structurés. Un dessin est en effet constitué d'une collection d'objets de base qui ont des relations entre eux[48].
Les objets de base sont les segments de droite, les rectangles, les ellipses, les polygones, les courbes et les textes. Ces objets ont des *propriétés* qui définissent l'épaisseur de leur trait ou la texture de remplissage. Ils peuvent être groupés pour former des ensembles qui sont alors manipulés d'un bloc. Chaque objet ou groupe d'objets peut subir de nombreuses opérations comme le changement de ses propriétés, de sa taille, de sa position ou de son orientation. Il peut également être copié.
A l'éditeur graphique s'ajoute parfois un éditeur d'images, pour le traitement des photographies numérisées. Cet éditeur permet de retoucher les images avec toute une série de pinceaux différents ou pixel par pixel, grâce à un zoom. Il permet aussi de faire varier le contraste des images ou de les inverser (positif-négatif) en jouant graphiquement sur la fonction de transfert. Enfin, les images peuvent être agrandies, réduites, tournées ou déplacées.
Une différence importante par rapport aux outils couramment offerts sur Macintosh est que les différents éditeurs sont complètement intégrés. On peut éditer un graphique à l'intérieur même d'un document et non pas dans une autre fenêtre et il

[48] Cette approche est comparable à celle de Draw (P. C. Baudelaire, « Draw », *Alto User's Handbook*, (B. W. Lampson and E. A. Taft, eds.), Xerox Palo Alto Research Center, novembre 1978, 97–128) ou plus encore à celle décrite par V. Joloboff, « An Interactive Graphics Editor for Document Preparation », *ACM SigPC Notices*, 1983, vol. 6, n° 6, 45–53.

n'est pas nécessaire de copier le graphique dans un autre environnement lorsqu'on veut le modifier.

Edition du document Pour créer un document, on peut partir de zéro en créant de nouveaux types, avec toutes leurs propriétés, au fur et à mesure des besoins. On crée ensuite des composants conformes à ces types. Mais la méthode la plus adaptée consiste sans doute à créer des documents particuliers, qui sont des squelettes de documents typiques, comportant au moins un composant de chacun des types possibles. On peut avoir ainsi un squelette de lettre, un squelette de rapport, etc. La création d'un document commence alors par la copie du document squelette correspondant au document à créer. Il reste à remplir les composants vides en tapant leur texte, et à créer de nouveaux composants avec les types prédéfinis. La structure du document définie par le squelette n'est qu'indicative : l'utilisateur peut parfaitement créer de nouveaux types, modifier les propriétés d'un type existant, supprimer des composants qui étaient dans le squelette, ou encore changer leur ordre ou leur présentation.
Il y a plusieurs façons de se déplacer dans un document. Une méthode évidente est de faire défiler le document sur l'écran, dans un sens ou dans l'autre : une barre sur le côté du document permet, avec la souris, de contrôler ce défilement. Il est aussi possible de faire des recherches de chaînes de caractères en avant ou en arrière. On peut enfin utiliser les noms de type des composants et ainsi sauter au prochain (ou précédent) composant d'un type donné.
Plusieurs opérations de manipulation du document supposent qu'un composant ou plusieurs soient sélectionnés. Cette sélection se fait grâce à la souris. Une fois que des composants sont sélectionnés, différentes opérations peuvent leur être appliquées :

— Couper : la partie sélectionnée est retirée du document et rangée dans le presse-papiers. Elle pourra ensuite être « collée » ailleurs.
— Copier : la partie sélectionnée est simplement copiée dans le presse-papiers sans être supprimée du document.
— Changer : le composant sélectionné change de type et les propriétés attachées au nouveau type lui sont appliquées.
— Propriétés : certaines (ou toutes) les propriétés du composant peuvent être modifiées et les modifications peuvent s'appliquer à ce seul composant ou à tous les composants de même type.

Le texte des composants est manipulé d'une façon désormais classique dans ce type de système. Pour insérer, on place le point d'insertion avec la souris et tout le texte frappé au clavier vient s'insérer à cet endroit. Au fur et à mesure de la frappe, le texte reste présenté suivant les propriétés du composant où il s'insère. Pendant cette insertion, et quelles que soient les propriétés du composant, on peut modifier les attributs typographiques des caractères entrés (style, corps, graisse, italique). Pour les modifications, la sélection du texte s'opère avec la souris et la partie sélectionnée s'affiche sur fond noir. On peut « couper » ou « copier » la partie sélectionnée, on peut aussi changer ses attributs typographiques.
Les commandes « couper » et « copier » permettent, on l'a vu, de ranger des extraits du document dans le presse-papiers. Le contenu de ce tampon peut ensuite être réinséré dans le texte par une simple commande « coller ».

Conclusion Ce type de système se place dans la lignée des outils d'édition développés initialement sur l'Alto et diffusés ensuite avec le Star. Les représentants de cette catégorie sont par exemple Interleaf, Frame Maker, ou Xerox Documenter[49]. Ils se caractérisent par une interface utilisateur particulièrement soignée, où les menus et les feuilles de propriétés tiennent une grande place bien que les commandes les plus courantes puissent également être entrées au clavier. Cette interface facilite grandement l'apprentissage et l'utilisation du système.
Le modèle de document est simple, donc facile à appréhender par l'utilisateur, qui, grâce aux notions de composant et de propriété, se trouve déchargé d'une large partie de la tâche de mise en page. Les propriétés autorisent une certaine uniformité de la présentation, sans toutefois garantir une homogénéité absolue. L'utilisateur ne se sent pas contraint par un modèle limité ou trop rigide, et peut, à tout moment, tout modifier : le contenu du document, mais aussi les types et les propriétés.

3.4 Éditeurs structurés

3.4.1 Éditeurs syntaxiques

Bien qu'ils soient principalement destinés à la manipulation de programmes, les principes mis en œuvre dans les éditeurs syntaxiques sont d'un grand intérêt pour d'autres types de documents. D'ailleurs, on verra par la suite qu'ils ont largement influencé la conception des systèmes interactifs traitant des documents structurés. A la différence des éditeurs de texte, les éditeurs syntaxiques connaissent le langage dans lequel est écrit le programme qu'ils traitent. Certains de ces éditeurs ne peuvent traiter qu'un langage[50]. D'autres, au contraire, peuvent être paramétrés. Ils acceptent une description formelle de la syntaxe et de la sémantique des langages de programmation et traitent alors des programmes écrits dans les langages spécifiés de cette façon[51].
Les éditeurs syntaxiques utilisent leur connaissance de la syntaxe du langage pour guider l'utilisateur dans la construction d'un programme correct. Ils peuvent bâtir un squelette de programme que l'utilisateur complète ensuite. Au cours de ce travail, ils proposent les instructions valides dans chaque environnement et vérifient que l'ensemble reste conforme à la grammaire du langage. De cette façon, ils réalisent la synthèse d'un programme à partir des éléments fournis progressivement par l'utilisateur. Mais ils sont également souvent capables d'analyse, acceptant des fragments de programme sous leur forme textuelle dont ils extraient la structure syntaxique.

---

[49] On trouvera une comparaison de ces systèmes dans Bernard Girard, « Banc d'essai de cinq chaîne d'édition électronique pour la documentation technique », *Burotica*, 1988, 171–188.

[50] C'est le cas, par exemple, d'Adèle (J. Estublier, S. Krakowiak, J. Mossière and Y. Rouzaud, « Design Principles of the Adèle Programming Environment », *International Computing Symposium on Application Development*, A.C.M. mars 1983) qui ne connait que Pascal ou du Cornell Program Synthesizer (T. Teitelbaum and T. Reps, « The Cornell program synthesizer: a syntax directed programming environment », *Communications of the ACM*, vol. 24, n° 9) au moins dans sa première version destinée à un sous-ensemble de PL/1

[51] Dans cette catégorie on trouve Mentor (V. Donzeau-Gouge, G. Kahn, B. Lang, B. Mélèse and E. Morcos, « Outline of a tool for document manipulation », *IFIP 83*, Elsevier Science Publishers B.V., 1983, 615–620), Pecan (S. P. Reiss, « Graphical Program Development with Pecan Program Development System », *Proceedings of the ACM Sigsoft/Sigplan Software Engineering Symposium on Practical Software Development Environments, ACM Software Engineering Notes*, vol. 9, n° 3, may 1984) ou Aloe (R. Medina-Mora, *Syntax directed editing: toward integrated environments*, CMU-CS-82-113 report, Carnegie-Mellon University, 1982).

En fait les éditeurs syntaxiques ne constituent qu'un des outils proposés dans l'ensemble d'un environnement de programmation ou d'un atelier logiciel. Ils sont donc souvent accompagnés d'autres outils dont certains sont destinés à la production des documents qui accompagnent normalement les programmes, comme les spécifications ou les manuels[52].

Comme pour les formateurs, nous ne présentons que des systèmes interactifs qui prennent en compte l'aspect logique des documents, même s'il ne s'agit pas toujours d'une structure très forte. Alors que les formateurs présentés plus haut sont pour la plupart commercialement disponibles, les systèmes interactifs retenus ici sont, sauf exception, des prototypes de recherche. Cela est dû au développement plus tardif des outils interactifs. Il était en effet plus simple, dans un premier temps, d'appliquer les concepts des documents structurés dans des formateurs. Ce n'est que lorsque les formateurs de ce type ont fait leurs preuves, et que les concepts de structuration logique ont été mieux compris, qu'ont été entrepris les premiers développements de systèmes interactifs structurés.

3.4.2 Quelques systèmes

Faisons abstraction, ici, des systèmes précurseurs[53]. Voici quelques systèmes importants :

— Grif[54] est un système interactif pour la production de documents complexes. Il est essentiellement destiné au traitement des documents structurés, qui peuvent eux-mêmes contenir des objets structurés tels que des tableaux, des formules mathématiques, des schémas ou des programmes. L'approche retenue pour les manipulations des documents et de ces objets est voisine de celle utilisée en génie logiciel pour l'édition des programmes : le document et les objets sont d'abord vus comme des structures construites dynamiquement selon des modèles génériques ; c'est à partir de ces structures que le système engendre l'image affichée sur l'écran. Grif a été conçu comme un système largement paramétrable qui permet à l'utilisateur de définir ses propres types de documents et d'objets, ainsi que leur aspect visuel sur l'écran. Il imprime les documents en utilisant les formateurs existants (TEX, Scribe, Mint, Troff...) mais il peut également effectuer une mise en page à l'écran avec une sortie directe sur imprimante, en PostScript.

---

[52] Ces outils de production de documents peuvent être obtenus directement à partir de l'éditeur syntaxique lui-même, comme c'est le cas dans Mentor (B. Mélèse, « Édition de Documents Multilangages sous Mentor-Rapport », *T.S.I.*, vol. 4, n° 5, 1985, 267–277.). Ils peuvent aussi s'appuyer seulement sur l'environnement de l'atelier et constituer un outil plus indépendant, comme celui proposé par Concerto (J. Paris and L. Delamare, « L'environnement document sur le poste de travail Concerto », *Actes des journées Concerto*, CNET, Février 1986). On trouvera une comparaison entre génie logiciel et manipulation de documents dans : V. Quint, M. Nanard et J. André, « Towards Document Engineering », *Proceedings of the EP90 conference*, R. Furuta ed., Cambridge University Press, 1990 (sous presse).

[53] On en aura une idée en consultant la bibliographie « historique » donnée en section 3.1.

[54] V. Quint and I. Vatton, « Grif: An Interactive System for Structured Document Manipulation », *Text Processing and Document Manipulation, Proceedings of EP86 International Conference*, (J. C. van Vliet ed.), Cambridge University Press, 1986. ; V. Quint, I. Vatton and H. Bedor, « Grif: an Interactive Environment for TEX », *TEX for Scientific Documentation* (J. Désarménien ed.), Lecture notes in computer Science, vol. 236, 145–158 ; V. Quint, I. Vatton et H. Bedor, « Le système Grif », *T.S.I.*, 337–341.

— Tioga est l'éditeur et le système de préparation de documents de l'environnement de programmation Cedar[55]. Développé au centre de recherche de Xerox à Palo Alto, il organise les documents selon une structure arborescente, mais sans contraintes sur les relations entre les composants de l'arbre. Chaque composant porte un nom de type et à chaque nom de type est associé un format qui détermine son aspect graphique. Les formats peuvent être regroupés dans des styles, qui sont partageables entre plusieurs documents, assurant ainsi une certaine cohérence de la présentation de ces documents.

— Quill[56], est développé au centre de recherche de IBM à Almaden. C'est un système WYSIWYG qui affiche une représentation formatée des documents et qui autorise une manipulation directe. Le modèle de document suit le standard SGML ; un document est donc représenté comme une hiérarchie d'éléments typés. Plusieurs éditeurs constituent le système, qui est extensible. Ainsi de nouveaux éditeurs peuvent être ajoutés à ceux qui sont déjà disponibles : éditeurs de texte, de graphique, de tableaux et de formules mathématiques. Les documents sont stockés sous la forme de fichiers SGML.

— Alors que la plupart des systèmes précédents utilisent des stations de travail, Author/Editor[57] est l'un des rares éditeurs structurés disponibles sur Macintosh. C'est avant tout un système conçu pour l'édition de documents SGML. Il accepte une structure logique générique (DTD dans la terminologie SGML) quelconque et peut donc éditer n'importe quel type de document. Il est toutefois limité au texte pur : les tableaux ou les formules ne peuvent pas être affichés sous leur forme naturelle. Pour le texte, l'interface utilisateur est de type WYSIWYG et permet la manipulation directe sur un document formaté à l'écran. Il peut montrer ou cacher les balises SGML et la structure du document peut être vue dans une fenêtre spécifique, à un niveau de détail choisi par l'utilisateur. Une commande `Insert Element` offre, à chaque instant, la possibilité de créer, par l'intermédiaire d'un menu, uniquement les éléments autorisés en fonction de la position courante dans le document.

3.4.3 La structure logique

Tous ces systèmes prennent en compte la structure logique des documents qu'ils manipulent, mais pas tous de la même façon, ni dans le même but, ni au même niveau.

Certains systèmes, comme Lara et Mentor-Rapport, imposent un modèle unique et général, que tout document est censé suivre. Ce modèle définit les types des éléments qui doivent constituer un document et n'est pas extensible par l'utilisateur. Il est probablement mal adapté et trop contraignant dès qu'on veut traiter autre chose que des documents techniques. Cette approche est à l'opposé de celle des éditeurs-formateurs présentés plus haut. où l'utilisateur final a toute latitude pour spécifier la structure du document au moment de sa création. Il peut créer de nouveaux types d'éléments selon ses besoins ou sa fantaisie et il les agence à sa

---

[55] W. Teitelman, « A tour through Cedar », *IEEE Transactions on Software Engineering*, vol. 11, n° 3, march 1985, 285–302.

[56] D. D. Chamberlin, H. F. Hasselmeier and D.P. Paris, « Defining Document Styles for WYSIWYG Processing », *Document Manipulation and Typography* (J.C. van Vliet, ed.), Cambridge University Press, 1988, 121–137

[57] M. Maloney, Y. Rubinski, P. Sharpe, B. Shiff and R. Spencer, *Author/Editor Advance version User's Guide*, SoftQuad Inc., Toronto, Canada, 1988.

guise. Il n'y a pas de structure générique. Ce modèle définit une syntaxe mais n'y associe aucune sémantique, ce qui limite les traitements applicables aux documents. Pratiquement, seule la présentation peut se faire à partir de cette structure, puisque l'utilisateur fournit les règles de formatage de chacun des types qu'il crée.
La majorité des systèmes se fondent sur la même notion de document : une structure spécifique construite selon le modèle d'une structure logique générique. Les langages de définition de structure présentent des différences d'ordre syntaxique, comme on peut le voir sur les exemples de la figure 15. Ces exemples montrent comment, dans différents systèmes, on décrit la structure générique d'une même classe de documents appelée « Article ». Alors que la plupart des systèmes utilisent une syntaxe de style BNF, Grif propose un langage un peu plus riche, où il est notamment possible de spécifier les cardinalités minimum et maximum des listes : `LIST [2..*] OF (Section)` représente une séquence d'au moins deux sections, l'étoile indiquant qu'il n'y a pas de limite supérieure au nombre de sections. Il est a signaler qu'un nombre croissant de systèmes, comme Author/Editor, utilisent maintenant le langage SGML pour décrire les structures logiques génériques.
Bien que la plupart des systèmes utilisent le même modèle, la finesse de la structure qu'ils considèrent varie. Pour certains, le paragraphe constitue un atome, alors que d'autres voient différents éléments dans les paragraphes[58]. Ces systèmes permettent de définir des structures très fines, mais qui risquent d'être lourdes à manipuler. C'est pourquoi certains[59] offrent une alternative à ce raffinement de la structure avec les attributs, qui permettent d'ajouter de la sémantique à certaines parties d'éléments de la structure et qui sont plus faciles à manipuler.
Ainsi, pour une certaine classe de documents, on définit les attributs « Langues » (qui prend les valeurs « Anglais » et « Français ») et « Type de mot » (qui prend, par exemple, les valeurs « mot-clé », « concept » et « identificateur »). L'utilisateur peut alors mettre l'attribut « Langué » sur certaines parties d'un document de cette classe et l'attribut « Type de mot » sur certaines chaînes de caractères. Ces attributs sont utilisés lors de l'affichage ou de l'impression pour sélectionner la police de caractères ou pour appliquer le bon algorithme ou dictionnaire de coupure de mots. Ils peuvent aussi servir à d'autres applications comme, par exemple, un système documentaire.
Une autre notion, peu courante dans ces systèmes, est la référence, qui permet d'établir des relations non hiérarchiques entre les différentes parties d'un document. Les références sont notamment utilisées pour représenter les renvois à toutes sortes d'objets, tels que notes de bas de page, citations bibliographiques, figures, formules, chapitres ou sections.
En revanche les structures arborescentes se retrouvent dans pratiquement tous les systèmes. Ces structures sont décrites au niveau générique par trois constructeurs :

— l'agrégat qui rassemble des éléments de type déterminé (il est noté par une simple juxtaposition de symboles dans Speed et pedtnt, par les mots-clés `BEGIN` et `END` dans Grif),
— la liste qui est une séquence d'éléments tous de même type et en nombre variable, (elle est noté par le signe * dans Speed, les signes * ou + dans

[58] Par exemple les citations, les parenthèses, etc. Voir Hamlet, « A Disciplined Text Environment », *Text Processing and Document Manipulation*, van Vliet ed., Cambridge University Press, 1986, p. 78–89 ; ou encore, on vient de le voir sur les exemples, Speed et pedtnt.

[59] Outre Grif, on peut citer le travail de D. D. Cowan and G. De V. Smit, « Combining Interactive Document Editing with Batch Document Formatting », *Text Processing and Document Manipulation*, van Vliet ed., Cambridge University Press, 1986, p. 140–153,

La syntaxe utilisée par Speed :

```
Article      = Entete Corps .
Entete       = Titre Auteurs .
Titre        = Char * .
Auteurs      = Char * .
Corps        = Section * .
Section      = TitreSection Paragraphe * Section * .
TitreSection = String Char * .
Paragraphe   = Entite * .
Entite       = Texte | EntiteTable | Liste .
Texte        = Char * .
...
```

La syntaxe utilisée par pedtnt :

```
<article>      = titre auteurs <section> +
<section>      = section-id titre-section
                 <paragraphe> + <section> +
<paragraphe>   = [ text-block | <entite-table> | <liste> ] +
<entite-table> = tbl-block legende
<liste>        = <item> +
<item>         = item-marker <entite> +
```

La syntaxe utilisée par Grif :

```
Article    = BEGIN
             Titre = Text;
             Auteurs = Text;
             Corps =  Sections;
             END;
Sections   = LIST [2..*] OF (Section =
                 BEGIN TitreSection = Text;
                       Paragraphes = LIST OF (Paragraphe);
                       Sections;
                  END );
Paragraphe = LIST OF (Entite = CASE OF
                 Texte = Text;
                 EntiteTable = BEGIN
                                 Table; Legende = Text;
                               END;
                  Liste = LIST [2..*] OF (Item = Paragraphes);
                  END );
```

Figure 15: Différentes spécifications d'une structure générique.

pedtnt, par le mot-clé `LIST` dans Grif),

— le choix qui permet de choisir le type d'un élément parmi un ensemble limité de type possibles (il est noté par le signe | dans Speed et pedtnt, par le mot-clé `CASE OF` dans Grif).

Pedtnt se distingue en traitant en plus les structures tabulaires dans son éditeur spécialisé pour les tableaux.
L'usage qui est fait de la structure dans tous ces systèmes varie beaucoup. Pour certains la structure permet seulement d'associer des règles de présentation aux différents éléments du document. Pour d'autres, elle permet en plus de guider l'utilisateur dans la construction d'un document conforme à un modèle. Quelques systèmes effectuent de plus des numérotations automatiques à partir de la structure. Enfin quelques systèmes ajoutent des traitements plus élaborés, notamment des renvois et des index.

3.4.4 L'image des documents

L'image présentée sur l'écran pendant l'édition, comme celle qui est imprimée en fin de traitement, est construite de façon automatique d'après la structure logique du document. Cette construction s'appuie généralement sur des règles de présentation qui spécifient la façon dont chaque type d'élément dans la structure du document doit être affiché ou imprimé. C'est un principe qui ne souffre pas d'exception dans le domaine des éditeurs de documents structurés.
La façon dont les règles de présentation sont spécifiées est différente d'un système à l'autre. Certains, surtout les moins structurés, travaillent uniquement en mode interactif, avec des feuilles de propriétés, qui sont des formulaires interactifs où apparaissent tous les paramètres accessibles à l'utilisateur. Celui-ci peut, par ce moyen, définir de nouvelles propriétés ou modifier des propriétés existantes.
Speed procède différemment pour la spécification de la présentation des documents. L'ensemble des règles de présentation est défini dans une « table d'attributs » qui contient, pour chaque type d'élément défini dans la structure générique de la classe, un ensemble d'attributs caractéristiques du formatage de ce type d'élément. La figure 16 contient un extrait d'une table d'attributs possible pour les documents de la classe « Articlé » définie plus haut.
Le point d'interrogation indique des attributs dont la valeur peut être changée par l'utilisateur, la valeur qui suit le point d'interrogation étant une valeur par défaut. On peut remarquer que tous les attributs ne sont pas définis pour chaque type d'élément. Cela est dû au mécanisme d'héritage qui affecte à un attribut la valeur de ce même attribut dans l'élément de niveau supérieur si l'attribut n'est pas défini explicitement pour ce type d'élément.
Grâce aux tables d'attributs, l'éditeur peut construire automatiquement l'image du document. Mais l'utilisateur a également la possibilité de modifier tous les attributs qui sont modifiables, et il peut le faire soit dans le document lui-même, soit dans un fichier séparé.
Dans Grif la présentation d'une classe de documents est, comme dans Speed, spécifiée avant la création d'un document, par des « schémas de présentation », qui correspondent aux tables d'attributs. Ces schémas sont décrits en termes abstraits qui rendent la description indépendante de l'appareil d'affichage ou d'impression. Ainsi le corps (`Size`) est défini par un niveau relatif (1 = tout petit, 2 = petit, 3 = moyen...) et les dimensions et distances sont exprimées en unités liées à la taille

```
Article      ~: FontFamily = ? (Times),
                MainTextFontShape = ? (Roman),
                MainTextFontSize = ? (10),
                MainTextLineSpacing = ? (13),
                MainHeadingFontShape = ? (Bold),
                ...
                LeftMargin = ? (100),
                RightMargin = LeftMargin + TextAreaWidth,
                Language = French,
                PAGINATE (AbsoluteTopMargin, TextAreaHeight);

Entete       : LeftMargin = ? (LeftMargin + 36),
                FormattingMode = LeftAdjusted,
                MaxVertSpace = ? (182-26);

Titre        : FontShape = ? (Italic),
                FontSize = ? (14),
                LineSpacing = ? (17),
                TopSpace = ? (4 * LineSpacing),
                ...
Corps        : FontShape = MainTextFontShape,
                FontSize = MainTextFontSize,
                LineSpacing = MainTextLineSpacing,
                FormattingMode = Justified;

Section      : SectionNumber = COUNTIN(Article);

TitreSection : FontShape = HeadingFontShape,
                BuiltString = SectionNumber & '.',
                ...
Paragraphe   : ParagraphNumber = COUNTIN(Section),
                Indentation = IF ParagraphNumber=0 THEN 0
                              ELSE em(FontSize),
                JUSTIFY(LeftMargin, RightMargin, Indentation,
                        FormattingMode, LineSpacing, Language);
```

Figure 16: Extrait d'une table d'attributs.

des caractères (1 = hauteur des caractères). Un schéma de présentation possible (ou plutôt un extrait) pour les « Articles » est donné dans la figure 17.

Dans la plupart des systèmes, le concept de boîte est largement utilisé pour la construction des images affichées ou imprimées, mais seuls certains le rendent directement accessible dans les règles de présentation, qui permettent de spécifier les positions relatives des différents éléments qui composent un document. Dans beaucoup de systèmes, les règles de présentation sont exprimées en termes d'attributs typographiques conventionnels, comme dans Speed : corps, style des caractères, marges, retraits, centrages, etc.

Dans Grif, une boîte est associée à chaque type d'élément défini dans la structure générique et les règles de présentation indiquent des relations entre les dimensions et les positions des boîtes des différents types d'éléments. Ces relations sont respectées pendant l'édition des documents. Les boîtes sont utilisées d'une façon systématique : elles décrivent aussi bien les éléments de structure que les éléments de présentation, tels que les numéros (voir `NumSection` dans l'exemple de la figure 17).

Pour certains systèmes, le principe WYSIWYG est appliqué jusqu'au bout, et chaque document n'a qu'une image, la même sur l'écran d'édition et sur le papier. D'autres systèmes considèrent que plusieurs images sont possibles pour le même document, et que le changement de présentation doit se faire simplement en changeant les règles de présentation appliquées au document et non pas en modifiant le document lui-même.

La question des pages est traitée d'une façon variée dans tous ces systèmes. Certains affichent des pages à l'écran ; ce sont ceux qui lient étroitement l'édition et le formatage. D'autres, au contraire donnent une image assez lointaine de ce qui sera imprimé. Grif combine les deux approches, en ignorant les pages lorsque le découpage en pages pose plus de problèmes qu'il n'en résoud, mais en les prenant en compte lorsque l'utilisateur veut obtenir sur imprimante une image fidèle de ce qu'il voit sur l'écran.

De nombreux systèmes proposent un mécanisme de vues qui permet de montrer le même document selon des aspects différents. Certains systèmes prédéfinissent une vue montrant le squelette du document. D'autres, comme Grif, offrent un mécanisme général permettant de définir, dans les règles de présentation, l'aspect et le contenu d'un nombre quelconque de vues. Celles-ci peuvent servir à montrer, par exemple, la table des matières en même temps que le texte intégral, ou la partie anglaise d'un document bilingue, ou encore uniquement les notes qui se trouvent dispersées dans un document.

Bien que les vues constituent une façon de visualiser la structure des documents, d'autres moyens sont proposés dans la plupart des systèmes pour montrer la structure logique. Certains affichent le chemin dans l'arbre depuis la racine jusqu'à l'élément courant, d'autres écrivent systématiquement le type des éléments dans une zone spécifique, en regard de l'image du document ou dans l'image même du document. Plusieurs systèmes permettent d'ajouter des éléments complémentaires dans la présentation des documents, par exemple un mot comme « Résumé » en tête du résumé, ou un caractère (étoile, tiret, boulet...) au début de chaque élément d'une énumération, ou encore un filet pour séparer deux éléments. On notera que Speed et pedtnt spécifient ces éléments déjà dans la structure générique (par exemple `MarqueItem` dans Speed et `item-marker` dans pedtnt, comme on le voit sur la figure 15), alors que Grif les traite uniquement au niveau de la présentation.

```
COUNTER          CptSection : RANK OF Section;

DEFAULT          BEGIN
                 Size : Enclosing = ;
                 Width : Enclosing . Width;
                 HorizPos : Left = Enclosing . Left;
                 Height : Enclosed . Height;
                 VertPos : Top = Previous . Bottom;
                 END;

Article        : BEGIN
                 Size : 2;
                 Width : Enclosing . Width - 6;
                 HorizPos : Left = Enclosing . Left + 3;
                 VertPos : Top = Enclosing . Top + 2;
                 END;

                 ...
Corps          : VertPos : Top = Entete . Bottom + 2.5;

NumSection     : Content: (Value(CptSection, Arabic)
                          Text '. ');

TitreSection   : BEGIN
                 Size : Enclosing +1;
                 VertPos : Top = Enclosing . Top;
                 Create (NumSection);
                 Line (Left);
                 END;

Paragraphe     : BEGIN
                 IF First THEN Indent : 0;
                 IF NOT First THEN Indent : 1;
                 VertPos : Top = Previous . Bottom + .5;
                 Line (Left);
                 END;
```

Figure 17: Les règles de présentation de Grif.

3.4.5 Les objets inclus

Pour certains systèmes, le seul contenu possible des documents est le texte. Andra autorise l'insertion d'images dans les documents, mais celles-ci ne sont pas traitées par l'éditeur, elles sont seulement insérées au moment de l'impression du document. La plupart des autres systèmes autorisent l'inclusion d'éléments de différentes natures et traitent ces objets avec des outils spécifiques qui font partie de l'éditeur. Grif et W traitent les formules mathématiques de la même façon que le reste de la structure du document, mais seul Grif manipule de façon homogène toutes sortes d'objets, définissables par les utilisateurs, avec les mêmes outils que pour les documents eux-mêmes : structures génériques et règles de présentation. Ceci est dû en partie au modèle basé sur les boîtes, retenu pour la présentation. La notion de boîte permet en effet de décrire aussi bien du texte mis en lignes que des formules mathématiques ou des tableaux.
Ainsi, dans les exemples de structures génériques de la figure 15, les mots `Table` dans Speed et `tbl-block` dans pedtnt désignent des objets connus du système, et qui sont traités par un éditeur spécifique. Au contraire, dans Grif, le mot `Table` désigne une autre structure générique définie de la même façon que celle des Articles. Ceci permet d'adapter l'éditeur à différents types d'objets, de la même façon qu'on l'adapte à différents types de documents.

3.4.6 Le mode d'interaction

L'interface utilisateur reflète d'assez près la source d'inspiration des concepteurs de ces systèmes. Ceux qui subissent le plus directement l'influence des environnements de programmation conservent un dialogue utilisateur semblable à celui des programmeurs, avec des commandes explicites de traitement d'arbres. Ce sont d'ailleurs les mêmes qui utilisent un éditeur de texte préexistant[60], pour la manipulation des atomes de la structure. Les autres sont plus orientés vers des interfaces à base de menus et souris et intègrent plus étroitement le traitement de texte.
Le mode d'interaction varie également selon le type d'image affichée et le type de structuration des documents.

— On a vu qu'avec les éditeurs-formateurs, l'utilisateur peut créer en n'importe quel point du document un élément de n'importe quel type. Il peut aussi changer le type ou les attributs de présentation de n'importe quel élément.
— Au contraire, dans les systèmes où les documents sont définis par une structure générique, la liberté de l'utilisateur est plus réduite : il ne peut créer, modifier ou supprimer des éléments que conformément au modèle de la structure générique. Il est en général guidé par le système, qui suit ce modèle. Il est ainsi sûr de construire un document conforme au modèle.

Plusieurs techniques sont mises en œuvre pour conduire l'utilisateur dans la construction d'un document correct.

— Le prototype pedtnt engendre automatiquement des « boutons » pour tous les éléments qui peuvent être créés d'après la structure générique. Ces boutons apparaissent sous la forme du nom de l'élément qui peut être créé, affiché en inversion vidéo. Il suffit de déplacer le curseur sur ces boutons et d'activer

[60] Par exemple Emacs : R. M. Stallman, « EMACS, The Extensible, Customizable, Self-Documenting Display Editor », *ACM SIGPLAN Notices*, vol. 16(6), 1981, p. 147–156.

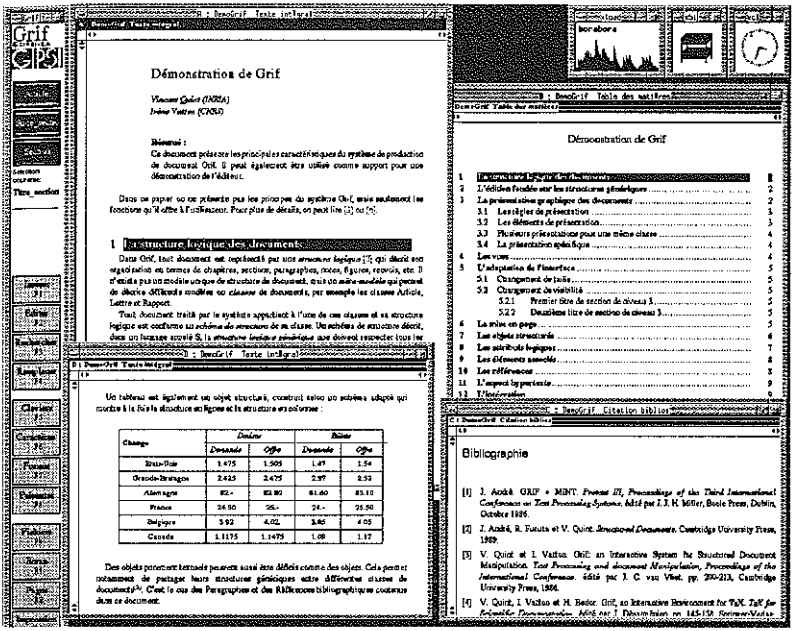


Figure 18: L'écran de Grif.

une commande pour créer une occurence de l'élément, et du même coup les boutons représentant les voisins potentiels.

— Grif au contraire n'introduit pas d'information supplémentaire dans l'image des documents (voir figure 18). Il affiche dans une fenêtre spécifique (en haut de l'écran sur la figure) le type des éléments voisins de la partie sélectionnée (celle-ci est déterminée à l'aide de la souris et est affichée en inversion vidéo) et le type de ceux qui peuvent être créés. Il suffit de sélectionner l'une de ces cases grises avec la souris pour créer un élément du type voulu. La commande `INSERER` permet également de créer de nouveaux éléments : elle propose un menu des éléments qui peuvent être créés après (ou avant) la partie sélectionnée, selon la structure générique. Chaque fois qu'un nouvel élément est créé, les éléments qu'il doit contenir sont également créés, vides, et affichés sous la forme de rectangles gris ayant les dimensions et la place définies par les règles de présentation. Si ces rectangles gris ne sont pas suffisamment parlants, l'utilisateur les désigne avec la souris et leur type s'affiche en haut de l'écran.

Le degré d'interactivité est assez variable d'un système à l'autre. Certains s'efforcent de réfléchir immédiatement sur l'écran toute action de l'utilisateur, y compris la frappe d'un seul caractère de texte, d'autres ne reformatent l'image du document que dans certaines circonstances, telles qu'une commande explicite de l'utilisateur ou la fin de la saisie d'un élément. Grif utilise les deux techniques à la fois, en effectuant un réaffichage au caractère près dans la vue choisie par l'utilisateur, et un réaffichage différé dans les autres vues.

Tous les systèmes ne donnent pas la même importance à l'interface utilisateur. C'est un point fort pour Etude où les aides en ligne et l'annulation de la dernière com-

mande, par exemple, sont disponibles à tout moment, quel que soit le contexte[61].

3.4.7 Les autres caractéristiques

Certains systèmes prévoient l'adjonction à un éditeur structuré des outils d'aide à l'écriture déjà disponibles pour certains formateurs : détection de fautes d'orthographe, contrôle du style, dictionnaires de synonymes, etc.
Le mode d'impression des documents change d'un système à l'autre. Ceux qui attachent la plus grande attention au formatage en se conformant de près à l'approche WYSIWYG prennent eux-même en charge l'impression des documents. Cette conception conduit à une architecture relativement complexe pour Speed, où, lors de l'édition du document, le formatage se fait par parties et en parallèle. Dans d'autres systèmes, certaines concessions sont faites sur la qualité du formatage : les algorithmes utilisés sont le résultat d'un compromis entre l'efficacité (le formatage se fait en temps réel à l'écran) et la qualité du résultat obtenu. Mais la plupart des systèmes utilisent les formateurs disponibles : Troff, TEX, Scribe ou LATEX.

3.5 Conclusion

Que ce soit dans les systèmes interactifs ou dans les formateurs, la structuration des documents offre à l'utilisateur une grande puissance d'expression. Il peut se concentrer sur le contenu et l'organisation de son document, sans se préoccuper de la mise en forme. Mais une différence importante entre ces deux catégories de systèmes, à part le mode d'utilisation, réside dans le fait que les systèmes interactifs peuvent guider l'utilisateur et lui proposer les structures possibles dans l'environnement où il se trouve, en garantissant que le document produit sera correct, au moins vis à vis de sa structure générique. C'est une des raisons qui poussent au développement des systèmes interactifs.
Il faut aussi souligner que la structuration des documents, dans tous les types de systèmes, offre d'autres avantages encore, qui ne concernent pas uniquement la création et l'édition. A cause du haut degré d'abstraction du modèle utilisé, plusieurs sortes de traitements peuvent être appliqués aux documents structurés. Les informations nécessaires à un système documentaire peuvent être extraites automatiquement, sans passer par une phase d'indexation manuelle. D'une manière générale, les bases de données peuvent travailler efficacement sur de tels documents.
Une autre catégorie d'applications concerne le formatage ou la photocomposition. Grâce aux systèmes de production de documents structurés, il devient possible de faire passer un même document dans différents formateurs, sans intervention manuelle[62]. Ceci est une première étape vers le traitement des documents électroniques dans toute la chaîne éditoriale.
Cette évolution sera favorisée par la convergence des systèmes interactifs et des formateurs. Déjà de nombreux prototypes d'éditeurs interactifs de documents struc-

[61] Pléiade adopte également ce point de vue, bien que d'une manière moins systématique.

[62] Voir par exemple V. Quint, I. Vatton and H. Bedor, « Grif: an Interactive Environment for TEX », *TEX for Scientific Documentation*, (J. Désarménien ed.), Lecture notes in computer Science, vol. 236, Springer-Verlag, 1986,145–158 ; J. André, « GRIF + MINT, or how to abide by a layout », *Protext III, Proceedings of the Third International Conference on Text Processing Systems*, J. J. H. Miller ed., Dublin, Boole Press, 1986.

turés produisent des sorties compatibles avec les meilleurs formateurs, mais on commence aussi à voir des environnements interactifs de plus en plus complets bâtis autour des formateurs, et notamment de TEX.

4. Conclusion

La notion de structure est donc fondamentale pour les documents.
Comme on le verra dans le reste de ce cours, la notion d'hypertexte vient un peu en contre-point à celle de structure. La seule rigueur des structures ne suffit pas ; nous l'avons montré. La liberté apparente des hypertextes n'est pas non plus viable, du moins tant que ne seront pas résolus les problèmes de navigation par exemple. La solution est sûrement dans un équilibre des deux.