

# le cobol structuré : un modèle de programmation

IST

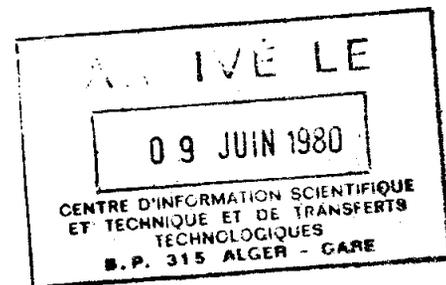
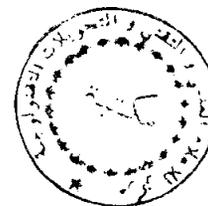
1017

par  
**Marie-Thérèse BERTINI**  
et  
**Yves TALLINEAU**



Editions d'Informatique

# le cobol structuré : un modèle de programmation





*« Pour atteindre le réel, il faut d'abord écarter le vécu »*

Claude Levy Strauss.

---

# table des matières

<b>Présentation du Cobol structuré</b>	<b>7</b>
<b>1. L'arbre programmatique</b>	<b>9</b>
1.1. PRESENTATION	10
1.2. COMPOSITION DE L'ARBRE	11
1.3. LA STRUCTURE REPETITIVE	12
1.4. LA STRUCTURE ALTERNATIVE	12
1.5. REPRESENTATION COMPLETE	12
1.6. LES NIVEAUX	16
1.7. STRUCTURES COMPLEXES	17
1.7.1. STRUCTURES CONSECUTIVES	17
1.7.2. STRUCTURES ENCHASSEES	21
1.8. REPRESENTATION DES ARBRES PROGRAMMATIQUES	22
1.8.1. REGLES GENERALES	22
1.8.2. PRESENTATION DE L'ARBRE	23
1.8.3. CAS PARTICULIER	24
1.9. CONCEPTION D'UN ARBRE PROGRAMMATIQUE	28
<b>2. L'écriture des programmes</b>	<b>33</b>
2.1. PRESENTATION	34
2.2. PROGRAMME A UNE REPETITIVE	34

---

2.3. PROGRAMME A UNE ALTERNATIVE	36
2.4. STRUCTURES COMPLEXES	38
2.4.1. STRUCTURES CONSECUTIVES	38
2.4.2. STRUCTURES ENCHASSEES	39
2.4.3. CAS REELS	41
2.5. REGLES D'ECRITURE	41
2.5.1. SEQUENCE DES PARAGRAPHES DE TRAITEMENT	41
2.5.2. SEQUENCE DES OPERATEURS	42
2.6. OPTIMISATION DE L'ECRITURE	46
2.7. ECRITURE MODULAIRE	51

### **3. Étude de cas** **57**

3.1. FCS001	58
3.1.1. BUT	58
3.1.2. PRESENTATION	58
3.1.3. A.P.	59
3.1.4. PROGRAMME COMPLET	60
3.1.5. PROGRAMME OPTIMISE	63
3.1.6. JEU D'ESSAI ET RESULTATS	65
3.1.7. REMARQUES	66
3.2. FCS002	68
3.2.1. BUT	68
3.2.2. PRESENTATION	68
3.2.3. A.P.	68
3.2.4. PROGRAMME	69
3.2.5. JEU D'ESSAI ET RESULTATS	71
3.2.6. REMARQUES	71

---

3.3. FCS003	72
3.3.1. BUT	72
3.3.2. PRESENTATION	72
3.3.3. A.P.	72
3.3.4. PROGRAMME	74
3.3.5. JEU D'ESSAI ET RESULTATS	79
3.3.6. REMARQUES	80
3.4. FCS004	82
3.4.1. BUT	82
3.4.2. PRESENTATION	82
3.4.3. A.P.	83
3.4.4. PROGRAMME	84
3.4.5. JEU D'ESSAI ET RESULTATS	87
3.4.6. REMARQUES	88
3.5. TRI	92
3.5.1. BUT	
3.5.2. PRESENTATION	
3.5.3. A.P.	
3.5.4. PROGRAMME	94
3.5.5. JEU D'ESSAI ET RESULTATS	98
3.5.6. REMARQUES	99
3.6. LES 8 REINES	102
3.6.1. BUT	102
3.6.2. PRESENTATION	102
3.6.3. ANALYSE	102
3.6.4. A.P.	105
3.6.5. PROGRAMME	110
3.6.6. RESULTATS	115

# présentation du cobol structuré

La programmation passe par :

- l'analyse d'un problème
- la conception précise d'un algorithme
- l'écriture dans un langage de programmation de cet algorithme.

L'écriture du programme est finalement à la discrétion du cheminement logique de l'homme. La pratique la plus courante consiste à élaborer le programme lentement aussitôt que le but à atteindre est à peu près défini. Cette approche conduit le programmeur à mettre en œuvre, « en temps réel », tout son savoir-faire. Elle correspond à une avance lente et pensée de l'esprit du programmeur dans un processus de synthèse constructive. Il rassemble les pièces, essaie, remplace, etc. en vue de constituer le puzzle. Cette démarche, de type heuristique, entraîne à une débauche d'efforts parce qu'il faut créer de toutes pièces une fonction en partant de détails ; cet effort de synthèse est très ardu parce que sans cesse interrompu par l'intervention des contextes. Le but atteint, il est d'ailleurs facile de voir que le programmeur a dû faire de fréquents retours en arrière, des remplacements, des ratures, des rajouts, etc. Ce type de programmation correspond à une approche dite « BOTTOM-UP PROGRAMMING » par les anglo-saxons.

Une autre pratique, pour construire le puzzle, consiste à analyser l'image que l'on veut obtenir, à déterminer les parties de cette image, repérer les sous-ensembles d'objets, en bref à découper l'image jusqu'à domination parfaite de sa structuration ; ceci étant fait, on cherchera les cubes que l'on ajustera aux bons endroits dans la structure, au premier coup d'œil. Cette méthode revient à décomposer la fonction globale en sous-fonctions hiérarchisées par le processus d'analyse et ce, en descendant niveau par niveau. Elle se fonde sur une mise en évidence de la structure du problème. C'est la programmation analytique ou « TOP-DOWN PROGRAMMING ». A l'origine de ces idées Dijkstra et son école ont lancé le concept de programmation structurée. En France, Warnier a abordé le problème de la structuration par le biais de l'analyse.

Dans ce livre, nous donnons les bases d'un modèle de programmation structurée. Cette structuration est en fait réalisée à partir d'un arbre programmatique. Ce graphe est un outil mieux adapté à la représentation des algorithmes que l'organigramme parce qu'il en donne une vision spatio-temporelle. La structure donnée par l'arbre programmatique se retrouvera photographiée dans le programme.

Ce traité comprend deux parties :

- la première concerne les règles et manières de construire l'arbre programmatique,
- la seconde donne les règles d'écriture, en Cobol, du programme à partir de l'arbre programmatique.

Le résultat est un programme très structuré donc lisible, clair et facilement extensible.

Si ce traité a vu le jour à la suite d'études sur la structuration logique appliquée au Cobol, il veut aussi être un modèle, un « Pattern » de structuration qui peut convenir avec plus ou moins d'adéquation à tout langage de programmation.