II. PRINCIPES GENERAUX

1 - LE GRAFCET : OUTIL DE MODELISATION

Au chapitre I, le Grafcet a été abordé sous 2 angles différents. Il a d'abord été le résultat d'une observation. Il a ensuite été présenté comme langage de programmation des automates. Cette distinction est de première importance car l'analyse d'un problème ne doit en aucun cas reposer sur un langage de programmation : chacun de ces outils doit être utilisé à bon escient, dans son propre contexte.

Pour éviter toute confusion, on nommera *Grafcet* l'outil de modélisation d'une manière générale (sa sémantique, ses règles d'évolution, sa syntaxe) et *grafcet* un modèle pour un système particulier (un diagramme spécifique). *SFC* désignera le langage de programmation inspiré du Grafcet.

1 - 1. L'essentiel sur le Grafcet

a) La sémantique

Ce paragraphe définit le vocabulaire lié au Grafcet ainsi que sa représentation graphique. La figure A-13 illustre les différents concepts d'une manière synthétique.

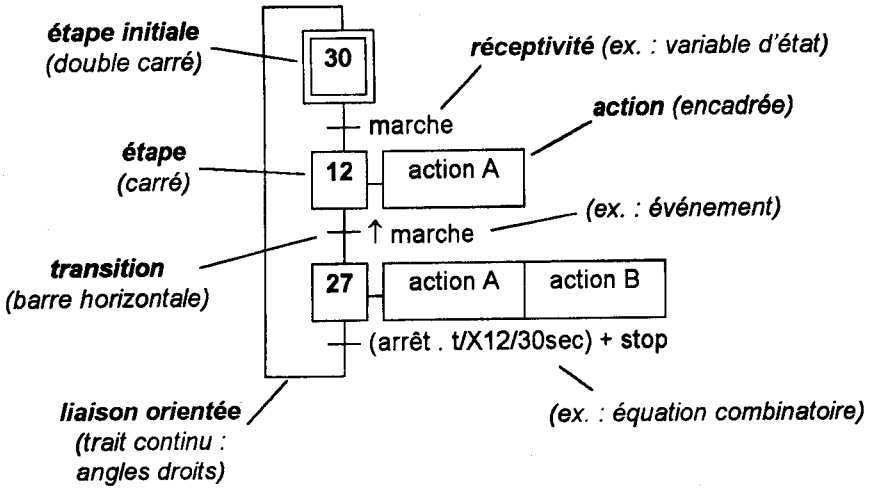


figure A-13

- ***Etape : active / inactive***

• Une étape représente un état particulier du système à un moment donné de son cycle de fonctionnement. L'étape qui correspond à l'état du système à un instant *t*, est dite active.

• Un état donné peut être déterminé par plusieurs étapes qui sont actives en même temps

• Afin de faciliter le repérage, chaque étape porte son propre numéro. Lorsqu'on désigne une étape, on fait précéder ce numéro de la lettre X majuscule.

• L'ordre des numéros est quelconque et ne traduit en aucune façon le déroulement du cycle (c'est le rôle des liaisons orientées).

• La variable Xi est = 1 lorsque l'étape i est active. Xi = 0 lorsqu'elle est inactive.

• L'étape initiale est l'étape qui est active au moment de la mise en marche du *système* (à ne pas confondre avec la mise en marche du *cycle*). Un grafcet peut comporter plusieurs étapes initiales.

- ***Situation***

• L'ensemble des étapes actives définit la situation du grafcet. La situation est représentée par un vecteur de la forme {i, j...} où i, j... sont les étapes actives.

• La situation est dite *vide* lorsqu'aucune étape n'est active. Elle est dite *initiale* lorsque seule l'étape initiale (ou les étapes initiales) est (sont) active(s). La situation *courante* est définie par l'ensemble des étapes actives à l'instant considéré.

- ***Action : produite / non produite***

• A chaque étape on peut associer une ou plusieurs actions.

• Il se peut également qu'aucune action ne soit associée à une étape.

• Une action donnée peut être associée à plusieurs étapes.

• Par défaut, les actions sont produites tant que les étapes auxquelles elles sont associées sont actives (ce sont les actions *continues*).

• Les actions peuvent en outre être *conditionnelles*, *mémorisées*, ou encore *limitées* dans le temps ou au contraire *retardées*.

types d'actions	représentation NF C 03-190 sept 95	principe
action continue	3 — action	l'action est produite tant que l'étape associée est active
action conditionnelle	condition 3 — action *ou* 3 — action si condition	*2 représentations possibles* l'action est produite tant que l'étape associée est active et que la condition est vraie
action mémorisée	3 — action = 1 27 — action = 0	l'action est produite dès que l'étape X3 est activée jusqu'au moment où l'étape X27 est activée
action retardée	3 — action si t/3/délai	l'action est produite après un certain délai suite à l'activation de l'étape associée, et ensuite, tant que cette étape est active
action limitée	3 — action si pas t/3/durée	l'action est produite pendant une certaine durée suite à l'activation de l'étape associée

figure A-14

Les normes CEI 848 et UTE C 03-191 divergent quant à ces symboles : le rectangle qui encadre l'action comporte une ou plusieurs sections supplémentaires.
Exemple :

3 — S | action — pour début d'action mémorisée (Set)

27 — R | action — pour fin d'action mémorisée (Reset)

Différentes combinaisons de S avec les lettres D, L, et C définissent les actions retardées, limitées et conditionnelles (avec mémorisation ou non). Cette symbolique est effectivement liée au modèle Grafcet *mais peut être confondue avec celle du langage SFC, pratiquement identique*.

- ***Liaison orientée, entrance / sortance***

• Les étapes sont reliées entre-elles par des liaisons. Ces liaisons sont orientées et définissent l'ordre d'activation des étapes. L'évolution générale du grafcet se fait du haut vers le bas, aussi ne représente-t-on pas obligatoirement de flèches sur les liaisons.

• L'entrance d'une étape (par où arrive une liaison) est *toujours* sur son côté supérieur. La sortance *toujours* sur son côté inférieur.

• On parle également de côté aval ou côté amont d'une étape.

- ***Transition : validée / non validée***

La liaison entre deux étapes comporte une et une seule transition. Cette transition permet l'évolution du grafcet de l'étape précédente vers l'étape suivante. Lorsque l'étape précédente est active, on dit que la transition est validée.

- ***Réceptivité : vraie / fausse***

De même que des actions sont associées aux étapes, des réceptivités sont associées aux transitions. Les réceptivités sont toujours des propositions logiques dont le résultat ne peut être que vrai ou faux. On distingue d'une part les niveaux 0 et 1, qui sont stables, et les passages d'un niveau à l'autre, qui sont par conséquent transitoires, et qu'on appelle événements. Ils sont notés de la manière suivante :

niveau		événement	
= 1	**v**	0 → 1	**↑v** : front montant
= 0	$\overline{\mathbf{v}}$ (ou **/v**) : v barre	1 → 0	**↓v** : front descendant

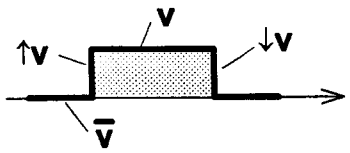


figure A-15

Le résultat logique peut également être formé d'après des opérations numériques, être la conséquence de l'évolution d'un temporisateur ou encore être toujours vrai.

résultat d'une opération	résultat d'une temporisation		résultat toujours vrai	
exemple (comparaison) : **[compteur < (1000+consigne)]**	exemple : **t/X12/30sec**	la temporisation qui a démarré à l'étape X12, et qui a duré 30 sec, prend fin (figure A-13)	**=1**	cette réceptivité est toujours vraie, aucune autre condition n'est attendue

figure A-16

b) Les règles d'évolution

L'évolution d'un grafcet se fait par franchissements successifs des transitions. Le modèle Grafcet définit d'une manière très précise comment s'opèrent ces franchissements.

- ***Franchissement d'une transition***

Plusieurs phases sont mises en oeuvre. On rappelle que l'activité d'une étape a comme effet de valider la transition en aval (figure A-17-b). Cette première condition est insuffisante (figure A-17-c) : la réceptivité doit être vraie pour que la transition soit franchie. La conséquence du franchissement d'une transition est d'activer immédiatement l'étape

suivante et simultanément de désactiver l'étape précédente. Une nouvelle transition sera donc validée sitôt la transition en question dévalidée (figure A-17-d).

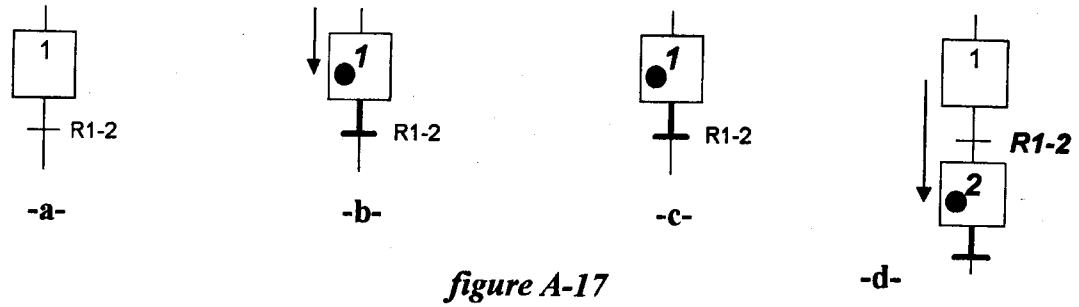
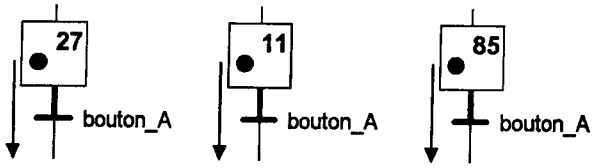


figure A-17

- ***Simultanéité des évolutions***

Si plusieurs transitions sont franchissables à un instant donné, elles sont toutes franchies en même temps. Aucune priorité n'est donnée à l'une d'entre elles. Ce cas se produit par exemple lorsque plusieurs transitions sont validées (parce que plusieurs étapes sont actives) et que la même réceptivité leur est associée.



1) X27, X11 et X85 sont actives

2) on appuie ensuite sur le bouton A

3) les 3 transitions sont franchies en même temps sans aucun ordre de priorité

figure A-18

- ***Situation initiale***

Selon les règles déjà énoncées, au moins une transition doit être validée pour permettre une première évolution. Cette situation peut être obtenue de 2 manières : par les transitions sources (voir le paragraphe suivant), ou par activation d'au moins une étape au démarrage du système : ce sont les étapes initiales, représentées par un carré double.

c) La syntaxe

- ***Alternance entre étapes et transitions***

Le grafcet évolue d'étape en étape à chaque fois qu'une transition est franchie. De ce fait, les liaisons orientées doivent comporter *une et une seule* transition entre 2 étapes.

- ***Source et puits***

Il arrive qu'une transition doit pouvoir être franchissable si la réceptivité associée est vraie *quelle que soit la situation du grafcet.* La validation de cette transition est alors permanente et ne dépend donc d'aucune étape précédente. C'est une transition source (comme une source de montagne, elle est toujours alimentée). Elle est utilisée lorsque l'*apparition* de la réceptivité qui lui est associée doit avoir un effet prioritaire sur tout le reste de l'application (appui sur le bouton arrêt d'urgence, perte de pression dans le circuit pneumatique, ouverture inopinée du capot de protection, etc.)

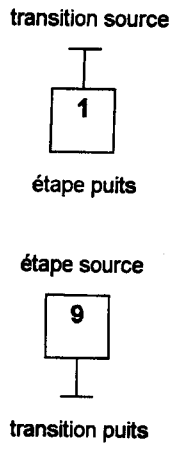


figure A-19

A l'inverse, le franchissement d'une transition peut ne pas activer d'étape suivante. L'étape précédente est cependant désactivée. La marque que véhicule le grafcet (le point noir qui symbolise l'activité des étapes) est alors perdue (comme dans un puits, d'où le nom de cette

transition particulière). De la même manière, les étapes peuvent être source ou puits.

- ***Structures de base***

Les structures sont les différentes manières d'organiser les liaisons orientées. Elles respectent toutes l'alternance entre les étapes et les transitions, ce qui explique les places occupées par les transitions. Ce point mérite une attention particulière. La figure A-20 illustre les structures de base. Le grafcet de la figure A-20-a comporte un aiguillage vers 2 séquences distinctes. L'une d'elles seulement pourra être exécutée. L'aiguillage peut être étendu à plus de 2 séquences. On parle de structure à sélection de séquence ou de divergence en *ou*. La figure A-20-b représente la convergence de telles séquences.

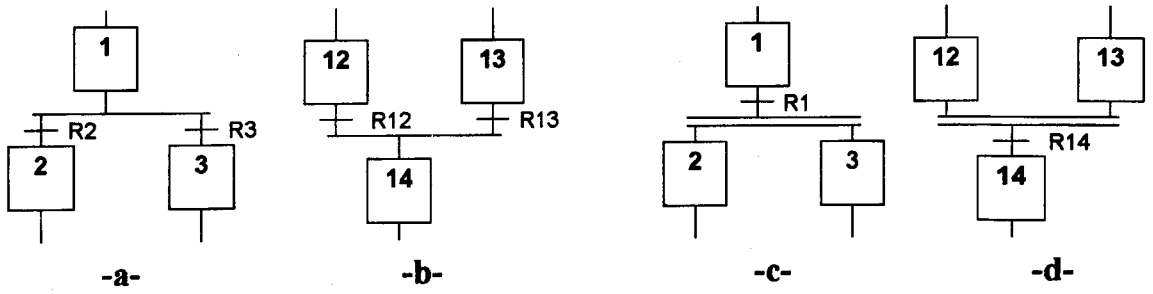


figure A-20

Les figures c et d représentent divergence et convergence (en *et*) de séquences simultanées. L'étape X1 valide la transition en aval. Si la réceptivité R1 est vraie, ce sont *toutes* les étapes suivantes qui sont activées (X2 *et* X3 dans ce cas). Par ailleurs, la transition en amont de X14 (figure d) n'est validée que si *toutes* les étapes qui la précèdent sont actives (X12 *et* X13 dans ce cas). Celles-ci seront *toutes* désactivées lors du franchissement de cette transition.

Dès la fin de chaque séquence simultanée, on ajoute une étape d'attente. Ainsi, chaque séquence évolue à son propre rythme et atteint son étape d'attente. La *resynchronisation* a lieu lorsque toutes les séquences sont terminées. Plus aucune autre condition n'est nécessaire pour poursuivre le cycle, c'est pourquoi la réceptivité en aval de la convergence est une réceptivité toujours vraie. Notons que ceci est le seul cas où cette réceptivité particulière est employée sans ambiguïté pour l'évolution du grafcet (ce point est développé au chapitre II.3).

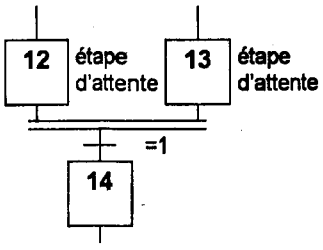


figure A-21

- ***Structures habituelles***

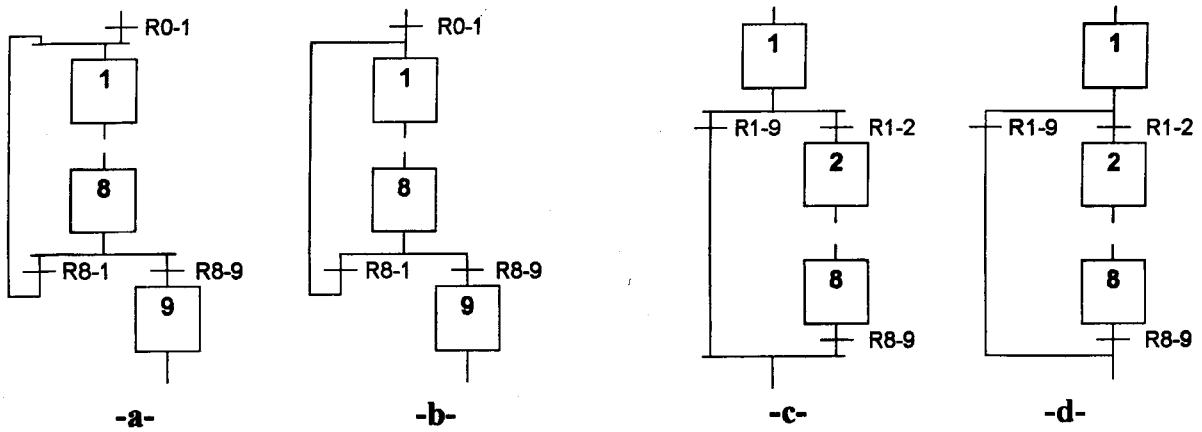


figure A-22

Les structures de base donnent lieu à des configurations souvent rencontrées (figure A-22). Il s'agit de la reprise de séquence (a et b) et du saut d'étapes (c et d).

Signalons la possibilité de graphismes simplifiés (figures b et d), que l'on rencontre la plupart du temps dans les logiciels d'édition.

- ***Gestion de priorités***

La structure à sélection de séquence pose des questions particulières. Dans le cas de la figure A-20-a par exemple, les réceptivités R2 et R3 peuvent très bien être vraies *avant* l'activation de l'étape X1. Cette activation valide les 2 transitions en question. Les réceptivités étant vraies, et, *les transitions simultanément franchissables devant être franchies simultanément*, les séquences qui commencent par les étapes X2 et X3 sont exécutées toutes les deux.

A propos :	Réponse :
Est-il exact que les 2 séquences seront réellement exécutées ?	Pour le modèle Grafcet : oui. Au niveau de la réalisation : c'est moins certain. En effet, les possibilités de réalisation d'un grafcet sont nombreuses. Le plus souvent, il est traduit en programme automate. Selon la marque de l'automate utilisé, son langage et la manière de programmer, il se peut que ce grafcet évolue effectivement vers les 2 séquences. Il se peut également qu'une seule séquence soit exécutée.

Si cette représentation peut créer un malentendu, le concepteur doit obligatoirement choisir une solution plus explicite. Dans le cas où R2 et R3 peuvent effectivement être vraies avant l'activation de X1, plusieurs possibilités se présentent :

- soit les 2 séquences doivent être exécutées,
- soit aucune séquence ne doit être exécutée,
- soit l'une d'elles seulement doit être exécutée.

Les grafcets de la figure A-23 illustrent ces divers fonctionnements sans équivoque.

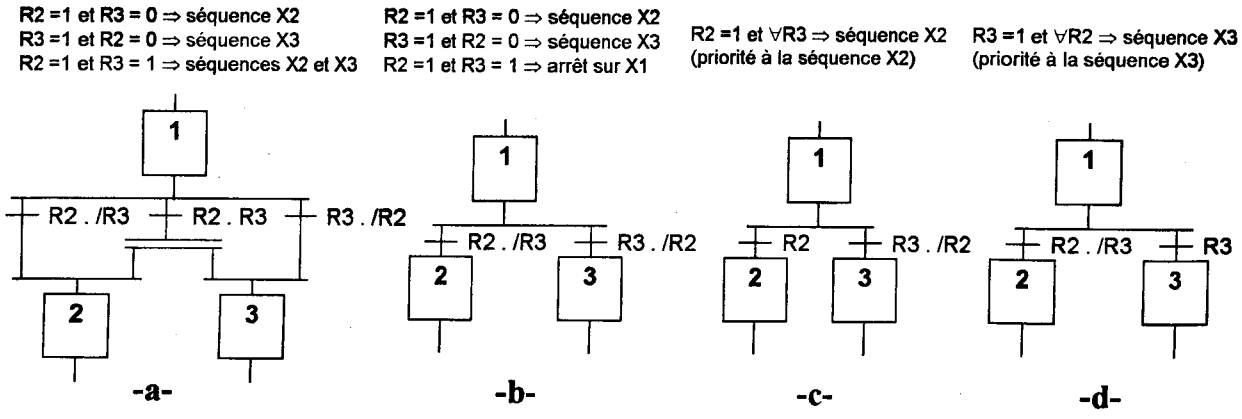


figure A-23

1 - 2. Dialogues entre grafcets

Une étude complète se décompose en général en plusieurs fonctions décrites chacune par un grafcet spécifique. Les modes de marches et d'arrêts se traduisent également par de multiples grafcets. Des dialogues entre eux devront donc s'établir.

a) Niveaux hiérarchiques

L'organisation de tous ces grafcets se fait sur plusieurs niveaux hiérarchiques (figure A-24). Au niveau supérieur, le *grafcet de surveillance* est chargé d'autoriser ou non le fonction-

nement du cycle de production. Le *grafcet de conduite* gère les modes de marches et d'arrêts et assure l'unicité du mode (un seul mode à la fois doit être actif). Chaque mode peut ensuite être décrit par un grafcet spécifique, le *grafcet de commande* du mode en question. Des tâches subalternes peuvent être définies par des grafcets de tâches de niveaux inférieurs.

b) Niveaux hiérarchiques

En principe, les dialogues entre ces grafcets ne se font que verticalement. Du haut vers le bas, ce sont des *ordres*, du bas vers le haut, des *comptes-rendus*.

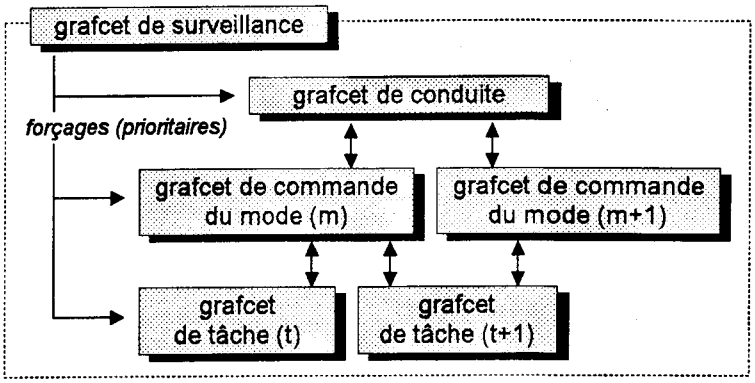


figure A-24 : exemple de grafcets hiérarchisés

c) Représentation et modes de fonctionnement

Plusieurs possibilités de représentations sont offertes. En premier lieu, il faut distinguer les dialogues liés à la gestion des tâches des ordres généralisés pour assurer la surveillance. Les ordres qui concernent cette seconde catégorie sont prioritaires sur les autres dialogues : ce sont des *forçages* de situations (figure A-24).

- ***Gestion des tâches***

Le tableau ci-dessous montre différentes versions d'une étape qui émet un ordre d'exécution vers un grafcet de niveau hiérarchique inférieur.

Tâche	Séquence	« Sous-programme »	Macro-étape
proposé à l'origine par le Grepa 1 T10	CEI 848 / UTE C 03-191 1 SEQUENCE 10 - 19	NF C 03-190 1 S-PRG 10 - 19	René David / UTE C 03-191 M1
Ce concept fait partie du Grafcet et n'introduit donc pas d'extention supplémentaire. Il permet toutes sortes de combinaisons en jouant sur les seules règles du Grafcet et est adapté à tous les langages d'automates. (voir ci-dessous)	Le symbole du rectangle, déjà attribué aux actions, peut de ce fait prêter à confusion, notamment lors de la programmation. Le concept est identique à celui de la tâche.	Ce terme est impropre car il fait référence à *quelque chose de semblable* dans le domaine informatique. Il est d'ailleurs systématiquement placé entre guillemets dans la norme. Les sous-programmes effectivement rencontrés lors de la programmation sont un concept différent : attention aux confusions possibles	Les macro-étapes permettent également de décomposer un grafcet en plusieurs parties. Elles s'ajoutent aux règles de base du Grafcet.

figure A-25

L'expansion d'une *macro-étape* se distingue des grafcets tâche, séquence et « sous-programme » par une formalisation particulière (figure A-26) :

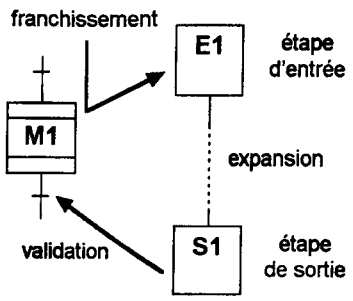


figure A-26

• si la macro-étape (du grafcet appelant) doit être activée, c'est en réalité l'étape d'entrée de son expansion (grafcet appelé) qui est activée,

• la transition en aval de la macro-étape est validée par activation de l'étape de sortie de son expansion.

L'expansion peut en fait s'insérer en lieu et place de la macro-étape. Elle n'a pas d'étape initiale.

Quant aux *tâches, séquences* et *« sous-programmes »*, les grafcets appelés répondent à la structure de la figure A-27-a. Le grafcet appelé nécessite une étape initiale et en fin de séquence, une étape de compte-rendu : sur l'exemple de cette figure, l'activation de l'étape X1 suffit à faire évoluer le grafcet de tâche car sa première transition est *validée* dès le démarrage du système. On voit qu'aucune nouvelle règle n'est nécessaire. De même, l'activation de l'étape de compte-rendu (X19) permet l'évolution du grafcet appelant. Dès lors, le grafcet de tâche est réinitialisé pour un usage ultérieur (réceptivité /X1).

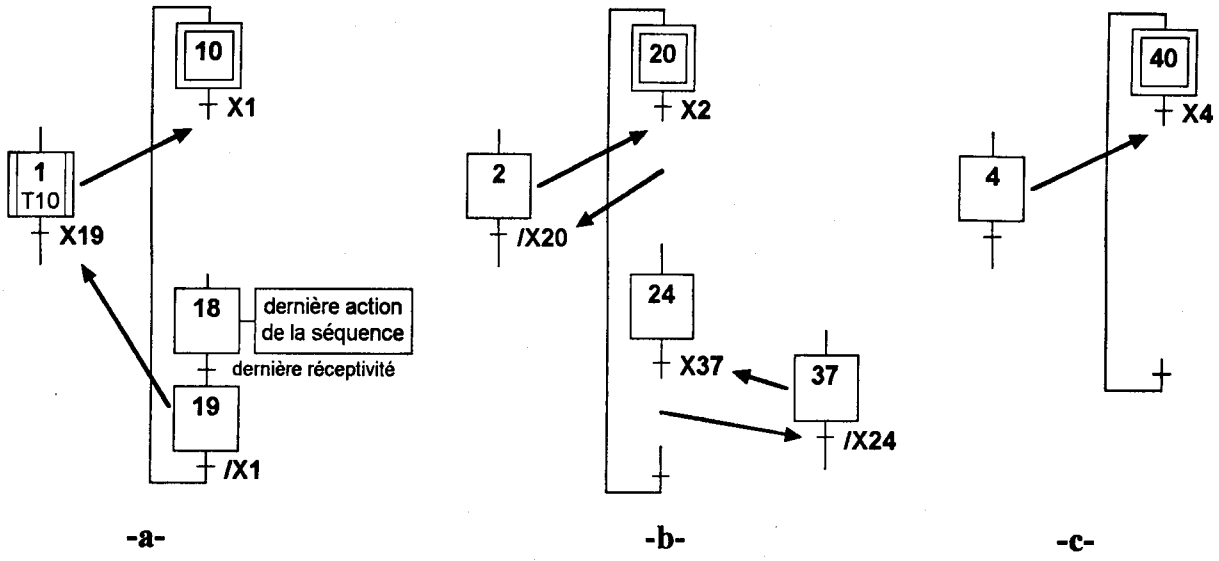


figure A-27

La synchronisation complète qui vient d'être présentée est la plus courante (le grafcet appelant n'évolue que si la séquence appelée est terminée). Mais basées sur ce principe général, d'autres formes de dialogues sont possibles.

• Le grafcet de tâche de la figure A-27-b est exécuté sur activation de l'étape X2. Les évolutions des deux grafcets sont ensuite indépendantes l'une de l'autre.

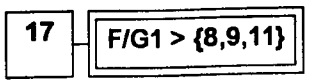
• Le grafcet de tâche de la figure A-27-c exécute sa séquence en boucle tant que X4 est active.

On rappelle que dans aucun de ces cas une quelconque action n'a été associée à l'étape appelante. Le seul fait de son activation suffit à exécuter la séquence.

- Forçages de situations

Le forçage consiste à imposer une situation déterminée à tous les grafcets d'une application quelle que soit leur situation courante. Du fait que ces ordres sont prioritaires par rapport aux autres dialogues, il est conseillé de les réserver au niveau *surveillance*. Des conflits de priorité risquent sinon de remettre en cause la sécurité. Par conséquent, un grafcet forcé

dans une situation donnée ne peut pas évoluer tant que l'ordre de forçage est présent. Le symbole défini par la norme UTE C 03-191 est le suivant :



Tant que l'étape X17 est active, les étapes X8, X9 et X11 du grafcet G1 sont actives et les autres étapes de ce grafcet sont inactives.

figure A-28

Cette notion permet de nombreux dérivés : figeage dans la situation courante, forçage des états des sorties, etc. Pour éviter d'encombrer le modèle Grafcet avec une iconographie supplémentaire, la norme se limite au symbole présenté ci-dessus. Les intentions particulières du concepteur peuvent donc être basées sur ce symbole et complétées *en clair*. En réponse aux cas souvent rencontrés, on peut admettre la symbolisation suivante :

F/G1 > { }	→	*forcer la situation vide de G1* :	aucune étape n'est active
F/G1 > {*}	→	*figer la situation courante de G1* :	les étapes conservent leur état
F/G1 > {INIT}	→	*forcer la situation initiale de G1* :	seules les étapes initiales sont actives

2 - NIVEAU DE PRECISION D'UN GRAFCET

Les notions de quantité et de qualité d'une information peuvent être illustrées par l'exemple suivant : quand on dit qu'il pleut, l'information *il pleut* exprime la même quantité d'informations pour un promeneur que pour un météorologue. Mais la qualité de cette information n'est pas la même pour tous : pour le promeneur, elle signifie *ne pas oublier de prendre le parapluie*, pour le météorologue, *mettre à jour les statistiques*. Ce qui est mis en évidence par cette représentation multiple d'un même phénomène est la notion de point de vue.

2 - 1. Structure d'une chaîne fonctionnelle

a) Notion de chaîne fonctionnelle

La figure A-29 montre la structure générale des liens à la fois physiques et informationnels qui mettent en relation la partie commande et la partie opérative pour la réalisation *d'une seule* fonction du système. La partie commande peut être considérée comme étant le *cerveau* du système, la partie opérative les *muscles* et les chaînes qui les relient seraient alors les *nerfs* : la chaîne d'action transmet les *ordres* alors que la chaîne d'acquisition véhicule les *comptes-rendus*. Ensemble elles constituent une chaîne fonctionnelle destinée à remplir une fonction. Un système complet fait intervenir plusieurs chaînes semblables.

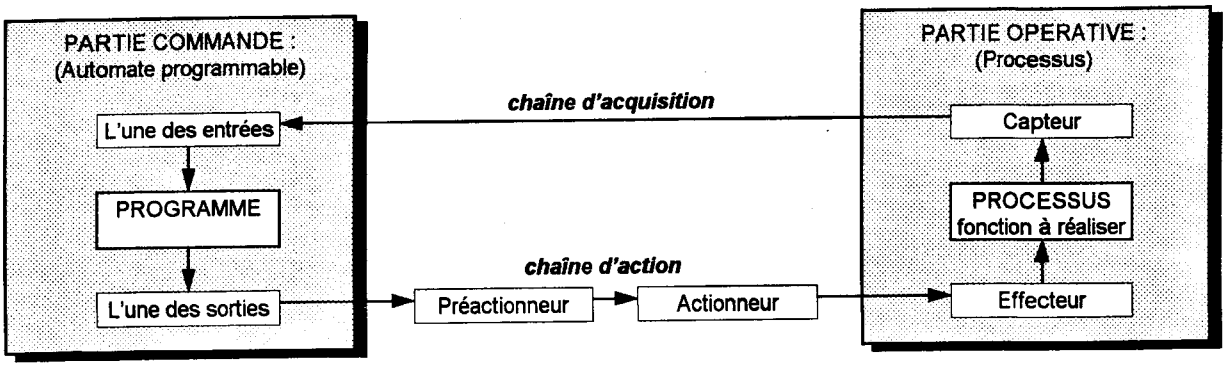


figure A-29

b) Chaîne d'acquisition

Le type et le niveau d'énergie de nombreux capteurs sont directement compatibles avec les cartes d'entrées des automates (par exemple 24V=). La liaison entre ces deux composants ne nécessite donc en général pas d'interfaçage particulier.

c) Chaîne d'action

Les actionneurs par contre utilisent des énergies de type et de niveau différents de ceux des cartes de sorties des automates (moteur 380V triphasé, vérin hydraulique, etc.) La conversion du type et/ou du niveau d'énergie doit donc obligatoirement être prise en charge par un composant supplémentaire : le préactionneur (contacteur, distributeur, etc.) En amont du préactionneur, on parle de *circuit de commande*, et en aval, de *circuit de puissance*.

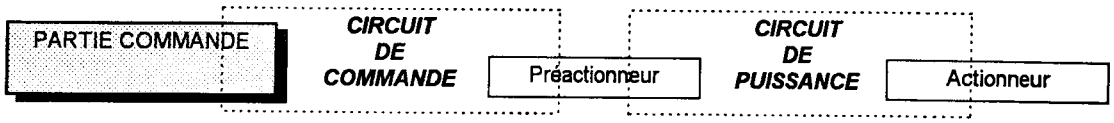


figure A-30

2 - 2. Etude détaillée d'une chaîne d'action

Soit l'exemple d'une caisse qu'il faut déplacer à un moment donné dans un cycle automatique de palettisation.

- Des phénomènes liés à ce déplacement se produisent en plusieurs endroits de la chaîne d'action (d'où les *niveaux* a, b, c, d et e ci-dessous).
- En chaque endroit, on peut porter un regard extérieur sur le système (c'est le *point de vue de l'observateur* dont les outils sont autant le voltmètre que la vue).
- On peut également considérer ces phénomènes du *point de vue du système* ; on se place alors « à l'intérieur » des composants.

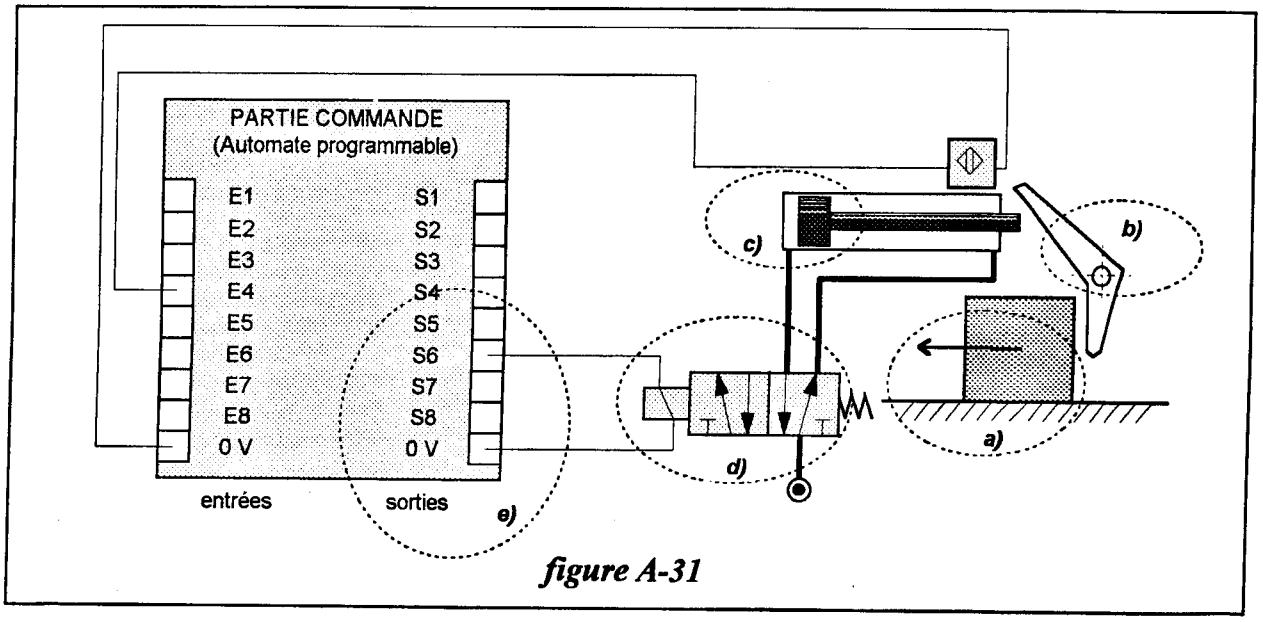


figure A-31

a) Au niveau de la fonction à réaliser

L'endroit observé est la caisse elle-même car le seul objectif de cette chaîne fonctionnelle est de la déplacer. Les 2 points de vue sont notés différemment : un verbe au présent pour exprimer que le phénomène est en train de se produire, un verbe à l'infinitif pour exprimer que l'action doit être produite par le système.

- ***du point de vue de l'observateur extérieur*** — la caisse se déplace
- ***du point de vue du système*** — déplacer la caisse

Pour l'observateur, la proposition *la caisse se déplace* est une *action* réalisée par le système. Par contre, pour qu'elle puisse se déplacer, le système doit donner un *ordre*, qui est *déplacer la caisse*. A ce niveau, la technologie employée n'est pas évoquée. La caisse pourrait aussi bien se déplacer sous l'effet de l'inclinaison de son plan d'appui que par la mise en marche d'un convoyeur à rouleaux.

b) Au niveau de l'effecteur

- ***du point de vue de l'observateur extérieur :*** — le levier pivote
- ***du point de vue du système :*** — faire pivoter le levier

Là aussi, l'observateur constate une action alors que le système émet l'ordre qui doit la provoquer. On a définit la solution technique qui permet d'agir sur la caisse : un levier.

c) Au niveau de l'actionneur

- ***du point de vue de l'observateur extérieur :*** — la tige du vérin avance
- ***du point de vue du système :*** — faire avancer la tige du vérin par amenée de la pression

A ce niveau de représentation, c'est la technologie de l'actionneur qui est définie, un vérin pneumatique à double-effet par exemple.

d) Au niveau du préactionneur

- ***du point de vue de l'observateur extérieur :*** — la bobine de l'électro-vanne qui pilote le distributeur est parcourue par un courant électrique et le tiroir du distributeur change de position
- ***du point de vue du système :*** — alimenter l'électro-vanne en courant électrique

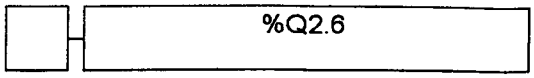
L'observateur peut mesurer la tension qui apparaît aux bornes de l'électrovanne. Le système doit fournir cette tension en établissant un circuit électrique. Le choix technologique concerne le préactionneur : par exemple un distributeur monostable à commande électrique.
L'écriture de cet ordre est souvent simplifiée en définissant un symbole équivalent.
L'étape du grafcet s'écrit alors comme ceci : — A

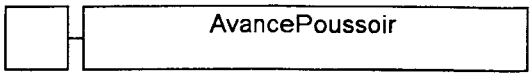
e) Au niveau de la sortie de l'automate

- ***du point de vue de l'observateur extérieur :*** — une tension apparaît à la sortie n°6 de l'automate (emplacement 2.6)
- ***du point de vue du système :*** — activer la sortie n°6 (adresse %Q2.6)

Cette fois, on peut mesurer la tension sur la borne de la sortie n°6 de l'automate. Du point de vue du système, c'est un ordre qui est généré par le programme. A ce niveau, on définit le type d'automate et le câblage de ses sorties.

Dans le grafcet destiné à l'écriture du programme d'automate, on note le numéro de l'emplacement défini pour le câblage. Ce numéro est également celui de l'adresse de la mémoire image associée. Les logiciels de programmation des automates permettent de définir un symbole pour chaque adresse, ce qui facilite la maintenance du programme (à condition que les symboles choisis soient explicites). L'étape ci-dessus devient alors :





figures A-32 a-e

2 - 3. Relations de cause à effet

Les observations faites successivement mettent en évidence la notion de chaîne dont les maillons sont des relations de cause à effet.

- *Toute action observable est l'effet d'un ordre émis par un élément du système.*
- *Tout ordre émis par un élément du système provoque une action observable.*

Sur la figure A-33, le rectangle 1 est la cause qui provoque l'effet décrit par le rectangle 2, qui est lui-même cause de l'effet du rectangle 3 et ainsi de suite jusqu'au rectangle 10. Le grafcet peut être écrit à l'un quelconque de ces niveaux et points de vue.

Si une confusion sur la nature des déclarations peut subsister,
on précise le point de vue et le niveau des grafcets.

Il faut également garder à l'esprit que le programme n'agit pas sur la caisse, ni sur le vérin, ni même sur le distributeur : seulement sur la sortie de l'automate.

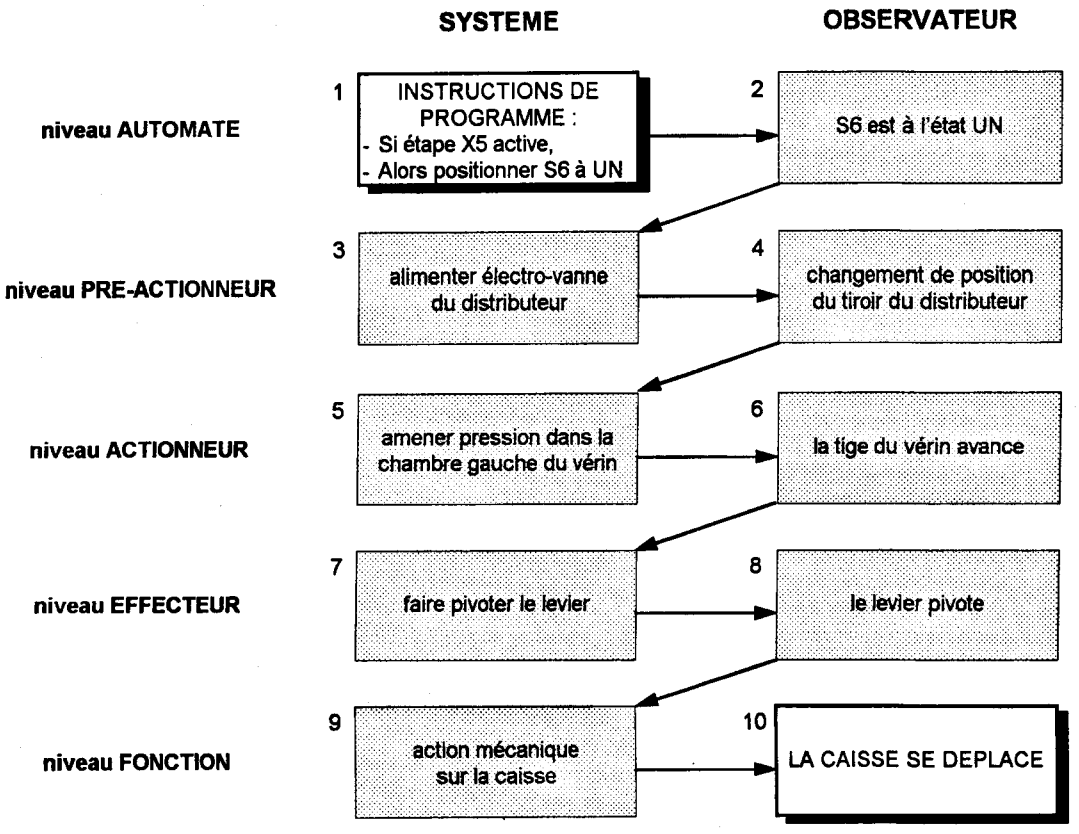


figure A-33

3 - GRAFCETS AU NIVEAU AUTOMATE

Sur certains modèles d'automates, l'algorithme de programmation est laissé à l'initiative de l'utilisateur. (L'algorithme est la manière d'organiser le programme, l'ordre dans lequel les instructions sont exécutées). Il doit être élaboré dans le respect des règles du Grafcet dont certaines sont difficiles à mettre en oeuvre d'une manière tout à fait rigoureuse. Il en est ainsi de la simultanéité de franchissement des transitions franchissables. En effet, le temps interne de traitement de l'automate, aussi court soit-il, ne peut jamais être réellement nul, ce qui empêche cette simultanéité. Une bonne maîtrise de ces phénomènes est primordiale, soit pour produire soit-même un algorithme correct, soit pour utiliser à bon escient celui implémenté par le constructeur le cas échéant.

3 - 1. Structure générale d'un programme

a) Cas d'un grafcet unique

Un grafcet complet peut être décomposé en 2 parties comme le montre la figure A-34. On a d'une part le grafcet proprement dit, à savoir la succession d'étapes-transitions (réceptivités comprises) et d'autre part les actions qui sont associées aux étapes. Les actions, ou les ordres qui les génèrent, dépendent directement des états des étapes. D'une manière générale, le programme doit donc mettre à jour les étapes du grafcet dans un premier temps, et dans un deuxième temps seulement, produire les actions en fonction de la nouvelle situation du grafcet.

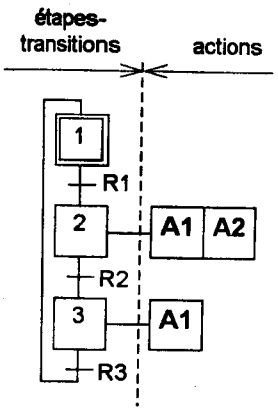


figure A-34

- ***Partie grafcet***

La mise à jour du grafcet se fait par analyse à la fois des étapes actives (situation courante) et des réceptivités qui sont vraies. Cette évolution se fait par franchissement des transitions franchissables. Pour *chaque transition* du grafcet, on réalise l'algorithme général suivant (*) :

Si étape précédente active
Et si réceptivité vraie
Alors activer l'étape suivante
Alors désactiver l'étape précédente
(ordre quelconque)

1 module par transition dans un ordre quelconque

Si le langage de l'automate n'intègre pas la représentation graphique du grafcet, cet algorithme peut facilement être codé sous forme de liste d'instructions ou sous forme de schéma à contacts.

- ***Partie actions***

Le grafcet ayant évolué, la 2° partie du programme doit produire les actions associées aux étapes actives. Dans l'exemple de la figure A-34, l'équation logique de l'action A1 s'écrit :

A1 ← X2 + X3

Si l'action A1 est une sortie qui agit sur la partie opérative, son équation est souvent enrichie de conditions supplémentaires : sécurité, modes de marches, affinement du fonctionnement, etc.

(*) D'autres techniques de programmation peuvent être envisagées. L'objet de ce chapitre étant de présenter les principes généraux, elles ne seront pas abordées.

b) Partie préliminaire

La formation de la réponse à un front ou le positionnement du résultat d'une équation logique particulièrement ardue peuvent être traités dans une partie supplémentaire du programme (exemple de la figure A-35). Cette partie doit être exécutée *avant toute autre* afin que ses résultats puissent être exploités ensuite. Le regroupement de ces calculs permet une meilleure structuration du programme.

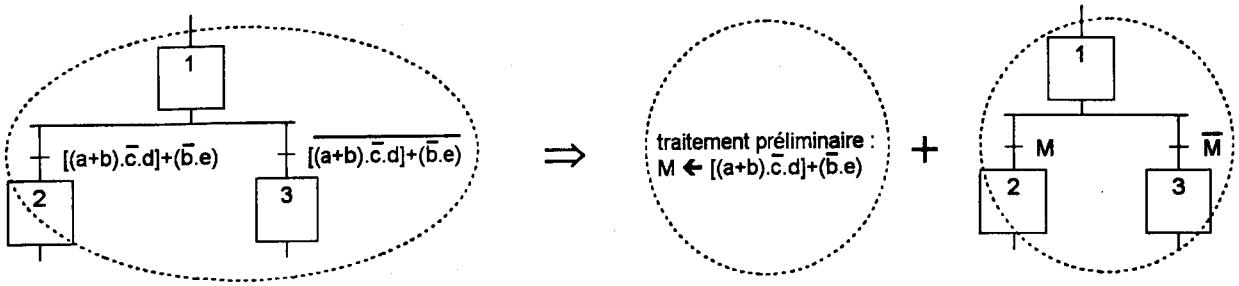


figure A-35

c) Cas d'un programme qui comprend plusieurs grafcets

Une sortie peut très bien être produite par des étapes qui appartiennent à des grafcets différents. Une sortie ne devant par ailleurs être positionnée que par une seule équation, on préfère d'une manière générale structurer le programme comme suit :

Traitement préliminaire
Traitement du grafcet g-1
Traitement du grafcet g] dans un ordre quelconque
Traitement du grafcet g+1
Traitement des sorties

Cette généralisation n'est pas utile pour toutes les sorties lors de l'étude initiale du système, mais elle garantit la pérennité des instructions existantes en cas de modifications ultérieures (ajout de grafcets en particulier). La mise à jour du programme est plus sûre du fait de sa modularité.

3 - 2. Programmation des actions

a) Stabilité d'une étape

Soit la figure A-36. Supposons que la réceptivité *r2* soit déjà vraie au moment de l'activation de l'étape X1. La transition associée est validée dès cette activation et donc *immédiatement* franchie. Le temps d'activation de l'étape X1 est proche de zéro : l'étape X1 est dite *instable*.

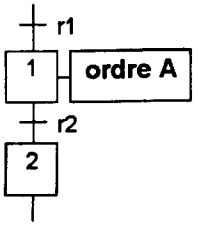


figure A-36

Ce qui compte dans le grafcet au niveau automate, c'est moins le <u>temps</u> de cette activation que la <u>possibilité</u> pour le programme de produire l'ordre qui est associé à l'étape X1.

b) Conséquence

- ***Si l'on considère le modèle Grafcet***

Le modèle Grafcet précise que la durée d'activation d'une étape et la durée de franchissement d'une transition ne peuvent jamais être tout à fait nulles : *les ordres associés à des étapes instables doivent donc être exécutés.*

- ***Si l'on considère l'algorithme général de programmation ...***

... qui consiste à ne traiter les actions qu'après *complète* mise à jour des étapes du grafcet, le résultat est le suivant : le traitement du grafcet a comme effet d'activer directement l'étape X2 dès vérification de la réceptivité ***r1*** ; au moment du traitement de l'équation de *A* (qui est A ← X1), cet ordre n'est pas produit car X1 *n'est plus* active. *Les ordres associés à des étapes instables ne sont donc pas éxécutés.*

- ***Première conclusion***

S'il s'agit d'un grafcet descriptif ou de spécification, des ordres peuvent être associés à des étapes instables. Plus de prudence est de rigueur si le grafcet est destiné à la programmation. Pour simplifier, la règle suivante s'applique à tous les cas de figure :

En règle générale, on ne peut associer aucune action à une étape instable.

- ***Si l'on considère les solutions des constructeurs***

En principe, la norme CEI 61131-3 standardise les démarches, mais le renouvellement du parc des automates s'étendra encore sur plusieurs années car la lourdeur des investissements en matière de maintenance freine souvent l'acquisition de nouveaux modèles. L'utilisateur doit donc prendre en compte la variété des principes de programmation, d'autant que la connaissance d'un seul type d'automate risque de le conduire à des habitudes fâcheuses.

c) Exemple 1 : gérer des données

Soit à gérer des données lorsque X12 est active (figure A-37). Envisageons les deux types de mises en oeuvre : sur étape instable, et sur étape stable.

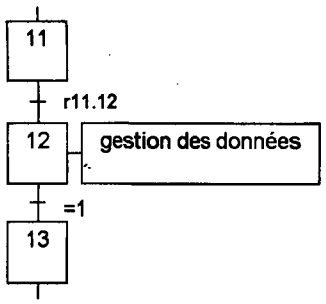


figure A-37

- ***Sur étape instable***

La réceptivité toujours vraie en aval de X12 autorise l'évolution immédiate du grafcet *dès* activation de X12. Pour que l'ordre de gestion des données soit pris en compte, l'algorithme général de programmation doit être aménagé comme suit :

```
Si étape X11 active
Et si [r11.12] vraie
     Alors activer l'étape X12
     Alors désactiver l'étape X11
     Alors gérer les données  <-------------  insertion du traitement interne

Si étape X12 active
     Alors activer l'étape X13
     Alors désactiver l'étape X12
```

Notons que la désactivation de X12 a lieu seulement après la gestion effective des données. De plus, les données n'auront été traitées *qu'une seule fois*, ce qui est suffisant d'une manière générale, et même obligatoire dans certains cas (empilement d'une information dans une file d'attente par exemple).

Cette insertion se traduit de manière différente selon le langage de l'automate utilisé.
A titre d'exemple, le grafcet de la figure A-37 est transcrit ci-dessous en langage Step5 (automates Siemens Simatic S5) et en langage Pl7-Micro (automates Modicon Télémécanique TSX 37/57).

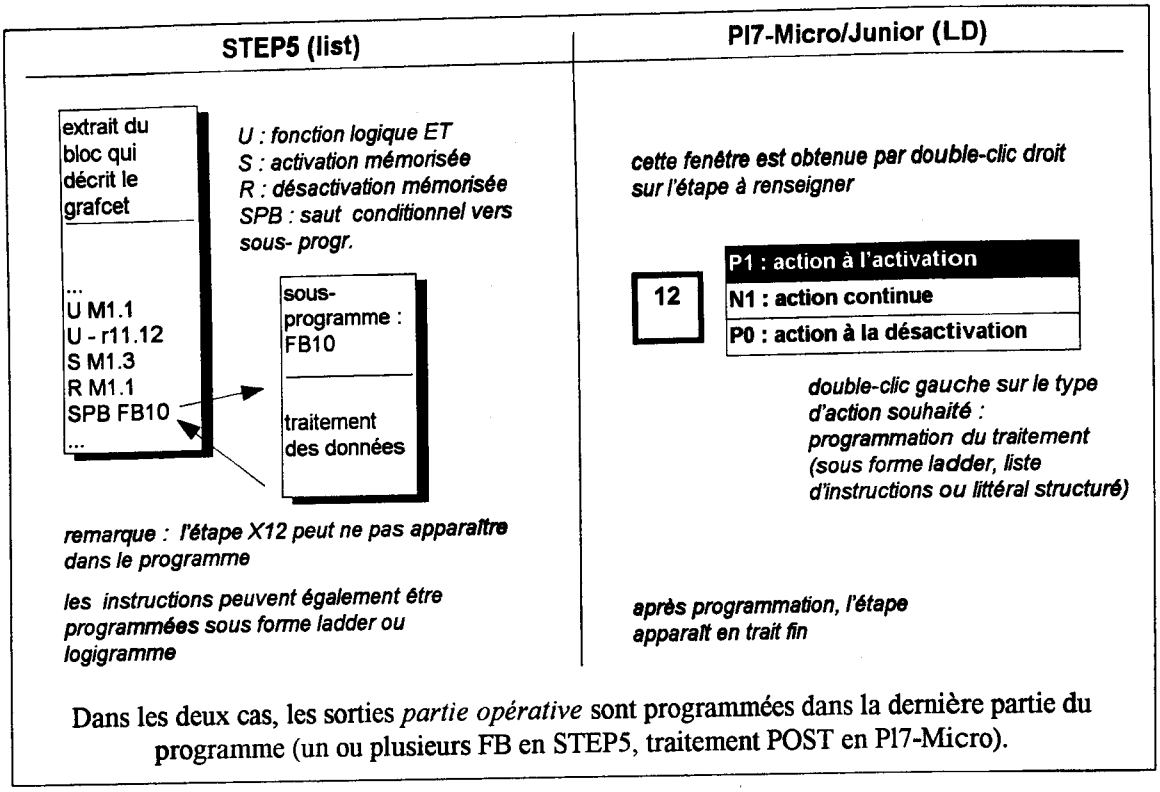


figure A-38

- ***Sur étape stable***

Dans ce cas, on applique l'algorithme général : la gestion des données est traitée après évolution complète du grafcet, c'est-à-dire dans la partie postérieure du programme, celle qui positionne également les sorties liées à la partie opérative (X12 doit être stable dans ce cas). L'étape n'est active que pendant une seule scrutation, le traitement n'a donc lieu qu'une seule fois. Cette technique s'applique d'une manière universelle, quel que soit l'automate (figure A-39).

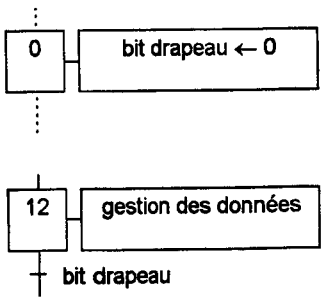


figure A-39

Si étape X12 active **Et si bit drapeau = 1** **Alors activer l'étape X13** **Alors désactiver l'étape X12**	partie grafcet
Si étape X12 active **Alors gestion des données et mise à UN du bit drapeau** **Si étape X0 active** **Alors mise à ZERO du bit drapeau**	partie sorties

Scrutation (n)

Partie grafcet : lors de l'activation de X12, la transition suivante ne peut pas être franchie car le bit drapeau est = 0. Le grafcet n'évolue pas : X12 est stable.

Partie sortie : X12 étant active ⇒ gestion des données et mise à UN du bit drapeau.

Scrutation (n+1)

Partie grafcet : X12 est active *et* le bit drapeau est =1 ⇒ évolution du grafcet.

Partie sortie : X12 n'est pas active, les données ne sont plus traitées.

- ***Remarque***

Le modèle Grafcet ne définit pas la représentation à ce niveau de finesse car sa vocation première est de définir uniquement les fonctions. La réceptivité [=1] notamment doit être utilisée avec précaution. On rappelle que la seule utilisation non ambiguë de cette réceptivité particulière est la resynchronisation après des séquences simultanées (convergence en ET).

d) Exemple 2 : incrémenter un compteur

Soit à positionner un compteur à une valeur déterminée (par exemple 0) sur l'étape X10 et à incrémenter ce compteur à chaque activation de l'étape X15.

- ***Le compteur est un mot quelconque***

Attribuons à ce compteur, dans un premier temps, l'adresse d'un mot quelconque : %MW4 par exemple.
Sous peine de voir le compteur augmenter d'un pas à chaque scrutation tant que l'étape X15 est active, le programmeur doit prendre l'une des précautions suivantes.

Sur étape instable :

- insérer l'incrémentation du compteur dans le traitement du grafcet (en Step5 sur la figure A-40)

Sur étape stable :

- ajouter une étape supplémentaire et un bit drapeau (non illustré),
- ou former un front sur activation (ou désactivation) de X15, avant incrémentation (en PL7- Micro sur la figure A-40).

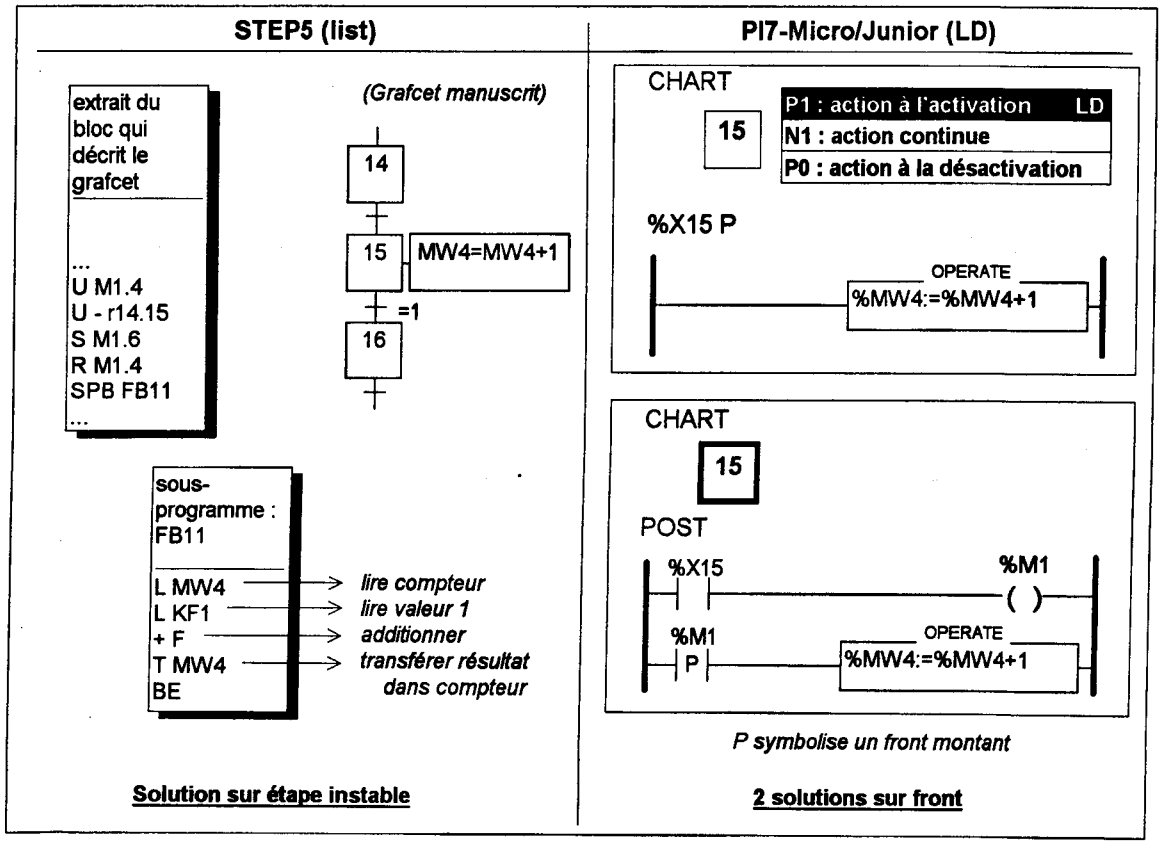


figure A-40

- ***Le compteur est une adresse spécifique ou un bloc fonctionnel***

Il est également possible d'attribuer à ce compteur une adresse spécifique, Z8 par exemple, ou un bloc fonctionnel, %C8 par exemple (selon l'automate utilisé). Les automates gèrent en général le fait que l'évolution des compteurs se fait sur front de la condition, ce qui simplifie la tâche du programmeur.

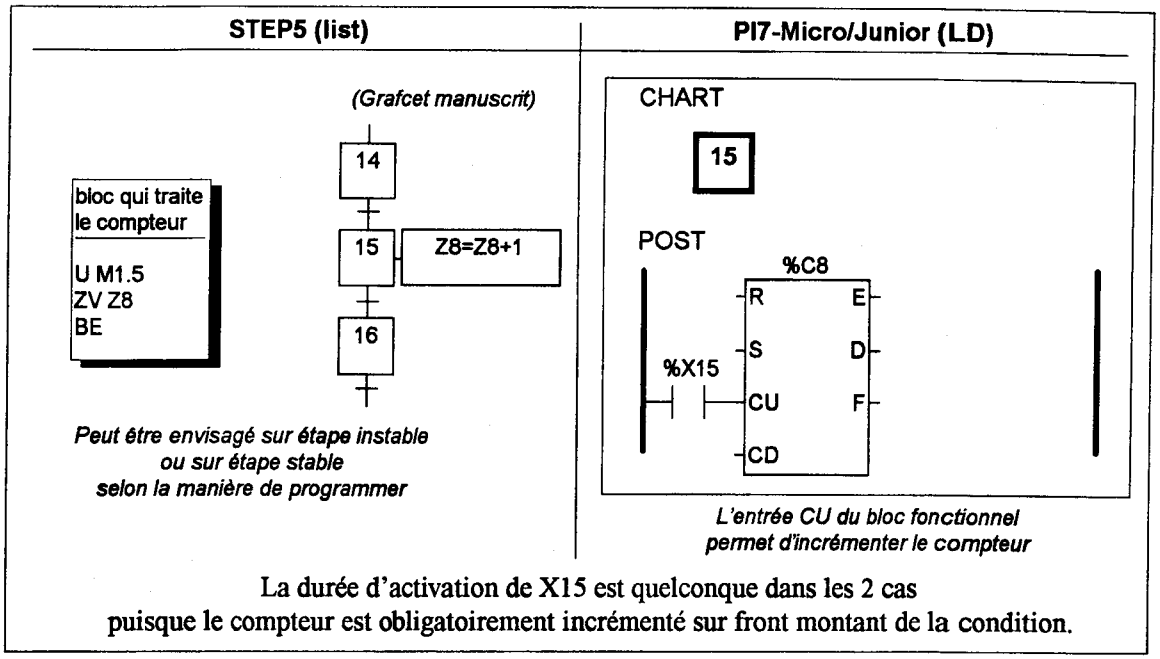


figure A-41

3 - 3. Application des fronts

Soit à allumer un voyant différent à chaque nouvelle impulsion sur un bouton poussoir. Les voyants sont au nombre de 2. La troisième impulsion a comme effet d'éteindre le dernier voyant. Ce cycle peut redémarrer à volonté.

On peut se satisfaire du grafcet au *niveau fonction* de la figure A-42-a.

Le grafcet au *niveau automate* doit cependant être aménagé. Une impulsion sur Bp est figée pour au moins une scrutation complète dans la MIE. L'étape X1 est alors activée et la transition suivante validée.

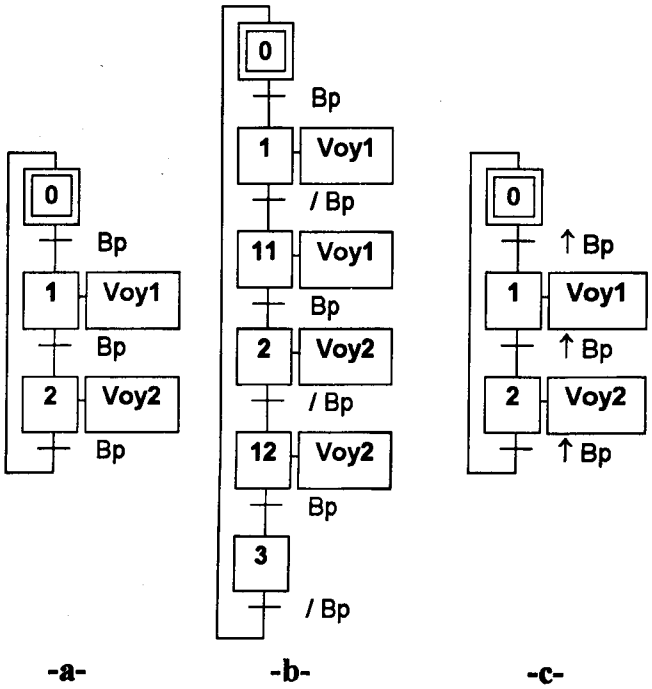


figure A-42

L'information Bp provenant de la MIE étant toujours =1, l'étape X2 est immédiatement activée et ainsi de suite jusqu'à épuisement de tous les modules de programme qui décrivent l'évolution du grafcet. Lorsque les équations des sorties sont traitées, il est probable que les étapes X1 et X2 ne sont plus actives : aucun voyant ne s'allume. Le grafcet n'a en réalité pas répondu au cahier des charges.

Les étapes doivent en fait être rendues stables. Cette situation se produit à chaque fois qu'une réceptivité est déjà vraie au moment de la validation de la transition à laquelle elle est associée et que seule l'occurrence (l'apparition ou la disparition) de cette réceptivité doit être prise en compte.

Le grafcet développé peut s'écrire sous 2 formes : soit en détaillant l'évolution des variables d'état (0 ou 1 seulement, figure A-42-b), soit en prenant en compte les états transitoires du signal (front montant ou front descendant, figure A-42-c).

4 - LES MODES DE MARCHES ET D'ARRETS

4 - 1. Les arrêts

a) Les arrêts pour causes normales

Ils peuvent être :

- demandés par l'opérateur : bouton d'arrêt
- émis par une autre partie commande
- générés automatiquement suite à épuisement du stock de pièces, etc.

Il sont traités au même titre que les modes de marches (voir § 4 - 2).

b) Les arrêts de sécurité

Il s'agit :

- de l'arrêt d'urgence : bouton coup de poing
- de l'ouverture du dispositif matériel de protection : capot, barrière, etc.
- des dysfonctionnements : procédures de surveillance automatiques, etc.

Les arrêts de sécurité doivent toujours être prioritaires par rapport aux marches. Afin de répondre à cette règle, issue des normes EN292 et EN418 sur la construction des machines, ces arrêts sont systématiquement intégrés par câblage.

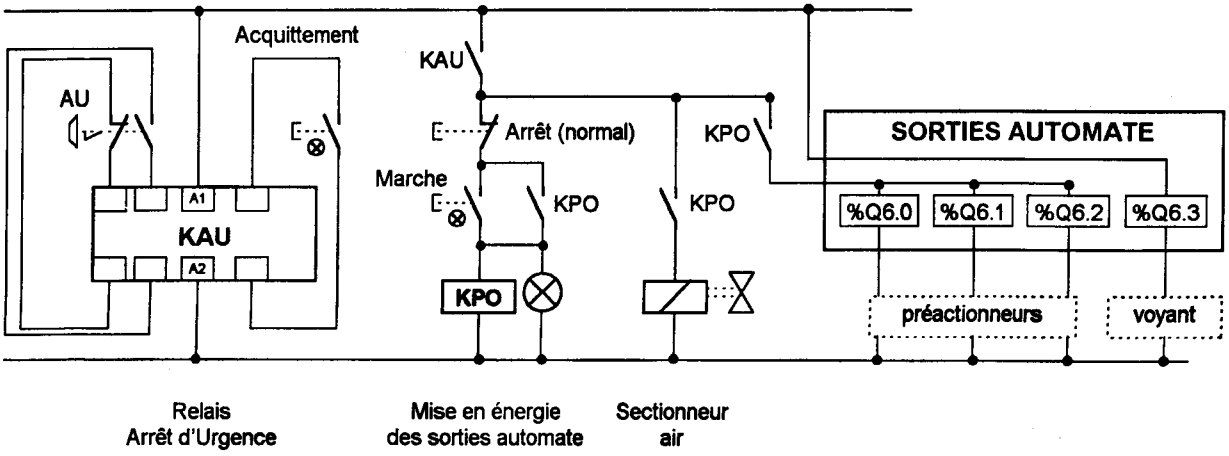


figure A-43

Le schéma de principe de la figure A-43 montre que si l'on appuie sur le bouton d'arrêt d'urgence, les sorties de l'automate ne sont plus alimentées. Les préactionneurs sont ainsi privés de leur alimentation *sans qu'aucune intervention du programme ne soit nécessaire*. Notons que les voyants conservent leur alimentation afin de permettre la signalisation des défauts par programme.

Le rôle du programme, en cas d'arrêt d'urgence, est en outre de forcer le grafcet dans une situation conforme à la réalité, c'est-à-dire de garantir la concordance entre le modèle et le

système modélisé. Pour que le redémarrage après acquittement se fasse en toute sécurité, il convient également de positionner tous les éléments de type bistable :

- les sorties qui pilotent un distributeur bistable par exemple (afin d'éviter un mouvement inattendu du vérin au retour de l'air),
- les sorties dont la mise à Un est mémorisée (afin d'éviter par exemple qu'un moteur ne se remette en marche dès l'acquittement).

4 - 2. Les marches

a) Modes de démarrage de l'automate

Le démarrage de l'automate peut avoir lieu dans des conditions très différentes :

- un nouveau programme vient d'être chargé dans l'automate : les variables prennent alors leurs valeurs initiales ⇒ on parle de *démarrage à froid*,
- par ailleurs, plusieurs cas de *reprise à chaud* sont possibles : soit l'automate a été arrêté puis redémarré avec le même programme, soit une coupure puis un retour du secteur se sont produits alors que l'automate était en fonctionnement, etc. Dans ces cas les variables peuvent conserver ou non leurs états, les étapes de grafcets peuvent être maintenues ou non. Les conditions de redémarrage sont paramétrables et/ou programmables selon les choix du constructeur de l'automate.

b) Exemple de modes de marches et Gemma

Après démarrage, plusieurs modes de marches peuvent en général être exécutés, soit sur demande de l'opérateur (passage en mode manuel par exemple), soit d'une manière automatique, illustrée par l'exemple de remplissage de pots de yaourts de la figure A-44.

Lorsqu'une nouvelle bande vient d'être introduite dans la machine, seul le formage doit avoir lieu dans un premier temps. En cours de production, les trois postes agissent simultanément après avance d'un pas de la bande. Lorsque la production doit se terminer, le formage et le remplissage s'arrêtent *avant* le poste de bouchage-découpe. On a donc 3 modes de marche : la préparation, la production normale et la clôture. Chacun d'eux peut être décrit par un grafcet indépendant (figure A-45).

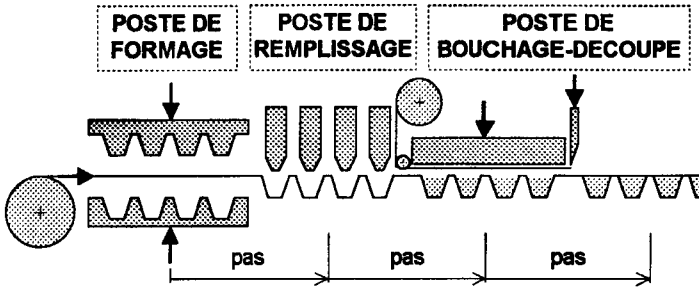


figure A-44

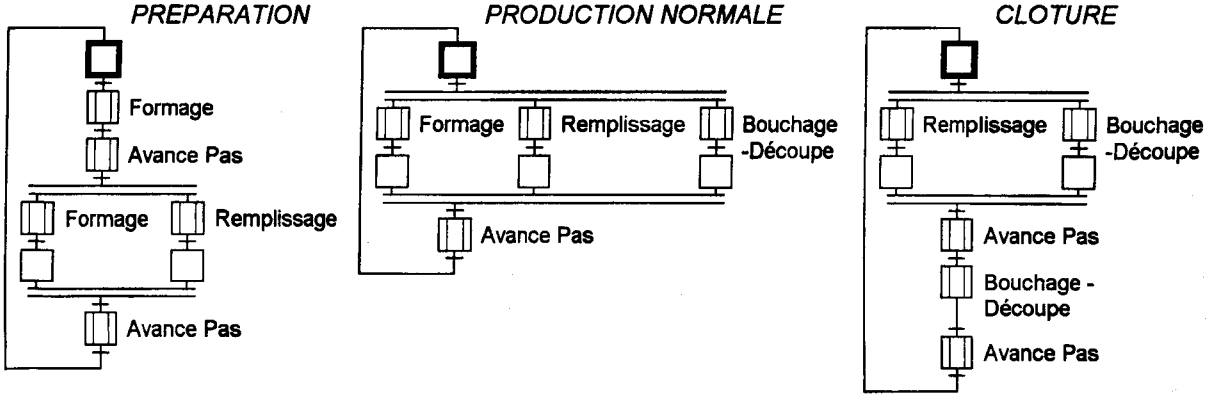


figure A-45

L'enchaînement de ces modes est organisé de la manière suivante :
Préparation → Production Normale tant que nécessaire → Clotûre. Un grafcet de gestion des modes de marches peut décrire ce cycle. On parle souvent de *grafcet de conduite* ou de *Gmma (grafcet de gestion des modes de marches et d'arrêts)*. La figure A-46 représente ce grafcet ainsi que les correspondances avec les grafcets des différents modes. Ceux-ci sont schématisés à l'intérieur de *rectangles-états* qui forment avec les liaisons un diagramme. Il n'est pas obligatoire de représenter les grafcets : un simple texte explicatif peut suffire. L'ADEPA a généralisé ce type de diagramme à tous les modes habituellement rencontrés sur les machines industrielles : le *Gemma (Guide d'étude des modes de marches et d'arrêts)*. Sa version la plus récente est représentée figure A-47. Ce document doit être complété par le concepteur : son but est de proposer, d'une manière similaire à une check-list, un canevas pour l'étude des modes de fonctionnement d'une machine. Le vocabulaire adopté est précis et le concepteur est guidé dans l'approche du problème.

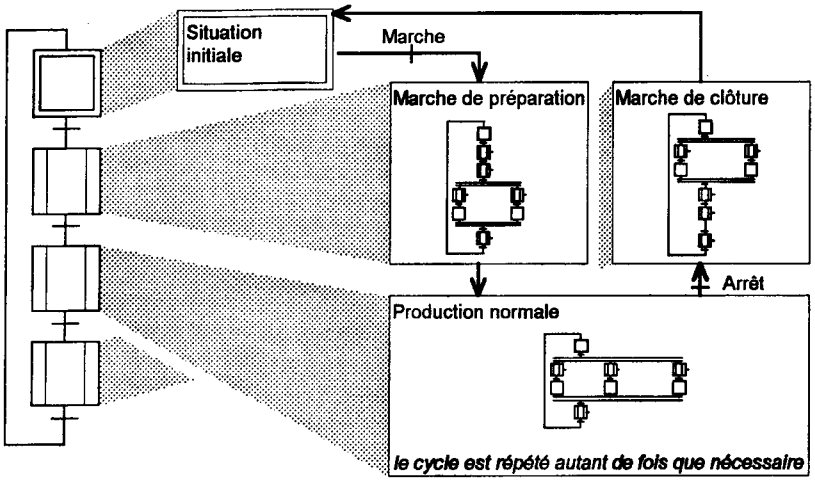


figure A-46

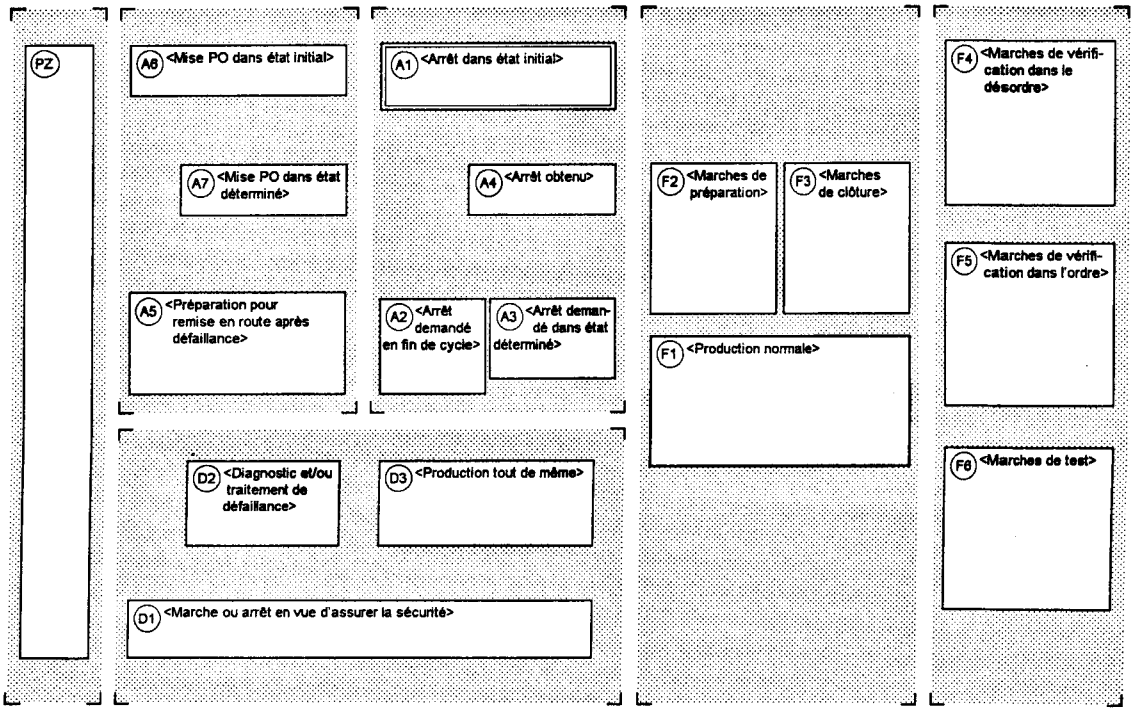


figure A-47

Les concepts du Gemma sont cependant bien moins rigoureux que ceux du Grafcet. Les liaisons entre les rectangles-états par exemple peuvent ne pas comporter de « transition ». Par ailleurs, des rectangles peuvent nécessiter soit aucune, soit plusieurs étapes. La transcription du Gemma en grafcet de conduite n'est de ce fait pas tout à fait systématique.

4 - 3. Conduite de la machine

Le Gemma ayant d'abord guidé le concepteur dans son travail, il sert ensuite à définir les interfaces utilisateurs liés aux modes de fonctionnement et éventuellement à tracer le synoptique de commande par simple analogie. Lors des phases d'exploitation et de maintenance, le Gemma facilitera ainsi la conduite de la machine. Le panneau de commande de la machine peut combiner différentes technologies : des boutons, des commutateurs, des voyants, un terminal opérateur programmable du commerce (voir la figure A-48). Ces appareils disposent d'un certain nombre de touches et de voyants ainsi que d'un afficheur textuel ou graphique. Tous ces éléments sont paramétrables en fonction de l'application.

Il est très important de prévoir toutes les signalisations utiles à la bonne prise en main de la machine. L'ajout de cette fonction ne nécessite que des frais limités et peu de programmation mais augmente considérablement la qualité générale du système. La signalisation doit cependant rester pertinente car la surabondance de voyants et autres dispositifs nuit à la perception de *chaque* signal.

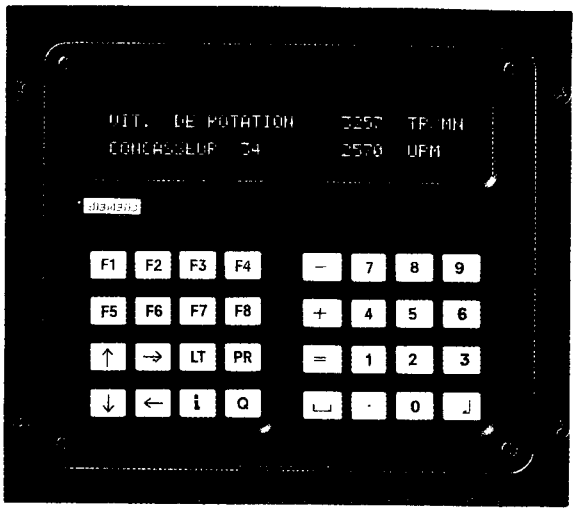


figure A-48

4 - 4. Programmes spécifiques

Quelquefois, certains modes ne sont utilisés que très épisodiquement, voire une seule fois, par exemple pendant les tests préliminaires avant mise en exploitation de la machine. Dans ces cas, plutôt que de prévoir une pléthore de modes différents dans un même programme dont la densité serait mal exploitée, on préfère réaliser des programmes spécifiques et les charger dans l'automate aux moments opportuns.