

1. INTRODUCTION : LES SYSTEMES AUTOMATISES DE PRODUCTION

Un Système de Production a pour rôle de conférer une valeur ajoutée à un ensemble de produits et matériaux bruts pour obtenir un produit fini ou un produit intermédiaire qui va, à son tour, participer à l'élaboration d'un produit fini. Le système de production reçoit donc un flux de matières d'oeuvre (brutes), consomme divers types d'énergie (électrique, pneumatique, hydraulique, etc.) et consomme un certain nombre de composants auxiliaires (eau, liquide de refroidissement, huile, etc.). Il génère un flux de produits élaborés et un flux de déchets et de nuisances diverses.

Tout système de production nécessite la présence d'opérateurs humains d'exploitation, de réglage et de maintenance. Automatiser un système de production consiste donc à réduire l'intervention humaine tout au long du processus de fabrication, à optimiser l'utilisation des matières et des énergies par l'emploi de nouvelles technologies, à innover en terme de performances, de qualité. Cette automatisation permet donc globalement de rendre plus compétitif le produit, l'entreprise et les hommes. Nous parlons alors de Systèmes Automatisés de Production (SAP).

Un Système Automatisé de Production est globalement composé d'une Partie Opérative (P.O.) et d'une Partie Commande (P.C.).

1.1. Partie Opérative

La Partie Opérative (P.O.) est formée de l'ensemble des divers organes physiques qui interagissent sur le produit pour lui conférer une valeur ajoutée : les pré-actionneurs, les actionneurs et les capteurs. Les pré-actionneurs servent de relais de puissance entre la commande et les actionneurs qui agissent et transforment le produit. Les capteurs recueillent les informations : état ou position du produit, alarmes, etc. traduisant un changement d'état du procédé. Ces changements induisent le calcul de la commande du procédé de production par son Système de Contrôle/Commande ou Partie Commande.

1.2. Partie Commande

La Partie Commande (P.C.) ou Système de Contrôle/Commande (S.C.C.) est l'ensemble des moyens logiciels et d'informations concernant le pilotage et la conduite du procédé. Notons qu'elle comporte aussi une partie purement matérielle (câblage), imposée par les normes de sécurité.

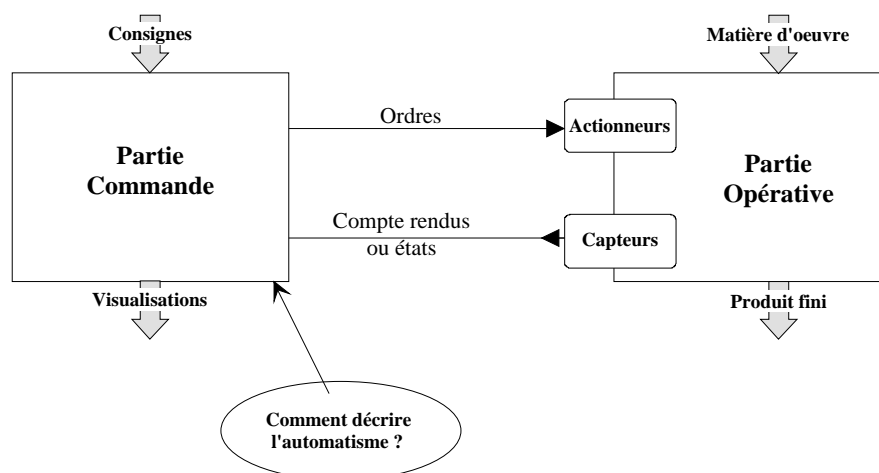


Figure 1 : Découpage fonctionnel PC/PO

Un procédé industriel passe d'un état initial à un état final via une succession d'états intermédiaires qui marquent son évolution. Cette évolution est surveillée grâce aux capteurs qui renvoient des comptes-rendus de l'état du système. La Partie Commande traite ces informations et élabore les ordres renvoyés vers les actionneurs.

La notion de séquence (d'états) et la notion de temps sont donc implicites.

Le problème est donc de décrire l'automatisme qui va gérer et provoquer ces changements d'états, c'est à dire le comportement global du système. On peut alors employer divers outils [10][11]:

- Décrire littéralement le comportement en **langage naturel**, c'est à dire énumérer au moyen du langage 'de tous les jours' ce que l'automatisme doit faire. Malheureusement, ce langage est interprétable de différentes manières et génère des cahier des charges difficiles et ambigus à exploiter lorsque l'on aborde des systèmes complexes comme c'est de plus en plus souvent le cas.

exemple : Comment interprétez-vous 'La porte du four doit rester fermée jusqu'à ce que la température atteigne 60°C seulement si l'alarme ne s'est pas déclenchée et lorsque la pièce est présente' ?

- La description plus symbolique basée sur l'utilisation de variables Tout Ou Rien (TOR), de fonctions logiques (**algèbre de Boole**) combinatoires ou séquentielles.

- La description graphique semble être la plus facile à utiliser puisque la plus facile à appréhender. Pour cela, plusieurs modèles existent comme :

-- le **schéma électrique à contacts** ou les **diagrammes en échelle (Ladder Diagram)** qui utilisent les mêmes principes de représentation mais avec des symboles un peu différents.

exemple:

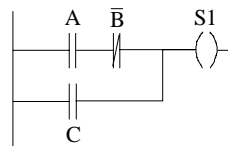


Diagramme en échelle

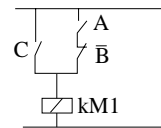
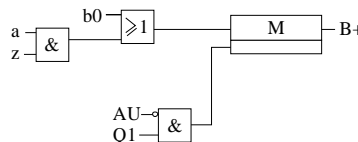


Schéma à contacts

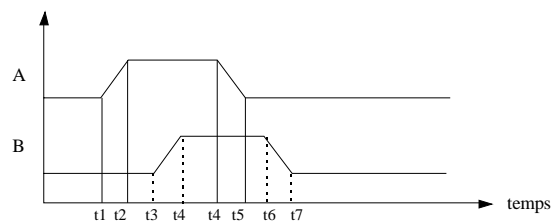
-- le **logigramme** qui est une représentation graphique des fonctions logiques éventuellement complexes.

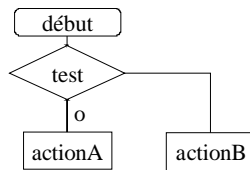
exemple:



-- les **chronogrammes** qui permettent de représenter dans le temps l'évolution des différentes variables utilisées dans les systèmes.

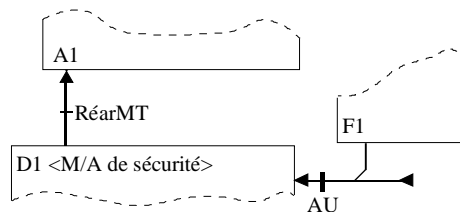
exemple:





-- le **GEMMA** (Guide d'Etude des Modes de Marches et d'Arrêt) propose un vocabulaire de base et un graphisme qui permettent de clarifier les procédures de conduite et le dialogue opérateur relatifs à un SAP sans pour autant décrire réellement le contenu de la commande.

exemple (détails):

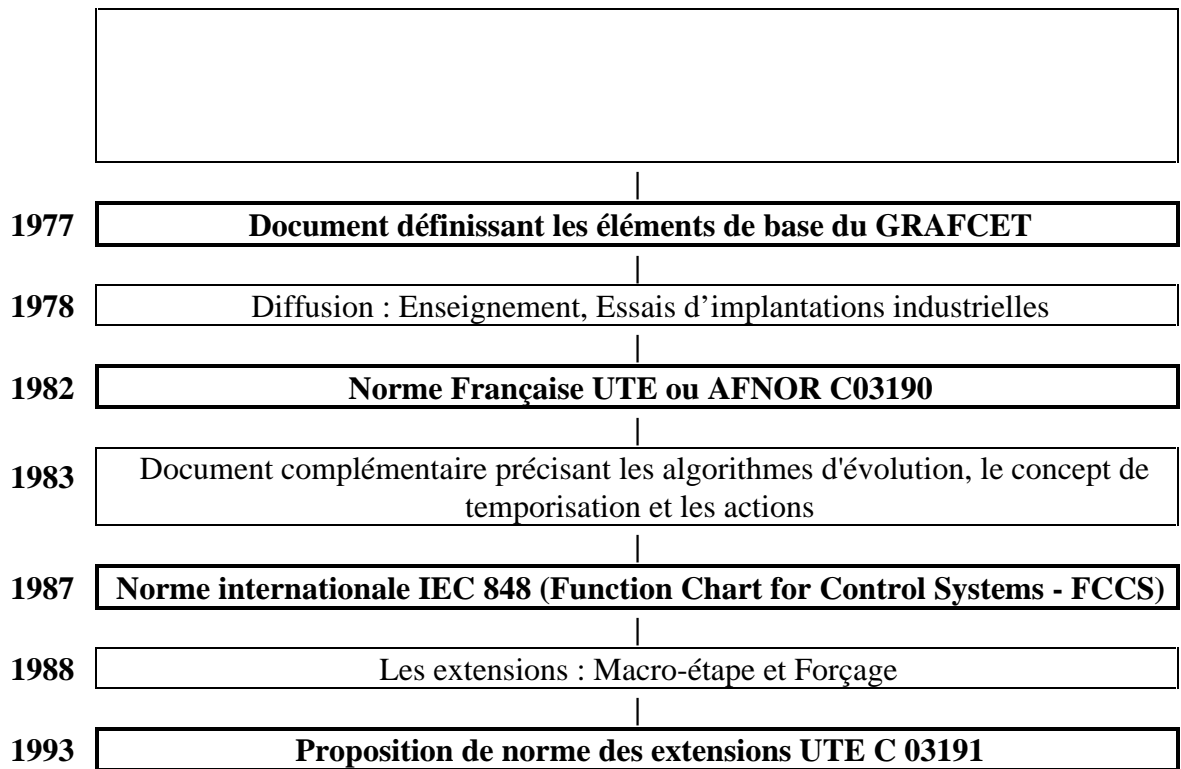


-- le **GRAFCET** (GRAphe Fonctionnel de Commande Etape-Transition), enfin, que nous allons décrire dans la suite de ce document

2. LE GRAFCET

Le Grafcet est la symbiose parfaite des différents éléments vus ci-dessus. C'est un modèle permettant de spécifier et concevoir un automatisme séquentiel. C'est un outil graphique, puissant, directement exploitable par la plupart des Automates Programmables Industriels (API) existants actuellement sur le marché. On voit ici l'avantage de disposer d'un modèle de spécification qui possède, de plus, des possibilités d'implantation technologique, c'est à dire de passer très rapidement d'une spécification à une implantation exécutable. Le, ou les, Grafcet de commande sont alors matérialisés directement sur une machine ou sur un ensemble de supports (câblage, séquenceurs, etc.).

Le tableau suivant [6] montre les différentes phases de l'évolution du GRAFCET.



Notons enfin l'existence d'un Groupe de Recherche, le GR-GRAFCET comprenant des industriels et des universitaires, qui, au sein de l'AFCEC, continue à travailler sur le GRAFCET et sur ses évolutions futures.

Un GRAFCET est un graphe (diagramme) constitué par une alternance d'étapes et de transitions reliées par des arcs orientés.

3.1. Eléments de base

Ce sont :

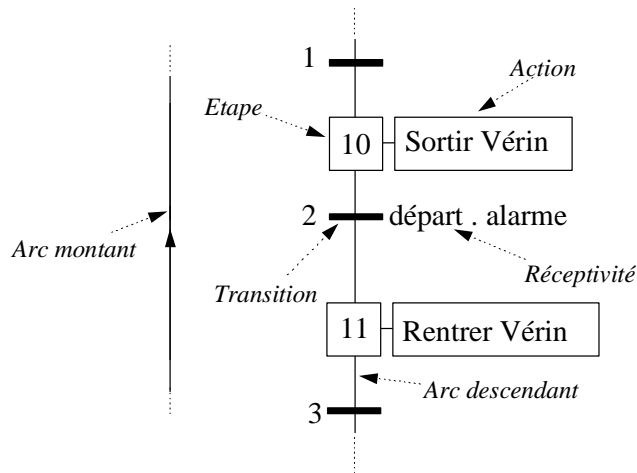


Figure 2 : Eléments de base du GRAFCET

3.1.1. Etapes

L'étape correspond à une situation dans laquelle le comportement de tout ou partie du système par rapport à ses entrées et ses sorties est invariant.

A chaque étape est associée une action, c'est à dire un ordre vers la partie opérative ou des communications vers d'autres graphes.

L'étape possède deux états possibles: active ou inactive. Si elle est active, l'action associée peut être exécutée. L'étape, repérée numériquement, possède ainsi une variable appelée variable d'état d'étape et notée X_i où i est le numéro de l'étape. Cette variable est une variable booléenne valant 1 si l'étape n^o est active, 0 sinon.

ACTIVE		L'action s'exécute et le piston du vérin sort X_{10} est la variable d'état associée à l'étape 10 et qui vaut 1
INACTIVE		L'action ne s'exécute pas X_{10} vaut 0

Un étape peut être qualifiée d'étape initiale ou d'étape normale selon qu'elle est active ou non à l'instant initial. Le formalisme de représentation est alors différent.

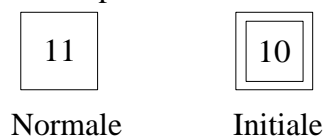


Figure 3 : Etapes

$A+B$
 Départ . Alarme
 Auto . (VerinSorti + PiècePrésente)
 = 1 *indiquant une réceptivité toujours vraie*
 X11
 t / X20 / 5 s

Figure 4 : Exemples de réceptivités

Les changements d'état du GRAFCET se font donc par franchissement des transitions à la suite de certains événements dans la Partie Opérative ou dans la Partie Commande.

Par exemple, un vérin doit, en l'absence de signal d'alarme, si une condition particulière notée E est vraie et lorsque l'on appuie sur un bouton poussoir Départ, pousser une pièce vers un centre d'usinage. Il est piloté par une vanne V comme le montre la figure suivante. Il dispose de deux capteurs de fin de course tout ou rien (TOR) FC0 (qui renvoie 1 lorsque le vérin est rentré) et FC1 (qui renvoie 1 lorsque le vérin est sorti).

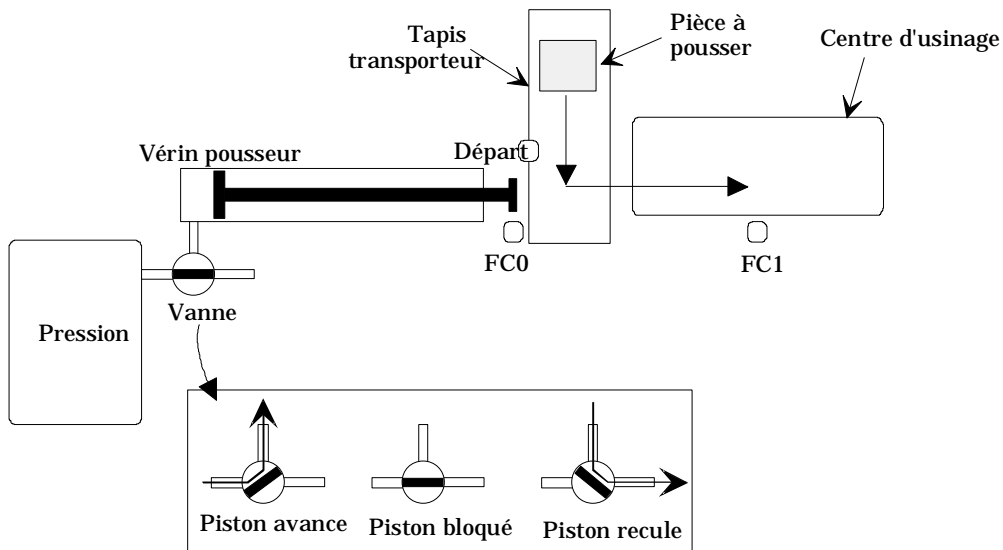


Figure 5 : Exemple : pilotage d'un vérin

Le graphe de commande de la vanne se présente comme suit :

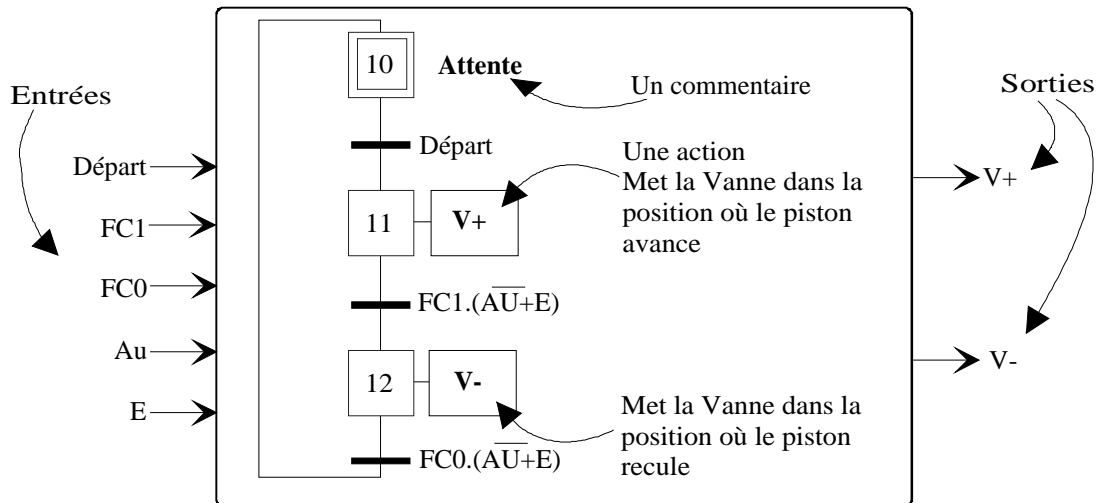


Figure 6 : Graphe de commande du vérin

Cinq entrées proviennent des capteurs FC0 et FC1 (ce sont des variables booléennes valant 0 lorsque le capteur ne détecte rien et 1 dans le cas contraire), d'état du bouton poussoir Départ et de signaux booléens AU et E.

Les états d'étapes Xi marquent les phases d'activation/désactivation des étapes en fonction de ces entrées lors d'un cycle aller/retour du vérin comme le montre le chronogramme suivant :

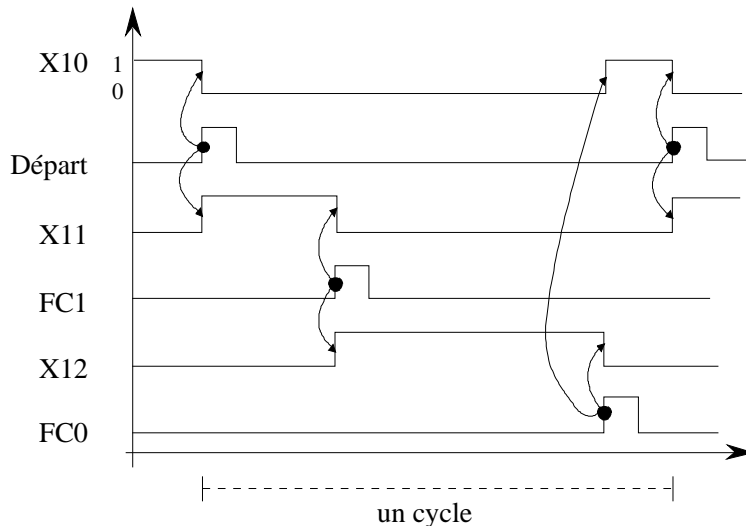


Figure 7 : Chronogramme d'un cycle ALLER/RETOUR du vérin

3.1.3. Liaisons orientées

Une arc lie toujours une étape à une transition ou une transition à une étape. Il respecte cependant un certain formalisme lors des alternances particulières décrites page suivante. Ces alternances sont des structures particulières du GRAFCET que nous décrivons plus loin.

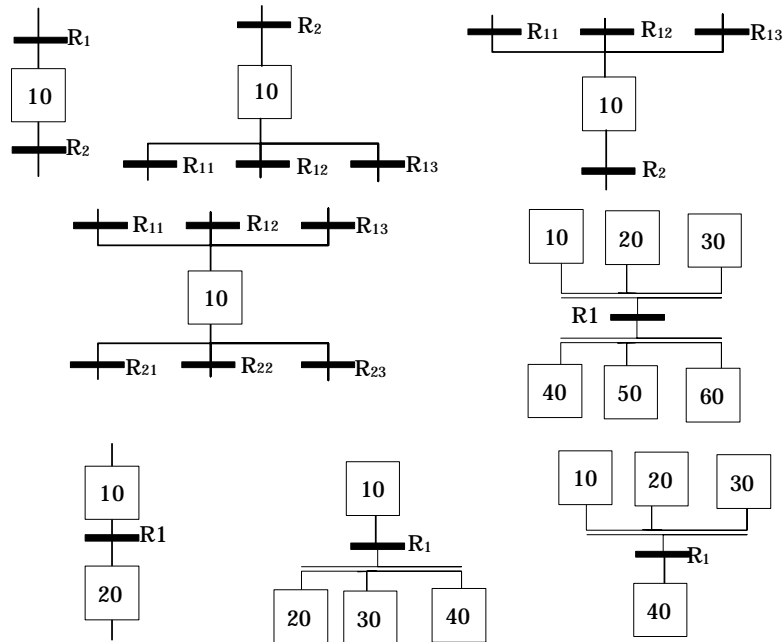


Figure 8 : Liaisons orientées

L'exemple suivant montre des constructions de graphe où l'on met en évidence les éléments amont et aval des transitions et des étapes.

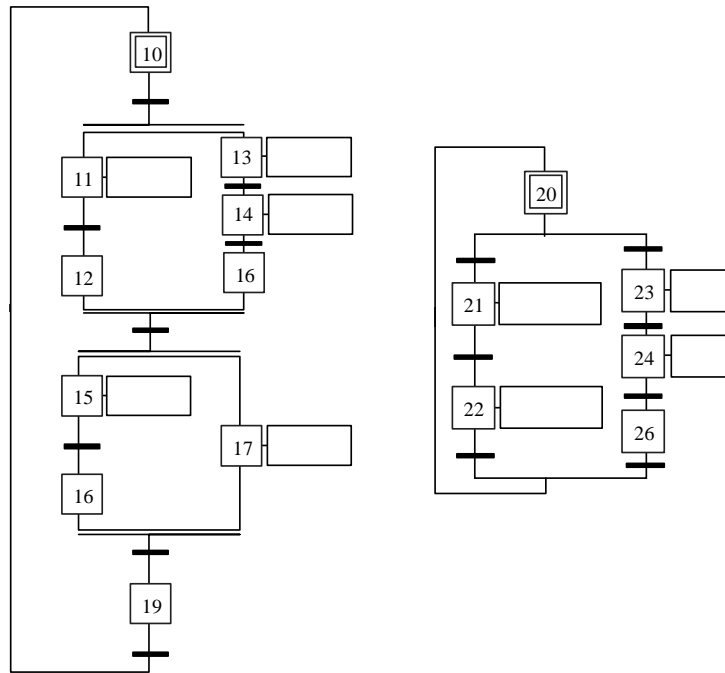


Figure 9 : Exemples

Présentation du GRAFCET

3.2. Les réceptivités

Une réceptivité est associée à chaque transition. C'est une fonction booléenne calculée à partir des entrées du graphe qui peuvent être des entrées booléennes, des états d'étape ou des temporisations.

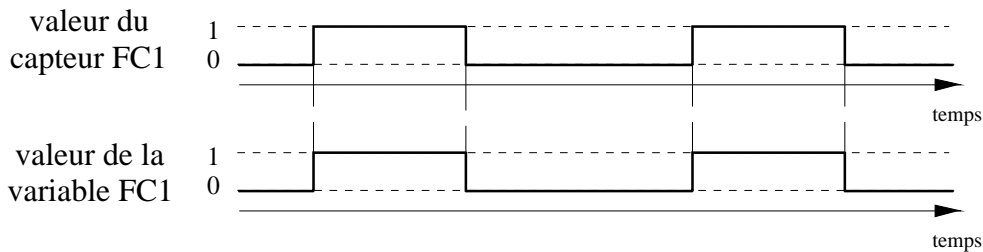


Figure 10 : Variable à niveau

Une réceptivité peut alors être écrite comme une fonction booléenne utilisant les opérateurs ET, OU, NON et front. Le front montant ou le front descendant d'une variable booléenne permettent de situer dans le temps le changement de valeur de ce capteur. On les note respectivement $\uparrow E$ et $\downarrow E$.

Les fronts ne sont à 1 que durant un délai d qui correspond au temps de prise en compte de l'événement, c'est à dire le temps de franchissement d'une transition.

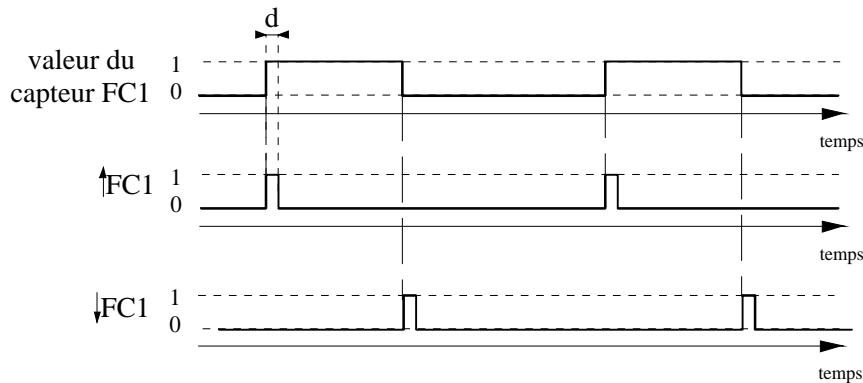


Figure 11 : Variable à front

3.2.2. Variable d'état

Ce sont les variables booléennes X_i qui valent 1 dès que l'étape n^o_i associée est active et 0 si cette étape n'est pas active. Ces variables sont, d'une part, utilisées dans le calcul des temporisations, d'autre part, pour établir une communication simple entre des graphes.

Dans l'exemple page suivante, le graphe G_1 modélise le comportement global du système face à un défaut. En l'absence de défaut, l'étape 10 est active. Lorsqu'un défaut est constaté, l'entrée booléenne défautsConstatés passe à 1 et la transition devient franchissable. L'étape 11 devient active et permet de franchir la transition d'un autre graphe G_2 qui décrit une partie du comportement du système et de rendre l'étape 22 active. L'étape 11 mémorise le défaut tant que la maintenance n'est pas intervenue. Le comportement de G_2 dépend donc de l'état de G_1 par l'intermédiaire de la variable d'état X_{11} . G_1 et G_2 sont respectivement des grafjets partiels composant le grafjet global (G_1, G_2).

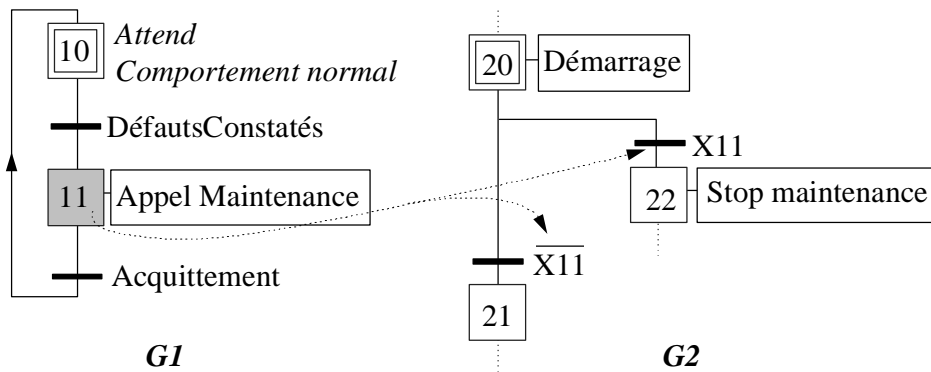


Figure 12 : Etat d'étape utilisé entre deux graphes communicants

3.2.3. Temporisations

Une variable temporisation S_n est une variable booléenne dont le mode d'évaluation permet de prendre en compte le temps. Elle peut s'écrire, de manière générale [3], par $S_n = t_1 / En / t_2$ où :

- En désigne une variable d'entrée.
- t_1 désigne le retard apporté au changement de l'état logique 0 vers l'état logique 1 de la variable d'entrée En.
- t_2 désigne le retard apporté au changement de l'état logique 1 vers l'état logique 0 de la variable d'entrée En.

L'évolution de S_n en fonction de ces caractéristiques est illustrée figure 13.

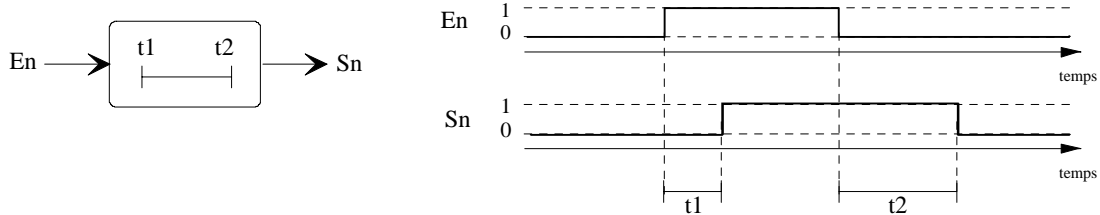


Figure 13 : Définition de la variable temporisation

La variable En doit être présente pendant un temps supérieur ou égal à t_1 pour que S_n puisse être évaluée. Les délais t_1 et t_2 peuvent être nuls. Dans ce cas, on préférera écrire respectivement En/t_2 ou t_1/En . Enfin t_1 et t_2 sont remplacés par des valeurs exprimées dans une unité de temps donnée.

L'utilisation et la notation les plus courantes consistent à prendre En égale à une variable d'état, par exemple X_m , associée à l'étape n^o m et à choisir $t_2 = 0$. La variable S_n est alors notée $t/X_m/t_1$ où t_1 signifie la durée de la temporisation, c'est à dire le retard mis par S_n pour passer à 1 après que X_m soit passée à 1.

L'exemple, décrit dans la figure suivante, montre l'évolution d'une variable temporisation ainsi notée. La variable $t/X_{20}/3s$ vaut 0 à l'instant initial, 0 dès que l'étape 20 devient active et passe à 1 après un délai de 3 s qui suit la dernière activation de l'étape 20. Elle repasse à 0 dès la prochaine ré-activation de l'étape 20

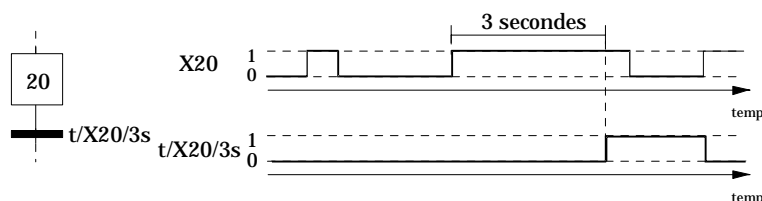


Figure 14 : Utilisation de la variable temporisation

Vous désirez augmenter la température d'une pièce de votre maison. Pour passer de la température actuelle à la température voulue de 20°C et avec un radiateur d'une puissance donnée, il faut alimenter ce radiateur durant 1 h. Cette attente peut se modéliser de la manière suivante :

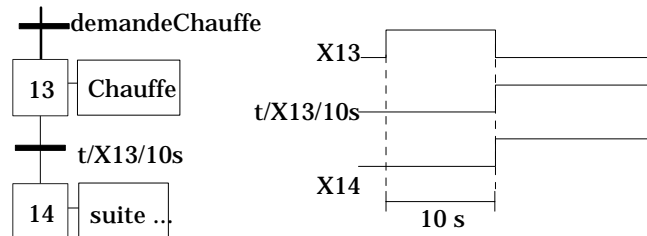


Figure 15 : Attente temporisée

3.3. Les Actions (ou ordres)

Le GRAFCET possède trois types d'actions:

- Les actions continues
- Les actions conditionnelles qui peuvent elles-mêmes être décomposées en trois cas particuliers:
 - Action conditionnelle simple
 - Action retardée
 - Action limitée dans le temps
- Les actions mémorisées

Elles peuvent être implantées systématiquement puisque reconnues dans la norme IEC 848 qui en donne le formalisme suivant :

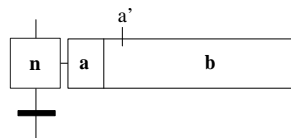


Figure 16 : Action

Plusieurs actions simultanées peuvent être associées à une même étape.

3.3.1. Continues

Symbole	<div style="border: 1px solid black; padding: 2px; display: inline-block;"> n Ordre A </div>
Diagramme Temporel	
Equation Booléenne	$A = Xn$

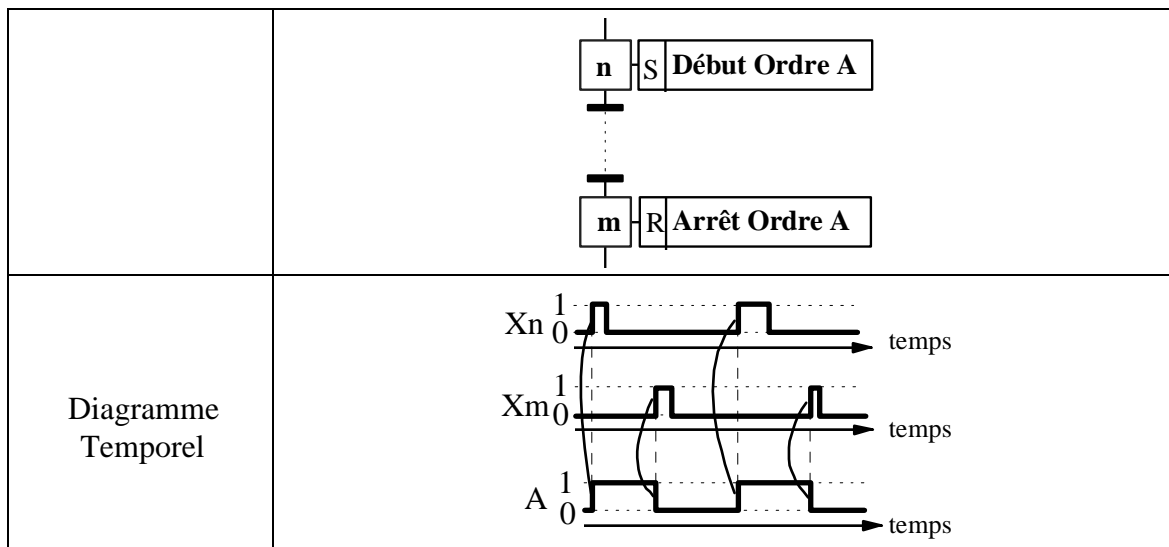
Diagramme Temporel	
Equation Booléenne	$A = \text{Cond} \cdot Xn$

Cas particulier: l'ordre A est Retardé :

Symbole	
Diagramme Temporel	
Equation Booléenne	$A = t/Xn/d \cdot Xn$

Cas particulier: l'ordre A est limité dans le temps :

Symbole	
Diagramme Temporel	
Equation Booléenne	$A = \overline{t/Xn/L} \cdot Xn$



3.3.4. Equation logique

On peut ainsi donner l'équation générale d'un ordre A en fonction des états des étapes, des conditions éventuelles et du temps.

Exemple :

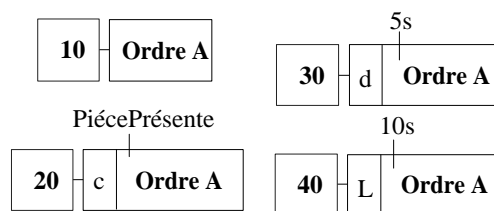


Figure 17 : Exemple d'équation logique d'un ordre A

L'équation de sortie est alors :

$$A = X10 + X20.PiècePrésente + X30 . t/X30/5s + X40 . \overline{t/X40/10s}$$

3.4. Règles d'évolution du GRAFCET

Le GRAFCET possède cinq règles d'évolution précisant les causes et les effets du franchissement des transitions.

3.4.1. Règle 1 Condition initiale

A l'instant initial, seules les étapes initiales sont actives.

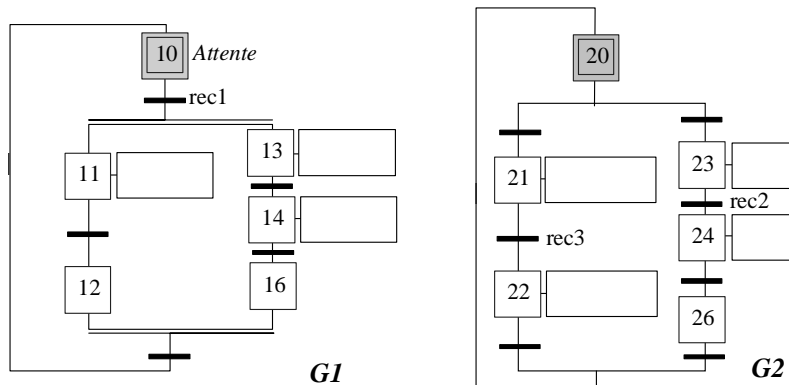


Figure 18 : Règle 1

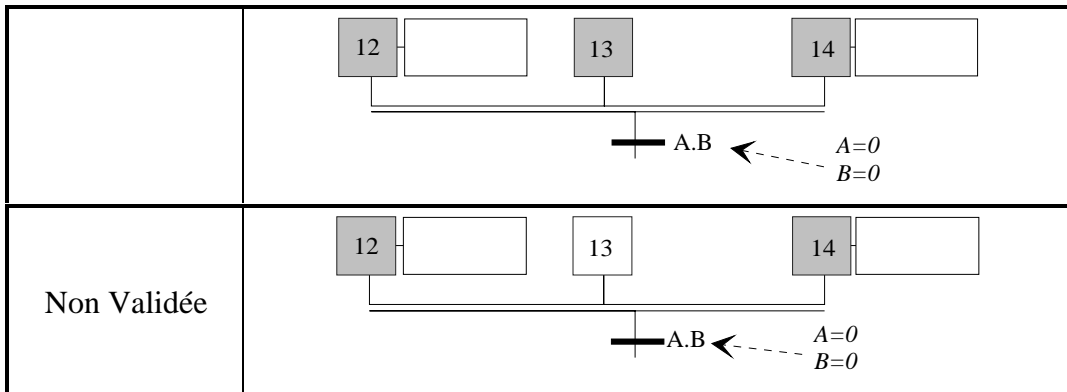


Figure 19 : règle 2

3.4.3. Règle 3 : Condition de franchissement

Une transition n'est franchissable que si elle est validée, et seulement si la réceptivité associée est vraie.

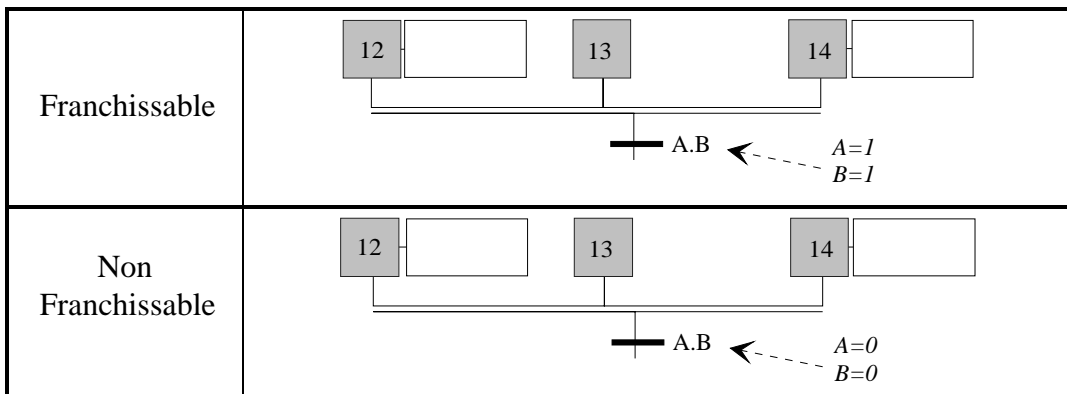


Figure 20 : Règle 3

3.4.4. Effets du franchissement

Pour franchir une transition, il faut, obligatoirement dans cet ordre, désactiver toutes ses étapes amont puis activer ses étapes aval.

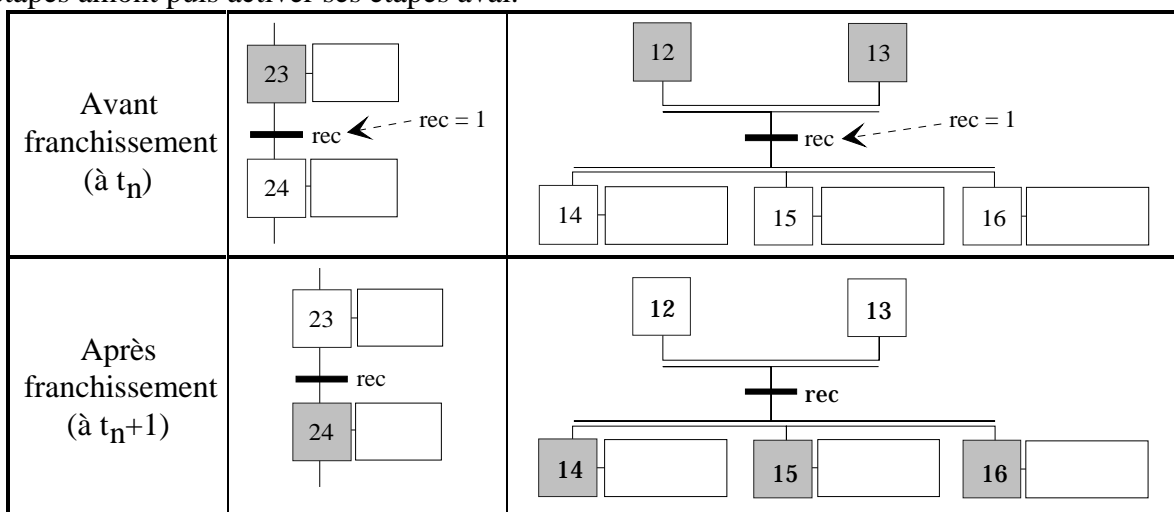


Figure 21 : Effets du franchissement

3.4.5. Règle 4 : Franchissement simultané

Toutes les transitions franchissables à un instant donné, sont franchies simultanément.

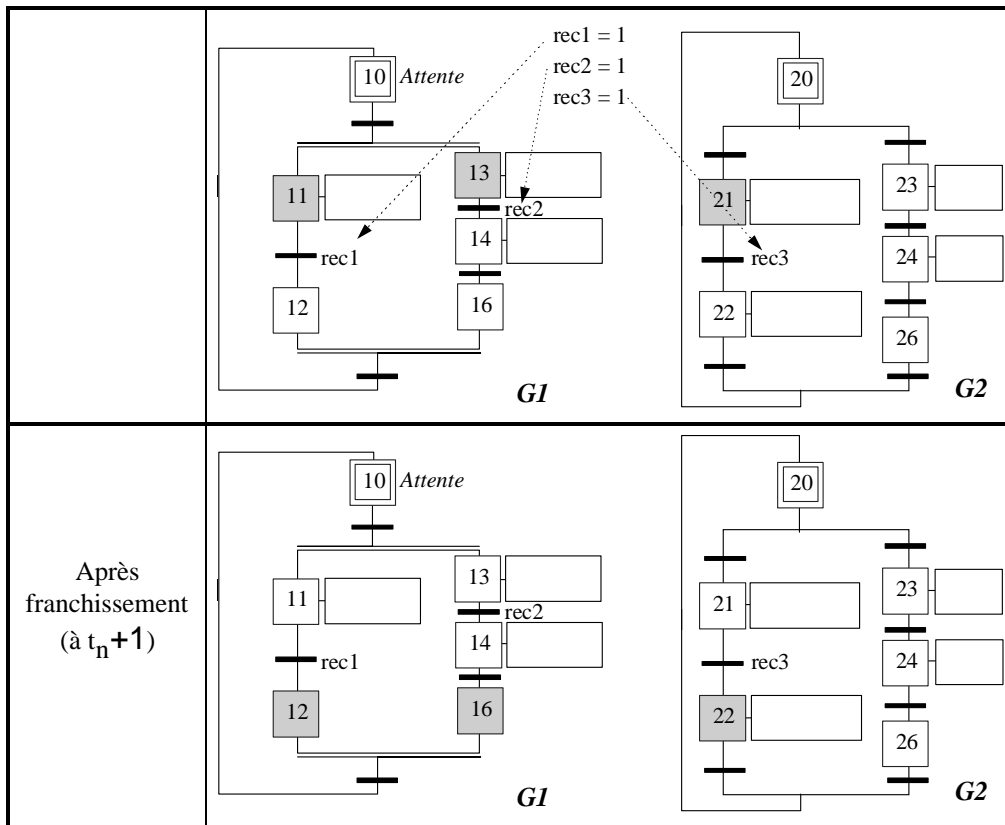


Figure 22 : Règle 4

3.4.6. Règle 5 : Conflit d'activation

Si une étape doit être simultanément désactivée par le franchissement d'une transition aval, et activée par le franchissement d'une transition amont, alors elle reste active.

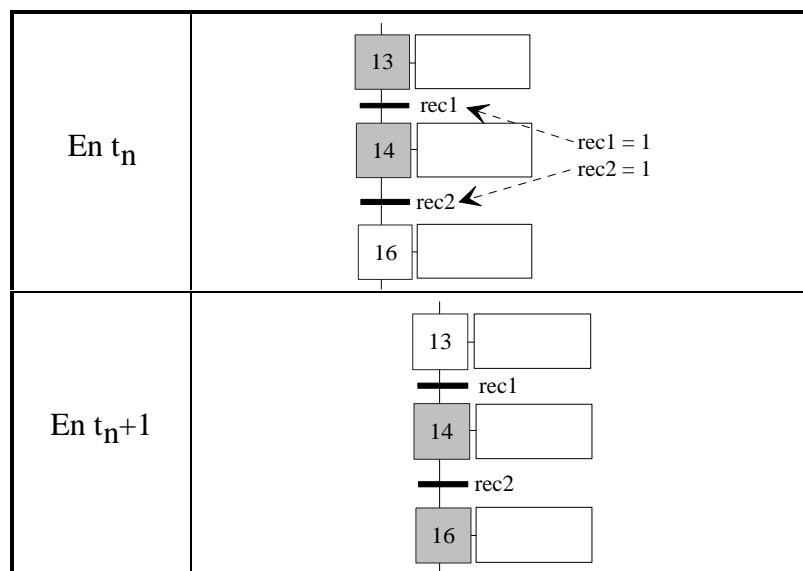


Figure 23 : Règle 5

3.5. Structures de base

Un certain nombre de constructions usuelles du GRAFCET sont aussi appelées structures de base.

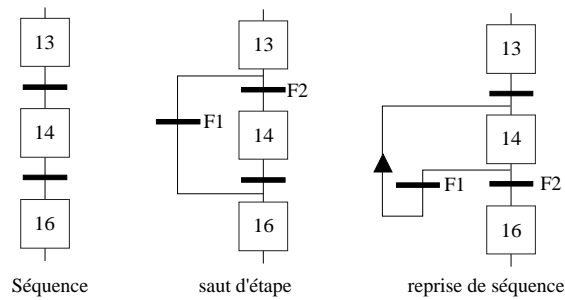


Figure 24 : Séquence, saut et reprise de séquence

3.5.2. Choix (aiguillage)

Le choix indique la possibilité de passer de l'activité d'une étape en amont à celle d'une étape parmi plusieurs en aval.

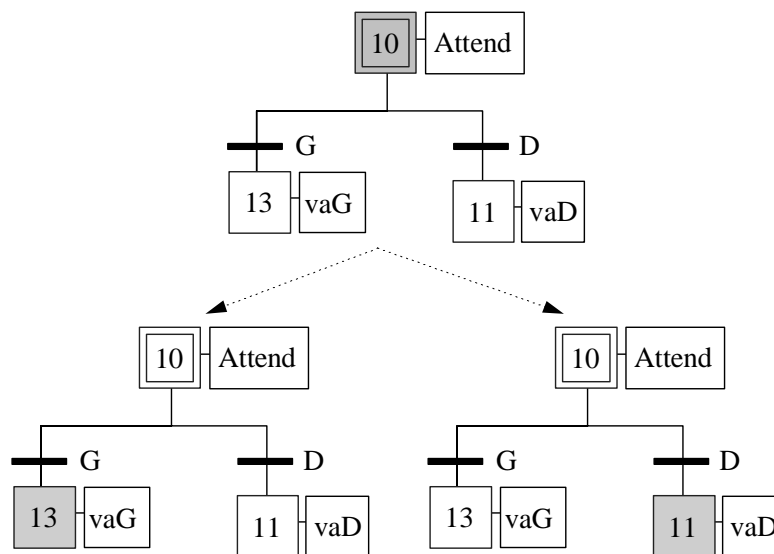


Figure 25 : choix

Dans l'exemple ci-dessus, on peut activer soit l'étape 13, soit l'étape 11, si respectivement la réceptivité D ou G est vraie. Attention, si D et G sont vraies simultanément alors nous nous trouvons dans le cas d'un **parallélisme interprété**. Les deux étapes 13 et 11 vont devenir actives simultanément et donc provoquer l'exécution d'actions en parallèle, situation non voulue par le concepteur. D et G doivent donc être exclusives : une ou l'autre sera vraie mais pas les deux à la fois.

Exemple de modification à apporter pour assurer cette exclusivité :

$$D = \text{ChariotAlimenté} \cdot \bar{G}$$

$$G = \text{ChariotAlimenté}$$

3.5.3. Convergence

La convergence est le dual du choix. Deux activités doivent se terminer de la même manière alors elles sont réunies à nouveau.

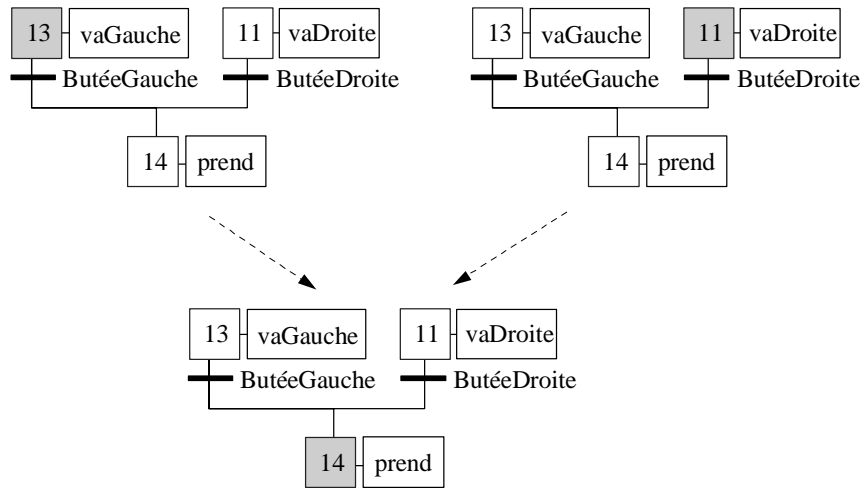


Figure 26 : Convergence

3.5.4. Parallélisme

Au contraire du choix où une seule activité à la fois peut se dérouler, il peut y avoir plusieurs activités indépendantes pouvant se dérouler en parallèle. On parle alors de **parallélisme structurel**

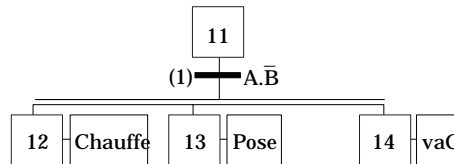


Figure 27 : Parallélisme

Après franchissement de R, nous obtenons ainsi trois activités simultanées ou parallèles.

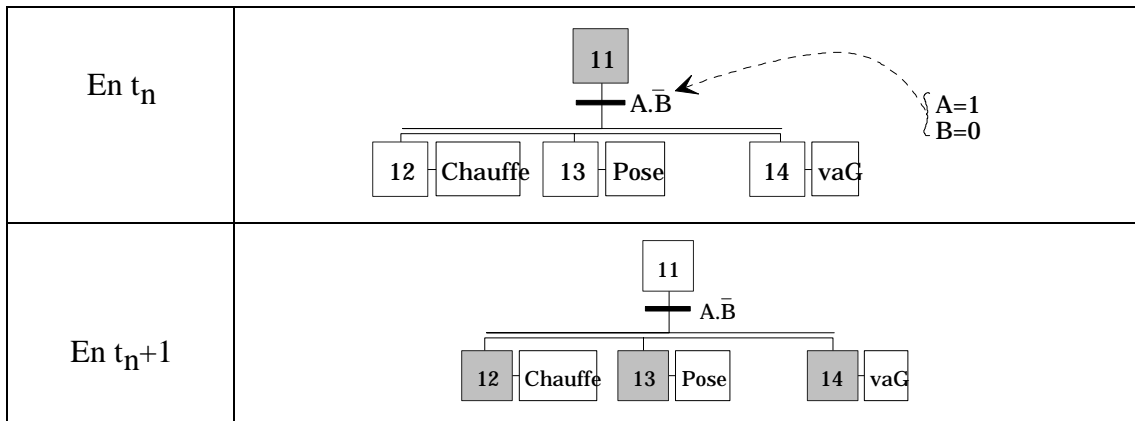


Figure 28 : Evolution du parallélisme

3.5.5. Synchronisation

La synchronisation permet d'attendre la fin plusieurs activités se déroulant en parallèle pour continuer par une seule. On parle alors de synchronisation puisque, par exemple, l'activité P1 qui se termine avant P2 et P3 doit les attendre et se synchroniser avec elles.

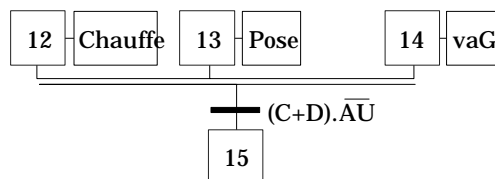


Figure 29 : Synchronisation

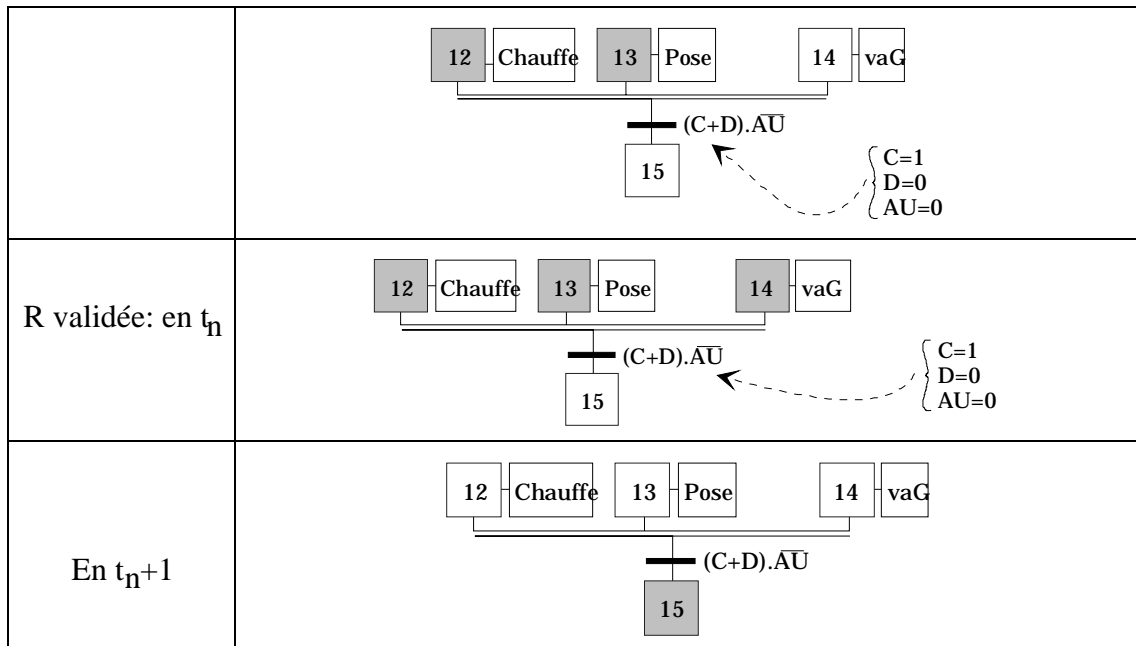


Figure 30 : Evolution de la synchronisation

3.5.6. Le sémaphore d'exclusion mutuelle

Le sémaphore d'exclusion mutuelle, décrit figure 31, est un mécanisme basé sur l'emploi d'une synchronisation puis d'un parallélisme et d'une étape initiale. Il permet de rendre deux activités réellement séquentielles ou encore protège deux activités d'une exécution simultanée parcequ'elles ne doivent en aucun cas être parallèles.

Par exemple, une perceuse ne peut percer qu'une pièce à la fois, un chariot ne peut pas en même temps aller d'un point A vers un point B et d'un point C vers un point D. La perceuse et le chariot sont alors appelés **ressources critiques** car elles ne peuvent pas participer à plusieurs activités à la fois.

Par contre, l'ordre d'exécution des activités (A1 avant A2 ou le contraire) importe peu. C'est la première activité sollicitée qui sera exécutée la première si la ressource est libre. L'autre devra attendre que cette ressource critique se libère pour continuer.

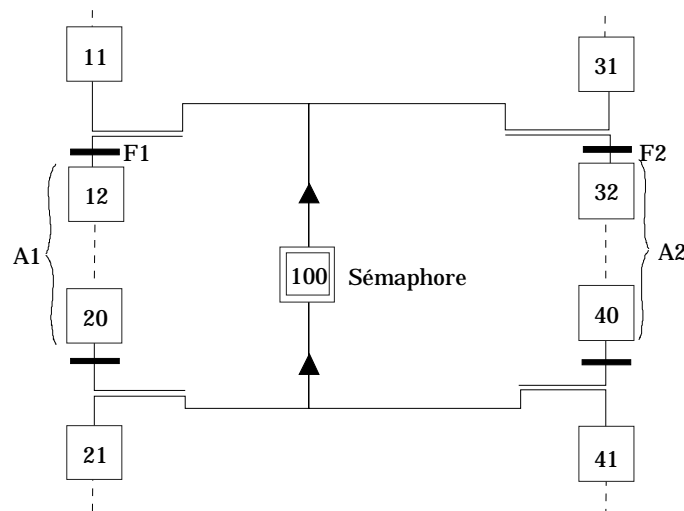


Figure 31 : Sémaphore

Il fonctionne comme le montre la figure suivante :

1) les deux étapes 11 et 31 sont actives ce qui signifie deux demandes simultanées de réservation de la ressource pour exécuter soit l'activité de gauche soit l'activité de droite

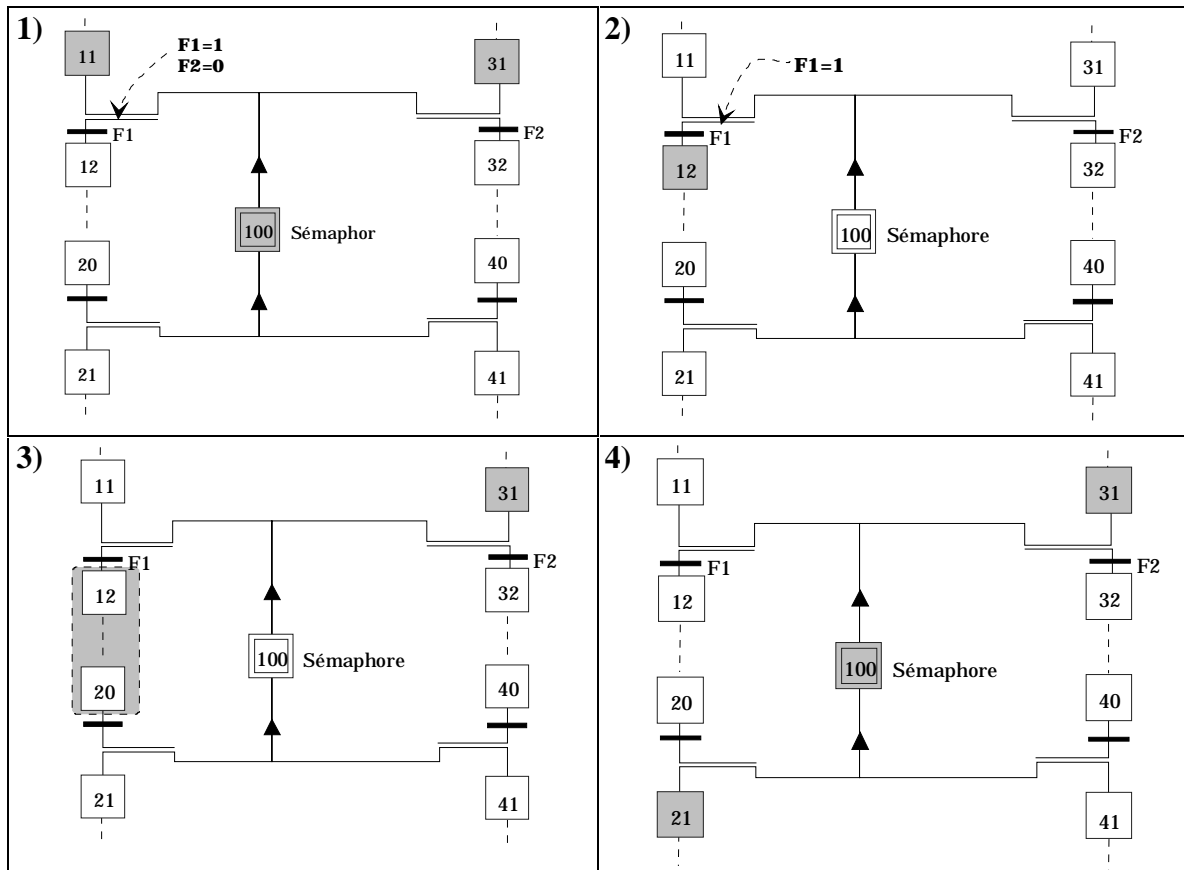
2) F1 est choisie judicieusement afin d'être plus prioritaire que F2. La transition portant F1 est donc franchie ce qui provoque l'activation de l'étape 12 et la désactivation des étapes 100 et 11

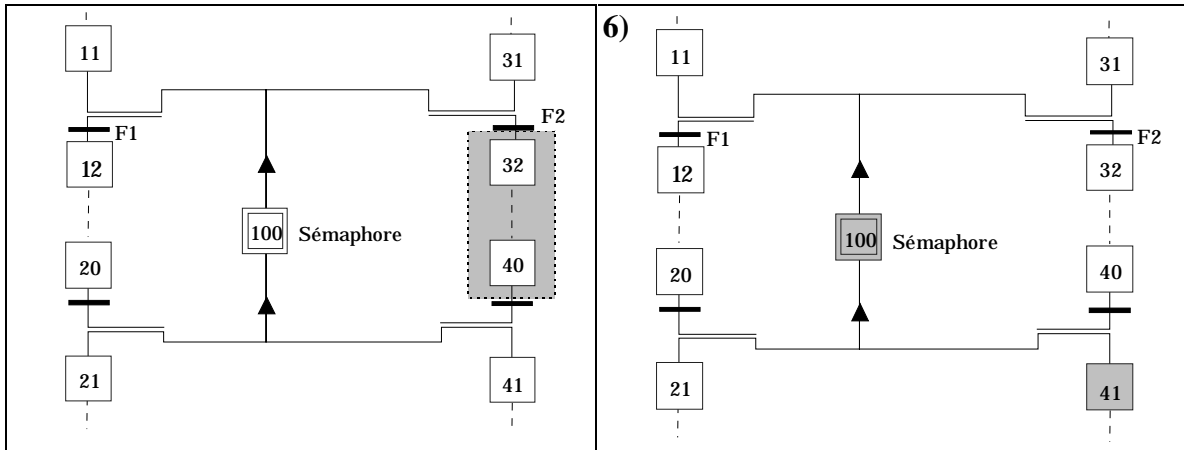
3) L'activité de gauche est exécutée et la ressource lui est attribuée

4) L'activité de gauche vient de se terminer ce qui provoque, entre autre, la réactivation de l'étape 100 : la ressource est donc libre à nouveau pour exécuter maintenant l'activité de droite. Si l'étape 11 redevient active avant que cette ressource ne soit occupée alors revenir en 1.

5) L'activité de droite est exécutée et la ressource lui est attribuée. Si l'étape 11 redevient active pendant ce temps, elle reste active jusqu'à ce que la ressource soit libérée.

6) L'activité de droite vient de se terminer ce qui provoque, entre autre, la réactivation de l'étape 100 : le ressource est donc libérée à nouveau.





*Remarque : **attention**, il faut absolument que F1 et F2 soient **exclusives** sinon on ne peut pas garantir l'exclusion mutuelle des processus A1 et A2.*

Nous pouvons aussi trouver des structures plus complexes permettant de mettre en exclusion mutuelle plusieurs activités. Une seule des réceptivités F1, F2, ..., Fn doit être vraie à la fois.

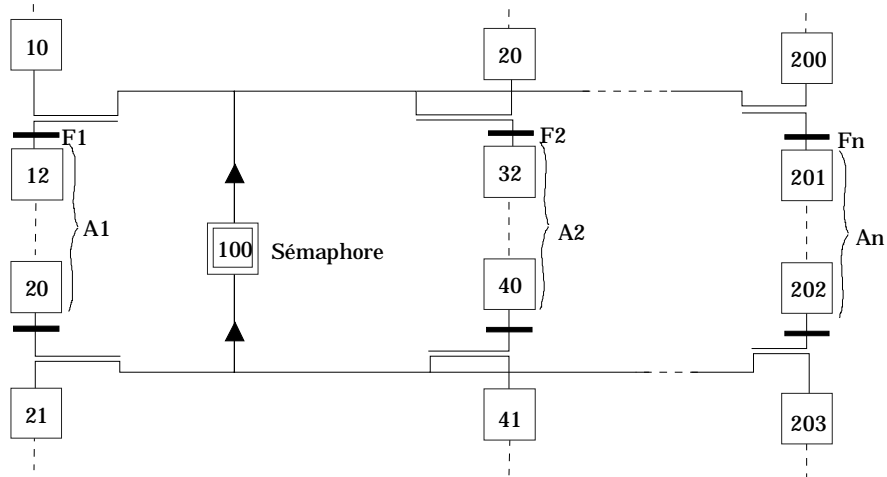


Figure 32 : Sémaphore Multiple

Comme pour le sémaphore simple, on peut assurer la priorité à l'activité la plus à gauche par le choix judicieux des réceptivités comme suit :

$$\begin{cases} F1 = 1 \\ F2 = \overline{X10} \\ F3 = \overline{X20} \cdot \overline{X10} \\ \dots \\ Fn = \overline{X\dots} \dots \overline{X10} \end{cases}$$

3.5.7. Communication

Deux processus P1 et P2 désirent dialoguer. Pour cela, il faut respecter certaines règles. Supposons que P1 soit l'émetteur du message et P2 le receveur.

P2 doit donc attendre le message envoyé par P1. Il traite ce message et produit une réponse attendue par P1. C'est donc une suite de parallélismes/synchronisations qui sont mis ici en oeuvre.

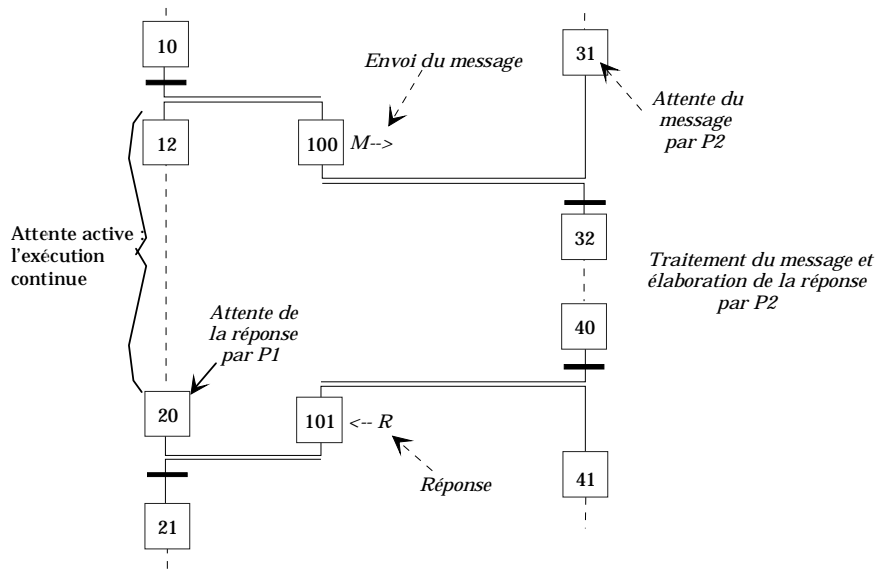


Figure 33 : Communication

Durant les phases d'attente de messages, chacun des deux processus peut avoir une attente active, c'est à dire continuer une activité.

3.6. L'algorithme d'évolution

L'algorithme d'évolution a pour but de calculer la situation future d'un graphe en fonction de la situation courante et des entrées de celui-ci à cet instant.

La situation courante (ou plus simplement la situation) d'un GRAFCET à un instant donné est la liste de ses étapes actives. La situation initiale correspond ainsi à la liste des étapes initiales du graphe. La situation future d'un graphe est la liste des étapes qui seront activées au prochain cycle.

Exemple :

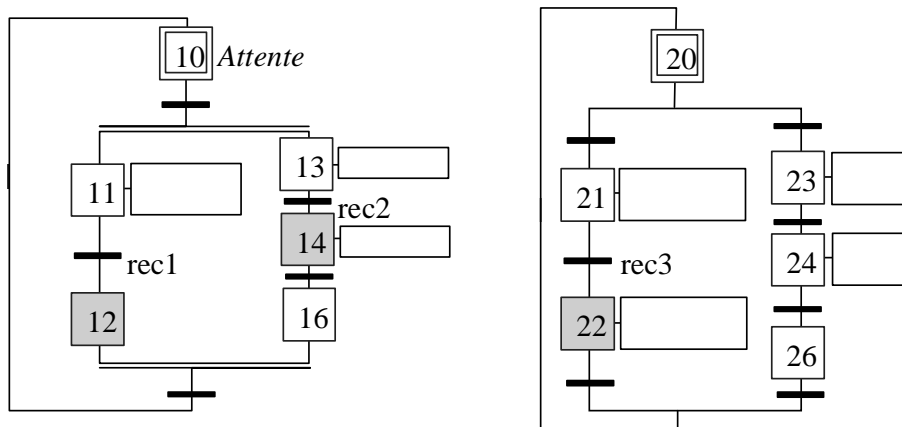


Figure 34 : Situation d'un graphe ou d'un ensemble de graphes

La situation du graphe G1 est {12,14}, celle de G2 est {22}. La situation initiale de G2 est {20} et la situation de G1+G2 est {12,14,22}.

Un algorithme possible du GRAFCET est donné dans l'algorithme simplifié suivant :

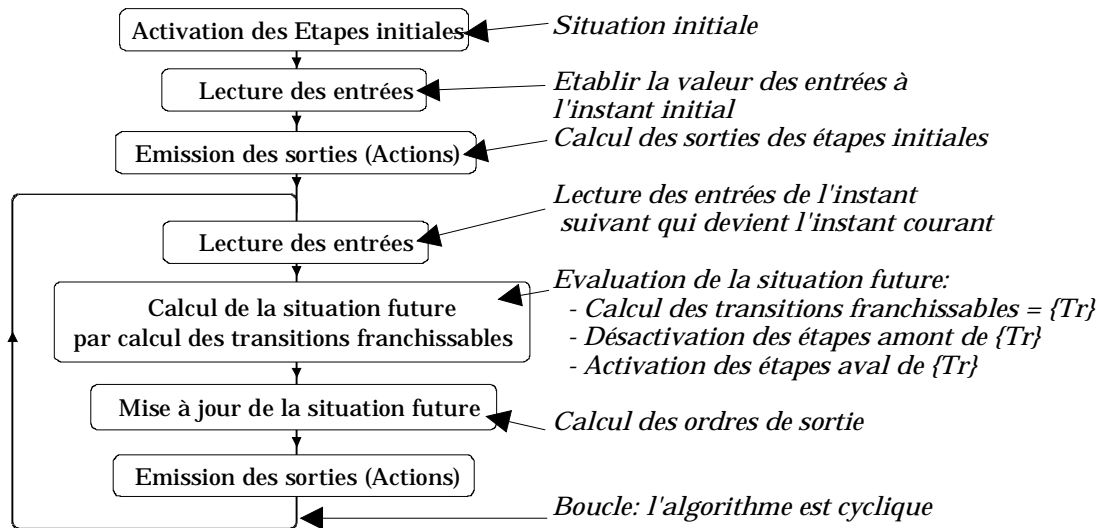


Figure 35 : Algorithme d'évolution possible simplifié

Afin de mieux comprendre cet algorithme, il est possible de représenter le modèle GRAFCET par un modèle générique équivalent représenté par le schéma bloc [4] figure36.

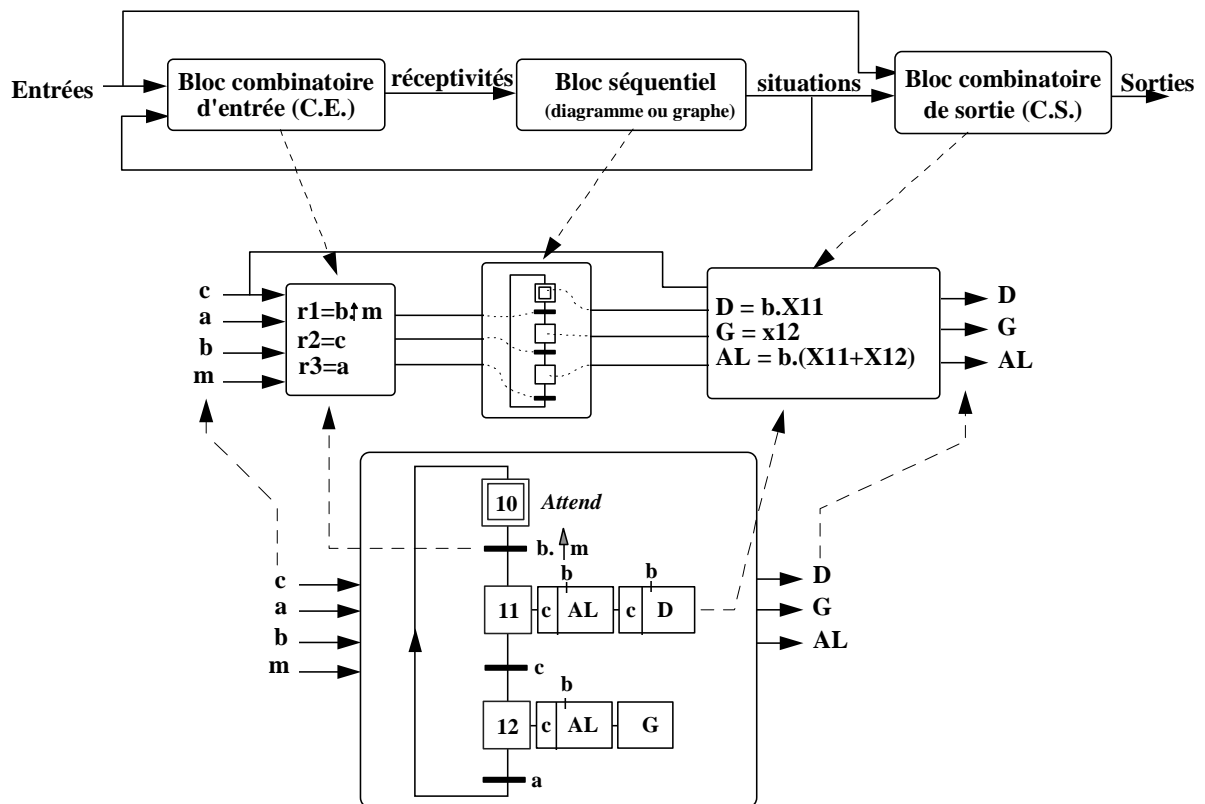


Figure 36 : Schéma bloc Schéma bloc du modèle GRAFCET

La macro-étape permet de structurer un grafcet complexe en plusieurs grafcets partiels comme le montre la figure 37. Chacun de ces grafcets, appelés expansion de macro-étape, décrit un aspect précis ou une fonction donnée du système. C'est à dire que la macro-étape permet une représentation plus claire et mieux structurée d'un grafcet initialement très étendu et complexe à lire.

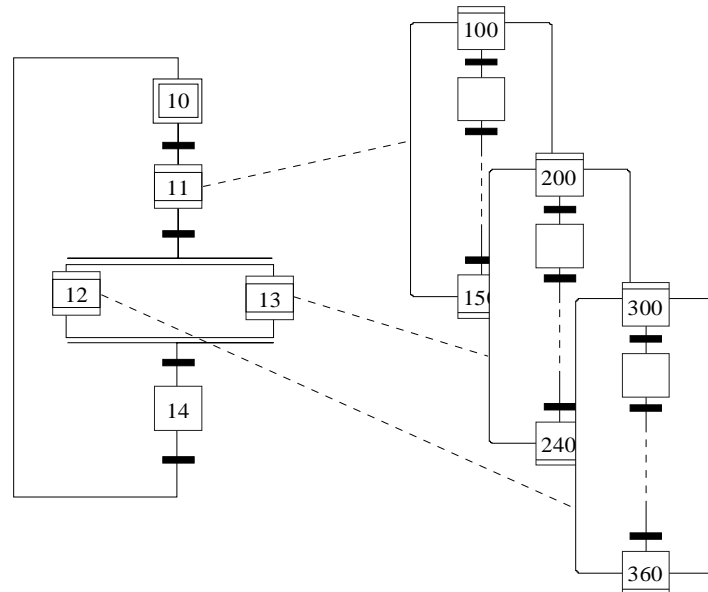


Figure 37 : Exemple de structuration employant des Macro-Étapes

Une macro-étape ME est ainsi une étape particulière (figure 38) qui se détaille dans son grafcet d'expansion. Un GRAFCET d'expansion débute par une macro-étape d'entrée et se termine par une macro-étape de sortie. Il peut contenir, entre ces deux étapes spécialisées une ou plusieurs étapes initiales, des séquences, des reprises, etc. conformément à l'usage du GRAFCET.

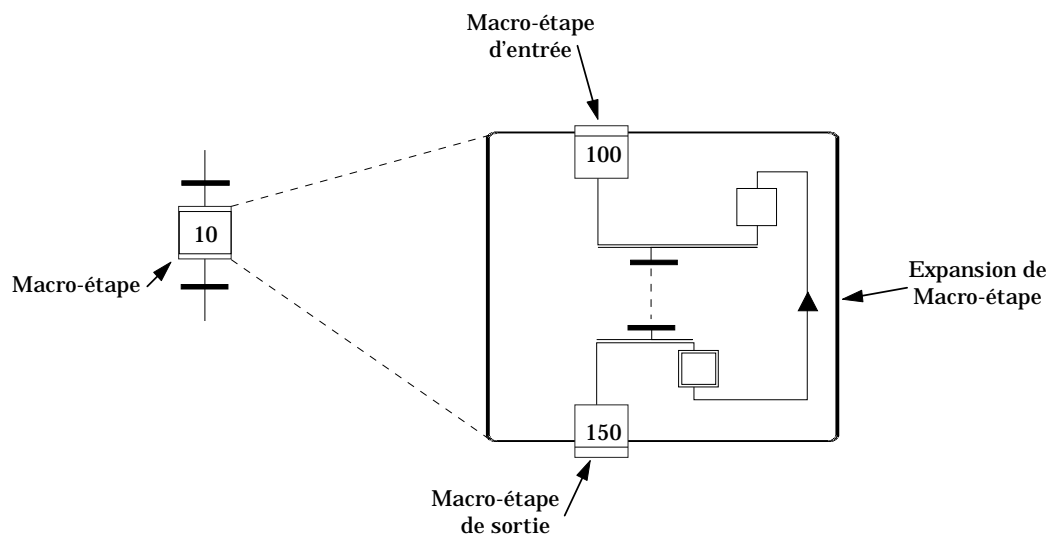


Figure 38 : Macro-étape

L'activation de la macro-étape ME entraîne l'activation de la macro-étape d'entrée de son expansion. ME est dite active si au moins une des étapes de l'expansion est active. La transition aval de ME n'est validée que lorsque l'étape de sortie de l'expansion est active.