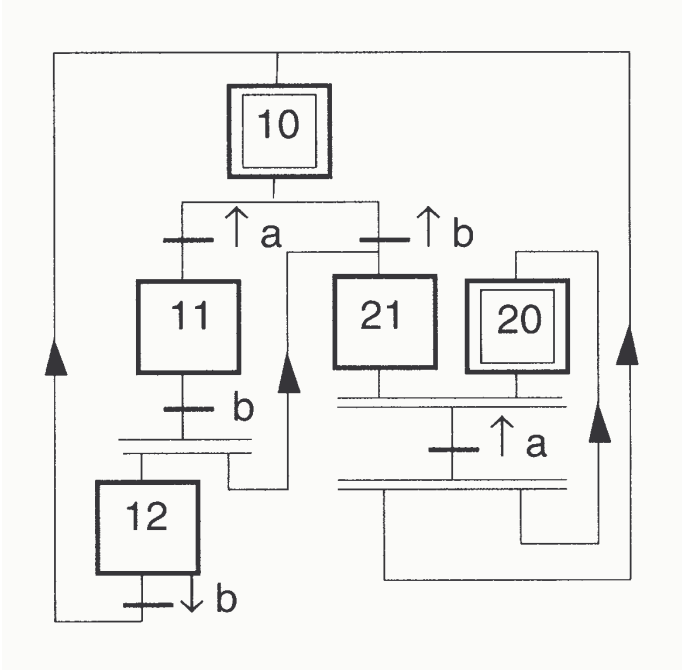
Lycée Vaucanson
Tours



Automatique – Le modèle GRAFCET

AUTOMATIQUE – LE MODELE GRAFCET

1. Introduction.

Le GRAFCET (GRAphe Fonctionnel de Commande Etape - Transition) est un modèle de représentation graphique des comportements successifs d'un système logique séquentiel, préalablement défini par ses entrées et ses sorties.
Il est, à ce titre, à rapprocher des autres modèles de représentation que sont les schémas à contacts, les logigrammes ou les chronogrammes.
Le GRAFCET est né en 1977 d'une réflexion menée par des universitaires et des industriels français désireux de trouver un langage commun pour modéliser un problème séquentiel de commande. Ils avaient crée dans ce but un groupe de travail : l'AFCET (Association Française pour la Cybernétique Economique et Technique). Ceci explique aussi l'origine du terme GRAFCET (GRoupe AFCET).
Maintenant normalisé (Norme française NF C03-190 - Norme internationale CEI 60848), le GRAFCET est reconnu comme le langage graphique le mieux adapté à la modélisation de la partie commande d'un automatisme séquentiel.
Dans la suite on écrira « GRAFCET » pour parler de l'outil de modélisation et « diagramme grafcet » pour parler d'un modèle particulier.

2. Les différents points de vue de description.

La conception et la réalisation d'un système automatisé est un processus complexe que l'on peut décomposer en trois grandes étapes :

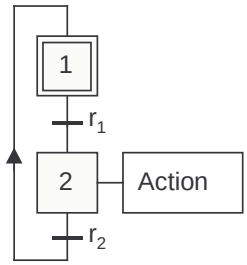
- ***Avant projet*** : On établie un cahier des charges précisant les objectifs, les fonctions et les contraintes du système à automatiser, les différents modes de marche et d'arrêt, la sécurité, la maintenance…
  Dans cette phase d'avant projet, les fonctions et les contraintes du système peuvent être représentées par un ***GRAFCET fonctionnel*** (ou **point de vue système**). Dans ce modèle apparaissent les fonctions à réaliser et les informations nécessaires à leur exécution. Ce modèle est purement descriptif. Le choix des actionneurs ou capteurs n'est pas encore fait.

- ***Pré-étude et étude*** : une analyse détaillée de faisabilité permet d'arrêter les choix techniques et les solutions technologiques ; le modèle devient complet et détaillé.
  Dans cette phase d'étude, les choix technologiques sont arrêtés. Un modèle détaillé du système de commande est réalisé. C'est le ***GRAFCET technologique*** (ou **point de vue partie opérative**).
  On peut alors distinguer trois niveaux :
  - point de vue "effecteur" : aucun actionneur ou pré-actionneur n'est supposé connu,
  - point de vue "actionneur" : *point de vue privilégié pour les systèmes connus,*
  - point de vue "pré-actionneur" : le grafcet décrit les comportements successifs des pré-actionneurs.

- ***Réalisation et exploitation*** : les différents modèles de commande sont implantés et testés ; la partie commande et la partie opérative sont reliées ; l'installation est mise en service progressivement.
  Dans la phase réalisation, le GRAFCET peut encore être utilisé (**point de vue partie commande**). De nombreux automates programmables industriels (API) disposent d'un langage de programmation qui permet de représenter les grafcets de commande et leurs éléments de structuration.
  Le GRAFCET décrit :
  - soit la succession des commandes des pré-actionneurs à partir des signaux des capteurs,
  - soit les ordres élaborés par la partie commande sous forme symbolique ou codée pour élaborer les sorties de la partie commande à partir des entrées capteurs.

3. Eléments de base.

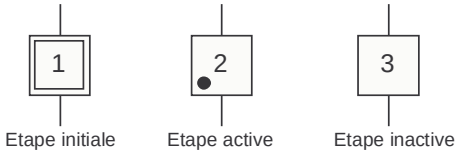
Un grafcet est un diagramme comportant deux types de symboles : les **étapes** et les **transitions**.
Des liaisons orientées (**arcs**) relient soit une étape à une transition, soit une transition à une étape.
Les arcs sont implicitement *orientés du haut vers le bas*. Si un arc est orienté du bas vers le haut il doit porter une flèche qui l'indique.



3.1. Les étapes.

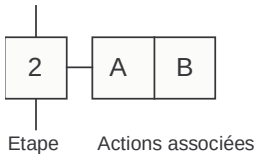
Les étapes modélisent les états du système.

Une étape est représentée par un carré et identifiée par un numéro.
Une étape peut être ***active*** ou ***inactive*** (un point à l'intérieur d'une étape signifie qu'elle est active).
Les étapes qui sont actives à l'instant initial sont appelées ***étapes initiales*** et sont représentées par un double carré.



La variable binaire $\mathbf{X_i}$ est associée à l'état de l'étape numéro **i** : X_i=1 si l'étape numéro i est active ; X_i=0 si l'étape numéro i est inactive.
À un instant donné, l'état du système séquentiel est représenté par l'ensemble des étapes actives à cet instant. Cet ensemble est appelé ***situation***.

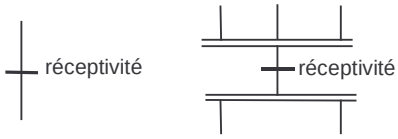
Les actions à exécuter sont liées aux états du système. Dans le GRAFCET, elles sont donc associées aux étapes. Une action est exécutée si l'étape correspondante est active. Plusieurs actions peuvent être associées à la même étape.



3.2. Les transitions.

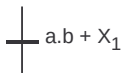
Les transitions permettent de modéliser les conditions de changements d'états du système.

Une transition est représentée par un trait horizontal placée entre une (plusieurs) étape(s) d'entrée, située(s) en amont, et une (plusieurs) étape(s) de sortie située(s) en aval.



Le passage d'un comportement du système au comportement suivant, respectivement d'une étape à l'étape suivante, correspond au franchissement de la transition.
Une transition représente une, et une seule, possibilité dévolution.
Une transition est validée lorsque toutes les étapes à partir desquelles la transition peut s'effectuer sont actives.

La proposition logique (ou variable logique) qui conditionne la transition est appelée ***réceptivité***. Cette réceptivité est une fonction logique combinatoire des variables d'entrée, éventuellement de l'état interne du système et du temps.
Une réceptivité est soit vraie, soit fausse.



4. Les règles d'évolution du GRAFCET.

L'alternance **étape - transition** et **transition - étape** doit toujours être respectée quelle que soit la séquence parcourue.

Règle 1 : Situation initiale.

La situation initiale d'un grafcet caractérise le comportement initial de la partie commande vis-à-vis de la partie opérative ou/et des éléments extérieurs. Elle correspond à l'ensemble des étapes actives au début du fonctionnement. Elle traduit généralement un comportement de repos.

Exemple *: La situation initiale du grafcet de la figure 1 est S_0={1,3}*

Règle 2 : Franchissement d'une transition.

Une transition est franchissable et obligatoirement franchie si les deux conditions suivantes sont remplies :

- La transition est validée (toutes les étapes d'entrée de cette transition sont actives) ;
- La réceptivité associée à cette transition est vraie.

Exemple *: Dans le grafcet de la figure 1, seule la transition t_1 est validée. Elle est franchissable si m=1.*

Règle 3 : Evolution de la situation

Le franchissement d'une transition entraîne simultanément l'activation de toutes les étapes immédiatement suivantes et la désactivation de toutes les étapes immédiatement précédentes.

Exemple : *Dans le grafcet de la figure 1, le franchissement de la transition t_1 conduit à la situation S_1={2,3}. Dans cette situation la transition t_2 est validée et son franchissement (lorsque a=1) conduit à la situation S_2={4}.*

Remarques *:*

- *La durée de franchissement d'une transition est considérée comme infiniment petite mais non nulle.*
- *Pour un grafcet donné, il est possible de construire le graphe des situations (graphe d'états) représentant l'ensemble des situations atteignables par le système (voir exemple figure 2).*

Règle 4 : Evolutions simultanées.

Plusieurs transitions simultanément franchissables sont simultanément franchies.

Exemple *: le franchissement simultané des transitions t4 et t5 dans le grafcet modifié de la figure 1 (si $\overline{c}$ est la réceptivité associée à t4) conduit à la situation S={1,3,6}.*

Règle 5 : Activation et désactivation simultanées d'une étape.

Si au cours du fonctionnement, la même étape est simultanément activée et désactivée, elle reste active.

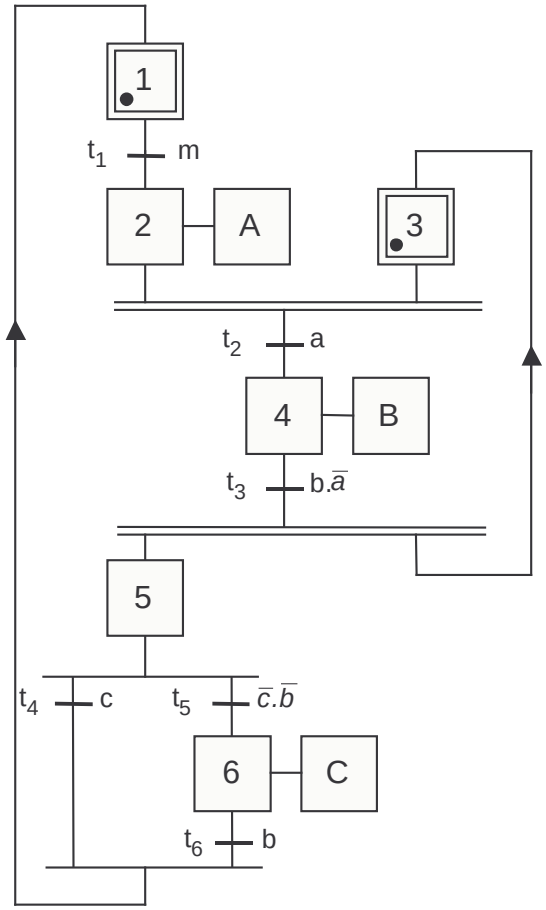


Fig 1 *: Exemple de grafcet*

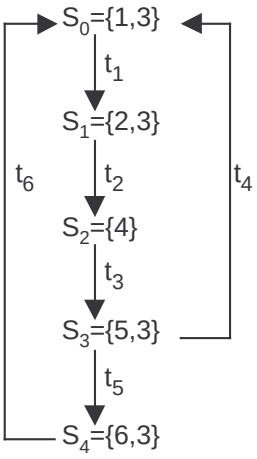


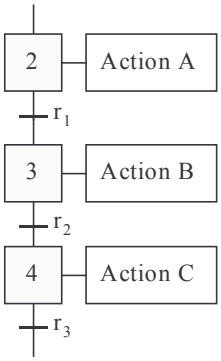
Fig 2 *: graphe des situations*

5. Structure de base.

Séquence.

On appelle séquence tout ensemble d'étapes successives où chaque étape est suivie d'une seule transition et chaque transition n'est validée que par une seule étape.

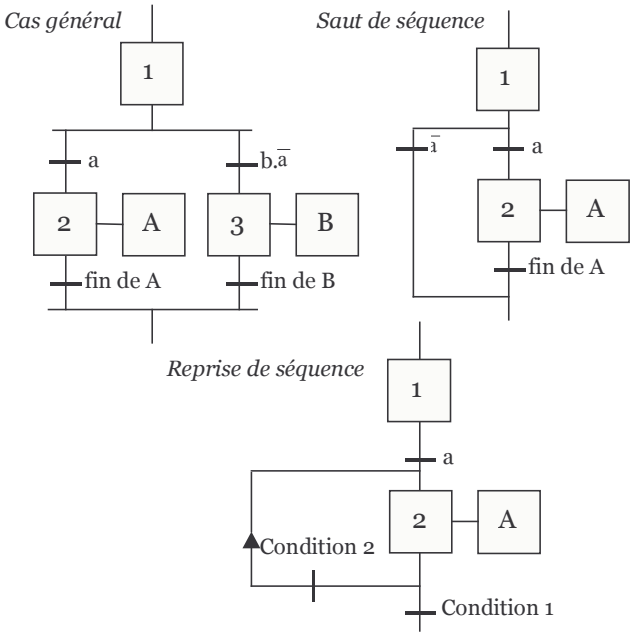
remarque *: un grafcet ne comporte qu'une seule séquence est un grafcet à séquence unique ou grafcet linéaire.*



Sélection de séquence.

Il représente une alternative d'évolution vers plusieurs étapes.
Les réceptivités associées aux transitions d'un aiguillage doivent être exclusives (leur ET logique doit être nul).

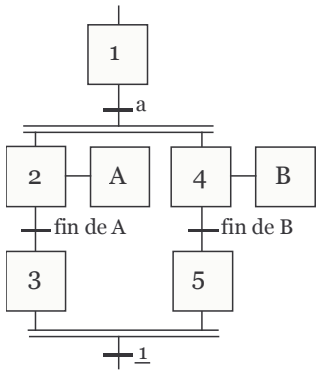
Définition : La sélection de séquence est aussi appelée ***divergent OU*** ou ***divergence de sélection de séquences***.
La convergence de plusieurs séquences est appelée ***convergent OU*** ou ***convergence de sélection de séquences***.



Parallélisme de séquences.

C'est un ensemble de séquences pouvant évoluer indépendamment, à partir du franchissement d'une transition activant simultanément plusieurs étapes.

Définition : Une transition qui possède plusieurs étapes de sortie représente l'exécution en ***parallèle*** de plusieurs séquences. On appelle cette structure ***divergent ET*** ou ***divergence de séquences simultanées.***
Une transition qui possède plusieurs étapes d'entrée représente la ***synchronisation*** de plusieurs séquences. On appelle cette structure ***convergent ET*** ou ***convergence de séquences simultanées***.



6. Compléments sur les réceptivités et les actions.

Les variables.

Une variable externe est soit :

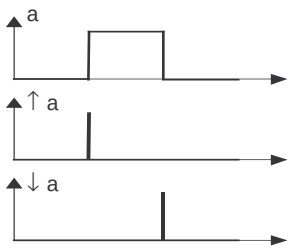
- Une variable binaire délivrée par la partie opérative à commander (état des capteurs) ou par son environnement (état d'un bouton manipulé par l'opérateur),
- Une variable binaire relative au temps.

Une variable interne est soit :

- Une variable binaire relative à la partie commande, c'est à dire à la situation dans laquelle se trouve le grafcet (l'état de l'étape **i** est représenté par la variable d'étape **Xi** : Xi=1 si l'étape i est active),
- Une variable générée par le modèle GRAFCET (compteur, variable de calcul…).
- Une variable binaire relative à un prédicat (ex : [t > 8℃] signifie que lorsque la proposition logique « t > 8℃ » sera vérifiée alors la variable binaire [t > 8℃] sera égale à 1).

Les événements.

Un événement est un changement d'état d'une variable binaire.



- On notera ***↑a*** l'événement d'entrée ***front montant*** de ***a*** associé au passage de la valeur 0 à la valeur 1 de la variable a. On notera ***↓a*** l'événement d'entrée ***front descendant*** de ***a*** associé au passage de la valeur 1 à la valeur 0 de la variable a.
- On notera **Act (2)** l'événement interne qui conduit à l'activation de l'étape 2.
- On notera **Dac (2)** l'événement interne qui conduit à la désactivation de l'étape 2.
- On notera **Clr (t8)** l'événement interne qui conduit au franchissement de la transition t8.

Les réceptivités.

Une réceptivité est soit :

- Une fonction logique de variables externes et internes, appelée condition (exemple : $(a+\overline{b}).X_2$)
- Un événement externe ou interne (exemple ↑a)
- Un événement et une condition (exemple ↑a.(b+X_2)

remarques :

Le temps peut être associé à une réceptivité en utilisant la notation t1/✳/t2.(exemple : $3s/X2/7s$) ;

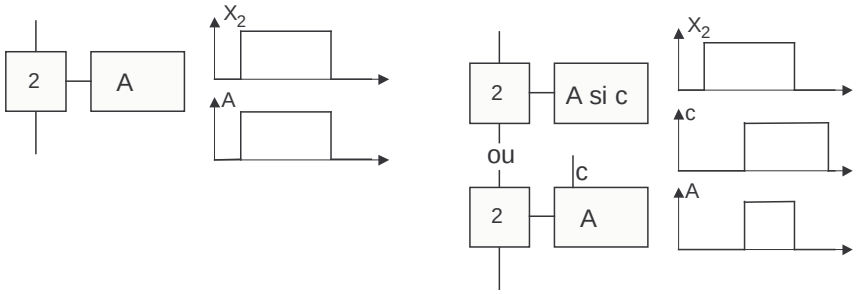
Lorsque aucune condition, ni aucun événement ne limite le franchissement d'une transition, alors par convention on note « $\underline{1}$ » sa réceptivité. C'est la condition ***toujours vraie***.

Les actions.

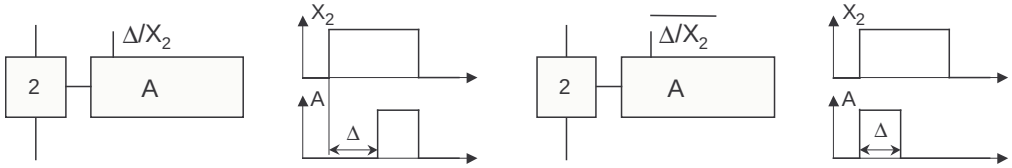
♦ Action continue et action conditionnelle.

- Une action ***continue*** est maintenue tant que l'étape est active.
- Une action ***conditionnelle*** est maintenue lorsque l'étape est active **et** que la condition associée est vraie.

action continue *action conditionnelle*

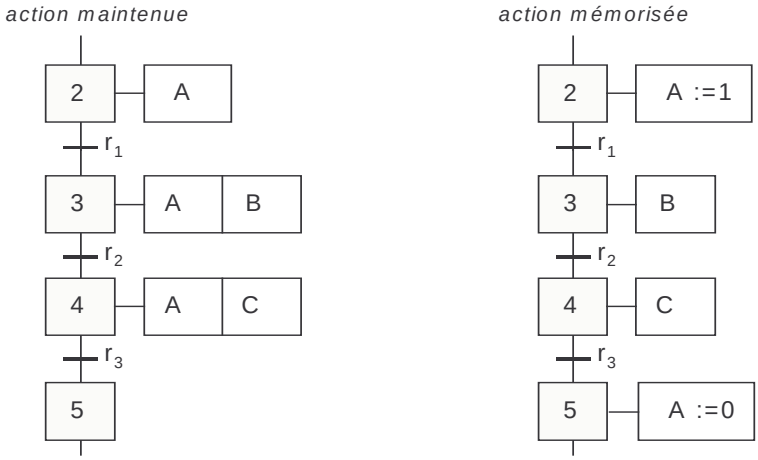


♦ Action temporisée → action conditionnée à une mesure de temps.



Action retardée Action limitée dans le temps

- Action maintenue et action mémorisée.
  - Une action ***maintenue*** A est une action exécutée tant que l'une au moins des étapes à laquelle elle est associée reste active. Une telle action est aussi appelée action à niveau.
  - Par opposition, une action ***mémorisée*** est une action qui est mise en mémoire jusqu'à ce qu'on l'annule.

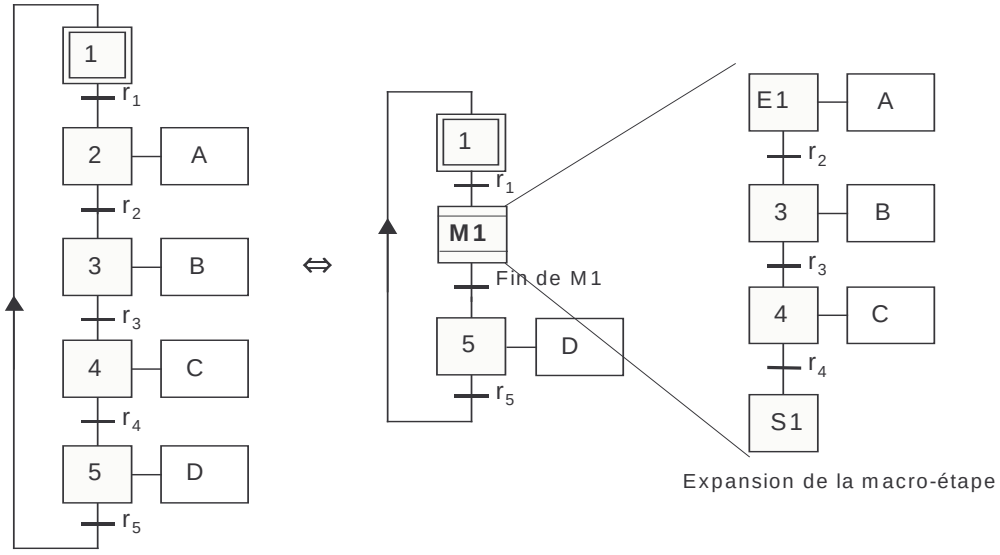


Si le choix entre ces deux modes de représentation est arbitraire pour des représentations du point de vue système, il peut être imposé, selon les points de vue partie opérative ou partie commande, par le choix technique, monostable ou bistable, des pré-actionneurs.

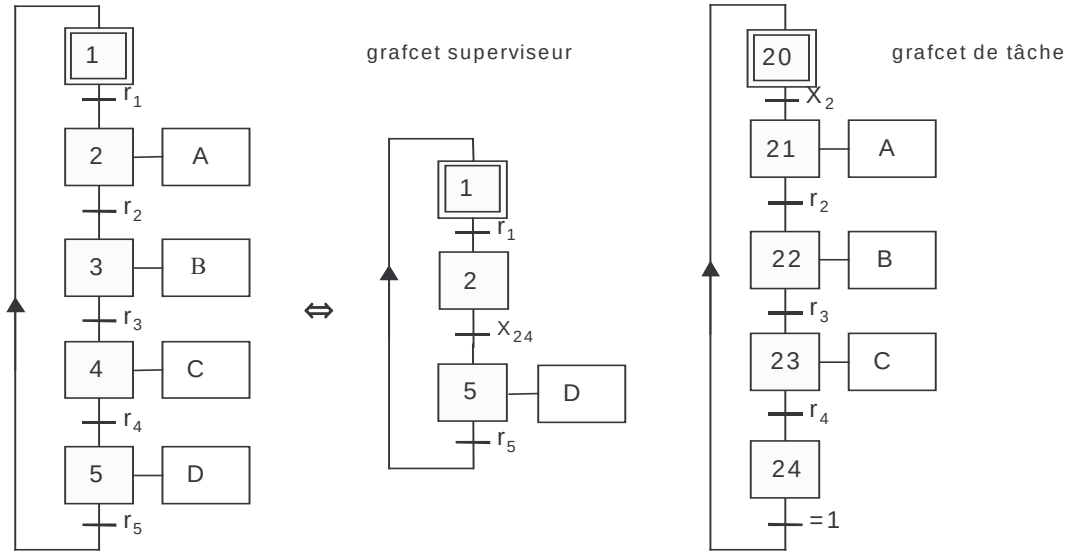
7. Macro-étape et tâche.

Macro-étape.

Une macro-étape Mi est une représentation par une seule cellule d'un ensemble unique d'étapes et de transitions. Cet ensemble est appelé ***expansion*** de Mi. L'expansion commence par une seule étape d'entrée, notée Ei, et se termine par une seule étape de sortie, notée Si. Il n'existe aucun arc qui arrive de l'extérieur sur cet ensemble, ni aucun arc qui en parte.
Une macro-étape est représentée par un carré séparé en trois parties par deux traits horizontaux comme le montre la figure ci-contre.



Tâche.
la notion de tâche peut être considérée comme la transformation de l'expansion d'une macro-étape en un grafcet indépendant. Dans la situation initiale, l'opération attend d'être déclenchée. Ce déclenchement est contrôlé par un autre grafcet indépendant appelé superviseur. Les échanges entre les tâches et le superviseur s'effectuent à l'aide des variables d'étape.



8. Réalisation de la partie commande.

8.1 Logique câblée - logique programmée.

Définition.

La réalisation de la partie commande peut se faire par association de fonctions logiques (technologie électromagnétique → relais, contacteurs ; technologie pneumatique → clapets).
Chaque fonction utilisée a un rôle différent dans l'élaboration des ordres de commande.
Ces fonctions logiques sont raccordées entre elles par des connexions (fils électriques, circuits imprimés, tubes) d'où le nom de **logique câblée** utilisé pour toutes les parties commandes ainsi réalisée.
Une autre possibilité existe : utiliser une fonction de chaque type (ET, OU, mémoire, etc.) pour réaliser le fonctionnement recherché mais il est alors nécessaire d'utiliser plusieurs fois chaque fonction en la "raccordant" différemment à chaque utilisation.
La description de la suite des fonctions à utiliser et des raccordements à effectuer à chaque utilisation constitue un programme.
Une partie commande réalisée selon ce principe est dite en **logique programmée**.

Avantages et inconvénients

☞ ***Volume de traitement***

La logique câblée n'utilise que les fonctions logiques nécessaires au fonctionnement de la partie opérative alors que la logique programmée nécessite tout un ensemble de fonctions indispensables au fonctionnement interne de la partie commande.
Il existe donc une limite inférieur dans le volume du traitement au-dessous de laquelle la logique programmée est plus coûteuse et parfois plus volumineuse que la logique câblée.
Les progrès rapides des technologies font que cette limite s'abaisse rapidement et il est possible de trouver aujourd'hui des automates programmables équivalents à quelques dizaines de fonctions logiques en coût et en volume.
Inversement, la logique câblée impose l'utilisation d'autant de fonctions logiques que le fonctionnement le nécessite alors que la logique programmée utilise un nombre de fonctions logiques limité quel que soit le fonctionnement.

Il existe donc une limite supérieure au-dessus de laquelle la logique câblée est plus coûteuse et plus volumineuse que la logique programmée.

☞ ***Souplesse***

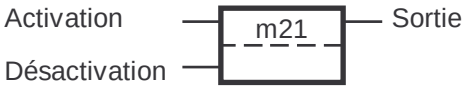
Le fonctionnement d'une logique programmée est déterminé par le programme.
Celui-ci peut être modifié sans intervention sur le câblage, d'où une souplesse à laquelle ne peut pas prétendre la logique câblée.

☞ ***Vitesse de fonctionnement***

En logique câblée, le temps de modification d'un ordre de sortie ne dépend que des temps de commutation des fonctions logiques, alors qu'en logique programmée, ce temps est donné par celui de la lecture du programme.
En effet, il faut "recalculer" chaque équation logique régulièrement, pour obtenir la modification d'un ordre de sortie.
La logique câblée peut donc être beaucoup plus rapide que la logique programmée.

8.2 Logique câblée : le séquenceur.

Chaque étape du GRAFCET peut être matérialisée par une mémoire (prioritaire à l'arrêt) appelée *mémoire d'étape*.

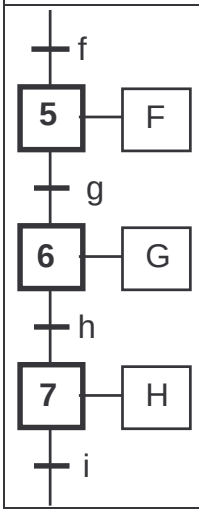
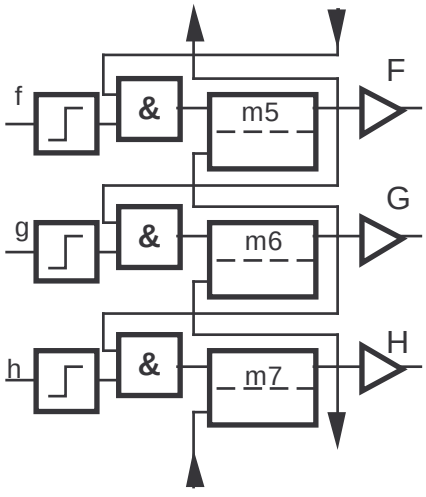


Outre les fonctions logiques combinatoires classiques, il est nécessaire d'introduire des fonctions d'adaptation des signaux d'entrée et d'amplification des signaux de sortie.

Adaptateur d'entrée

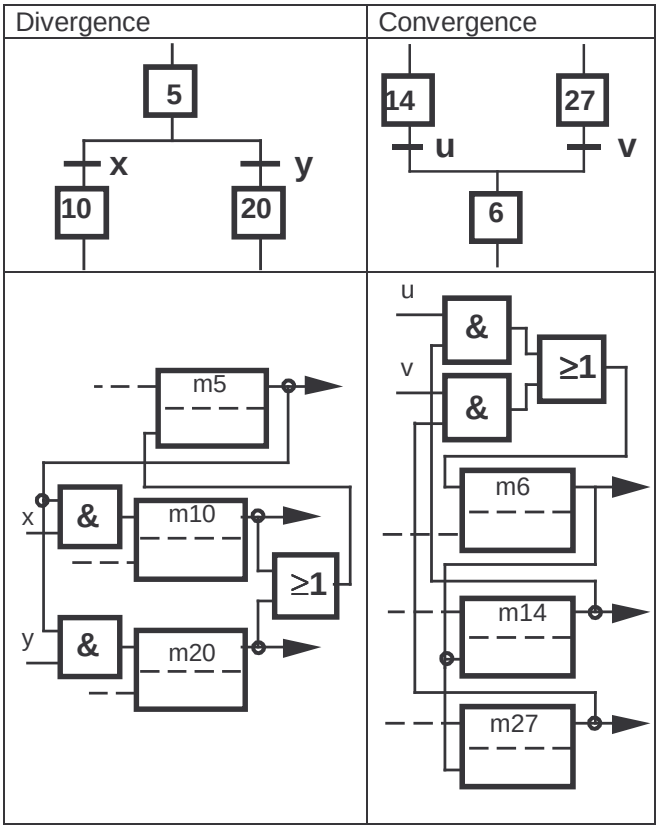
Amplificateur de sortie

♦ *Grafcet à séquence unique*

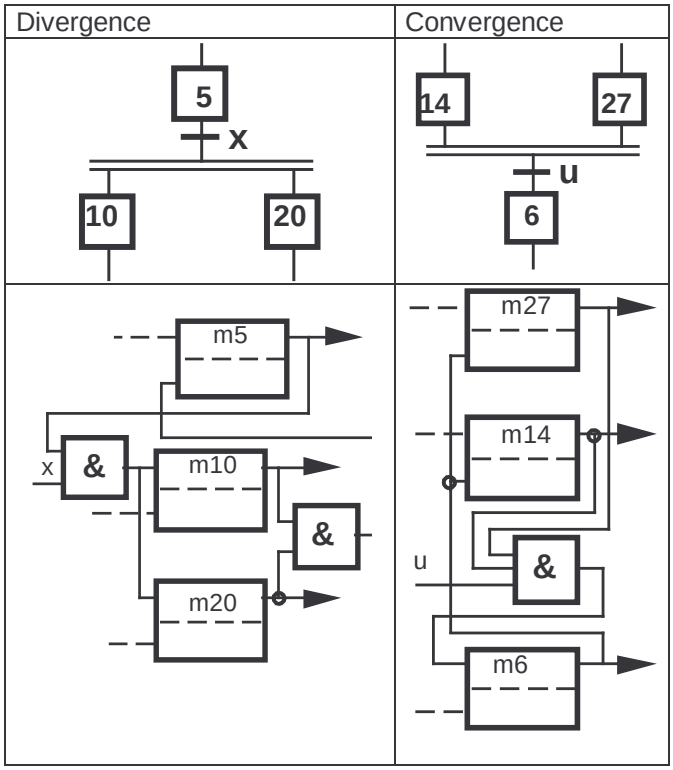
GRAFCET	Logigramme Correspondant
	

Afin de simplifier les logigrammes qui suivent, les symboles associés aux adaptateurs et aux amplificateurs ne seront plus représentés.

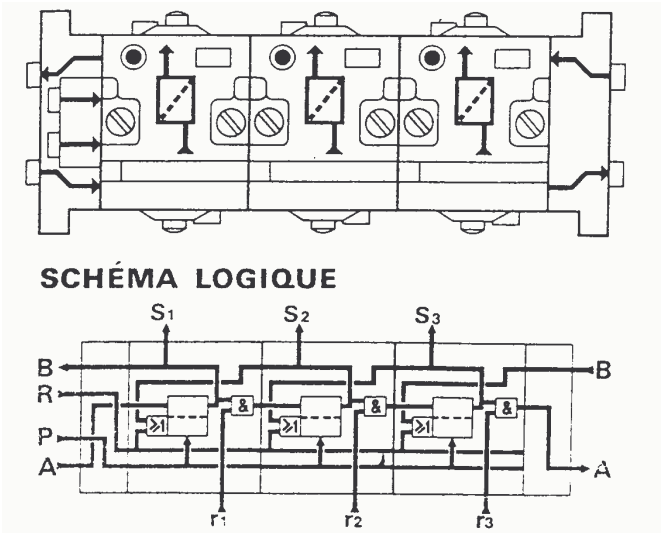
♦ ***Grafcet avec sélection de séquence***

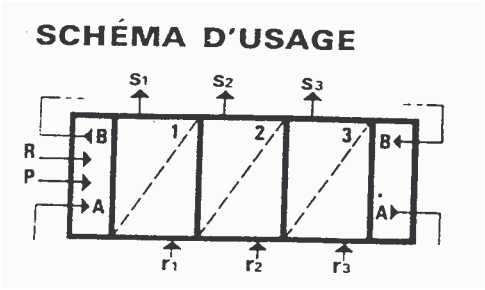


♦ ***Grafcet avec parallélisme de séquence***



♦ ***Exemple : le séquenceur TELEMECANIQUE***





8.3 Automate Programmable Industriel (API)

Définition

L'automate programmable industriel (en abrégé API) est le constituant de base des équipements automatisés. Il est apparu vers les années soixante dix, à la demande des constructeurs automobiles, qui souhaitaient disposer pour l'automatisation des usines d'un matériel pouvant s'adapter à l'évolution des fabrications plus simplement et à un coût moindre que les ensembles câblés. Initialement destiné au traitement des signaux logiques, il est capable de traiter des tâches de plus en plus complexes (calculs, asservissements, gestion, etc.).
C'est un micro-ordinateur spécifique, qui se distingue des micro-ordinateurs de bureau par plusieurs caractéristiques :

- il est conçu pour fonctionner dans des ambiances industrielles qui peuvent êtres sévères ;
- il peut gérer un grand nombre de signaux entrées/sorties en temps réel ;
- il dispose de langages adaptés aux fonctions d'automatisme et qui ne réclame pas de connaissances particulières en informatique.

Exemples d'Automates Programmables Industriels

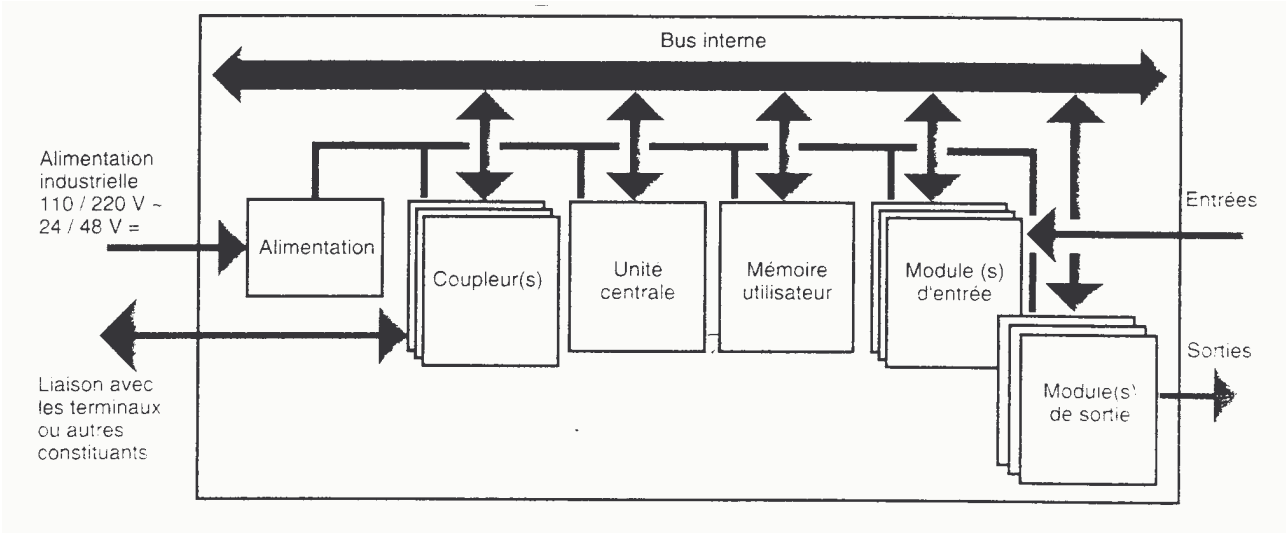
Architecture

La structure de base d'un automate programmable comprend plusieurs ensembles fonctionnels :

- l'unité centrale bâtie autour d'un microprocesseur (ou plusieurs) qui gère le fonctionnement de l'automate ;
- la mémoire utilisateur qui sert de stockage du programme et des données ;
- les modules d'entrées/sorties, interface entre les signaux électriques issus du processus et les variables informatiques ;
- les coupleurs de liaison avec les organes de dialogue ou d'autres constituants programmables (automates, ordinateurs, etc.).

Ces unités échangent des informations par l'intermédiaire d'un ensemble de conducteurs : **le bus**, qui est généralement réalisé par un circuit imprimé.

Les tensions nécessaires au bon fonctionnement des composants sont fournies par une alimentation qui peut comporter des dispositifs de surveillance de la qualité des tensions pour garantir un niveau de sûreté requis.



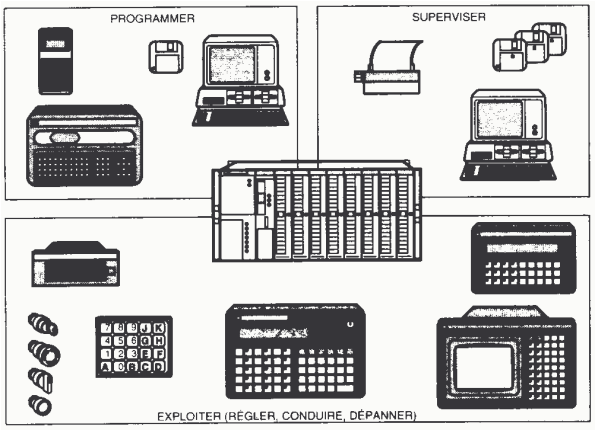
Mise en œuvre

La mise en œuvre d'un automate programmable nécessite : le raccordement aux divers chaînes d'acquisition et d'action ainsi qu'aux sources d'énergie, et l'implémentation d'un programme et son exécution.

Il est possible de distinguer pour la mise en œuvre d'un automate programmable, trois fonctions de liaison avec l'environnement qui concernent :

- ***la programmation*** pour une première mise en œuvre ou des évolutions. Celle-ci a pour but d'écrire dans la mémoire le programme de l'application à relire soit dans une mémoire RAM en phase de mise au point, soit dans une mémoire EPROM en phase définitive ;
- ***l'aide à l'exploitation*** ou dialogue d'exploitation (conduite de machines, réglage des paramètres, dépannage...) ;

la supervision (dialogue avec d'autres équipements périphériques à des fins de coordination et de gestion).



La tendance est à l'heure actuelle à l'utilisation de matériels universels qui permettent par le biais de passerelles logicielles de programmer et d'exploiter des matériels de différents constructeurs.

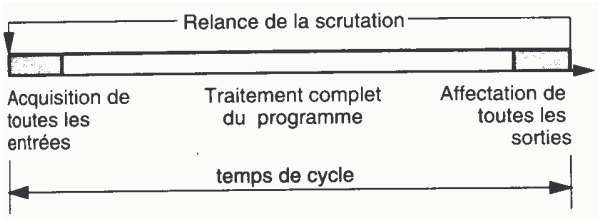
L'exécution d'un programme

La mise en mémoire étant réalisée (implémentation), la phase d'exécution est alors possible.

♦ Cycle d'un automate programmable : traitement monotâche.

Le traitement est cyclique c'est à dire qu'il est relancé à la fin de chaque exécution. Le processeur exécute les instructions, une après l'autre, dans l'ordre de la liste. Ce cycle est réalisé en trois étapes principales :

- acquisition des entrées dans l'état qui reste figé durant toute la durée du cycle, évitant les aléas de fonctionnement ;
- traitement du programme écrit par l'utilisateur d'une durée variable selon les instructions ;
- mise à jour ou affectation des sorties.



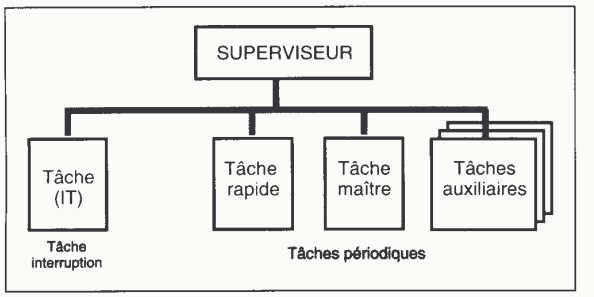
Le temps de cycle dépend de nombreux paramètres (type et importance du traitement, caractéristiques logicielles et matérielles de l'automate, etc.). Il peut atteindre plusieurs dizaines de millisecondes. Ce mode de fonctionnement de nombreux automates est appelé cyclique asynchrone.

Ce cycle de fonctionnement reste acceptable tant que la durée du cycle est compatible avec les exigences du processus. Il devient inacceptable quand les informations d'entrée et de sortie dont la durée est inférieure au temps de cycle ne sont pas prises en compte.

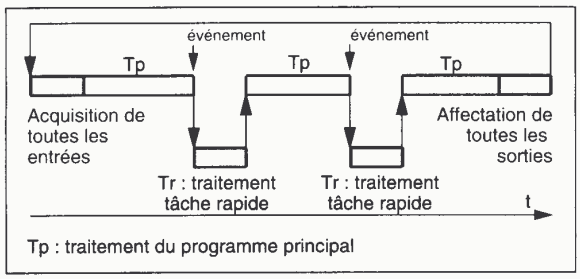
♦ Cycle d'un automate programmable : traitement multitâche.

Avec un automate à structure multitâche, le programme est organisé en tâches correspondant chacune à une fonction : positionnement d'un mobile, régulation d'un niveau, etc.

Ces tâches peuvent être périodiques (ordre et périodicité définis par l'utilisateur) ou d'interruption (du programme en cours d'exécution) en vue d'une exécution immédiate est prioritaire.



A coté d'une tâche principale ou tâche maître exécutée de manière cyclique, l'utilisateur peut programmer une tâche non périodique et événementielle (dite tâche rapide). A l'apparition d'un état sur une entrée spécifique ou quand la valeur de présélection d'un compteur rapide est atteint, le programme principal est momentanément interrompu, un programme court est alors exécuté. Puis le programme principal est relancé de là où il s'était arrêté.



Structure multitraitement.

Les structures multitraitement permettent de séparer les fonctions de traitement est d'affecter à chacune un processeur spécialisé. Cette solution augmente la performance d'un système grâce à la simultanéité des traitements.