Ministère de l’enseignement supérieur et de la recherche scientifique.

Université Mohamed Boudiaf de Msila,
Département d ‘électronique faculté de technologie

Support de cours

Les automates programmables industriels

Cours et TD Master 2

Filière Electronique

Réalisé Par :

LALAOUI Lahouaoui

Année universitaire 2016 /2017

Table des matières

Chapitre I Le réseau de pétri

I.1 Introduction

Les systèmes de commande en temps réel sont souvent très complexes, et leurs conceptions posent des problèmes délicats dus essentiellement à l'interaction entre les différentes taches exécutées en parallèle, et aux conflits d'accès aux ressources du système. Il est donc nécessaire de disposer d'outils puissants qui permettent de représenter dune façon réaliste l'évolution des taches, de façon a facilité la conception du système. En pratique, la représentation par réseaux de pétri (Pétri Nets) s'est avérée être un bon compromis entre la souplesse d'emploi et la puissance de représentation. Ils permettent de décrire dune façon relativement simple l'évolution des processus ainsi les relations entre eux et leur usage s'est répandu dans de nombreux domaines tels que les automatismes logiques, les systèmes multiprogrammes ou les protocoles de communication.

Les réseaux de pétri peuvent être considères sous plusieurs angles différents. Un premier point de vue consiste à définir dune façon formelle les réseaux de pétri et a en déduire les règles d'évolutions et les principales propriétés. Une seconde approche consiste à étudier les différentes manières d'appliquer ce modèle théorique à la spécification fonctionnelle de systèmes fonctionnant dans un contexte de temps réel. Nous présenterons ici ceux deux aspects des réseaux

de pétri et nous montrerons ensuite comment il est possible de valider le modèle a base de réseaux de pétri d'un système et quelles sont les techniques employées pour passer d'un réseau de Pétri a une réalisation matérielle ou logicielle.

I.2 Définitions

Carl Adam Pétri est un mathématicien allemand qui a défini un outil mathématique très général permettant de décrire des relations existant entre des conditions et des événements, de modéliser le comportement de systèmes dynamiques a événements discrets.

¤ **Début des travaux 1960-1962** : ont donne lieu a de nombreuses recherches.

¤ **1972-1973**, utilisation de cet outil pour la description d'automatismes logiques, ce qui a débouche sur le Grafcet. Cet outil permet l'analyse qualitative. Il existe différents types de réseaux de Pétri : temporises, interprètes, stochastiques, colores, continus et hybrides.

• **Un réseau de Pétri (RdP)** est un graphe biparti constitué de 2 sortes de nœuds : Les places (représentées par des ronds) et les transitions (représentées par des barres).

• **Le graphe est orienté** : Des arcs vont d'une sorte de nœuds à l'autre (jamais de places à places, ou de transitions à transitions directement).

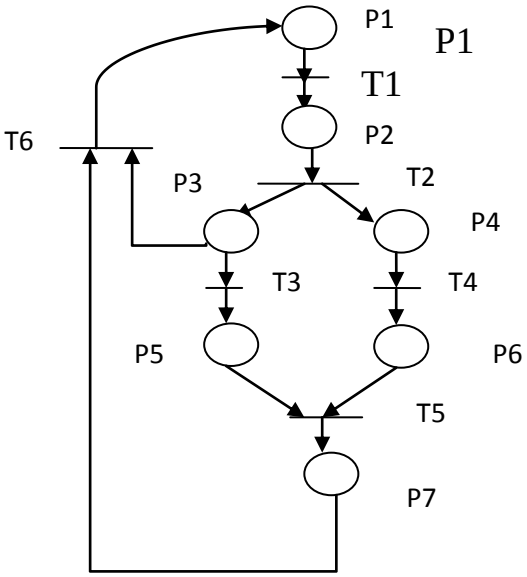
• Graphe formé de :

- ensemble de places d'un réseau de Pétri est note par P = {P1, P2, P3,….}

- ensemble de transition d'un réseau de Pétri est note par T= {T1, T2, T3,….}

- marquage initial M= {m_1, m_2, m_3,….}

Exemple :



FigureI.1: Réseaux de Pétri non marque

On dit que la place P_2 est en **amont** ou est une **entrée** de la transition T_2.

On dira que la place P_7 est en **aval** ou est une **sortie** de la transition T_5.

Une place peut être reliée par plusieurs arcs à une transition. Ces arcs multiples peuvent être représente par un arc simple affecte dune pondération égale au nombre d'arcs multiples (voir la figure I.2). Par défaut les arcs non pondérés sont considérés comme ayant un poids égal à 1.

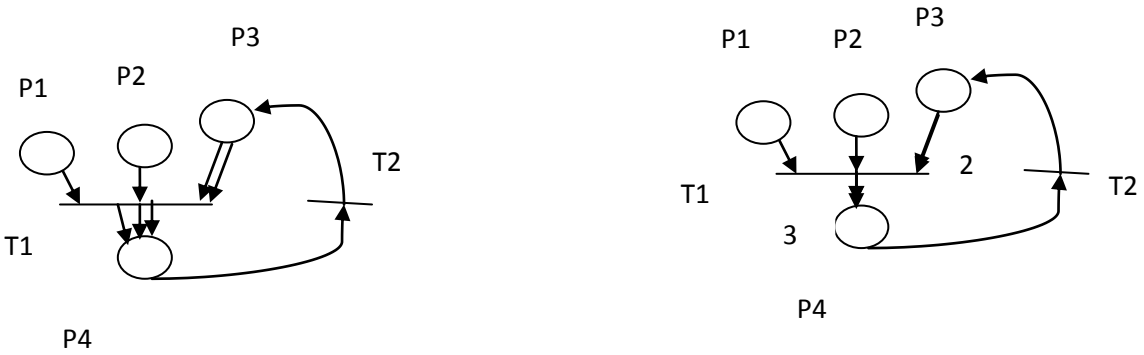


Figure I.2 : Réseaux de pétri a arcs multiples

I.3 Matrice d'entrée et matrice de sortie

La structure de réseau de pétri est décrite par deux matrices notées I et O qui spécifient respectivement les poids des arcs reliant les places aux transitions et ceux reliant transitions aux places.

Dans le cas d'un réseau de pétri de N places et K transitions, les matrices I et O sont de dimension NxK et elles sont définies par :

$I(NxK) = [e_{ij}] + et \quad O(NxK) = [o_{ij}]$

I.3.1. Marquage des places

• Les places sont marquées par des jetons ou marques (points noirs), un nombre entiers positifs ou nul.

• Les jetons circulent dans les places selon certaines règles (définies ci-dessous). Cette circulation symbolise l'évolution dynamique du système. Le marquage initial (celui indiqué sur le dessin) donne la position initiale des jetons.

Même Rdp que précédemment avec marquage M=(1,0,1,0,0,2,0).

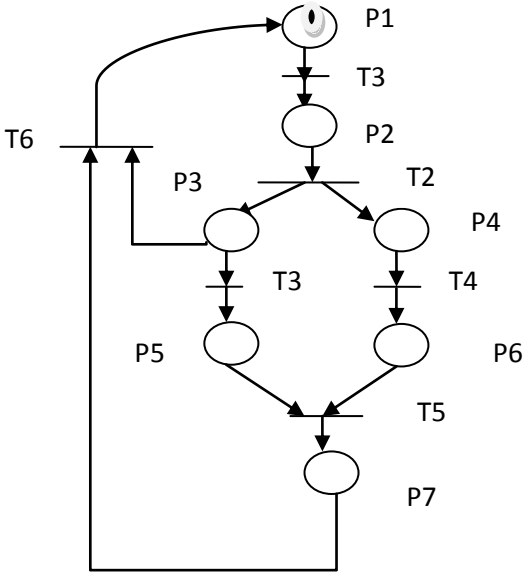


Figure I.3 : Réseaux de pétri initialement marqué

I.3.2. Franchissement de transition

Règles d'évolution d'un RDP sont :

L'évolution du réseau de pétri est définie par le déplacement par le déplacement des jetons a partir de l'état initial. Les jetons, qui matérialisent l'état du réseau à un instant donne peuvent passer dune place a une autre par franchissement (ou tir) dune transition.

Les règles d'évolution d'un RDP sont les suivantes :

Une transition est franchissable ou sensibilisée, lorsque chacune de ses places d'entrée (en amont) de cette transition possède un nombre de jetons au moins égal au poids de l'arc qui la relie à cette transition.

Le réseau ne peut évoluer que par franchissement dune seule transition à la fois.

Celle-ci étant choisie au hasard parmi toutes celles qui sont sensibilisées à cet instant.

Le franchissement dune transition est un phénomène instantané, qui se traduit par les opérations indivisibles suivantes :

On enlève de chaque place d'entrée de la transition, un nombre de jetons égal au poids de l'arc qui la relie à celui-ci.

On ajoute de chaque place de sortie de la transition, un nombre de jetons égal au poids de l'arc qui la relie à celui-ci.

Remarque : si une transition est validée, cela n'implique pas qu'elle sera immédiatement franchie. Ces règles introduisent en effet un certain

indéterminisme dans l'évolution des réseaux de Pétri, puisque ceux-ci peuvent passer par différents états dont l'apparition est conditionnée par le choix des transitions tirées. Ce fonctionnement représente assez bien les situations réelles où il n'y a pas de priorité dans la succession des événements.

Note : Il y a conflit si plus d'une transition peuvent être franchies pour une même place d'origine, on choisit l'une des transitions, de manière non-déterministe.

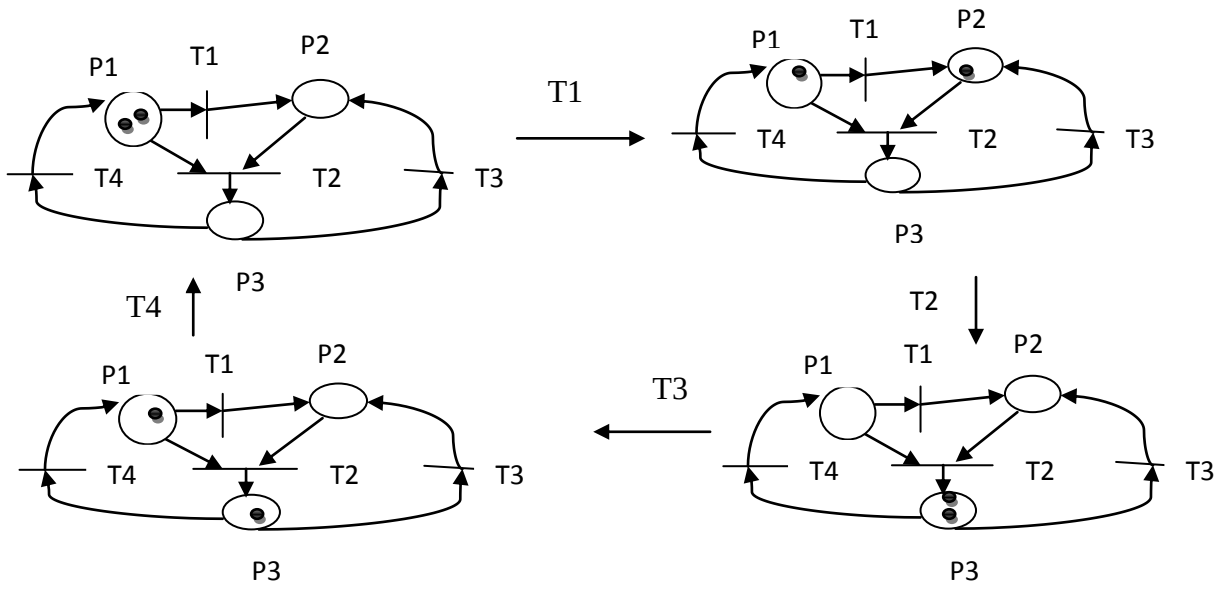


Figure I.4 : exemple d'évolution d'un RDP

Dune manière formelle :

--- une transition Tj est sensibilisée si : $\forall P_i \in \{plances\ \ dentrees\ de\ Tj\}, m(P_i) \geq e_{ij}$.

Le tir de la transition Tj a partir du marquage $M_k = [m_k(P_1), \dots, m_k(P_N)]$ se traduit par un nouveau marquage $M_{k+1} = [m_{k+1}(P_1), \dots, m_{k+1}(P_N)]$tel que:$m_{k+1}(P_i) = m_k(P_i) - e_{ij} + o_{ij}$

Ou les poids"e_{ij} " et "o_{ij}" des arcs reliant, respectivement, la place P_i a la transition Tj et la transition Tj a la place P_i sont tel que :

$$e_{ij} = \begin{cases} x \geq 1 \text{ si } P_i \text{ est une place dentree de Tj.} \\ 0 \text{ si non .} \end{cases}$$

$$o_{ij} = \begin{cases} y \geq 1 \text{ si } P_i \text{ est une place sortie de Tj.} \\ 0 \text{ si non .} \end{cases}$$

Comme exemple considérons le cas de la figure I.4, les matrices d'entrées et de sortie correspondantes sont :

$$I = \begin{bmatrix} 1 & 0 \\ 1 & 0 \\ 2 & 0 \\ 0 & 1 \end{bmatrix} \begin{matrix} P1 \\ P2 \\ P3 \\ P4 \end{matrix} \qquad o = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ 0 & 1 \\ 3 & 0 \end{bmatrix} \begin{matrix} P1 \\ P2 \\ P3 \\ P4 \end{matrix}$$

I.3.3. Graphes de marquage

Le réseau de Pétri ainsi définie, permet la représentation statique d'un système. Pour modéliser l'évolution d'un système au cours du temps, on est amené a compléter le réseau de Pétri par un marquage M qui consiste a disposer un nombre de jetons $m(Pi) \geq 0$ pour chaque place Pi du réseau. Le marquage est donc symbolise par des jetons situes a l'intérieur des places. L'évolution temporelle d'un RdP peut être décrite par un graphe de marquage représentant l'ensemble des marquages accessibles et d'arcs correspondant aux franchissements des transitions faisant passer d'un marquage à l'autre pour un marquage initial M0.

Exemples :

1) $M_0 = [1, 3, 0]^T$

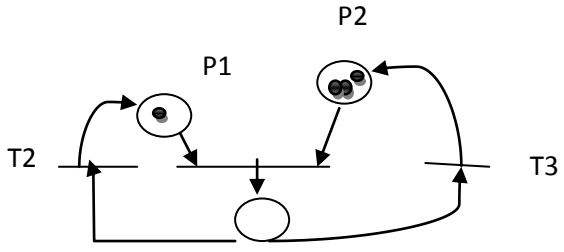


Figure. I.5 : Exemple de Rdp marque.

Soient M_j et M_k deux marquages d'un réseau de Pétri, on dit que :

$M_j \geq M_k, si\ \forall i, m_j(P_i) \geq m_k(P_i)$.

$$M_j \geq M_k, si\ \forall i, m_j(P_i) \geq m_k(P_i)\ et\ \ni i\ tel\ que\ m_j(P_i) \geq m_k(P_i)$$

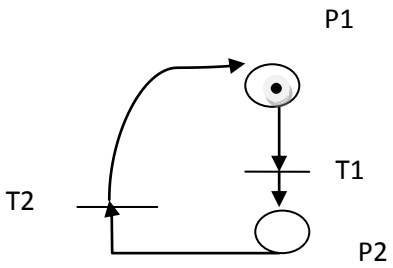
Ensemble des marquages accessibles

L'ensemble des marquages accessibles à partir du marquageM_0, qu'on note M0, est l'ensemble des marquages qu'on peut atteindre en utilisant toutes les séquences de franchissement possibles.

Exemple :

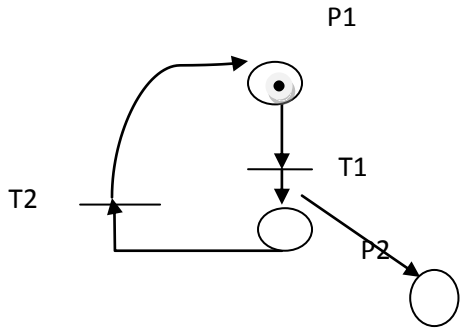
Illustrer cette notion, considérons l'exemple de la figure I.6 sur laquelle nous montrons la procédure de l'ensemble des marquages M0 d'un réseau de Pétri qui est, dans ce cas :

2)



$$\begin{bmatrix}1\\0\end{bmatrix} \rightleftarrows \begin{bmatrix}1\\0\end{bmatrix}$$

3)



$$\begin{bmatrix}1\\0\\0\end{bmatrix} \rightarrow \begin{bmatrix}0\\1\\1\end{bmatrix} \rightarrow \begin{bmatrix}1\\1\\2\end{bmatrix} \rightarrow \cdots$$

Figure I.6 : Ensemble des marquages d'un RDP

Par exemple si le graphe présente une zone non bouclé, cette partie du marquage une fois atteinte constitue un arrêt de l'évolution du RdP et celui-ci sera déclaré avec blocage.

Classification des réseaux de Pétri

I.3.4. RdP autonomes et non autonomes

Un RdP autonome décrit le fonctionnement d'un système qui évolue de façon autonome, c-à-d dont les instants de franchissement ne sont pas connus, ou pas indiqués.

(Exemple : Le cycle des saisons).

a. RDP ordinaire

Un réseau de Pétri ordinaire est un réseau dont les poids des arcs sont tous égaux a 1.

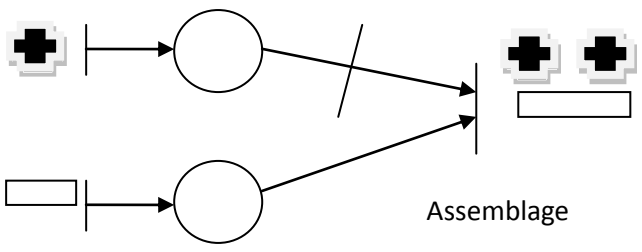
b. RdP généralisé

Un RdP dans lequel des poids (nombres entiers strictement positifs) sont associés aux arcs. L'arc Pi → Tj a un poids p. La transition Tj ne sera validée que si Pi contient au moins p jetons.

Lors du franchissement de cette transition, *p* jetons seront retirés de la place Pi.

Lorsqu'un arc Tj → Pi a un poids *p* cela signifie que lors du franchissement de Tj, *p* jetons seront ajoutés à la place Pi.

Exemple :



c. RdP à capacité

Un RdP dans lequel des capacités (nombres entiers strictement positifs) sont associés aux places. Le franchissement d'une transition d'entrée d'une place Pi dont la capacité est cap(Pi) n'est possible que si le franchissement ne conduit pas à un nombre de jetons dans Pi qui dépasse cette capacité.

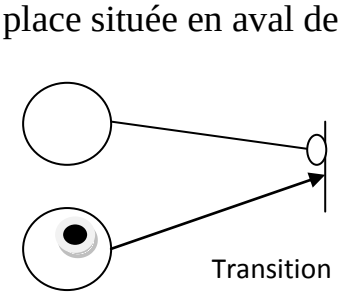
Note : Tout RdP à capacité peut être transformé en un RdP ordinaire. (À démontrer en classe).

d. RdP coloré

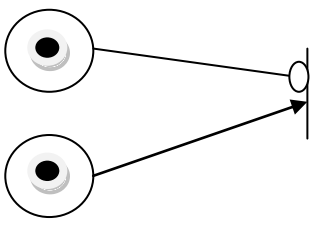
Un RdP coloré comporte des jetons auxquelles on attribue des couleurs (possibilité de représenter des processus parallèles)

e. RdP à arcs inhibiteurs

Un arc inhibiteur est un arc orienté qui part d'une place pour aboutir à une transition (et non l'inverse). Son extrémité est marquée par un petit cercle. La présence d'un arc inhibiteur entre une place Pi et une transition Tj signifie que la transition Tj n'est validée que si la place Pi ne contient aucun jeton. Le franchissement de la transition Tj consiste à retirer un jeton dans chaque place située en amont de la transition à l'exception de la place Pi, et à ajouter un jeton dans chaque place située en aval de la transition.



Transition Franchissable



Transition non Franchissable

f**. RdP à priorité**

Un tel réseau est utilisé lorsque l'on veut imposer un choix entre plusieurs transitions validées.

g. RDP à prédicats

Un réseau de Pétri à prédicats comporte des marques aux quelles sont associes des paramètres. En utilise ce type de réseaux lorsque les marques dune même place ne sont pas du même type.

Remarque :

- Les RdP généralisés, à capacité, coloré sont des abréviations des RdP ordinaires. Toutes les propriétés que nous allons voir dans ce qui suit peuvent donc être adaptées à ces modèles.

- Les RdP à arcs inhibiteurs, les RdP à priorités, les RdP non autonomes et les RdP continus sont des extensions des RdP ordinaires. Certaines propriétés des RdP ordinaires leur sont applicables mais pas toutes.

RdP non autonome

• Un RdP non autonome décrit le fonctionnement d'un système dont l'évolution est conditionnée par des événements externes ou par le temps (Exemple : Démarrage d'un moteur).

. Un RdP non autonome est Synchronisé et/ou Temporisé.

a. RDP Synchronisé

Un réseau de Pétri Synchronisé(RdPS) est un réseau de Pétri auquel on a associe un ensemble d'événements externes aces transitions.

b. RdP Temporisé

Un réseau de Pétri Temporise (RdPT) permet de décrire un système dont le fonctionnement dépend du temps.

c. **RdP continu**

Leur particularité est que le marquage d'une place est un nombre réel (positif) et non plus un nombre entier.

I.3.5. RdP particuliers

I.3.5.1 Structures particulière :

Ici on s'intéresse en premier à la structure du RdP.

a. Graphe d'états

Un RdP est un graphe d'états si est seulement si toute transition a exactement une place d'entrée et une place de sortie.

b. Graphe d'événements

Un Rdp est un graphe d'événement si et seulement si toute place a exactement une transition d'entrée et une transition de sortie. Un Graphe d'événement est donc dual d'un graphe d'états.

c. RdP sans conflit

Un RdP dans lequel toute place a au plus une transition de sortie. Un conflit (ou conflit structurel) correspond à l'existence d'une place P1 qui a au moins deux transitions de sortie T1, T2,…On notera <P1, {T1, T2,….}>

d. RdP à choix libre

Un RdP à choix libre est un RdP dans lequel pour tout conflit <P1, {T1, T2,….}> aucune des transitions T1, T2,…ne possède une autre place d'entrée que P1.

e. RdP simple

C’est un RdP dans lequel chaque transition ne peut être concernée que par un conflit au plus. S’il existe une transition T1 et deux conflits <P1, {T1, T2,…}> et <P2, {T1, T3,…}>, alors le RdP n’est pas simple.

Remarque :

- L’ensemble des RdP simples inclut l’ensemble des RdP à choix libre, qui inclut l’ensemble des RdP sans conflit, qui inclut lui-même l’ensemble des graphes d’événements

- L’ensemble des graphes d’états est inclus dans l’ensemble des RdP à choix libre.

f. RdP pur

C’est un RdP dans lequel il n’existe pas de transition ayant une place d’entrée qui soit également place de sortie de cette transition.

g. RdP sans boucle

Un RdP sans boucle est tel qu’il existe une transition Tj et une place Pi qui est à la fois place d’entrée et place de sortie de Tj, alors Tj à au moins une autre place d’entrée.

I.3.5.2. Abréviations et extensions :

Abréviations :

Des représentations simplifiées utiles pour alléger le graphisme mais auxquelles on peut toujours faire correspondre un RdP ordinaire (c.-à-d. un RdP autonome marqué fonctionnant selon les règles prédéfinies).

Extensions : Des modèles auxquels des règles de fonctionnement ont été ajoutées afin d’enrichir le modèle initial pour aborder un plus grand nombre d’applications.

Remarque : Toutes les propriétés des RdP ordinaires se conservent pour les abréviations moyennant quelques adaptations, tandis que ces propriétés ne se conservent pas toutes pour les extensions.

I.3.6. Propriétés des réseaux de Petri

I.3.6.1 *RdP borné :*

• Une place Pi est dite bornée pour un marquage initial M0 si pour tout marquage accessible à partir à partir de M0 le nombre de jetons dans Pi est fini.

• Un RdP est borné pour un marquage initial M0 si toutes les places sont bornées pour M0.

• Si pour tout marquage M appartement à l'ensemble des marquages accessibles à partir de M0 (noté *M0), on a $M(Pi) \leq K$ où K est un nombre entier naturel, on dit que Pi est **K-borné**. Si la propriété est vraie pour toute place on dit que ce RdP est **K-borné**.

I.3.6.2 RdP sauf :

C'est un RdP 1-borné (ou binaire).

I.3.6.3 *Vivacité :*

• Une transition **Tj est vivante** pour un marquage initial M0 si pour tout marquage accessible $Mi \in *M0$, il existe une séquence de franchissement S qui contient Tj à partir de Mi (c-à-d il subsistera toujours une possibilité de franchir Tj).

• Un **RdP est vivant** pour un marquage initial M0 si toutes ses transitions sont vivantes pour M0 (c-à-d aucune transition ne sera jamais définitivement infranchissable).

I.3.6.4 *Blocage :*

• Un blocage (ou état puits) est un marquage tel qu'aucune transition n'est validée.

• Un RdP est dit **sans blocage** pour un marquage initial M0 si ∀ marquage accessible Mi ∈*M0, il est sans blocage.

I.3.7.Graphe des marquages et arbre de couverture

Pour pouvoir trouver si tel RdP présente telle ou telle propriété, il existe principalement 3classes de méthodes :

a. Établissement du graphe de marquage ou de l'arbre de couverture

b. Utilisation des méthodes basées sur l'algèbre linéaire : résultats puissants.

c. Les méthodes de réduction des RdP.

I.3.7.1 Graphe des marquages :

Il est composé de nœuds qui correspondent aux marquages accessibles et d'arcs correspondant aux franchissements de transition faisant passer d'un marquage à l'autre.

I.3.7.2 Arbre de couverture :

Quand on ne peut pas construire le graphe des marquages (RdP non borné), on construit un arbre de couverture qui possède un nombre fini de noeuds. Un arbre est un graphe particulier dans lequel il n'y a pas de boucle ni de circuit.

I.3.8 Représentation matricielle

I.3.8.1 Matrice de description :

Considérons le RdP R={P, T, A,M0} comportant ***l*** places et ***m*** transitions.

R peut être représenté par une matrice dite de sortie S(P,T) et une matrice dite d'entrée E(P,T) définies comme suit :

- S(P,T) : matrice de dimension $l \times m$ dans laquelle :

- S(i,j)= poids de l'arc reliant Tj à Pi si Pi est une place de sortie de Tj.

- S(i,j)= 0 si Pi n'est pas une place de sortie de Tj.

- E(P,T) : matrice de dimension $l \times m$ dans laquelle :

- E(i,j)= poids de l'arc reliant Pi à Tj si Pi est une place d'entrée de Tj.

- E(i,j)= 0 si Pi n'est pas une place d'entrée de Tj.

Cas particulier :

Si aucune place du RdP n'est à la fois place d'entrée et place de sortie d'une même transition alors il est possible de décrire complètement le réseau avec la matrice d'incidence C(P,T) comme suit :

C(P,T) = S(P,T) – E(P,T) dans laquelle :

- C(i,j)= poids de l'arc reliant Tj à Pi si Pi est une place de sortie de Tj.

- C(i,j)= poids de l'arc reliant Pi à Tj précédé du signe (-) si Pi est une place d'entrée de Tj.

- C(i,j)= 0 si Pi n'est ni place d'entrée ni place de sortie de Tj.

I.3.8.2 *Validation des transitions :*

Si une transition Ti est validée pour un marquage initial M_0, il faut que :

$\forall Pj \in P, M(Pj) \geq E(Pj, Ti)$ Soit $|M_0| \geq E(P, Ti)$

En comparant successivement toutes les colonnes de E(P,T) au vecteur M_0, on peut trouver toutes les transitions validées par ce marquage.

On obtient aussi un vecteur de validation V de dimension égale au nombre de transitions du RdP (un 1 correspond à une transition validée, un 0 dans le cas contraire).

Cas particulier :

Si le RdP étudié **est sauf**, le vecteur V peut être obtenu en appliquant la relation suivante :

$V = E(P,T)^t \otimes M_0$

M_0 est le vecteur en complément à 1 de M_0.

$E(P,T)^t$: matrice transposée de E(P,T)

$\otimes$ est l'opérateur matriciel booléen obtenu en faisant terme à terme le produit des lignes de E(P,T)t et du vecteur colonne $\boldsymbol{M_0}$ puis la somme logique de ces produits.

I.3.8.3 ***Évolution de marquage :***

Les marquages successifs M1, M2, M3,… obtenus à partir du marquage initial M0 par les tirs successifs des transitions Ti, Tj, Tk,…peuvent être obtenus comme suit :

M1 = M0 + C(P, Ti)

M2 = M0 + C(P, Tj)

M3 = M0 + C(P, Tk)

Pour une séquence finie de tirs, il est possible de relier le marquage final Mn au marquage initial M0 par une relation de la forme :

|Mn| = |M0| = |C(P,T)| . |D|

Avec |D| : vecteur colonne de dimension égale au nombre total *m* de transitions du réseau.

Les composantes *di* de |D| sont des nombres entiers positifs correspondant au nombre de tirs de la transition Ti pour la séquence donnée aboutissant au marquage Mn.

Invariants

Composantes conservatives

Soit R un réseau de Pétri et P l'ensemble de ses places. On a un invariant de places (on dira simplement invariant) sil existe un sous ensemble de places $P' = \{P_1, P_2, P_3, \dots, P_r\}$ inclus dans P et un vecteur de pondération $q = (q_1, q_2, q_3, \dots, q_n)^T$, dont les poids q_i sont des nombres entiers positifs, tel que :

$$\sum_{i=1}^{r} q_i m_k(P_i) = Constante$$

Pour tout $M_k \in M_0$.

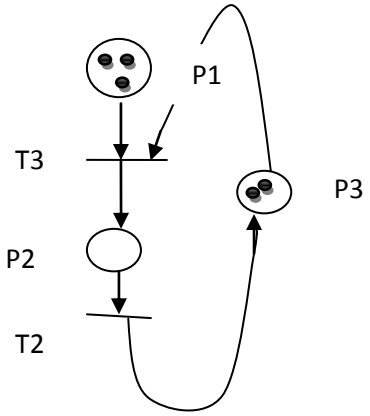
Dans ce cas l'ensemble des places P' est une composante conservative.

Le réseau R est dit conservatif si et seulement si P est une composante conservative.

Remarque :

La propriété d'être composante conservative est indépendante du marquage initial M_0. par contre, la constante d'invariant en dépend.

Exemple :



FigI.7 Exemple de composante conservative

I Composante répétitive

Soit R un réseau de Pétri, $T = \{T_1, T_2, T_3, \ldots., T_k\}$ l'ensemble des transitions de R et S une séquence de franchissements répétitive dans laquelle apparaissent r transitions de T. on note l'ensemble de ces transitions par $T' = \{T_1, T_2, T_3, \ldots., T_r\}$.

Une séquence de franchissements S est dite répétitive, si elle ramène à l'état initial :

$$S : \mathbf{M_0} \longrightarrow \mathbf{M_0}.$$

Alors, l'ensemble des transitions T' est une composante répétitive.

Si la séquence S contient toutes les transitions (c'est dire que $= k$), on dit quelle est complète.

Dans ce cas, T est une composante répétitive et le réseau R est dit répétitif.

NB : Le réseau R est répétitif si est seulement si T est une composante répétitive.

Si dans un réseau de Pétri, il existe deux séquences répétitives S1 et S2 minimales pour un marquage $\mathbf{M_0}$., l'union de ces deux séquences donne l'ensemble des séquences minimales pour $\mathbf{M_0}$ on note : $\boldsymbol{L(M_0) = S_1(M_0) + S_1(M_0)}$.

Propriété :

Soit $\boldsymbol{L(M0)}$ lensemble des sequences de franchissements minimales a partir du marqua

Si $M'_0 > \mathrm{M}_0$, alors $L(\boldsymbol{M'_0}) > L(\mathbf{M_0})$.

En particulier, si S est une séquence répétitive pour$\mathbf{M_0}$, elle est aussi une séquence répétitive pour $\boldsymbol{M'_0 > M_0}$.

Les Avances Synchroniques Maximales :

Les avances synchroniques maximales constituent des invariants relatifs aux franchissements des transitions du réseau de Pétri. Leurs valeurs dépendant du marquage initial $\mathbf{M_0}$. elles sont définis comme suite :

Soient :

S_k une sequence de franchissements et $\{s_k\}$ l'ensemble de ces sequences.

Tj et T1 deux transitions deS_k.

$N_k(T_j)$, le nombre d'occurrences de la transition T_j dans S_k.

Alors :

$A_k(T_j, T_i) = N_k(T_j) - N_k(T_i)$, est l'avance synchronique de la transition T_j par rapport a T_i.

$A_{max}(T_j, T_i) = max\{A_k(T_j, T_i)\}$, est l'avance synchrone maximale de la transition T_j par rapport a T_i.

Exemple :

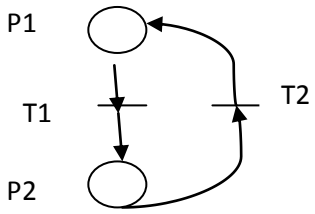


Fig I.8: RDP a 2 places.

Considérons le réseau de Pétri de la figure I.7

Si $\mathbf{M_0} = [\mathbf{1}, \mathbf{0}]^{\mathbf{t}}$, les sequences possibles sont :

$S_1 = T_1$ et $S_2 = T_1T_2$.

Donc,

$N_1(T_1) = 1$ et $N_1(T_2) = 0 \longrightarrow A_1(T_1, T_2) = 1$.

$N_2(T_1) = 1$ et $N_2(T_2) = 0 \longrightarrow A_2(T_1, T_2) = 0$.

$A_{max}(T_1, T_2) = max\{A_k(T_1, T_2)\} = 1$.

Si $M_0 = (N, 0)$, les sequences possibles sont :

$S_{10} = T_1$ et $S_{11} = T_1T_2 = T_1^2$.

$S_{20} = T_1^2$ et $S_{21} = T_1^2T_2$, .$S_{22} = T_1^2T_2^2$.

.

. $S_{N0} = T_1^N, \ldots, S_{NN} = T_1^N T_2^N$.

Donc,

$N_{10}(T_1) = 1$ et $N_{1o}(T_2) = 0 \longrightarrow A_{10}(T_1, T_2) = 1$.

$N_{11}(T_1) = 1$ et $N_{11}(T_2) = 1 \longrightarrow A_{11}(T_1, T_2) = 0$.

$N_{20}(T_1) = 2$ et $N_{20}(T_2) = 0 \longrightarrow A_{10}(T_1, T_2) = 2$.

$N_{21}(T_1) = 2$ et $N_{21}(T_2) = 1 \longrightarrow A_{11}(T_1, T_2) = 1.$

$N_{22}(T_1) = 2$ et $N_{22}(T_2) = 2 \longrightarrow A_{11}(T_1, T_2) = 0.$

.

$N_{N0}(T_1) = N$ et $N_{N0}(T_2) = 0 \longrightarrow A_{NN}(T_1, T_2) = N.$

.

$N_{NN}(T_1) = N$ et $N_{NN}(T_2) = N \longrightarrow A_{NN}(T_1, T_2) = 0.$

$A_{max}(T_1, T_2) = max\{A_{jk}(T_1, T_2)\} = N.$

Dans cet exemple, on Remarque que, T_1 $a\ une$ avance synchronique par rapport a T_2, mais T_2 ne peut pas avoir d'avance synchronique par rapport a T_1.

I.3.9 Modélisation par réseaux de Pétri

L'évolution d'un système se traduit par des séquences d'actions qui sont effectuées lorsque certaines conditions internes ou externes sont réalisées.

Lorsqu'un système est modélisé par un réseau de Pétri, on adopte un certain nombre de conventions qui font correspondre les places, les transitions et les jetons aux fonctions et aux conditions du système. Cette correspondance fait évidemment intervenir une part d'arbitraire, et c'est la raison pour la quelle il existe de nombreuses façons différentes de modéliser un système par réseau de pétri. La méthode de modélisation la plus simple et la plus naturelle consiste à associer :

Les places aux actions du système.

Les transitions aux conditions et ou aux actions en exploitant les conditions normales de sensibilisation des transitions.

Les jetons aux conditions et ou actions du système.

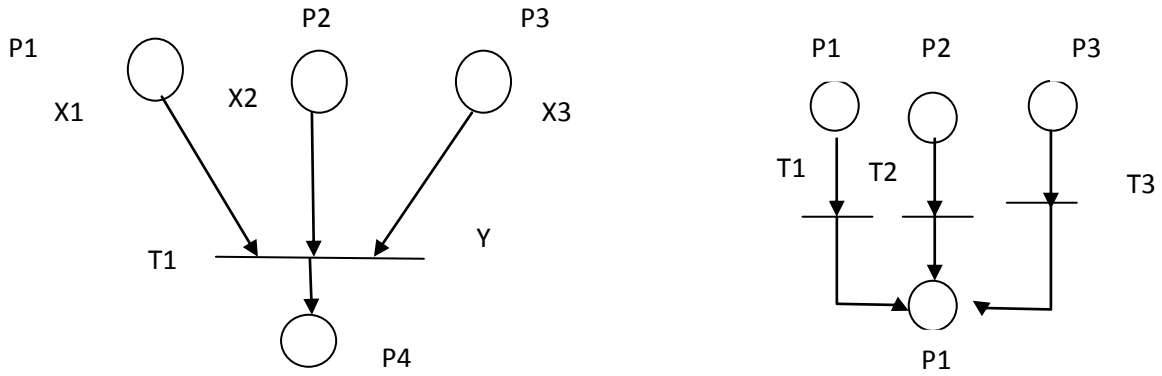
Compte tenu des propriétés des réseaux de pétri, ou la franchissement des transitions est suppose instantané, il est naturel de n'associer celles-ci que des actions de courtes durées. On aboutit ainsi a un réseau de Pétri étiqueté tel que celui de la figure I. dans lequel chaque étiquette spécifie un état ou une action, mais sans qu'il y ait de conditions extérieures imposées pour le franchissement des transitions.

Exemple de modélisation par réseaux de Pétri

a. Modélisation des opérations logiques 'ET' et 'OU'

Avec un réseau de pétri, les variables sont représentées par un nombre de jetons qui peut être positif ou nul. Si les arcs ont tous un poids égal a 1, le réseau de Pétri permet de modéliser des opérations qui peuvent être considérées dune façon simplifiée comme des opérations logiques. En effet, considérons par exemple le réseau de pétri de la figure 9.a, la transition T1est sensibilisée lorsque la condition logique ET sur les trois variables x1,x2, x3 est vérifiée. Par conséquent, si on associer a la transition T1 une variable binaire y qui prend la valeur 1 au moment du tir et 0dans le cas contraire, cette variable matérialisera la condition.

Dune façon similaire, et avec les mêmes restrictions que précédemment, une condition de type OU peut être modélisée comme indique sur la figure 9.b,



$y = x_1 x_2 x_3$ $y = x_1 + x_2 + x_3$

a **b**

figure I.9 : a) type ET et b) type OU

Autres exemples

A test avec pondéré

Dans le cas le plus général, les places d'un réseau de pétri peuvent contenir plus d'un jetons, et les arcs ont des pondérations différentes de 1. On associe alors aux places des variables entières qui prennent une valeur égale au nombre de jetons présents dans la place. Avec cette convention, les variables peuvent être modifiées par les deux primitives suivantes, qui correspondent respectivement à une opération d'entrée et à une opération de sortie (figureI.10)

a) Si x$X \geq a$, alors :

x : = x-a

Et y : = y+a

b) Si x$X \geq a$, alors :

y : = y+b Et y : = y

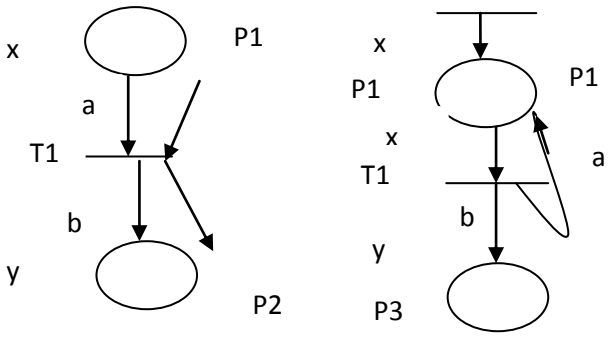


Figure I.10 : A test avec pondéré

a)RDP avec arcs pondérées b) test $x \geq a$

b) Test avec places complémentaires

La réalisation du test dune variable d'un réseau de pétri est plus délicate sil s'agit de provoquer une action lorsque la variable est inferieure a une constante donnée. Il faut alors mettre en œuvre des paires de places complémentaires telles que le nombre total de jetons présents dans les deux places dune paire est constant. Pour que deux places, P1 et P2 soient complémentaires, inferieur, il faut que chaque fois qu'un nombre ai de jetons est mis dans une des deux places, un nombre égal de jetons soit retire de l'autre place. Ceci conduit a un réseau de Pétri élémentaire du type de celui de la figure

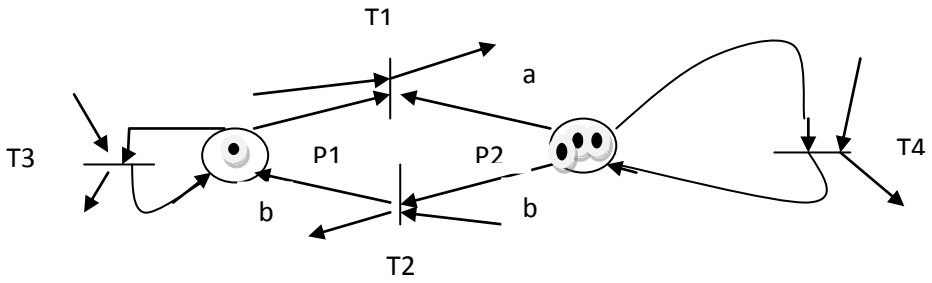


Figure I.11 Places complémentaires.

En utilisant cette technique des paires complémentaires, il devient possible de tester si une variable est inferieure a un constante donnée. Ainsi, le réseau de Pétri élémentaire de la figure avec un marquage initial de un jeton dans la place P2 permet de réaliser le test suivant :

Si $x > 0$, alors :$y_1 = y_1 + 1$

Si $x = 0$, alors :

$y_2 = y_2 + 1$

$y_3 = y_3 + k$

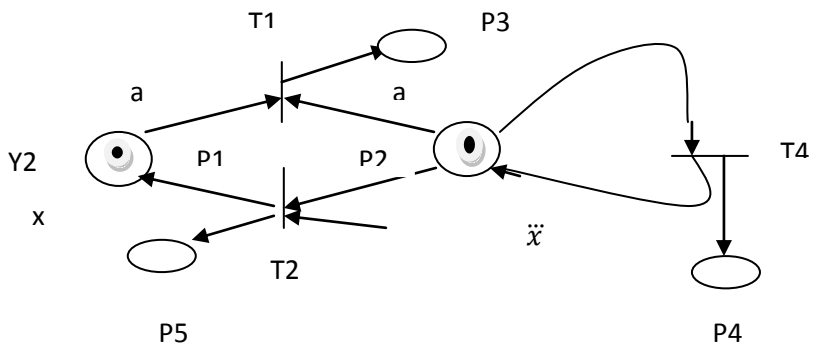


Figure I.12 : Représentation du test $x > 0, x > 0$.

I.3.10 RdP synchronisé

Un ensemble **d'événements externes** est associé au RdP ; ces événements permettent le franchissement de certaines transitions. Un tel RdP est dit **synchronisé**. Considérons le RdP modélisant la machine décrite ci-dessous. On associe à ce RdP l'ensemble d'événements *A, D, S* où *A* désigne l'événement « *Arrivée pièce* », *D* l'événement « *Démarrage service* », *S* l'événement « *Sortie pièce* ». La figure représente le système modélisé par un RdP synchronisé.

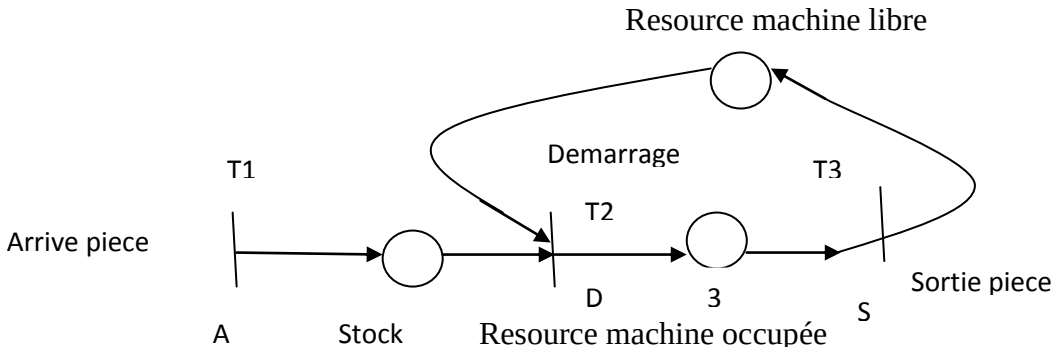


Figure I.13 : RdP synchronisé

• Le tir de la transition T1 est lié à l'occurrence de l'événement *A.*

• Le tir de la transition T2 est lié :

- A la validation de la transition, matérialisée par la présence d'au moins un jeton dans la place « *stock* » et d'un jeton dans la place « *ressource machine libre* » ;

- Au démarrage effectif du service (occurrence de l'événement *D*).

• Le tir de la transition T3 est lié à l'occurrence de l'événement *S.*

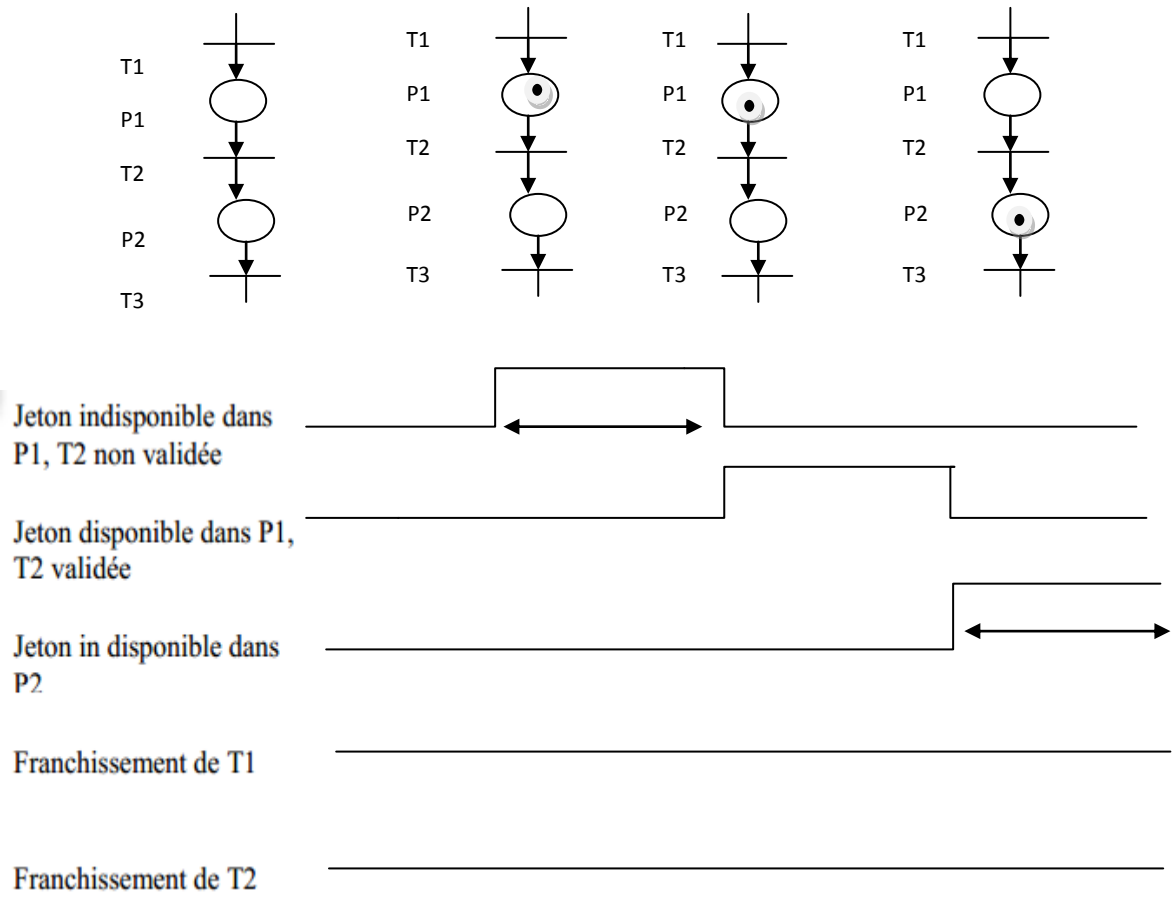
I.3.11 RdP temporisé

Décrit un système dont le fonctionnement dépend du temps.

a. RdP P-temporisé :

- Une durée *minimale* de séjour dans les places : Durée pendant laquelle tout jeton qui vient d'être produit dans une place ne peut pas encore servir à l'activation de transitions aval.

Illustration :
Jeton réservé, Jeton non réservé.



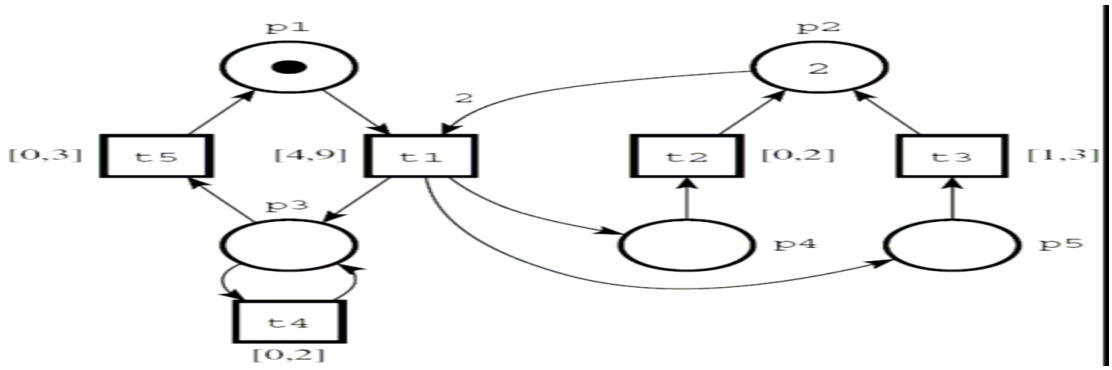
b. RdP T-temporisé :
- Une durée d'activation pour les transitions : Durée pendant laquelle un jeton situé dans chaque place amont de la transition activée est « réservé » pour cette transition (avant de disparaître), et au delà de laquelle un jeton apparaît dans chacune des places aval ;

Illustration :

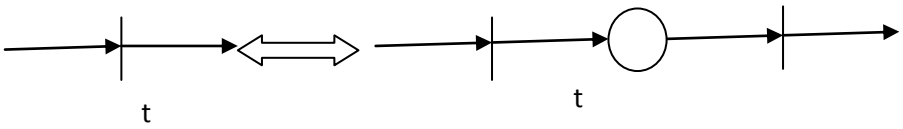
Jeton indisponible

Jeton disponible

***Remarque* :** Il y a un temps minimum et un temps maximum pour le franchissement d'une transition



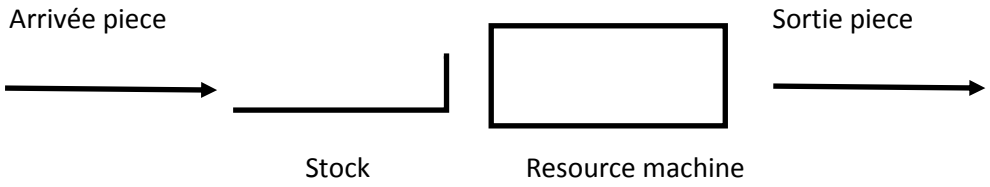
Remarque : Il existe une équivalence entre le RdP T-temporisé et celui P-temporisé. La figure suivante montre la transformation d'une transition de durée *t* en 2 transitions instantanées (le début et la fin) séparées par une place de temporisation *t*.



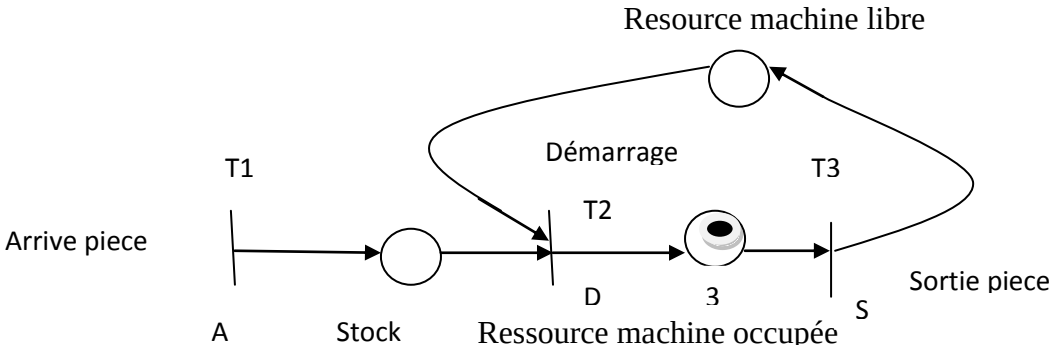
I.3.12. Exemples divers

Exemple 1 :

Soit une machine représentée dans la figure suivante. Chaque pièce qui arrive est, soit traitée immédiatement par la ressource machine, soit mise en attente dans le stock (à capacité infinie) jusqu'à ce que la ressource machine soit disponible. Le temps de traitement de la ressource machine est de 3 unités de temps. Après traitement, chaque pièce sort.



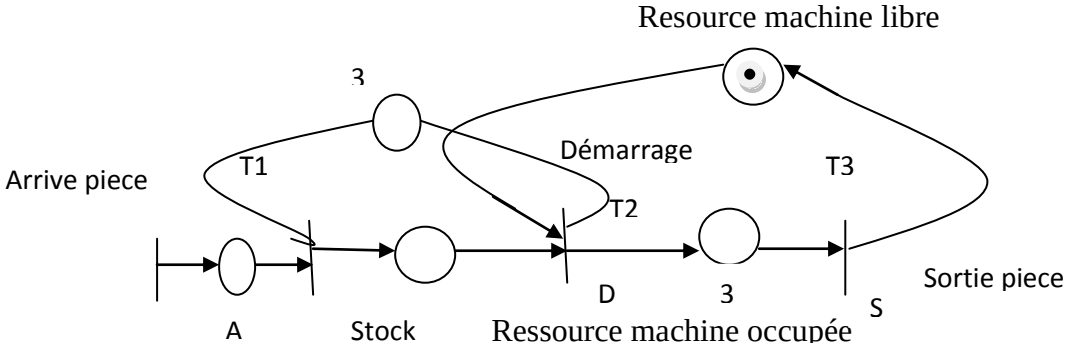
Le RdP suivant modélise ce système.



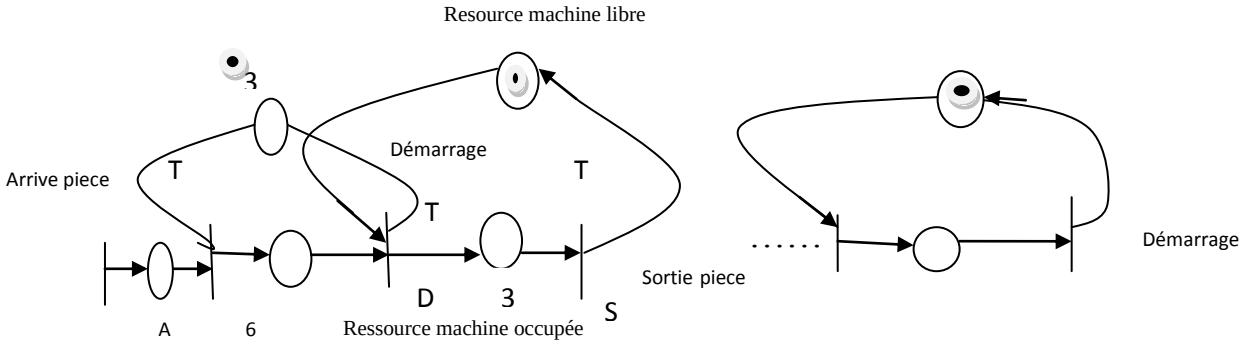
L'*état* du système modélisé par le RdP est représenté par le *marquage* définissant le nombre de jetons contenus dans chaque *place*. L'évolution de l'état (représentant la dynamique du système) correspond à l'évolution du marquage (produit par le franchissement de transitions).

Modifications :

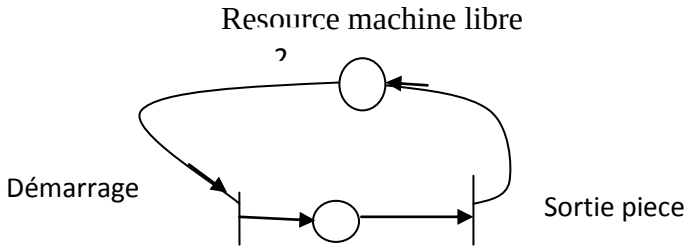
a) Le modèle RdP suivant indique une capacité de stockage limitée à 5 pièces.



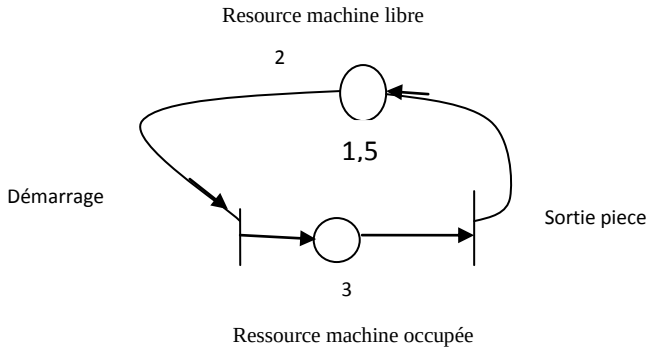
b) Le stock en amont de la ressource machine est remplacé par un convoyeur correspondant à une file composée de 5 compartiments (gestion *First-In, First-Out* du convoyeur). Le temps de déplacement du convoyeur est de 6 unités de temps. Le système est représenté par le modèle RdP suivant.



c) La machine a une capacité de traitement de 2 : Elle est capable de traiter 2 pièces simultanément. Le système est représenté par le modèle RdP suivant.

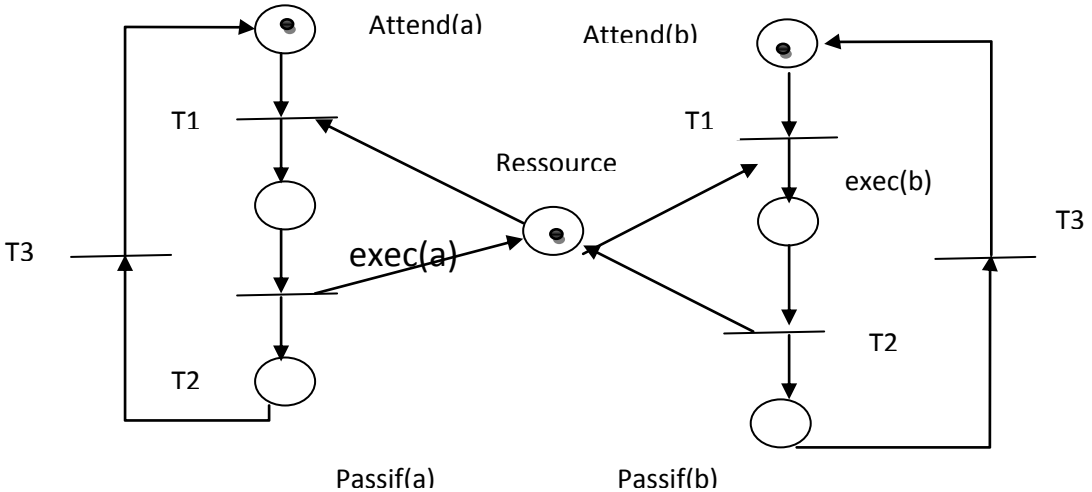


La machine a un temps d ₃ *tup* de 1,5 unités de temps. Le système est représenté par le mod



Exemple 2 :

Exclusion mutuelle : Nous voulons modéliser l'exclusion mutuelle de deux processus sur une source commune, ce qui donnera le RdP marqué suivant :



Exemple 3 :

Cet exemple est emprunté au domaine de l'intelligence artificielle et à la problématique de satisfaction de contraintes. L'énoncé est le suivant :

• John se rend à son travail en utilisant sa voiture (la durée de son trajet est alors de 30 à 40 minutes) ou le bus (au moins 60 minutes).

• Fred se rend à son travail en utilisant sa voiture (durée 20 à 30 minutes) ou un système de covoiturage (40 à 50 minutes).

• Aujourd'hui John a quitté son domicile entre 7h10 et 7h20, Fred est arrivé à son travail entre 8h00 et 8h10. De plus John est arrivé 10 à 20 minutes après que Fred ait quitté sa maison.

Il s'agit de répondre à trois questions :

1) Les contraintes temporelles figurant dans ce scénario sont-elles consistantes ?

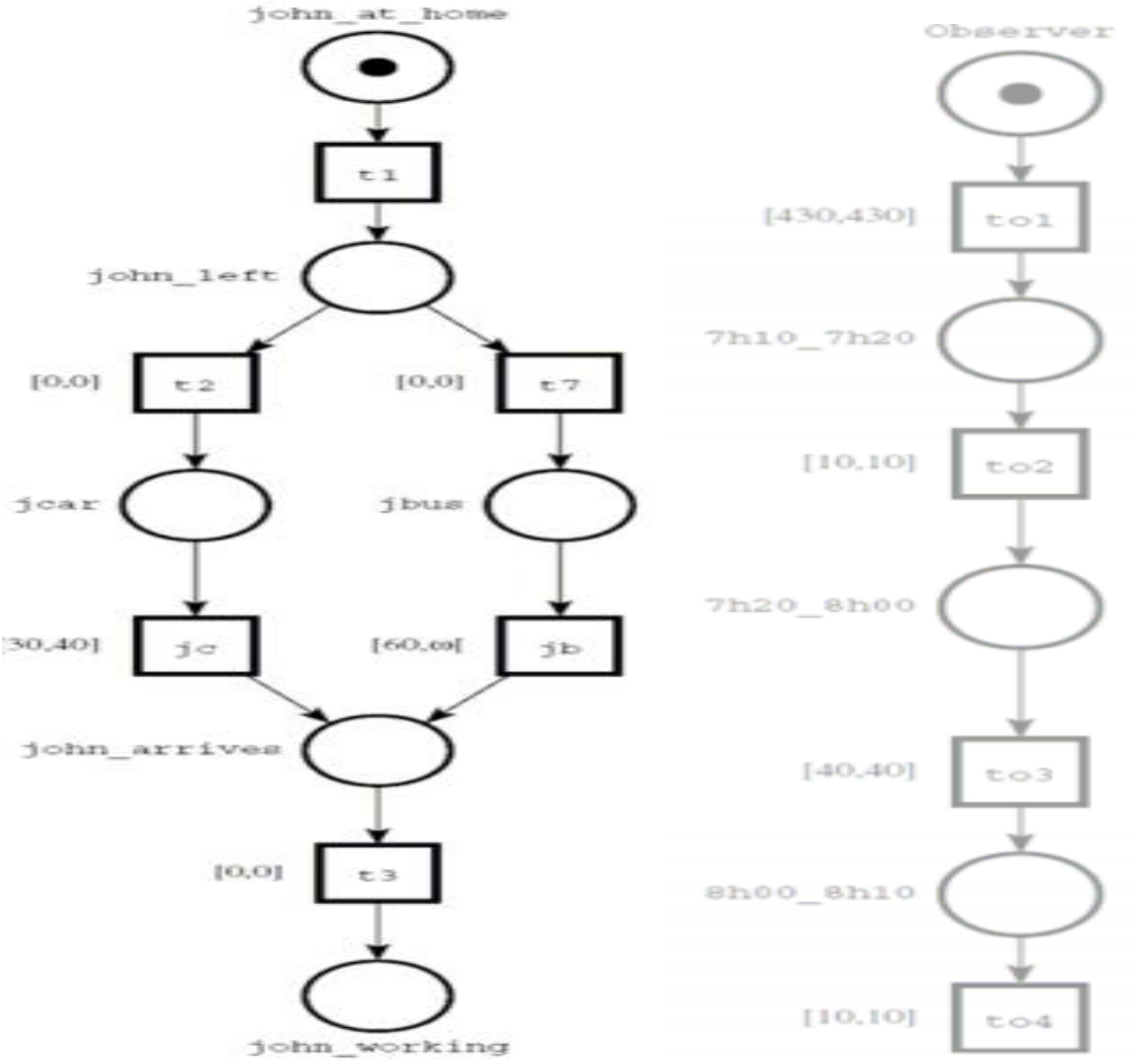
2) Est-il possible que Fred ait pris le bus et John utilisé le covoiturage ?

3) Dans quelle plage horaire Fred a pu partir ?

La modélisation du problème en réseau temporel est donnée figure ci-dessous Les parties non grisées correspondent aux comportements respectifs habituels de John et de Fred.

La partie grisée spécifie deux «observateurs» mis en place pour prendre en compte les contraintes temporelles spécifiques au scénario considéré. Sur la composante centrale

sont représentées les contraintes absolues : heure de départ de John, heure d'arrivée de Fred. La contrainte relative stipulant que «John est arrivé entre 10 et 20 minutes après que Fred ait quitté sa maison» est représentée par un second observateur qui complète le réseau de Fred en déclenchant un «chronomètre» après que Fred ait quitté sa maison.


john_at_home
t1
john_left
[0,0]
t2
[0,0]
t7
jcar
jbus
30,40]
jc
[60,∞[
jb
john_arrives
[0,0]
t3
john_working
Observer
[430,430]
to1
7h10_7h20
[10,10]
to2
7h20_8h00
[40,40]
to3
8h00_8h10
[10,10]
to4

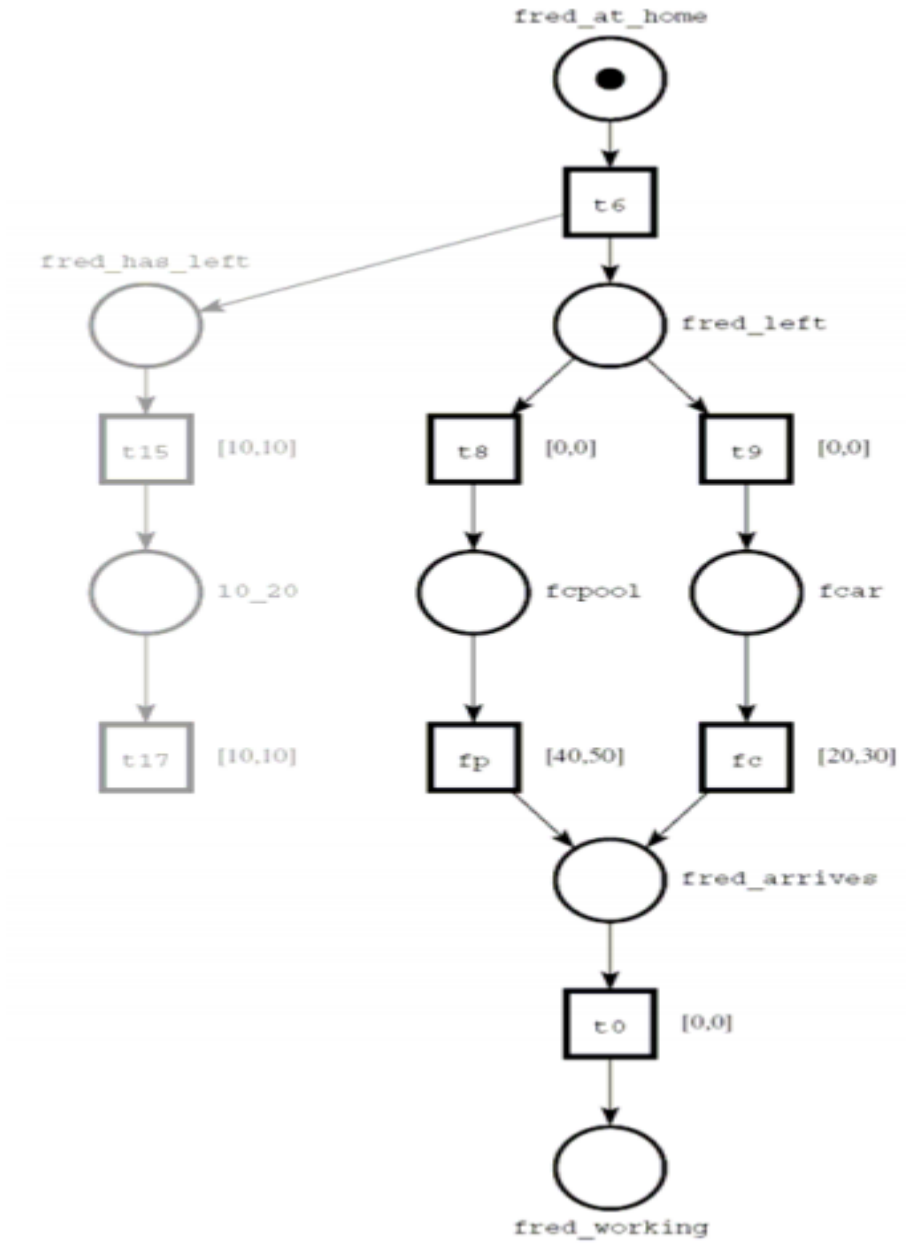


Figure I.14 : Modélisation en réseau temporel de John et Fred.

II .1 Introduction

Au début des années 70, le bilan de la conception des systèmes logiques n'était pas brillant. Chez les chercheurs, on constatait encore une importante dissipation d'énergie pour perfectionner les méthodes théoriques de synthèse (Huffman, expressions régulières...) afin de les rendre plus accessibles, plus efficaces et "optimales". Dans l'industrie, ces travaux étaient ignorés ou jugés avec quelque sévérité; ils avaient en tous cas peu d'impact sur les méthodes pratiques de synthèse qui restaient dans l'ensemble empiriques et rattachées à la technologie des automatismes à relais.

Avec l'arrivée des technologies nouvelles et l'accroissement de la complexité des systèmes étudiés. Les praticiens prirent conscience que l'approche empirique n'était ni sûre, ni adaptée à leurs besoins.

Le premier pas pour sortir de cette impasse fut accompli par P. GIRARD qui très tôt introduisit les notions de réceptivité et d'étape. A la même époque, aux Etats-Unis des équipes d'informaticiens totalement étrangers aux problèmes de synthèse des automatismes logiques, découvraient les réseaux de Pétri que celui-ci avait définis en 1962. La voie nouvelle dans laquelle s'engageait la conception des systèmes logiques se précisait alors. Deux mots clefs la définissent : cahier des charges et modélisation. C'est en 1975 que fut crée une "commission de normalisation de la représentation du cahier des charges d'un automatisme

logique" dans le cadre du groupe de travail "systèmes logiques" de l'AFCET1. L'objectif de la commission était clair : homogénéiser les différentes approches afin de dégager un outil unique de représentation d'un cahier des charges.

En 1977, deux ans après sa création, la commission dans son rapport final, définissait le contenu d'un cahier des charges et ses différents niveaux d'élaboration et proposait un outil pour sa représentation :

le GRAFCET2.

Le groupe de travail "Automatisation séquentielle "de ADEPA3 reprit le flambeau pour donner au GRAFCET, sans toucher au fond, une forme normalisée tenant compte des normes existantes et des usages généraux des symboles normalisés.

A la demande du GlMEE4 et au sein de l'UTE5, un groupe d'étude élargi réunissant les experts des diverses commissions intéressées auxquels se sont joints des experts du CNOMO6, de I'UNM7, de Education Nationale de L'AFNOR8 etc., a établi à partir des travaux effectués par l'AFCET et

l'ADEPA un projet aboutissant à la norme NF C03-190 de Mai 1982. Le GREPA9, équipe d'industriels et de formateurs, améliora le concept du GRAFCET en proposant la notion de Macro étape et de Forçage. Ceci fut l'objet d'un projet de norme référencé NF C03-191 en 1993. En 1988 le GRAFCET obtint ses galons internationaux avec la publication par la CEI10 de la norme CEI/IEC 848. Dès 1978, l'Inspection Générale des Sciences et Techniques Industrielles, convaincue du rôle fondamental de cet "outil" pour l'enseignement

des automatismes industriels créent une commission de travail en liaison avec l'ADEPA et lance parallèlement des expérimentations en Lycées techniques. En 1980 le GRAFCET est introduit dans les programmes d'enseignement. Il est également enseigné dans plusieurs pays comme l'Autriche, la Suisse, la Hollande, la Suède...

En résumé, voici sur quelles bases a été conçu le GRAFCET:

- Utilisation d'un procédé graphique, choix des symboles graphiques
- Mise en évidence de chacune des situations de l'automatisme séquentiel à un moment donné, notion de situation, d'étape et d'action associées
- Sélection des seules informations utiles à l'évolution de l'automatisme à partir d'une situation connue, notion de transition et de réceptivité
- Définition des conditions d'évolution entre les étapes, établissement des règles d'évolution
- Description progressive de l'automatisme, notion de niveau.
- Emploi d'un langage simple et accessible à l'ensemble des intervenants depuis le concepteur jusqu'à l'agent de maintenance, choix d'un vocabulaire.

II.2 NECESSITE D'UN OUTIL NORMALISE

Le cahier des charges d'un automatisme est un document régissant les rapports entre le fournisseur concepteur d'un matériel de commande et son client futur utilisateur de ce matériel. Un tel document peut donc faire intervenir des considérations judiciaires, commerciales, financières technico économiques ou

purement techniques. Nous nous placerons dans tout ce qui suit du seul point de vue du technicien : ce qu'il recherche dans un cahier des charges, c'est avant tout une description claire, précise, sans ambiguïtés ni omissions du rôle et des performances de l'équipement à réaliser.

Force est de constater que dans la plupart des cas cette description reste confuse, vague, ambiguë et incomplète d'ou un risque permanent de mauvaises interprétations qui si elles ne sont pas détectées à temps, peuvent conduire à des catastrophes, ne serait-ce qu'au plan financier.

A l'origine le Grafcet est donc un outil de communication qui s'attache exclusivement a la description fonctionnelle des automatismes.

Il a été normalisé en juin 1982 (NF C 03-190). En 1985 le GREPA a publié de nouveaux concepts.

Une norme européenne est publiée en 1988 (CEI 848)

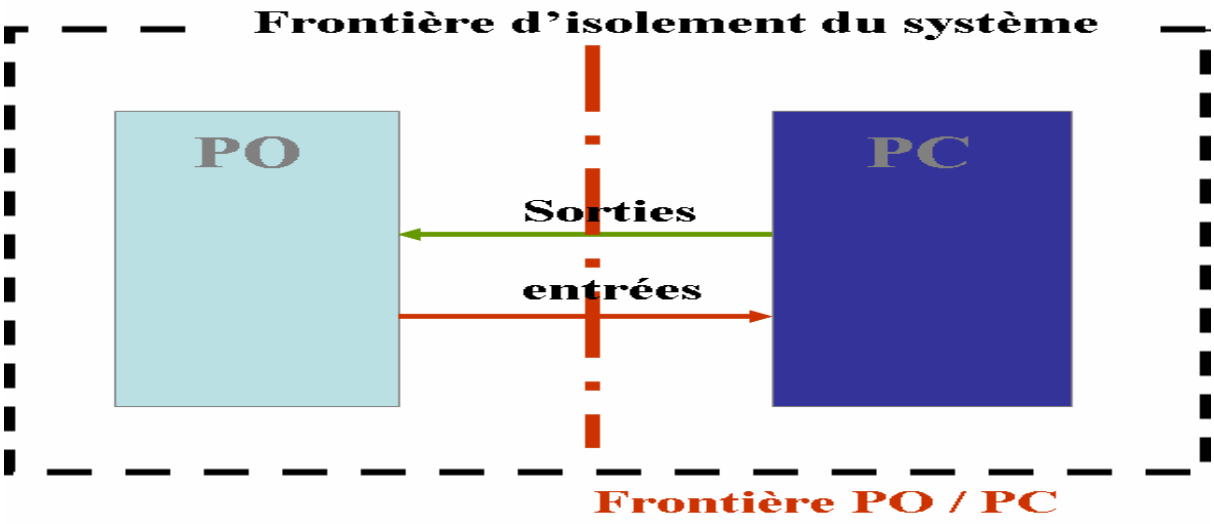
Une nouvelle norme française (suite à la publication internationale) est parue en 1990, complémentée en 1993. Une version complète sera publiée en 1995.

La dernière version, prenant en compte les concepts de description structurée et hiérarchisée est en vigueur depuis août 2002. Le concept de grafcet a été intégré dans la norme européenne sur les langages de programmation des automates (CEI 61131-3) sous la dénomination SFC11.

II. 3 PRINCIPES GENERAUX

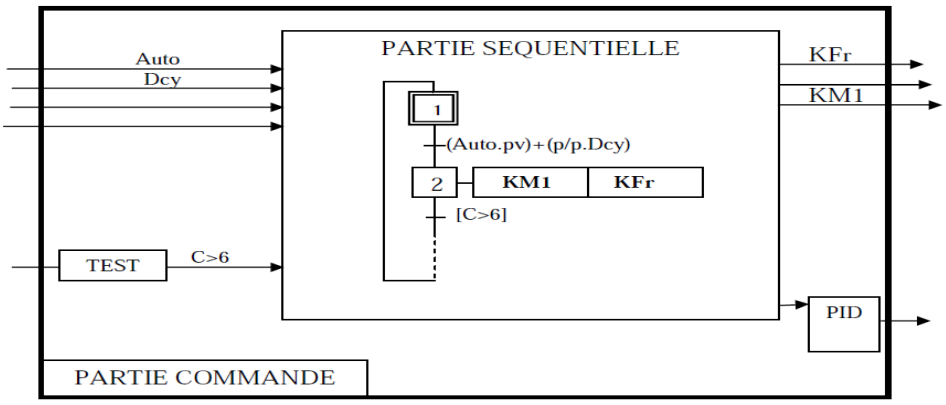
II.3.1 contexte

Le grafcet décrit les interactions entre la partie opérative (PO) et la partie commande (PC) d'un système isolé. Cette première frontière d'isolement permet de définir les limites de l'étude.



Dans la partie commande, la partie séquentielle du système est caractérisée par ses variables d'entrées, ses variables de sorties et son comportement.

La description par un grafcet de la partie séquentielle d'un système impose que les interactions entre la PO et la PC soient définies.



Autrement dit, il est nécessaire de caractériser toutes les entrées/sorties avant de pouvoir faire une description par grafcet.

II.3.2 Présentation sommaire

Le GRAFCET est utilisé pour concevoir des Grafcets donnant une représentation graphique du comportement des systèmes. La représentation distingue :

- **La structure** qui décrit les évolutions séquentielles possibles
- **L'interprétation** qui fait la relation entre les E/S et la structure.

II.3.2.1 la structure

La structure est constituée des éléments de base suivants

Etape: utilisée pour définir la situation de la partie séquentielle du système

Une étape est soit active soit inactive. L'ensemble des étapes actives représente la situation du grafcet.

- Transition : indique la possibilité d'évolution d'activité entre deux ou plusieurs étapes. Cette évolution se traduit par le franchissement de la transition (voir règles)
- Liaison orientée : indique les voies et le sens d'évolution.

II.3.2.2 l'interprétation

L'interprétation se fait grâce aux éléments suivants :

- **Réceptivité** : associée à chaque transition, elle est une condition logique qui est soit vraie soit fausse. Elle est composée de *variables d'entrées*
- **Action** : indique le comportement d'une *variable de sortie*, soit par assignation, soit par affectation.

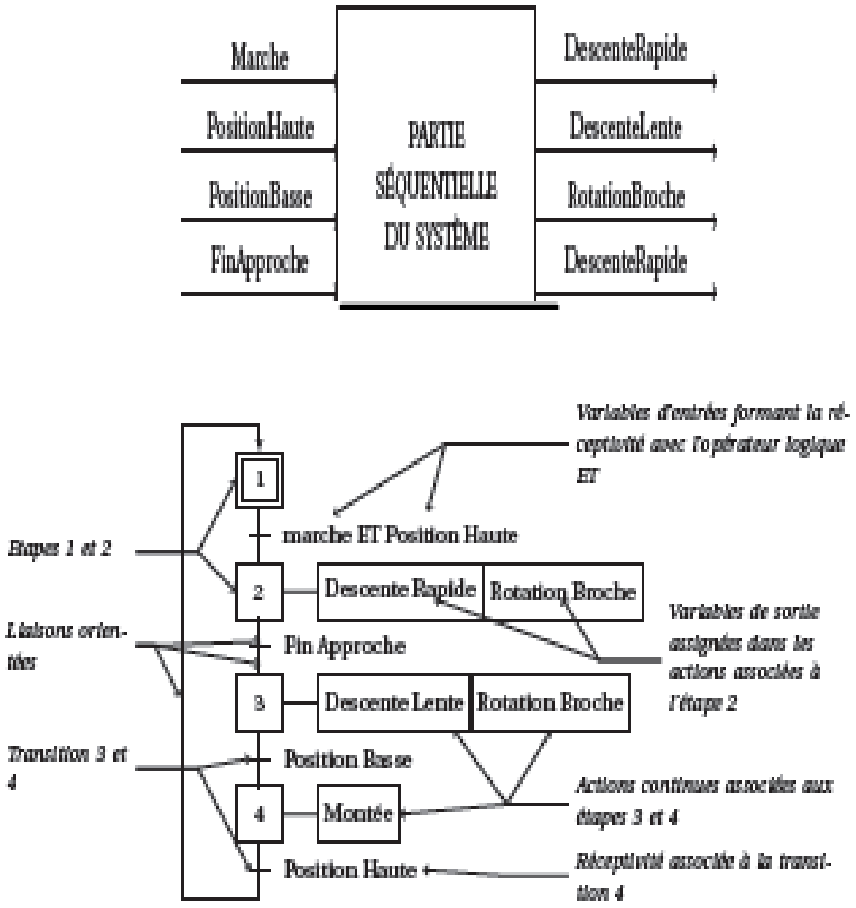


FIGURE 2.1 – Structure et éléments du GRAFCET

Le GRAFCET est un outil de description du comportement déterministe de la partie commande.

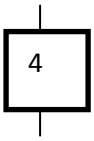
II. 4 Eléments graphique de base

Les étapes

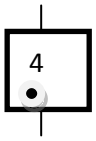
Une étape caractérise un état. Elle est représentée par un carre et un nom i . Elle est associée a la variable binaire Xi, dite variable détape. Une étape soit active, soit inactive.

Le nom de l'étape est inscrit dans le carre (au milieu de la partie haute). Un point peut caractériser une étape active (on peut aussi griser la carre). Un cercle dans une étape peut indiquer quelle vient d'être désactivée($X_i = 0$).

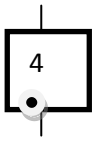
Etape 4 inactive (X4=0)



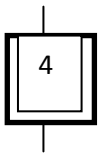
Etape 4 active (X4=1)



Etape 4 venant d'être désactivée



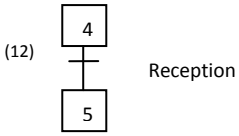
Etape initiale, active a l'initialisation du système



Les étapes encapsulantes est les macro - étapes seront vues plus loin.

Les transitions

Une transition est représentée par un petit trait horizontal coupant une liaison verticale : la réceptivité de cette transition est placée à droite. Une transition peut être exceptionnellement représentée par un petit trait vertical sur une liaison horizontale.

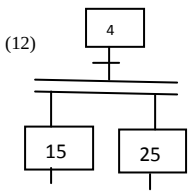


Une transition indique une et une seule possibilité d'évolution entre deux ou plusieurs étapes. Cette évolution s'accomplit par le franchissement de la transition si la réceptivité est vraie : la partie commande change alors de situation.

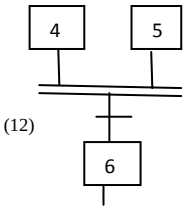
La transition est ici identifiée par (12).

Exemples

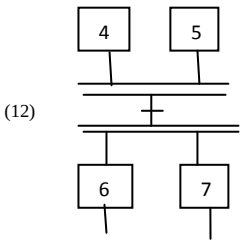
Divergence en ET ou activation de séquences parallèles



Convergence en ET ou synchronisations de séquences parallèles



Synchronisation et activation de séquences de séquences parallèles



Pour les convergences et divergences en ou.

Les liaisons orientées

Par convention, les évolutions se font du haut vers le bas. Dans le cas contraire, il est nécessaire d'indiquer le sens de l'évolution par une flèche.

Les lignes obliques peuvent exceptionnellement être utilisées.

Si une ligne horizontale croise une ligne verticale, il n'y a aucune relation entre elles.

Dans le cas de divergences ou convergences en **OU**, il faut donc :

Lorsqu'un grafcet est de taille trop importante pour être inscrit sur une seule feuille, on utilise des renvois :

Partition d'un grafcet

Un grafcet global peut être compose de plusieurs grafcets partiels qui seront notes Gi, par convention. La variable d'état d'un grafcet partiel Gi est notée XGi : elle vaut 1 quand au moins une des étapes du grafcet partiel Gi est active.

II.5 Règles de syntaxe et règles d'évolution

II.5.1 Règles de syntaxe

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

Deux étapes ne doivent jamais être reliées directement : elles doivent être séparées par une transition. Deux transitions ne doivent jamais être reliées directement : elles doivent être séparées par une étape.

II.5.2 Les cinq règles d'évolution

Règle 1 : situation initiale du GRAFCET

La situation initiale doit être précisée par une ou plusieurs étapes actives au début du fonctionnement, a la mise en en énergie de la partie commande. Si cette situation initiale est toujours identique (cas des automatismes cycliques), elle sera caractérisée par les étapes initiales.

Dans le cas ou cette situation initiale n'est pas provoquée par une mise en énergie, elle peut par exemple être convoquée par un ordre de forçage. Un ordre de forçage d'un grafcet partiel est représenté dans un double rectangle associe a une étape.

Quelques exemples de situations du grafcet partiel G2 provoquées par le forçage

$G2\{\ \}$ Situation ou le grafcet partiel G2 est totalement désactivé.

$G2\{INIT\}$ Situation ou le grafcet partiel G2 est initialise à ses étapes initiales.

$G2\{20, 22\}$ Situation ou les étapes 20 et 22 du grafcet partiel G2 sont actives et toutes les autres étapes sont désactivées.

$G2\{*\}$ Situation ou le grafcet partiel G2 est "fige" (maintien dans l'état).

Règle 2 : Transition franchissable

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

* Toutes les étapes qui précèdent immédiatement la transition sont actives.

* La réceptivité associée à la transition est vraie.

Règle 3: Franchissement dune transition

Le franchissement dune transition entraine simultanément la désactivation de toutes les étapes immédiatement précédentes et l'activation de toutes les étapes immédiatement suivantes.

Règle 4: Franchissement simultanés

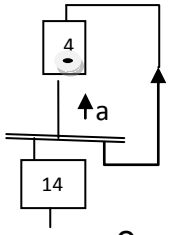
Plusieurs transitions simultanément franchissables sont simultanément franchies.

L'évolution entre deux situations actives implique qu'aucune situation intermédiaire ne soit possible, on passe donc instantanément d'une représentation de la situation par un ensemble d'étapes à une autre représentation.

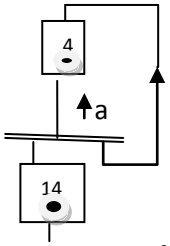
Règle 5 : Activation et désactivation simultanées

Si au cours du fonctionnement d'un automatisme, une même étape doit être désactivée et activée simultanément, elle reste active.

Cette règle exprime en fait que le système ne possède pas d'état instable, et que la durée du passage d'un état stable vers un autre est nulle. Il est à remarquer que dans la réalité, cette durée de franchissement ne peut être rigoureusement nulle, même si théoriquement elle peut être rendue aussi petite que l'on veut. Elle est fonction du temps de réponse des composants technologiques constituant l'automatisme.



a = 0



si a passe a 1

Ci-dessus l'étape 4 restera toujours active.

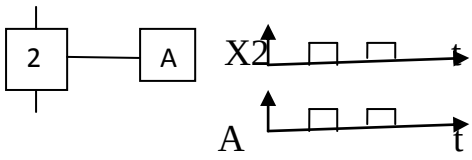
II.6 Les actions associées aux étapes

II.6.1 Actions continues (assignation sur état)

L'action continue, associée à une étape, dure tant que l'étape est active si aucune condition d'assignation ne l'interdit. Les conditions d'assignation sont définies dans les actions conditionnelles ainsi que dans les actions retardées et ou limitées dans le temps.

Plusieurs actions peuvent être associées à une même étape.

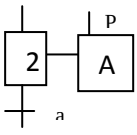
Dans les exemples ci-dessous, l'action"A " peut être remplacée par "ouvrir vanne" ; "monter", etc…



La sortie A est assignée à la valeur vraie quand l'étape 2 est active.

On note : A = X2

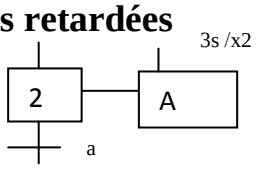
Actions conditionnelles



La sortie A est assignée a la valeur vraie quand l'étape 2 est active, a la condition que la variable p soit vraie. On note : A= X2.P

La condition d'assignation p ne doit jamais être un front de variable.

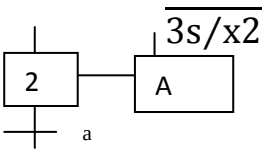
Actions retardées



La condition d'assignation est 3s/ x2. On note : A= 3s /x2.

Si la durée d'activité de l'étape 2 est inferieur a 3s, la sortie A ne sera pas assignée a la valeur vraie.

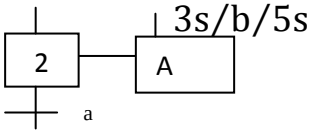
Actions limitées dans le temps



La condition d'assignation est $\overline{3s/x2}$. On note : A= $\overline{3s/x2}$.

La sortie A sera assignée a la valeur vraie pendant au plus 3s, des l'activation de letape2.

Actions retardées et limitées dans le temps



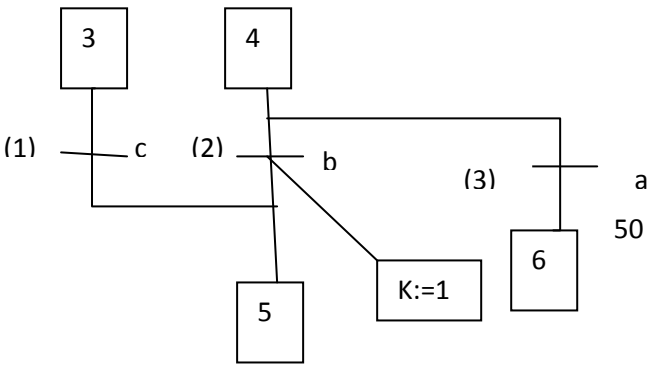
La condition d'assignation $3s/b/5s$ est vraie 3s après un front montant de b et 5s après le premier front descendant de b suivant. Si b passe a 0 avant 3s, alors la condition d'assignation ne peut pas etre vraie : il faudra attendre le prochain$\uparrow b$. On note A=$x2.3s/b/5s$ ne sera pas assignée a la valeur vraie.

Actions mémorisées

Une affectation mé

sortie qui la conserve. L'action mémorisée doit obligatoirement être associée à un événement interne. A l'initialisation la valeur de cette sortie est nulle.

Action au franchissement

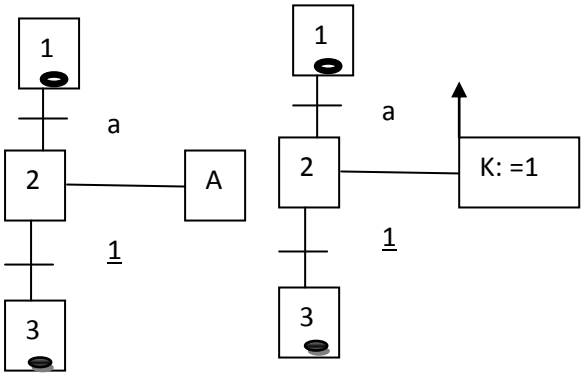


La valeur 1 est affectée à la variable booléenne K quand la transition (2) est franchie.

A noter qu'il est impossible d'obtenir le même résultat avec une action à l'activation de l'étape 5 ou avec une action avec la désactivation de l'étape 4.

II.6.2 Evolution fugace

Une évolution fugace se produit quand plusieurs transitions successives sont franchies sans événement d'entrée supplémentaire/ les deux exemples ci-dessous mettent aussi en évidence une évolution fugace liée a l'instabilité de l'étape 2.

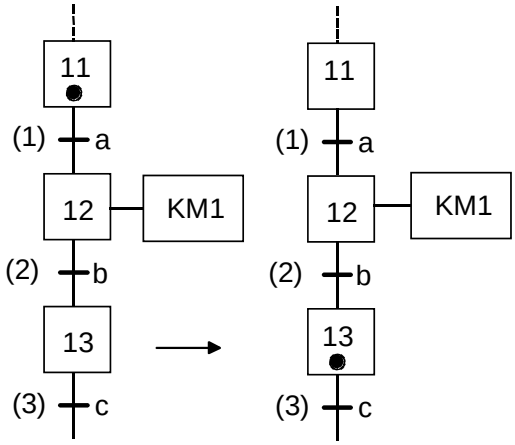


Si l'étape 1est active et que variable a passe de 0 a 1, l'étape 3 est directement activée. L'étape 2 est instable et les conséquences sont les suivantes :

Une action continue associée a une étape instable n'est pas assignée a la valeur 1 : A ne sera pas vraie.

Une action mémorisée associée à une étape instable permet d'assigner la valeur a la variable : on a K=1.

II.6.2.1 Conséquence d'une évolution fugace sur les assignations



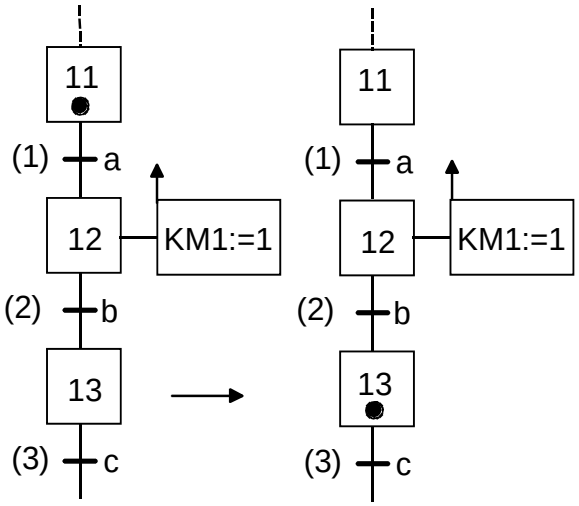
▪ Exemple d'action continue associée à une étape instable

Situation antérieure : étape 11 active, a=0, b=1 et c=0

Le changement de valeur de « **a** » conduit directement à la situation postérieure : étape 13 active.

La situation antérieure (étape 11 active) et la situation postérieure (étape 13 active) assignent la valeur 0 à la variable de sortie KM1.
L'étape instable 12 n'étant pas réellement activée, l'assignation à la valeur 1 de KM1 n'est pas effective au cours de cette évolution fugace.

II.6.2.2 Conséquence d'une évolution fugace sur les affectations



▪ **Exemple d'action mémorisée associée à l'activation d'une étape instable**

Situation antérieure : étape 11 active, a=0, b=1 et c=0

Le changement de valeur de « **a** » conduit directement à la situation postérieure : étape 13 active.

L'affectation de la valeur 1 à la variable de sortie KM1 est effective car elle est la conséquence de l'activation virtuelle de l'étape 12.

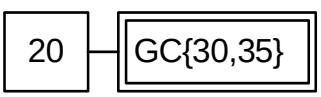
II.7 Structuration par forçage d'un GRAFCET partiel

L'ordre de forçage de situation émis par un GRAFCET hiérarchiquement supérieur permet de modifier la situation courante d'un GRAFCET hiérarchiquement inférieur, sans qu'il y ait franchissement de transition.

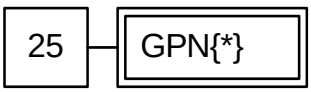
L'ordre de forçage est un ordre interne prioritaire sur toutes les conditions d'évolution et a pour effet d'**activer la** ou **les étapes** correspondant à la **situation forcée** et de **désactiver les autres étapes** du GRAFCET forcé.

L'ordre de forçage est représenté dans un double rectangle associé à l'étape pour le différencier d'une action.

Lorsque l'étape 2 est active, le GRAFCET nommé GPN est forcé dans la situation caractérisée par l'activité de l'étape 10 (l'étape 10 est activée et les autres étapes sont désactivées).



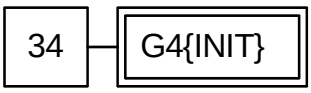
Lorsque l'étape 20 est active, le GRAFCET nommé GC est forcé dans la situation caractérisée par l'activité des étapes 30 et 35 (les étapes 30 et 35 sont activées et les autres étapes sont désactivées).



Lorsque l'étape 25 est active, le GRAFCET nommé GPN est forcé dans la situation où il se trouve à l'instant du forçage.

On appelle également cet ordre « **figeage** ».

Lorsque l'étape 22 est active, le GRAFCET nommé GPN est forcé dans la situation vide. Dans ce cas aucune de ses étapes n'est active.



Lorsque l'étape 34 est active, le GRAFCET nommé G4 est forcé dans la situation dans laquelle seules les étapes initiales sont actives.

II.8 Structuration par encapsulation

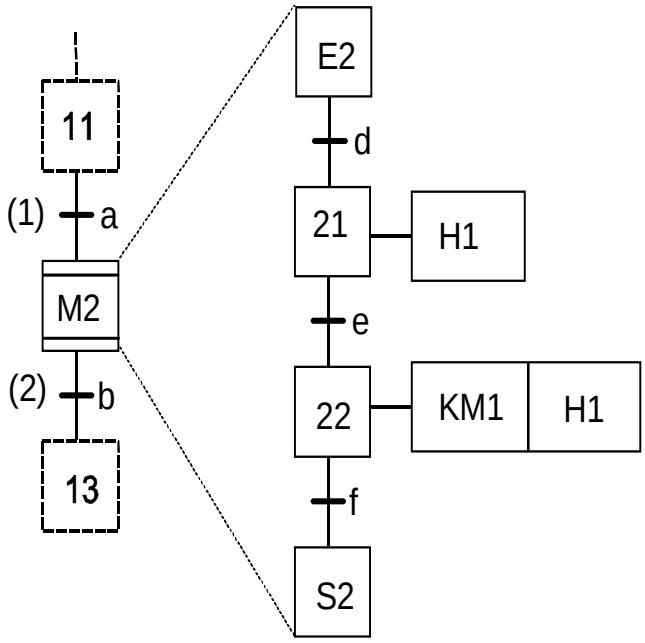
Cette nouvelle notion de la norme n'est pas reprise dans ce document.

Structuration par macro-étapes

Avec la notion de macro-représentation, on se donne le moyen de reporter à plus tard ou sur une autre page la description détaillée de certaines séquences.

La macro-étape est la représentation unique d'un ensemble d'étapes et de transitions nommé expansion de macro-étape.

▪ **Exemple d'une macro étape M2 représentée avec son expansion :**



L'expansion de la macro-étape commence par une seule étape d'entrée et se termine par une seule étape de sortie, étapes qui représentent les seuls liens possibles avec le GRAFCET auquel elle appartient.

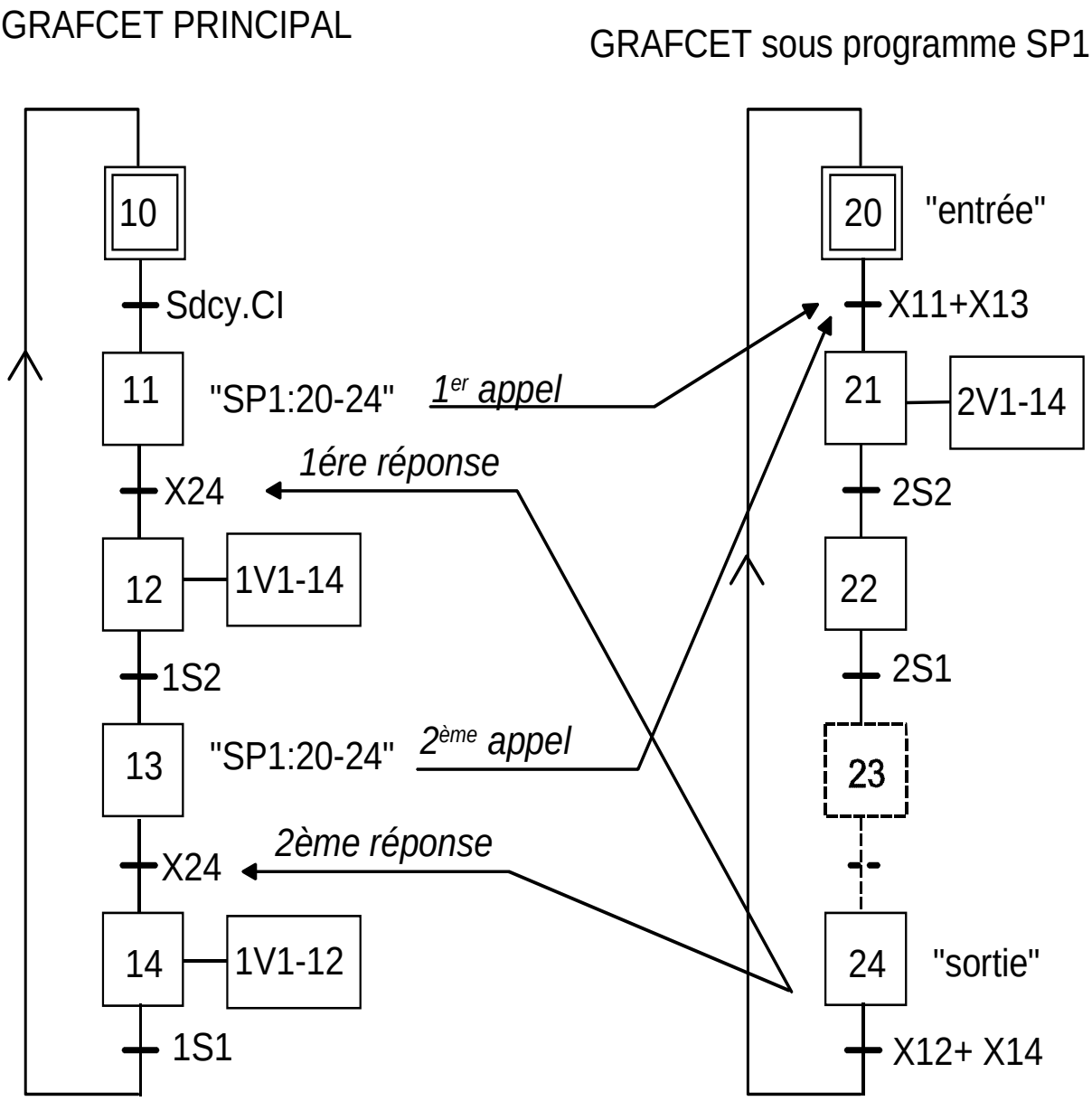
Le franchissement de la transition (1) active l'étape E2.

La transition (2) ne sera validée que lorsque l'étape S2 sera active.

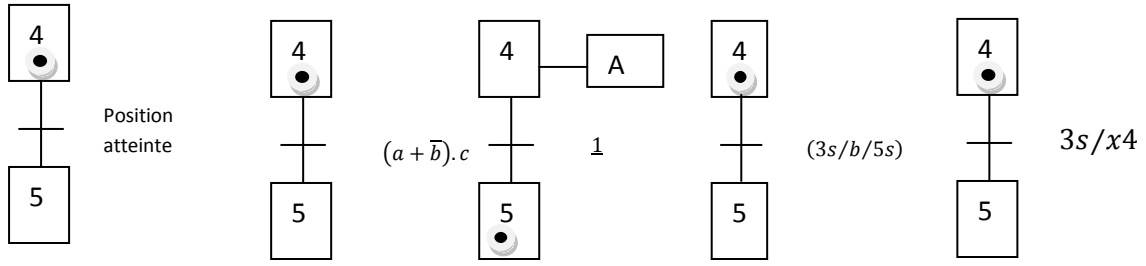
Le franchissement de la transition (2) désactive l'étape S2.

II.9 Structuration par GRAFCET de tâches et/ou sous-programme

La norme EN 60848 ne fait pas référence à ces notions et ne définit donc pas de symboles graphiques pour le GRAFCET de gestion des tâches. Nous pouvons continuer à utiliser la structuration par GRAFCET de sous-programme(s) en indiquant, entre guillemets (et pas dans un rectangle d'action), le nom du sous-programme appelé.



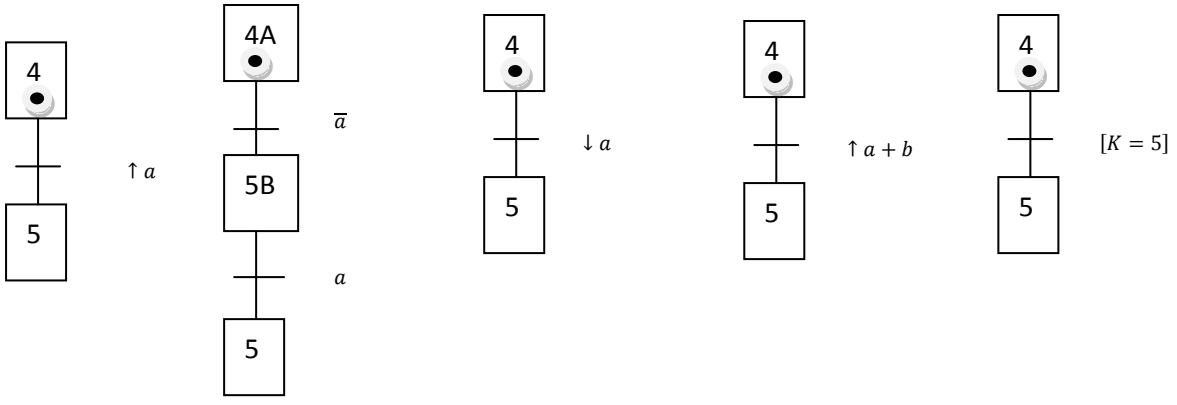
II.10 Les réceptivités associées aux transitions



Exemple 1 | Exemple2 Exemple3 Exemple4 Exemple5

Les réceptivités sont inscrites de Facon littérale (exemple 1) ou symbolique (exemple 2), à la droite de la transition. L'espace a gauche de la transition est réservé à une éventuelle identification ou numérotation. Si la réceptivité est toujours vraie (exemple 3) elle est note «1 » (1 souligne), dans ce cas l'évolution est fugace est l'action continue A n'est pas exécutée.

L'exemple 4 utilise une réceptivité 3s/b/5s qui vraie 3s après $\uparrow b$ est devient fausse 5s après $\downarrow b$: la variable b doit rester vraie pendant un temps supérieur ou égal a 3s pour que 3s/b/5s puisse être vraie.



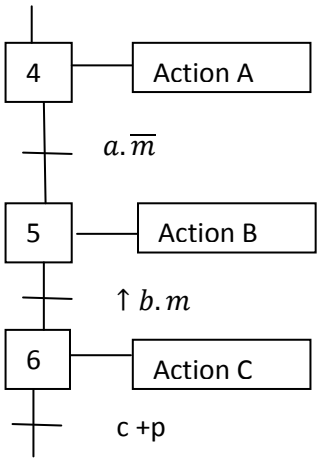
Exemple 6 Exemple7 Exemple8 Exemple9 Exemple10

Le « 5s » n'est utilise que si l'étape 4 devient active après↓ b. Dans une simplification usuelle (exemple 5), l'étape sera active pendant 3s : une autre façon de réaliser une action limitée dans le temps. L'exemple 6 utilise un front montant de a qui vaut 1 uniquement quand a passe de 0 à 1. L'exemple 7 propose un fonctionnement similaire a celui de l'exemple 6, l'étape 4 et alors divisée en deux étapes 4A et 4B. L'exemple 8 utilise un front descendant de a qui vaut 1 uniquement quand a passe de 1 a0. L'exemple 9 utilise une combinaison de variables de types différents. Enfin l'exemple 10 utilise la valeur booléenne d'un prédicat :

La réceptivité est vraie lorsque l'assertion K = 5 est vérifiée.

II.11 Structures particulaires

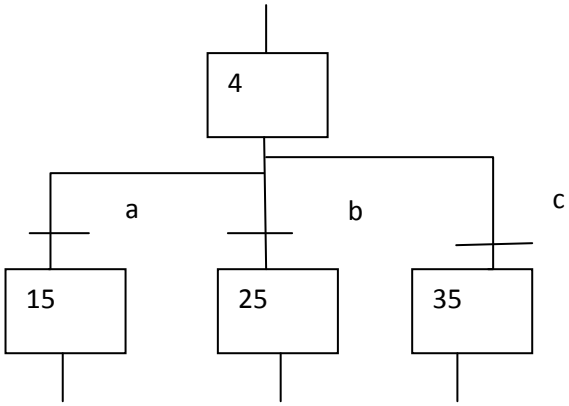
Séquence linéaire



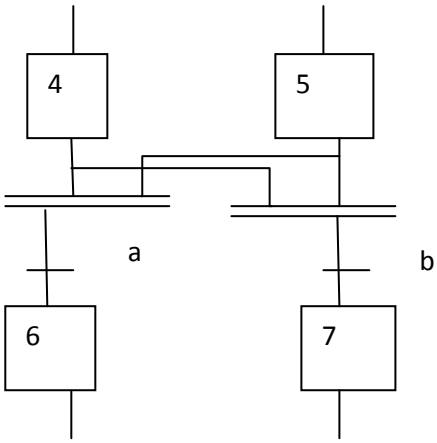
Une séquence linéaire est composée dune suite détape et de transitions. La séquence est dite active lorsqu'elle comporte une ou plusieurs étapes

Sélection de séquences

Une sélection de séquence est un choix dévolution entre plusieurs séquences à partir d'une ou plusieurs étapes.



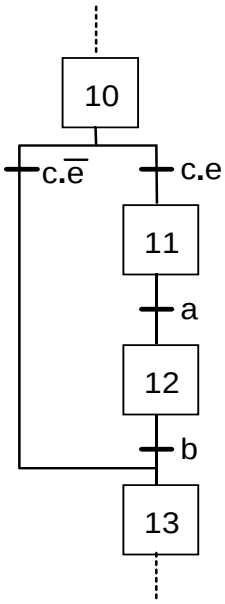
Aiguillage de 4 vers 15,25 ou 35



Aiguillage de (4et 5) vers 6 ou 7

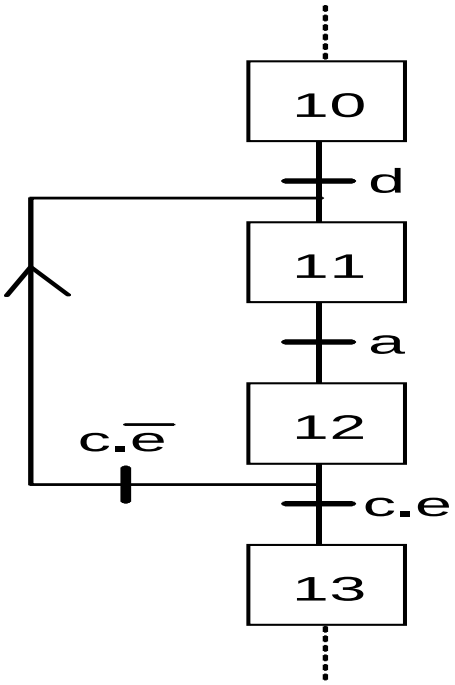
Saut d'étapes et reprise de séquence

Saut d'étapes



Le saut d'étapes permet de sauter une ou plusieurs étapes lorsque les actions associées à ces étapes deviennent inutiles.

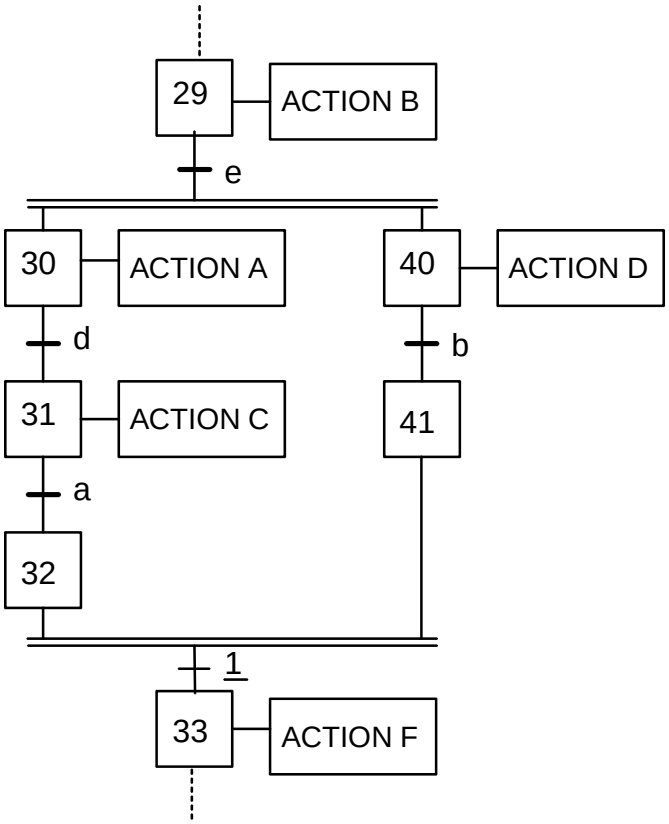
Reprise de séquence



La reprise de séquence permet de recommencer plusieurs fois la même séquence tant qu'une condition n'est pas obtenue.

Séquences simultanées (séquences parallèles)

Si le franchissement d'une transition conduit à activer plusieurs étapes en même temps, ces étapes déclencheront des séquences dont les évolutions seront à la fois simultanées et indépendantes.



Si l'étape 29 est active, la réceptivité « **e** » provoque, lorsqu'elle est vraie, l'activation simultanée des étapes 30 et 40.

Les deux séquences évoluent alors indépendamment l'une de l'autre.

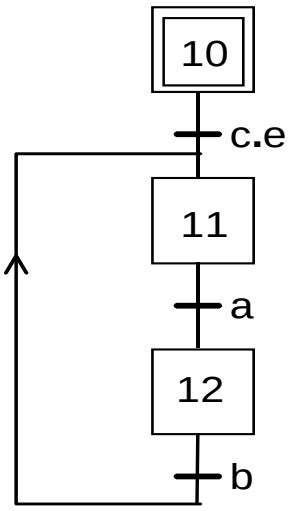
Les étapes 32 et 41 sont des étapes d'attente; dès qu'elles sont actives, la transition 32,41→33 est franchie ($\underline{1}$: réceptivité toujours vraie) ce qui entraîne simultanément, l'activation de l'étape 33 et la désactivation des étapes 32 et 41.

On remarque :

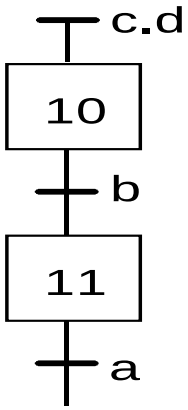
- que l'activation de l'étape 32 permet d'éviter que l'action C se poursuive lorsque a est vraie et que b ne l'est pas encore.

- que l'activation de l'étape 41 permet d'éviter que l'action D se poursuive lorsque b est vraie et que a ne l'est pas encore.

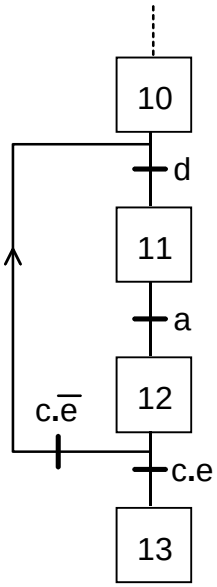
Les étapes et transitions sources ou puits



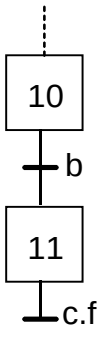
Les étapes sources sont souvent des étapes initiales (exemple 1, étape 10) par les quelles le système ne repasse plus en fonctionnement cyclique. Elles peuvent également être des étapes simple, forcées a l'activation par un grafcet hiérarchiquement supérieur (exemple 2, étape 10).



Transition source Une transition source et une transition qui ne possède aucune étape amont. Par convention, la transition source et toujours validée et est franchie dès que sa réceptivité est vraie. Dans l'exemple ci-dessous, l'étape 10 est activée dès que la réceptivité « **c.d** »est vraie.



Une étape puits est une étape qui ne possède aucune transition aval ; sa désactivation est possible par un ordre de forçage d'un GRAFCET de niveau supérieur.

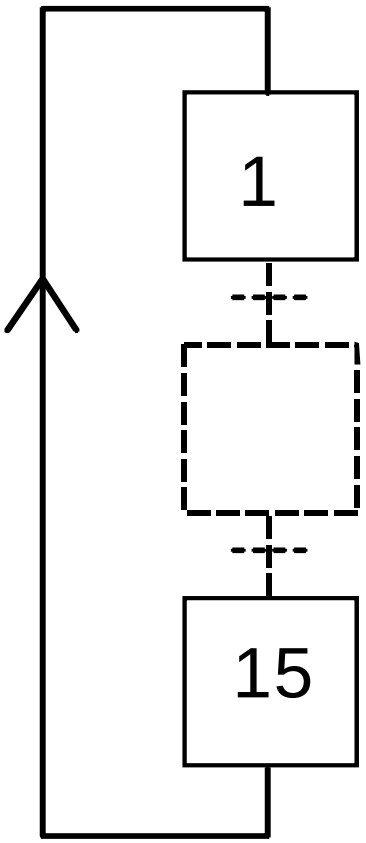


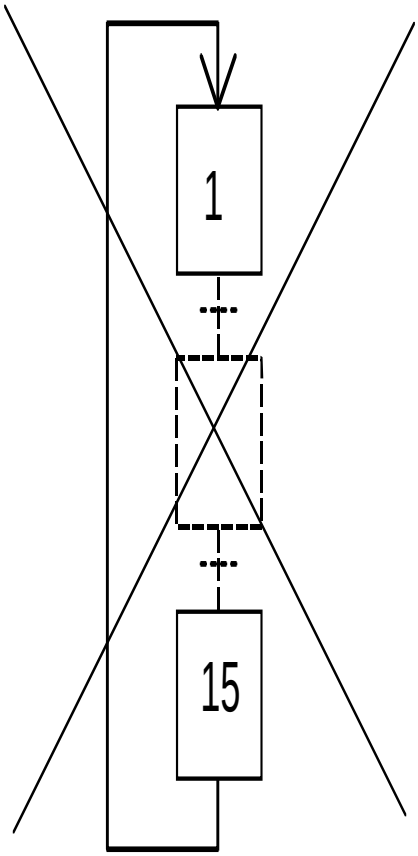
Une transition puits et une transition qui ne possède aucune étape aval. Dans l'exemple ci-dessous, lorsque la transition puits est validée et que « **c.d »** est vraie, le franchissement de cette transition a pour unique conséquence de désactiver l'étape 11.

II.12 Remarques sur les liaisons orientées

Liaison orientée de bas en haut

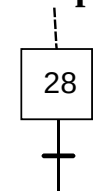
Par convention le sens d'évolution est toujours de bas en haut. Des flèches doivent être utilisées si cette convention n'est pas respectée ou si leur présence peut apporter une meilleure compréhension.





Solution conseillée **Solution à éviter**

Repère de liaison



Etape 29
page 2

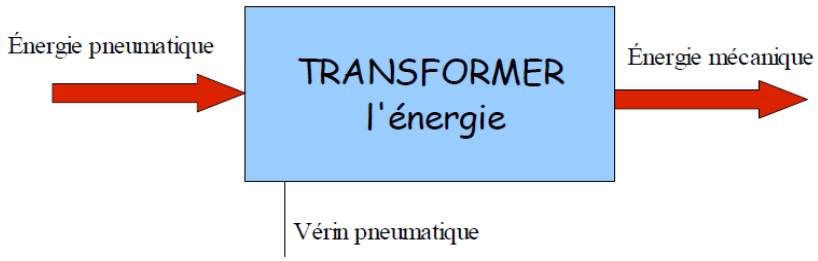
Lorsqu'une liaison orientée doit être interrompue, (dans des dessins complexes ou dans le cas de représentation sur plusieurs pages), le repère de l'étape de destination ainsi que le repère de la page à laquelle elle apparaît doivent être indiqués.

Le but est :

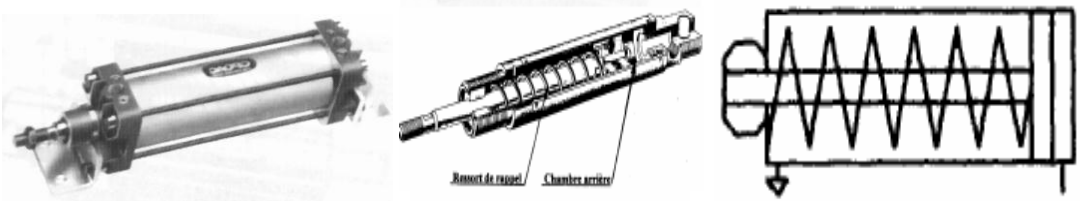
- De définir la fonction du vérin,
- De lister les différents types de vérins pneumatiques,
- De préciser leur mise en service et leur maintenance.

III.1. Principe

Un vérin pneumatique est un actionneur qui permet de transformer l'énergie de l'air comprimé en un travail mécanique.



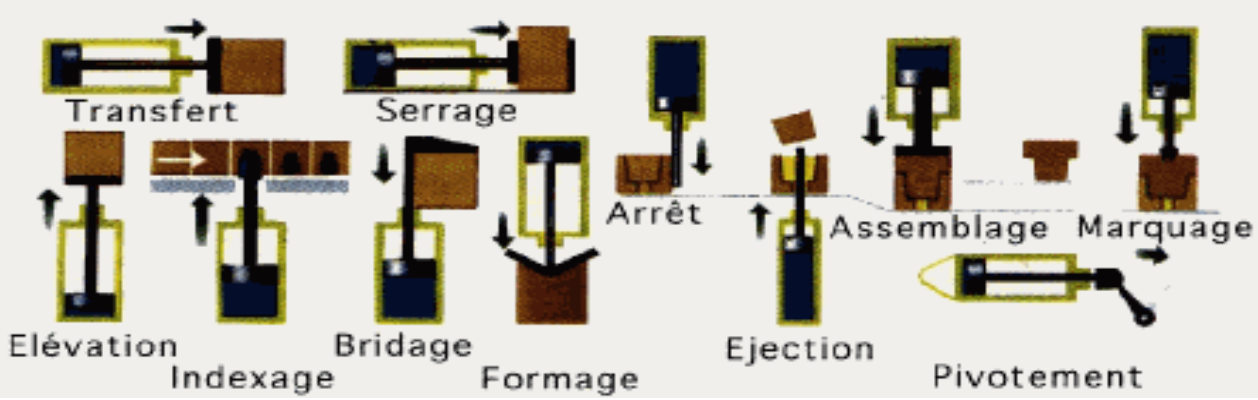
Un vérin pneumatique est soumis à des pressions d'air comprimé qui permettent d'obtenir des mouvements dans un sens puis dans l'autre. Les mouvements obtenus peuvent être linéaires ou rotatifs. Un vérin pneumatique ou hydraulique est un tube cylindrique (le cylindre) dans lequel une pièce mobile (le piston) sépare le volume du cylindre en deux chambres isolées l'une de l'autre. Un ou plusieurs orifices permettent d'introduire ou d'évacuer un fluide dans l'une ou l'autre des chambres et ainsi de déplacer le piston.



III.2. Applications

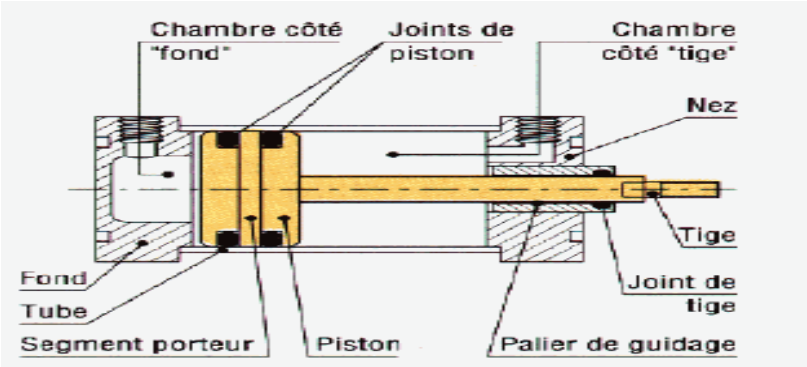
Cet actionneur de conception robuste et simple à mettre en œuvre est utilisé dans toutes les industries manufacturières. Il permet de reproduire les actions manuelles d'un opérateur telles que pousser, tirer, plier, serrer, soulever, poinçonner, positionner, etc...

Les croquis ci-dessous évoquent les principaux emplois des vérins pneumatiques en automatisation de production :



III.3. Constitution d'un vérin

Un piston muni d'une tige se déplace librement à l'intérieur d'un tube. Pour faire sortir la tige, on applique une pression sur la face avant du piston, et sur la face arrière pour faire rentrer la tige.



Amortissement

Certains vérins disposent d'amortisseurs afin d'obtenir un ralentissement en fin de mouvement de façon à éviter un choc du piston sur le nez ou le fond du vérin.

Auxiliaires implantés sur les vérins

Il est possible d'équiper les vérins de dispositifs de contrôle de mouvement tels que régleurs de vitesse et capteurs de position magnétique (ILS : Interrupteurs à Lames Souples).

Principe de fonctionnement

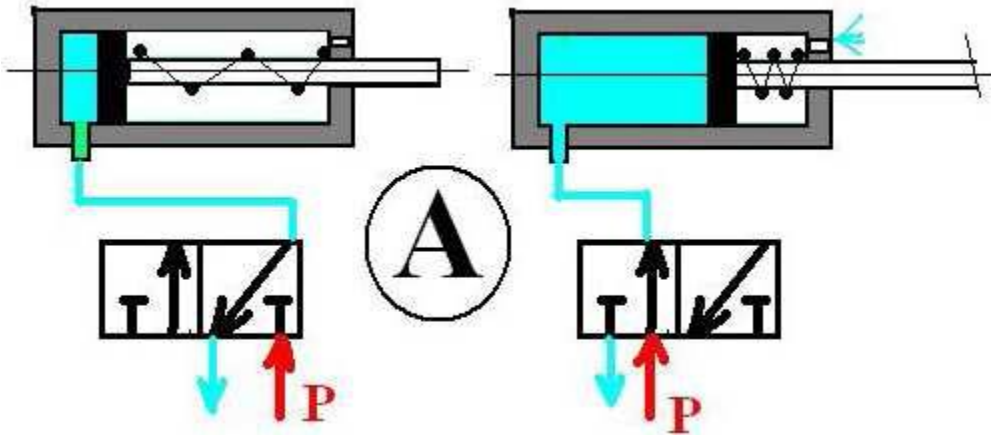
C'est l'air comprimé qui, en pénétrant dans l'une des chambres, pousse le piston. La tige se déplace. L'air présent dans l'autre chambre est donc chassé et évacué du corps du vérin. Le mouvement contraire est obtenu en inversant le sens de déplacement de l'air comprimé.

III.4. Différents types de vérins

Vérin simple effet

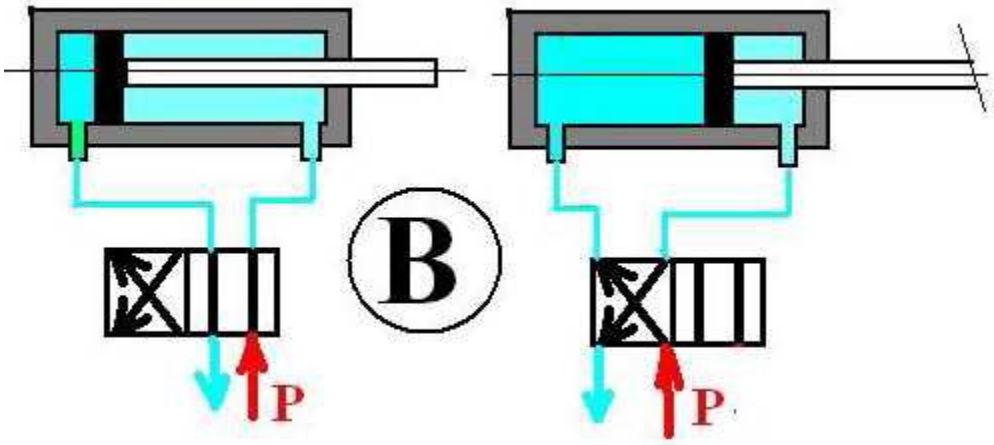
Un des deux mouvements de la tige est obtenu à l'aide d'un ressort de rappel qui se comprime lorsque s'effectue l'autre mouvement.

La position obtenue lorsque le ressort se détend (en absence d'air comprimé dans l'autre chambre) s'appelle la position repos.

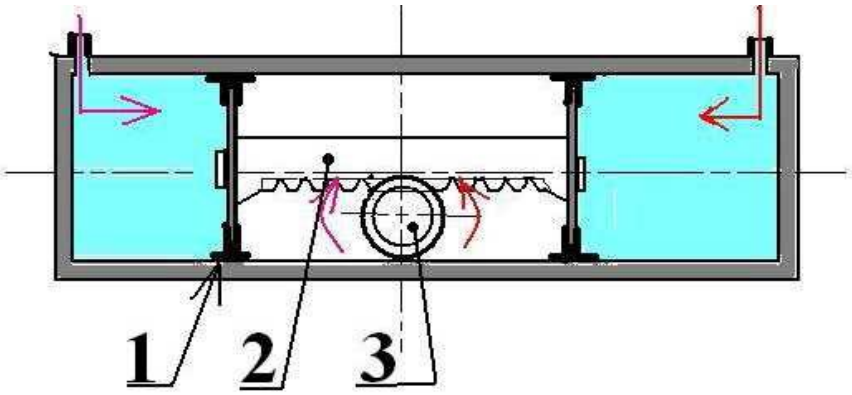


Vérin double effet

Le piston peut se déplacer librement dans le corps lorsqu'il est poussé par l'air comprimé. En l'absence d'air comprimé, il reste en position (tige rentrée ou sortie).



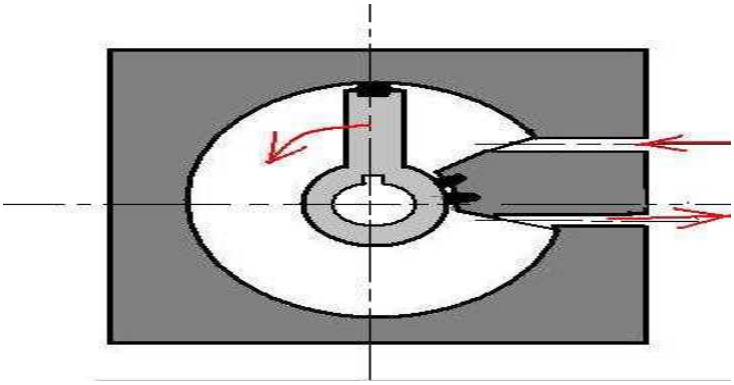
Vérin rotatif à pignon crémaillère



Vérin rotatif

Les vérins rotatifs peuvent être à pignon crémaillère (exemple ci-dessus), à simple palette (rotation limitée à 280°) ou à double palette (rotation limitée à 100°). L'angle de rotation sera

toujours réglable et inférieur à 360°.



Vérin électrique (système vis-écrou)

Plus facile à mettre en œuvre, le ***vérin électrique*** contrairement aux autres types de vérins est facile à utiliser. Son principe de fonctionnement est marqué par la simplicité, la rapidité, la fiabilité et la robustesse. Les résultats avec ce vérin sont satisfaisants.

III.5. Dimensionnement d'un vérin pneumatique linéaire

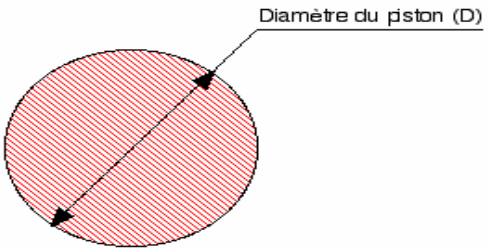
Critères de choix d'un vérin :

- Sa course : longueur du déplacement effectué par la tige de vérin,
- La vitesse de sortie de la tige : $V = \frac{Q}{S}$ (avec la vitesse v en m/s, le débit Q en m^3/s et la surface S en m^2),

- La force développée par le vérin, sachant que pour un vérin double effet cette force n'est pas la même en poussant et en tirant : $F = p \times S$| (avec la force F en Newtons, la pression p en Pa (1 bar = 10^5Pa) et la surface S en m^2)

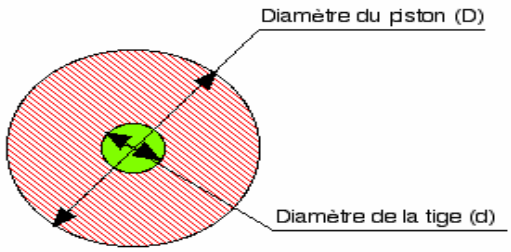
En sortie de tige, la surface du piston sur laquelle est appliquée la poussée est égale à

$S1 = \frac{\pi . D^2}{4}$ (voir **fig 5**)



En rentrée de tige, la surface n'est plus que:

$S2 = \frac{\pi . (D^2 - d^2)}{4}$ (voir **fig 6**)



Détermination du diamètre

Le diamètre du piston est en rapport direct avec l'effort axial développé par le vérin.

Effort théorique

L'air comprimé situé dans la chambre arrière applique une poussée sur toute la surface qui l'emprisonne entre autre, sur toute la surface du piston.

Il en résulte un effort axial théorique développé par le vérin et transmis en bout de tige.

$$F_t = P.S$$

F_t : effort théorique axial

p : pression de service à l'intérieur de la chambre du vérin

S : surface du piston sur laquelle la pression s'applique.

Effort réel

Lorsqu'un vérin est en conditions réelles d'utilisation, il développe un effort de poussée réel inférieur à l'effort théorique

car il faut tenir compte :

- Des frottements internes au vérin,
- De la contre pression qui est établie dans la chambre opposée pour obtenir un mouvement régulier.
- On estime, en usage général, les forces qui s'opposent à l'effort de poussée à environ 3 à 20% de l'effort obtenu (et 10% en général).

$$F_r = F_t - F_f \;\; et \;\; f_r = 90\% \, . F_t$$

Ft: effort théorique axial

Fr : effort réel

Ff : forces de frottement et divers

Calcul du taux de charge

Taux de charge = poussée réelle / poussée théorique Généralement de 0,5 (50%) pour les vérins dynamiques (travail en mouvement) et 0,8 (80%) pour les vérins statiques (travail à l'arrêt)

Exemple de calcul

Fn= ?, S = ? ,S' = ? ,Fr = 10% (valeur moyenne)

D = 50 mm

d = 12 mm

p = 6 bars

Calculs côté piston

Surface du piston

$S = \pi D^2/4$

$= (5 \times 5 \times 3.14) /4$

$= 19.625\ cm^2$

Poussée théorique

$Fth = S \times p = 19.625 \times 6$

$=1177.5\ N$

Résistance de frottement

$Fth / 10\% = 117.75\ N$

Poussée réelle en course avant

$Fn = S \times p – Fr = (19.625 \times 6) – 117.75 = 1060\ N$

Calculs côté tige

Surface du piston, côté tige

$S' = (D^2-d^2) \times \pi/ 4$

$= ((25 - 1.44) \times 3.14) / 4$

$= 18.5\ cm^2$

Poussée théorique

$Fth = S' \times p = 18.5 \times 6 = 1110\ N$

Résistance de frottement

$Fth / 10\% = 111\ N$

Poussée réelle en course arrière

Fn = S’ p – Fr = (18.5 x 6) – 111 = 999 N

Calculs et unités pratiques

La formule F = p.S permet de déterminer l’effort développé par un vérin donné ou de déterminer la section nécessaire pour développer un effort donnée.

$F = P.\pi.R^2$ Ou $R = \sqrt{\frac{F}{P\pi}}$

Attention aux unités :

En automatisme, l’unité de pression employée est le bar (et non le pascal), il en résulte un choix d’unités pratiques, permettant des calculs simples.

P est exprimé en bar

R est exprimé en cm

S est exprimé en cm^2

F est exprimé en daN (déca-newton)

Selon l’effort (Ft ou Fr) choisi dans le calcul, on déterminera un rayon théorique ou réel du piston.

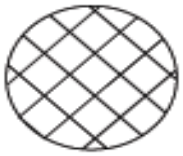
Lors du **calcul de la force développée par les vérins**, il faut tenir compte de la surface efficace du piston en contact avec la pression d'air :

- **lors de la sortie de la tige**, le travail s'effectue en poussée et agit sur la surface totale du piston ;

- **lors de la rentrée de la tige**, le travail s'effectue en traction et agit sur une surface réduite du piston, car il faut déduire la section de la tige.

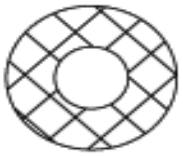
Surfaces de travail :

En poussée



$S = 0{,}7854 \times D^2$

En traction



$S = 0{,}7854 \times (D^2 - d^2)$

Dans lesquelles
S = surface de travail
D = diamètre du piston (alésage)
d = diamètre de la tige

La force (F, en newtons) est en relation directe avec la pression

(p, en pascals) et la surface (S, en mètres carrés)

$$F = P.S$$

Comme la plupart des vérins ont une forme circulaire, on calcule la surface du piston (cercle) à l'aide de l'une ou l'autre des formules suivantes : S = Pi × r^2 ou 3,1416 × rayon du cercle au carré

Ou encore : $S = \frac{\pi \times D^2}{4}$ ou $\frac{3{,}1416 \times \text{diamètre du cercle au carré}}{4}$

Ou encore : S = 0,7854 × D^2

Cette dernière équation s'avère la plus utile, car les constructeurs classent les vérins selon leur alésage (diamètre du vérin).

Par ailleurs, si le vérin est muni d'un ressort, il faut soustraire de la force de travail l'effort nécessaire pour combattre la force du ressort. Toutefois, on

néglige les forces de frottement, car elles représentent habituellement moins de 10 % de la force totale.

Exemple 1

Problème

Calculez les forces de poussée et de traction d'un vérin de 5 cm de diamètre muni d'une tige de 2 cm de diamètre et soumis à une pression de 500 kPa (5 bars).

a. Conversion des données

On convertit d'abord les données en unités de base :

F = ?

P = 500 kPa = 500 000 Pa

D = 5 cm = 0,05 m

d = 2 cm = 0,02 m

b. Calcul des surfaces efficaces du piston

On trouve ensuite les surfaces efficaces du piston :

En poussée :

$S_p = 0{,}7854 \times D^2$

$S_p = 0{,}7854 \times 0{,}05^2$

S_p environ 0,002 m^2

En traction :

$S_t = 0{,}7854 \times (D^2 - d^2)$

S_t environ 0,0017 m^2

c. Calcul des forces de poussée et de traction

La force de poussée équivaut alors à :

$F_p = p \times S_p$

$F_p = 500\,000 \times 0{,}002$

$F_p = 1\,000$ N

Et la force de traction équivaut à :

$F_t = p \times S_t$

$F_t = 500\,000 \times 0{,}0017$

$F_t = 850$ N

On conclut que la force de traction est effectivement plus faible que celle de la poussée à cause de la surface réduite par la tige du vérin.

La masse déplacée, d'un mouvement uniforme, suivant un axe vertical se définit à partir de la relation : P = m • g.

- P = F -> P = Poids du solide - F = force de poussée

- $m = \frac{F}{g}$

- F = 1 000 N - g = 10 m/s^2

- Masse m : $m = \frac{1\,000}{10} = 100$ kg

Exemple 2

Problème

Calculez la force de poussée si, dans les mêmes conditions, le diamètre du vérin a doublé.

a. Conversion des données

F = ?

P = 500 000 Pa

D = 10 cm = 0,1 m

b. Calcul de la surface de poussée

On trouve la surface de poussée du piston :

$S = 0{,}7854 \times 0{,}1^2$

S envion 0,008 m^2

On remarque qu'une augmentation du double du diamètre (de 5 à 10 cm) a fait quadrupler la surface de poussée (de 0,002 à 0,008 m^2).

c. Calcul de la force de poussée

La force de poussée du nouveau vérin devient :

$F = 500\,000 \times 0{,}008$

F = 4 000 N

Ce qui équivaut à déplacer une masse m de 400 kg, d'un mouvement uniforme, suivant un axe vertical.

On conclut que la surface et la force d'un vérin varient en fonction du carré du diamètre :

- Si le diamètre double (× 2), la force quadruple (× 4) ;

- Si le diamètre triple (× 3), la force augmente d'un facteur 9 (× 9) ; et ainsi de suite.

Effort dynamique développé par un vérin

Effort dynamique développé par un vérin :

F = Pression × Surface du piston × Rendement

Le rendement d'un vérin dépend du diamètre du vérin, de la pression et de paramètres d'ordre mécanique. Les abaques et tableaux ci-dessous définissent les efforts dynamiques développés par les vérins en sortie et rentrée de la tige, en fonction de la pression d'alimentation.

Taux de charge

C'est le **rapport**, exprimé en pourcentage, **entre la charge réelle à déplacer par le vérin et l'effort dynamique disponible en bout de tige**.

$$\text{Taux de charge (en \%)} = \frac{\text{Charge réelle}}{\text{Effort dynamique}} \times 100$$

Pour une utilisation optimale du vérin, il est recommandé de définir un vérin tel que le taux de charge soit inférieur ou égal à 75 %.

Exemple : Définition d'un vérin pour soulever une charge de 130 daN à une pression de 7 bar relatifs (manométriques).

Effort dynamique théorique = $\frac{\text{Charge réelle}}{\text{Taux de charge}} = \frac{130}{0,75} = 175 \text{ daN}$

Dans l'abaque **"sortie de tige"**, définir le point de rencontre entre l'effort dynamique ainsi calculé et la pression d'alimentation. Le diamètre du vérin nécessaire sera celui dont la courbe passe par ce point ou celui développant un effort immédiatement supérieur.

III.5. le distributeur # PILOTE#

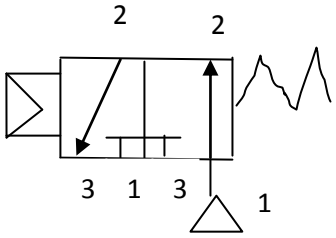
Le but d'un distributeur est de mettre en communication divers orifices deux à deux. Ceci peut être réalisé soit par rotation d'un disque distributeur ou d'un boisseau, soit par glissement d'un tiroir sont les plus répandus. Leur principale utilisation est celle de préactionneur pour vérin.

1. **Moyens de pilotage ou de commande**

   Fonctionnellement de distributeur a pour rôle de délivrer un débit d'air comprimé (pour commander un vérin), a la réception d'un signal de commande. Ce signal peut être mécanique, pneumatique ou électrique. Dans ce dernier cas on parle alors d'électro distributeur (le courant pouvant être continu ou alternatif). Parfois lune des 2 commandes du distributeur est effectuée par un ressort de rappel.

2. Symboles normalises

On parle de distributeur 2/2, 3/2, 4/2, 4/3,5/2. Le premier chiffre correspond au nombre d'orifices et le second au nombre de positions de travail. Il existe un troisième qui caractérise un distributeur, c'est le nombre de voies (ou chemins internes) possibles de passage de l'air imprimé. Les numéros des orifices sont normalises : 1 pour l'arrivée d'air, pairs (2 et 4 et 6) pour les sorties de commande, impairs (3 et 5) pour les sorties d'échappement.



Dans la représentation, le nombre de rectangles (ou de carrés) correspond au nombre de tiroirs, et donc au nombre de positions du distributeur.

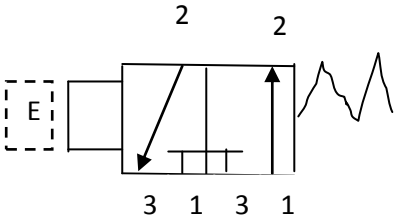
Généralement le distributeur 3/2 est utilise pour la commande des vérins simple effet, et les distributeurs 4/2 et 5/2 pour les vérins double effet. Un étrangleur réglable par une vis est généralement dispose sur l'échappement pour régler le débit (et donc la vitesse du vérin).

3.Electrodistributeurs

3. a. Distributeur 3/2 : monostable

Il possède un seul état stable qui est la position de repos. Il est donc commande par une bobine E et un ressort de rappel, et il faut maintenir

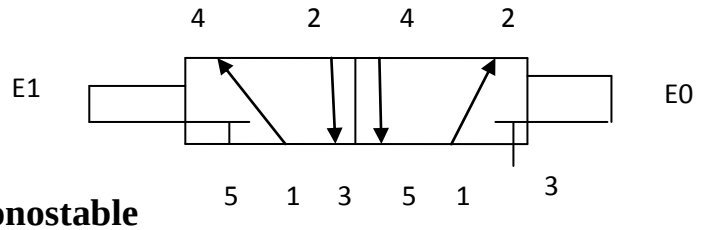
la commande, sinon le tiroir revient automatiquement a la position de repos sous l'effet du ressort.



Quand la bobine est alimentée, l'électrovanne s'ouvre laissant passer l'air comprime qui pousse le tiroir vers la droite mettant en communication les orifices 1 et 2(admission et commande). L'orifice 2 étant relie a l'entrée d'un vérin, ce dernier est alors commande. Quand l'alimentation de la bobine est coupée, électrovanne se ferme (coupant l'admission d'air), le ressort ramène le tiroir de la droite vers la gauche, mettant en communication les orifices 2 et 3, permettant ainsi à l'air (emmagasiné dans le vérin) de s'échapper par l'orifice 3.

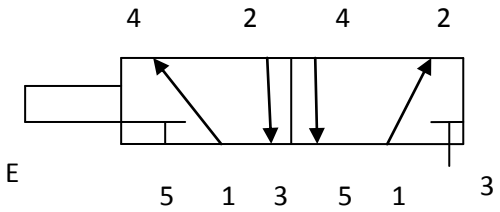
3. b. Distributeur 5/2 : bistable

Il possède deux états stables correspondant aux deux positions de travail. Il est commande par deux bobines E0 et E1, il a donc un fonctionnement a double pilotage. Même une impulsion sur E0 ou E1 suffit a le faire changer d'état.



3.c. Distributeur 5/2 : monostable

Il possède un seul état stable qui correspond a la position de repos. La commande s'effectue (comme pour 3/2) par une bobine et un ressort de rappel.



3.d. Distributeur 5/3 : monostable

La seul position stable est la position médiane : en l'absence de commande. Deux ressorts de centrage (gauche et droit) ramènent le tiroir en position centrale. En parle soit de 5/3 a point milieu bloque (entrée et sorties du distributeur bloquées), soit de 5/3 a point milieu libre (entrée bloquée et sorties reliées a l'échappement).ainsi en l'absence de commande sur le distributeur, le vérin commandé sera soit bloqué soit libre.

4. Auxiliaires de distribution

Sur le circuit de puissance, entre distributeur et vérin, des auxiliaires sont généralement nécessaires pour permettre :

- Le réglage de la vitesse du vérin, dans chacun des sens de déplacement,
- L'intégration de fonctions de sécurité, par blocage du vérin ou par purge de ses pressions d'air.

Pour toutes ces fonctions, les implémentations au plus prés du vérin est plus efficace. C'est pourquoi les auxiliaires sont prévus pour s'implanter, en général, directement sur les orifices de raccordement du vérin, en lieu et place des raccords qu'ils intègrent.

4.a. Le régleur de vitesse

La tige du vérin pneumatique A se déplace dans le sens A+ sous l'action de la différence des pressions entre chambres amont et aval. La vitesse du mouvement A+ est déterminée par la vitesse de purge de l'air contenu dans la chambre aval. Le régleur de vitesse laisse passer l'air à plein débit dans le sens admission, le clapet anti retour étant ouvert. Il régule le débit d'air, et donc la vitesse du mouvement du vérin, dans le sens échappement grâce à la restriction réglable, le clapet anti retour est ferme.

La restriction réglable est généralement réalisée par une vis pointeau verticale. L'anti retour est assure soit par une jupe en élastomère soit par une bille.

Afin que le vérin ne broute pas et que la vitesse de déplacement soit constante, il est préférable que le régleur de vitesse soit au plus prés du vérin.

4.b. le bloqueur 2/2

Particulièrement indiques pour arrêter les vérins en cours de mouvement ou pour les maintenir en position, les bloqueurs 2/2 assure un blocage efficace des que le signal de pilotage disparait.

Deux bloqueurs stoppent les débits d'air d'admission et d'échappement, immobilisant ainsi la tige du verin et sa charge, par mesure de sécurité.

4.c le sectionneur purgeur

Le sectionneur général dune installation peut ne pas suffire pour arrêter certains mouvements, la purge se trouvant ralentie par la restriction réglable du régleur de vitesse. Place entre l'orifice du vérin et le régleur de vitesse, le sectionneur purgeur vide localement et rapidement la chambre du vérin. Il doit être commande par le même signal que le sectionneur général.

III. 6. MAINTENANCE.

Les opérations de maintenance sur les actionneurs pneumatiques du type vérin sont très limitées. Sur les vérins de petites dimensions, le remplacement systématique en cas de panne est préférable. Sur les vérins de gros diamètres, la réparation peut être envisagée (changement des joints).

- Les pièces d'usure sont donc :
- Les joints de piston ;

- Le joint de tige ;
- La bague ou segment porteur ;
- Le palier de guidage de la tige.

L'ensemble de ces pièces est commercialisé par les constructeurs sous le nom de « pochette de réparation ». Pour éviter les défaillances successives, toutes les pièces d'usure sont remplacées lors du dépannage.

III.6.1. CONSIGNES ET PROCEDURES DE SECURITE.

Pour changer un vérin, il faut :

- Pallier les effets provoqués par une coupure d'air comprimé ;
- Consigner la machine ;
- Purger le circuit d'air comprimé ;
- Débrancher et démonter le vérin ;
- Remonter le nouveau vérin et vérifier manuellement (si possible) que la tige rentre et sort sans contrainte mécanique ;
- Régler les capteurs fin de course si nécessaire ;
- Rebrancher la tuyauterie, mettre sous pression et faire les essais d'usage.

III.6.2. MAINTENANCE PREVENTIVE DES VERINS.

On peut remplacer périodiquement les pièces d'usure d'un vérin, en comptabilisant le nombre de courses qu'il effectue.

La durée de vie moyenne sans défaillance d'un vérin, correspond à une course de 100 km, C'est à dire à 1 000000 de courses d'un vérin dont la longueur de course serait 100 mm.

Précautions à prendre pour le stockage :

Ne jamais stocker un vérin en laissant les orifices d’alimentation ouverts, risque de pénétration de pollution, poser des bouchons sur les orifices.

Protéger le tube et la tige des chocs accidentels. Huiler les parties métalliques.

Dans toutes les entreprises industrielles on utilise différents systèmes, ceux-ci ont pour fonctions de produire et d'améliorer les conditions de travail et d'hygiène des personnes.

Le fonctionnement global des entreprises est décrit de cette manière :

IV.1-Définition de l'automatisme :

L'automatisme consiste en l'étude de la commande de systèmes industriels. Les techniques et méthodes d'automatisation sont en continuelle évolution ; elles font appel à des technologies : électromécaniques, électronique, pneumatique, hydraulique. Les automatismes sont présents dans tous les secteurs d'activité (menuiserie, textile, alimentaire, automobile...).

La première amélioration des conditions de travail a été de remplacer l'énergie humaine fournie par l'ouvrier par une machine (Partie Opérative : P.O.). Les automatismes doivent améliorer :

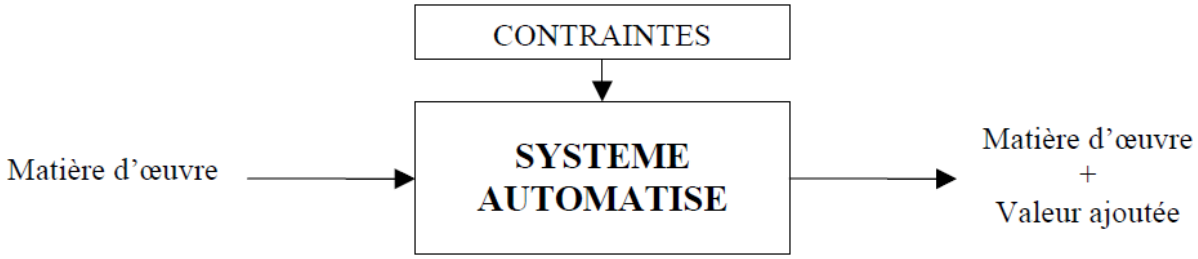
IV.2-Description d'un Système Automatisé de Production (S.A.P.) :

A. Les systèmes automatisés de production :

L'objectif de l'automatisation des systèmes est de produire, en ayant recours le moins possible à l'homme, des produits de qualité et ce pour un coût le plus faible possible.

Un système automatisé est un ensemble d'éléments en interaction, et organisés dans un but précis : agir sur une matière d'œuvre afin de lui donner une valeur ajoutée.

Le système automatisé est soumis à des contraintes : énergétiques, de configuration, de réglage et d'exploitation qui interviennent dans tous les modes de marche et d'arrêt du système.



L'automatisation des processus industriels a pour finalité de réaliser ces vœux ses objectifs principaux au nombre de quatre sont complémentaires et lies. Ils peuvent s'énoncer ainsi. :

a. **produire a qualité constante :** contrairement a l'être humain, il est clair que la machine n'est pas sujette a la fatigue de fin de journée, par conséquent la qualité des produits sortant des chaines est généralement la même.

b. **Fournir des quantités nécessaires :** cette notion fait référence à l'adaptativité, c'est dire pouvoir adapter l'offre à la demande. L'objectif est de produire juste les quantités nécessaires à un instant donné, de façon a tendre vert la notion de stock zero. Pour pouvoir adapter l'offre a la demande, cela se fait rapidement et efficacement

dans un enivrement automatise(arrêter par exemple une chaine de fabrication en période de faible demande, ou au contraire en mettre en route d'autres pour répondre a la forte demande).

c. **Augmenter la productivité :** il s'agit donc d'augmenter le rendement. Pour ce faire l'automatisation a consiste a remplacer une grande partie des operateurs humains par des machines, qui ont des cadences de travail élevées, ne connaissant ni les pauses café ni les congés payes.

d. **Améliorer les conditions de travail :** il s'agit dune part de remplacer L'homme par la machine pour les taches pénibles ou qu'il ne peut pas faire (pour l'affecter ailleurs ou il censé faire un travail plus noble).

B. Fonctions des automatismes :

Le degré d'automatisme d'un système varie selon la nature, la complexité, les objectifs assignés au projet. La surveillance dune tour d'immeuble est différente de celle des ascenseurs quelles comporte ou de son dispositif de climatisation.

Il existe trois degrés d'automatisation ou modes de fonctionnement des automatismes :

B.a. mode surveillance : dans ce mode l'automatisme a une fonction passive vis-à-vis procède qu'il pilote l'organe de contrôle acquiert les informations et les analyses pour fournir journaux de bord et bilans. L'objectif est la connaissance technique et économique du procédé.

B.b. mode guide operateur : les traitements sont plus élaborés que dans le cas précédent, et l'automatisme propose des actions pour conduire le procédé selon un critère donne. L'automatisme ne réagit pas directement sur les procédé, il a donc un fonctionnement en boucle ouverte.

B.c. mode commande : l'automatisme a une structure en boucle fermée. On a une automatisation complète de certaines fonctions : acquisition des informations, leur traitement, et enfin l'action sur le procédé.

Mode Fonctionnement	Surveillance	Guide operateur	Commande
Acquisition	X	X	X
Traitement		X	X
Action			X
Structure	Boucle ouverte	Boucle ouverte	Boucle fermée

Figure IV.1 Différentes fonctions d'un automatisme.

Les fonctions assurées dans chaque mode sont simples au complexes selon le procédé ou la partie de procédé aux quelles elles sont assignées.

Exemple : la surveillance dune installation de chauffage central d'un édifice quelconque. Si le niveau d'eau diminue, la pression augmente. Au delà dune certaine pression la chaudière risque d'exploser. Pour faire baisser la pression il faut tout simplement rajoute de l'eau. Dans le cas du mode surveillance seul un indicateur visuel a aiguille nous permet de savoir que la pression augmente.

Dans le mode guide operateur, on donne l'information sur l'action qu'il faut entre prendre afin de baisser la pression. Un indicateur visuel au sonore indique qu'il faut ouvrit la vanne d'eau.

Dans le mode commande l'automatisme commande l'ouverture de la vanne surveille le niveau d'eau puis ferme la vanne quand le niveau désiré est atteint.

C. Structure d'un système automatisé :

Tous les systèmes automatisés possèdent une structure générale composée de 3 parties fondamentales :

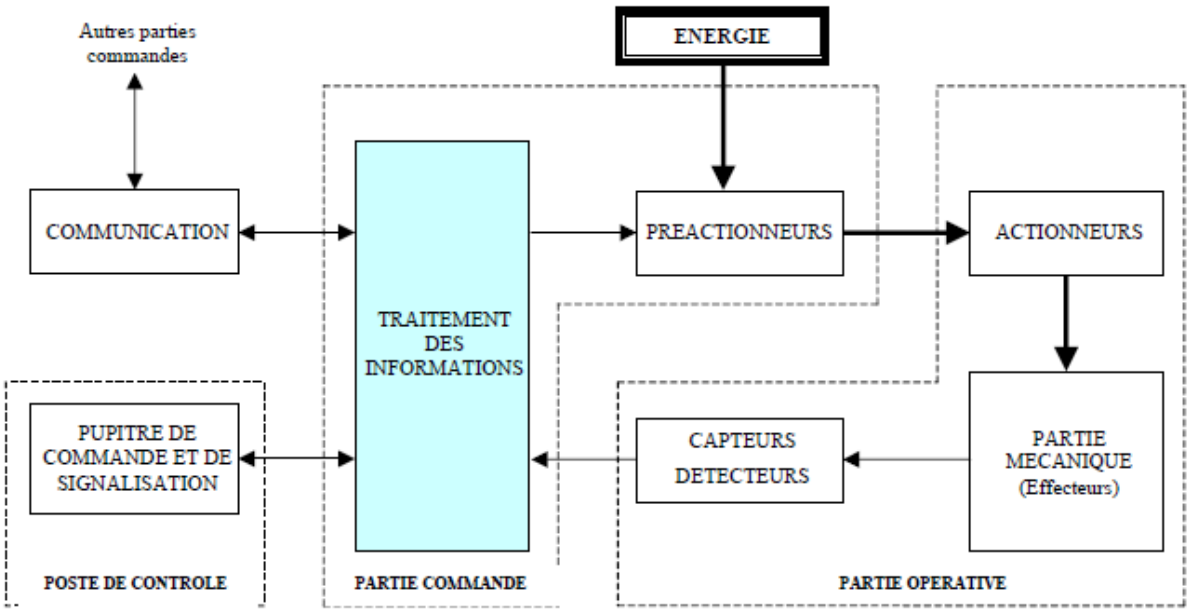
- la partie opérative (P.O.) : que l'on appelle également partie puissance, c'est la partie visible du système (corps) qui permet de transformer la matière d'œuvre entrante, elle est composée d'éléments mécaniques, d'actionneurs (vérins, moteurs), de préactionneurs (distributeurs et contacteurs) et des éléments de détection (capteurs, détecteurs). Par exemple, on va trouver des capteurs mécaniques, pneumatiques, ´électriques ou magnétiques montes sur les vérins. Le rôle des capteurs (ou détecteurs) est donc de contrôler, mesurer, surveiller et informer la PC sur l'´evolution du système.

Pour réaliser les mouvements il est nécessaire de fournir une énergie à la PO.

Dans le cadre des SAP nous étudierons principalement les trois suivantes :

- La partie commande (P.C.) : C'est la partie qui traite les informations, elle gère et contrôle le déroulement du cycle (cerveau). Ce secteur de l'automatisme gère selon une suite logique le déroulement ordonne des opérations à réaliser. Il reçoit des informations en provenance des capteurs de la Partie Opérative, et les

restitue vers cette même Partie Opérative en direction des pré-actionneurs et actionneurs.



- **Le pupitre** : permet d'intervenir sur le système (marche, arrêt, arrêt d'urgence...) et de visualiser son état (voyants).

IV.3. Structure interne d'un Automate programmable

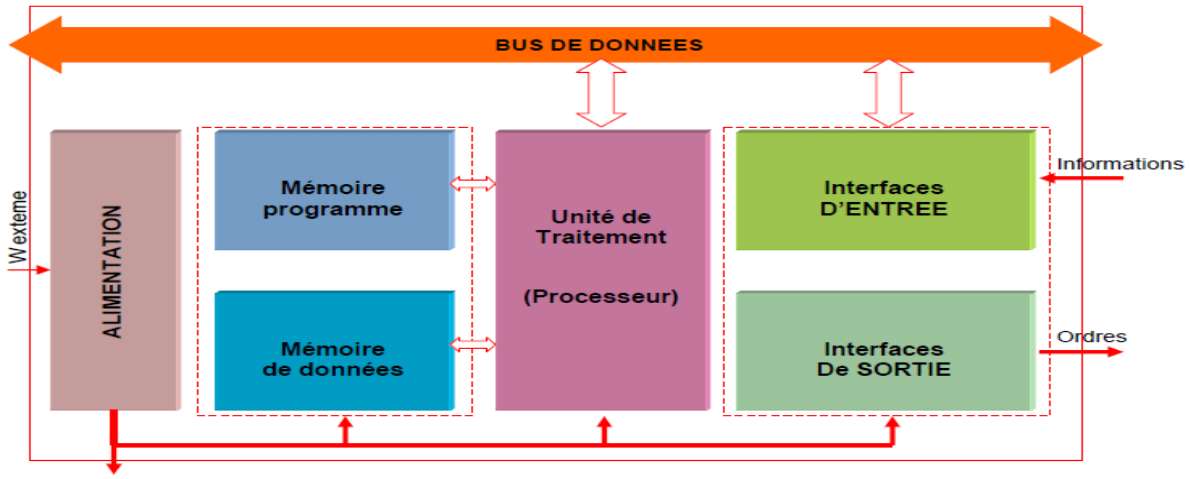


Fig. IV.2 – Structure interne d'un API

Un automate programmable est un appareil électronique qui comporte une mémoire programmable par un utilisateur automaticien à l'aide d'un langage adapté, pour le stockage interne des instructions composant les fonctions d'automatismes, par exemple :

- Logique séquentielle et combinatoire
- Temporisation
- Comptage, décomptage, comparaison
- Calcul arithmétique
- Réglage, asservissement, régulation,
- Etc…

Il permet de commander, mesurer et contrôler au moyen de modules d'entrées et de sorties (logiques, numériques ou analogiques) différentes sortes de machines ou de processus en environnement industriel.

La compacité, la robustesse et la facilité d'emploi des automates programmables industriels (A.P.I.) font qu'ils sont très utilisés dans la partie commande des systèmes industriels automatisés.

Un A.P.I. se compose :

Une unité de traitement ou processeur :

Le processeur gère l'ensemble des échanges informationnels en assurant :

- La lecture des informations d'entrée

- L’exécution des instructions du programme mis en mémoire
- La commande ou l’écriture des sorties

Une mémoire programme :

La mémoire programme de type RAM contient les instructions à exécuter par le processeur afin de déterminer les ordres à envoyer aux pré actionneurs reliés à l’interface de sortie en fonction des informations recueillies par les capteurs reliés à l’interface d’entrée.

Une mémoire de données

La mémoire de donnée permet le stockage de :

- l’image des entrées reliées à l’interface d’entrée
- L’état des sorties élaborées par le processeur
- Les valeurs internes utilisées par le programme (résultats de calculs, états intermédiaires ,…)
- Les états Forcés ou non des E/S

Une interface d’ENTREE

L’interface d’entrée permet la connexion à l’API d’une multitude de capteurs pouvant être :

- TOR (logiques ou Tout Ou Rien)
- Numériques
- Analogiques

Ces différentes entrées sont mises en forme par l'interface d'entrée avant d'être stockées dans la mémoire de données.

Une interface de SORTIE

L'interface de sortie permet la connexion à l'API d'une multitude de préactionneurs pouvant être :

- TOR (logiques ou Tout Ou Rien)
- Numériques
- Analogiques

Un module d'ALIMENTATION

Le module d'alimentation transforme l'énergie externe provenant du réseau en en la mettant en forme afin de fournir aux différents modules de l'API les niveaux de tension nécessaires à leur bon fonctionnement.

Plusieurs niveaux de tension peuvent être utilisés par les circuits internes (3v, 5v, 12v, 24v…) il sera dimensionné en fonction des consommations des différentes parties.

Description des éléments d'un API

Le processeur :

Son rôle consiste d'une part à organiser les différentes relations entre la zone mémoire et les interfaces d'E/S et d'autre part a gérer les instructions du programme.

Les interfaces :

L'interface d'Entrées comporte des adresses d'entrée, une pour chaque capteur relie. L'interface de Sorties comporte des adresses de sorties, une pour chaque preactionneur. Le nombre d'E/S varie suivant le type d'automate.

Les cartes d'E/S ont une modularité de 8, 16 ou 32 voies. Elles admettent ou délivrent des tensions continues 0 - 24 Vcc.

Nature des mémoires

La mémoire de l'API est l'élément fonctionnel qui peut recevoir, conserver et restituer des informations.

L'espace mémoire peut être divisé en deux parties :

La mémoire Programme qui permet le stockage des instructions à exécuter par l'API.

La mémoire de données qui permet le stockage de l'état des E/S et des variables internes.

Les mémoires utilisées dans un API peuvent être des types suivants :

R.A.M. (Random Access Memory) mémoire à accès aléatoire. Cette mémoire doit être alimentée électriquement pour pouvoir conserver les informations. On l'appelle également la mémoire vive.

Avant son exécution, le programme est transféré dans cette mémoire qui permet d'atteindre des vitesses en lecture et écriture très rapides.

R.O.M. (Read Only Memory) mémoire à lecture uniquement. Appelée également mémoire morte, elle permet de stocker des informations indéfiniment sans aucune alimentation électrique.

P.R.O.M. (Programable Read Only Memory) mémoire de type ROM mais Programmable. C'est une ROM que l'on peut programmer une seule fois.

E.P.R.O.M. (Erasable Programable Read Only Memory) mémoire de type PROM que l'on peut effacer par exposition du circuit aux rayons ultra-violets.

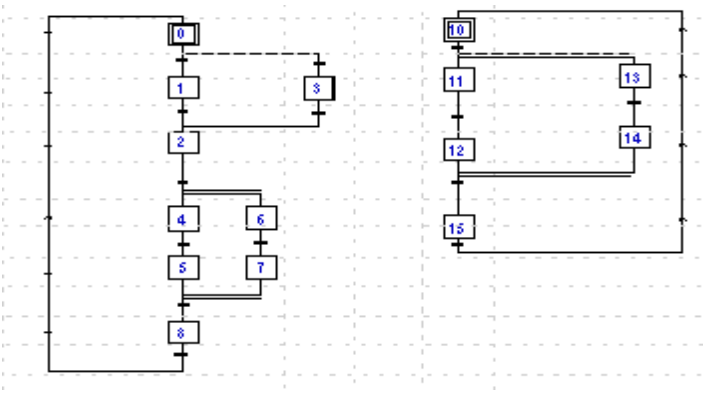
E.E.P.R.O.M. (Electrical Erasable Programable Read Only Memory) mémoire de type PROM que l'on peut effacer électriquement en écrivant à nouveau sur le contenu de la mémoire. Ce type de mémoire par sa simplicité de mise en œuvre tend à remplacer de plus en plus la mémoire EPROM.

Exemple de carte d'extension mémoire pour un API TSX 37.Format PCMCIA type1 - RAM 32K / 64K - EEPROM 32K / 64K

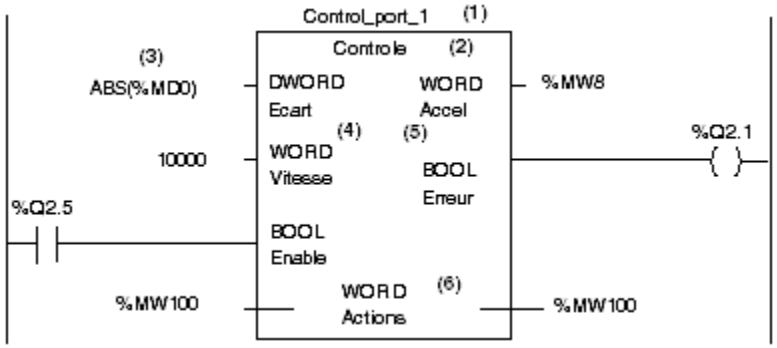
Langages de programmation pour API

Chaque automate possède son propre langage. Mais par contre, les constructeurs proposent tous une interface logicielle répondant à la norme CEI1 1131-3. Cette norme définit cinq langages de programmation utilisables, [15] qui sont :

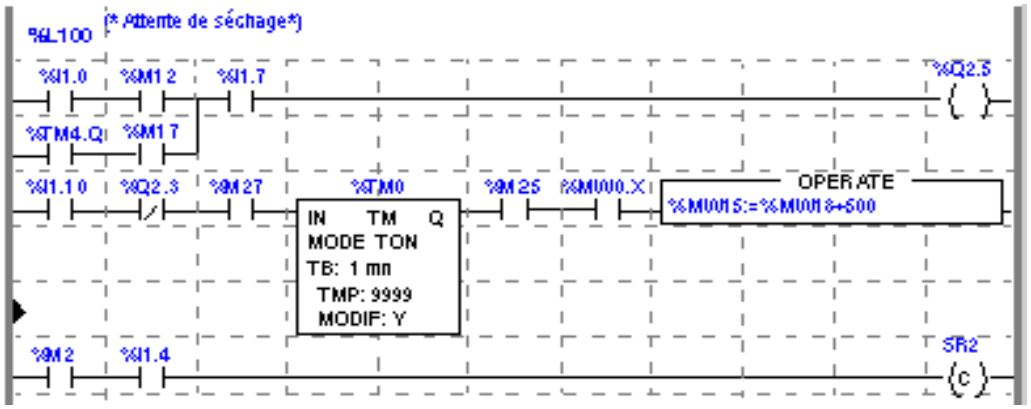
– GRAFCET ou SFC : Ce langage de programmation de haut niveau permet la programmation aisée de tous les procédés séquentiels.



– **Schéma par blocs ou FBD** : ce langage permet de programmer graphiquement à l'aide de blocs, représentant des variables, des operateurs ou des fonctions. Il permet de manipuler tous les types de variables.



– **Schéma a relais ou LD** : ce langage graphique est essentiellement dédie à la programmation d'´equations booléennes (true/false).



– **Texte structure ou ST** : ce langage est un langage textuel de haut niveau.

Il permet la programmation de tout type d'algorithme plus ou moins complexe.

```
IF %M0 THEN
    FOR %MW99 := 0 TO 31 DO
        IF %MW100 [%MW99] <> 0 THEN
            %MW10 := %MW100 [%MW99];
            %MW11 := %MW99;
            %M1 := TRUE;
            EXIT;              (*Sortie de la boucle FOR*)
        ELSE
            %M1 := FALSE;
        END_IF;
    END_FOR;
ELSE
    %M1 := FALSE;
END_IF;
```

– Liste d'instructions ou IL : ce langage textuel de bas niveau est un langage `a une instruction par ligne. Il peut être compare au langage assembleur.

Pour programmer l'automate, l'automaticien peut utiliser :

```
!  %L0 :
        LD          %I1.0
        ANDN        %M12
        OR (        %TM4.Q
        AND         %M17
        )
        AND         %I1.7
        ST          %Q2.5
!  %L5 :
        LD          %I1.10
        ANDN        %Q2.3
        ANDN        %M27
        IN          %TM0
        LD          %TM0.Q
        AND         %M25
        AND         %MW00:X5
        [ %MW15 :=  %MW18+500]
```

– une console de programmation ayant pour avantage la portabilité.

– un PC avec lequel la programmation est plus conviviale, communiquant avec l'automate par le biais d'une liaison série RS232 ou RS485 ou d'un réseau de terrain.

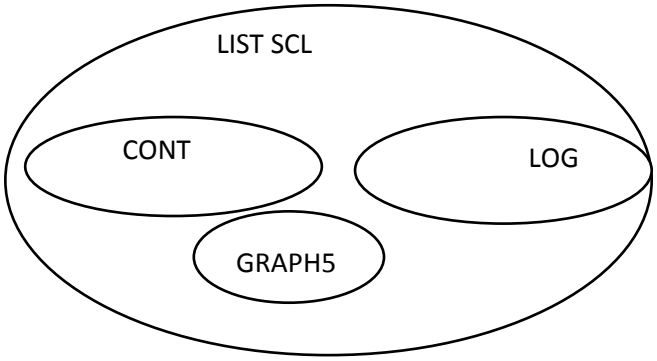
Exemple

PROGRAMMATION DE L'AUTOMATE SIEMENS S7-400

PRESENTATION

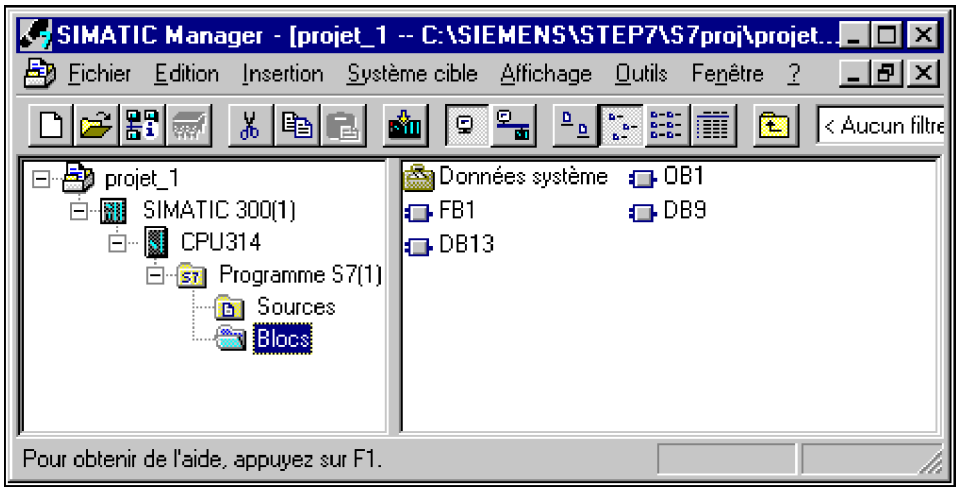
Les automates SIEMENS couvrent une gamme d'automates de 12E 8S à 4096 E/S plus des entrées et sorties analogiques, ainsi que des cartes spécifiques de régulation, comptage etc.. Ils sont programmables par l'intermédiaire de deux logiciels qui sont STEP 5 pour la série S5 (95U,100U,115U,135U, 155U) ou STEP 7 pour la nouvelle série 7(200,300,400).

La programmation STEP 7 est une programmation structurée dans des blocs qui sont les blocs d'organisation, les fonctions, les blocs fonctionnels, les blocs de données. L'écriture des programmes est possible sous plusieurs langages qui sont : le langage à contact **(CONT),** le logigramme **(LOG),** le langage en liste d'instructions **(LIST),** le grafcet **(graph S7),** le langage structuré **(SCL), etc.**

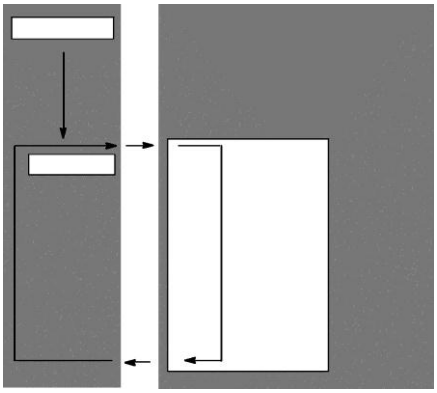


Gestionnaire de projets SIMATIC

Le gestionnaire de projets SIMATIC gère toutes les données relatives à un projet d'automatisation − quel que soit le système cible (S7/M7/C7) sur lequel elles ont été créées. Le gestionnaire de projets SIMATIC démarre automatiquement les applications requises pour le traitement des données sélectionnées.



IV.4 ORGANISATION GENERALE DES PROGRAMMES



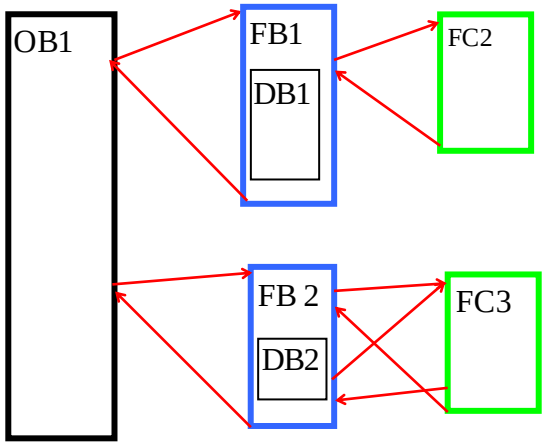
Le traitement de programme cyclique constitue le traitement normal pour les automates programmables. Cela signifie que Le programme utilisateur dans le bloc OB1 est donc exécuté cycliquement

L'**OB1** est le **b**loc d'**o**rganisation

Les **FC** sont les **b**locs **FONCTION**

Les **FB** sont les **b**locs **f**onctionnels

Les **DB** sont les **b**locs de **d**onnées



IV.4.1Traitement de programme cyclique

Blocs de code

Les blocs de code sont les blocs du programme utilisateur qui contiennent les instructions à exécuter. Il existe les blocs de code suivants :

Blocs d'organisation (a voir ultérieurement, ici seul OB1 pris en compte)

Le bloc d'organisation OB1 est le chef d'orchestre du programme ; il traite cycliquement le programme.

Fonction

Une fonction (FC) est un bloc de code **qui ne contient pas de données statiques**,

Bloc fonctionnel

Un bloc fonctionnel (FB) est un bloc de code **qui contient des données statiques**, (exemple programmation de graphe SFC dérivé du grafcet)

Fonction système (a voir ultérieurement)

Une fonction système (SFC) est une fonction intégrée au système d'exploitation de la CPU que vous pouvez appeler dans le programme utilisateur, si besoin est. Elle ne peut être programmée par l'utilisateur.

Les blocs de code (OB, FB, FC) du programme utilisateur peuvent être chargés dans la CPU S7. Ils sont soit créés et édités directement dans des éditeurs incrémentaux, soit ils résultent de la compilation de sources.

Blocs de données

Les blocs de données sont des blocs utilisés par les blocs de code de votre programme utilisateur pour enregistrer des valeurs.

(en langage graph on y retrouve toutes les données des variables des GRAFCET)

IV.4.2 LES LANGAGES

Langage de programmation LIST (liste d'instructions)

La langage de programmation LIST (liste d'instructions) est un langage textuel proche du langage machine.

Exemple :

Réseau 1 :

U(

O " Marche" // Bouton-poussoir "Marche"

O " Bobine" // Bobine de maintien

)

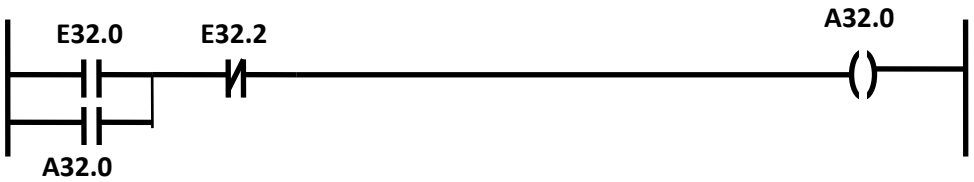
Langage de programmation CONT (schéma à contacts)

Dans le langage de programmation graphique CONT, la représentation est fondée sur des schémas à relais. Les éléments d'un tel schéma, comme par exemple les contacts à ouverture ou les contacts à fermeture sont reliés pour former des réseaux. Un ou plusieurs de ces réseaux forment la section d'instructions complète d'un bloc de code.

Le langage de programmation CONT fait partie du logiciel de base STEP 7.

Exemple :

Réseau 1 :



Dans le langage CONT, vous créez le programme en utilisant un éditeur incrémental.

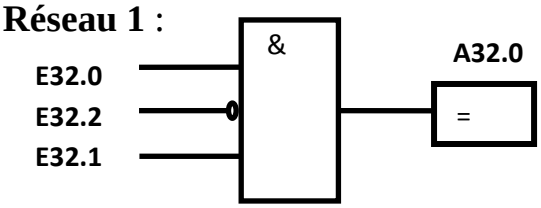
Langage de programmation LOG (logigramme)

Le langage de programmation LOG utilise les pavés logiques bien connus dans l'algèbre booléenne pour la représentation logique. Il permet en outre de représenter des fonctions complexes, telles que les fonctions mathématiques en les mettant directement en liaison avec ces pavés logiques.

Le langage de programmation LOG fait partie du logiciel de base STEP 7.

Exemple :

Réseau 1 :



Dans le langage LOG, vous créez le programme en utilisant un éditeur incrémental.

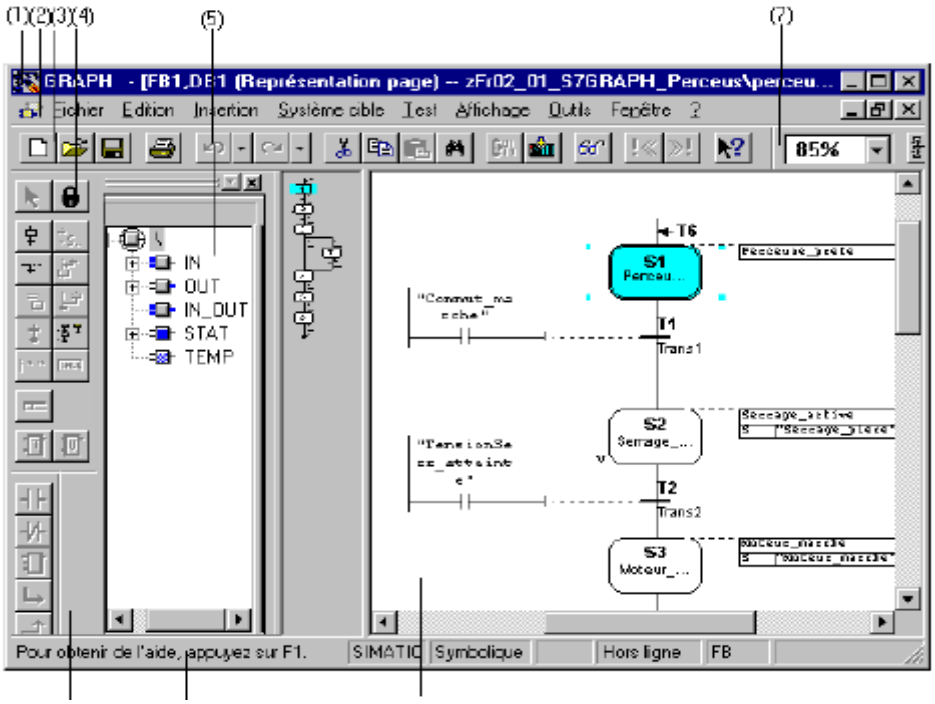
Langage de programmation Graph S7 (commande séquentielle)

Le langage de programmation graphique optionnel Graph S7 vous permet de programmer des commandes séquentielles. Ceci implique la création d'une succession d'étapes, la détermination du contenu respectif de ces étapes, de même que des conditions de transfert (transitions). Pour déterminer le contenu

des étapes, vous utilisez un langage de programmation spécial (similaire à LIST), alors que pour déterminer les transitions, vous utilisez une représentation sous forme de schéma à contacts ou de logigramme(langage de programmation CONT ou LOG restreint).

Graph S7 permet de représenter très clairement des séquences même complexes, ce qui favorise une programmation et une recherche d'erreurs effectives.

Exemple d’écran graph. 7



Graph. S7 ne crée que des blocs fonctionnels et les blocs de données d'instance correspondants, qu'il enregistre dans le programme utilisateur sous le programme S7. Vous créez le DB d'instance pour un FB de Graph dans l'éditeur de Graph S7, lorsque le bloc fonctionnel est ouvert. Il contient les données de la

série de séquences, comme par exemple les paramètres du bloc fonctionnel de même que la description des étapes et des transitions.

Dans Graph S7, vous créez le programme dans un éditeur incrémental. Un bloc fonctionnel de Graph peut cependant même être enregistré comme source Graph, s'il contient des erreurs. La source pourra à nouveau être compilée en bloc fonctionnel de Graph, une fois que les erreurs y auront été corrigées.

Mnémoniques de la table des mnémoniques

Lorsque vous créez votre programme, vous pouvez utiliser des mnémoniques auxquels vous pouvez aussi affecter des commentaires et des propriétés d'objets

Editer une table des mnémoniques

L'objet "Mnémoniques" (table des mnémoniques) est automatiquement créé sous un programme S7 Procédez de la manière suivante :

1. Cliquez deux fois sur le programme S7 ou le programme M7 dans la fenêtre du projet afin que l'objet "Mnémoniques" s'affiche dans la partie droite de la fenêtre.

2. Ouvrez l'objet "Mnémoniques", par exemple en cliquant deux fois dessus.

Dans la fenêtre qui s'ouvre, vous pouvez éditer la table des mnémoniques.

IV.5 PROJET S7

Créer un projet

Pour réaliser votre tâche d'automatisation au sein d'un gestionnaire de projets, vous devez d'abord créer un nouveau projet.

Le nouveau projet va être créé dans le répertoire que vous avez sélectionné pour les projets, lorsque vous avez choisi la commande Outils > Paramètres et l'onglet "Général".

Procédez de la manière suivante :

A/ Choisissez la commande Fichier > Nouveau .

B/ Dans la boîte de dialogue qui s'affiche, sélectionnez le chemin (lecteur et répertoire) pour le nouveau projet.

Nota : les noms de répertoire figurant dans le chemin ne doivent pas dépasser huit caractères, sans quoi des problèmes risqueraient de se poser lors de l'archivage et de l'utilisation de "C pour M7" (Borland Compiler).

C/ Tapez le nom du nouveau projet dans la boîte de dialogue.

D/ L'option "Type" vous permet d'indiquer si vous voulez créer le projet pour la version actuelle de STEP 7 (sélection par défaut) ou l'éditer dans une version plus ancienne de STEP7.

E/ Cliquez sur le bouton "OK" pour créer le projet.

Résultat : Une fenêtre du projet s'ouvre. Elle contient l'icône du projet ainsi que l'objet du sous-réseau MPI, qui est inséré automatiquement lors de la création d'un projet. La vue hors ligne est sélectionnée par défaut.

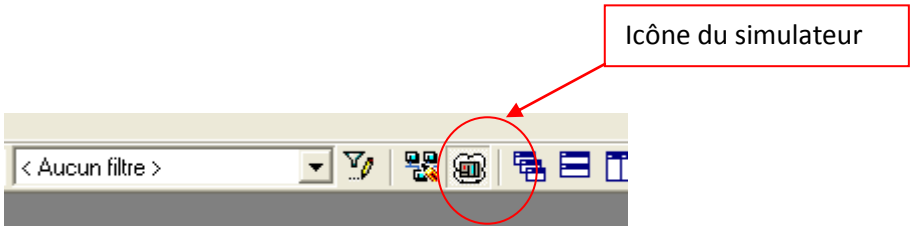
Nota : Pour ouvrir un projet existant, choisissez la commande Fichier > Ouvrir, puis sélectionnez le projet souhaité dans la boîte de dialogue qui s'ouvre. Si ce projet ne figure pas dans la liste de projets proposée, cliquez sur le bouton "Parcourir". L'arborescence de répertoires de la boîte de dialogue "Parcourir" s'affiche alors. Ce faisant vous pouvez chercher d'autres projets et reporter les projets trouvés dans la liste des projets. Vous pouvez modifier les entrées dans la liste de projets en choisissant la commande Fichier > Gérer.

Le gestionnaire de projets SIMATIC vous permet d'attribuer des noms dont le nombre de caractères est supérieur à 8. Le nom du répertoire du projet est tronqué après 8 caractères. Les noms de projets doivent de ce fait se distinguer dans leur 8 premiers caractères. Aucune différenciation n'est faite entre les majuscules et minuscules.

Test d'un programme utilisateur

Pour pouvoir tester vos programmes utilisateur, vous devez d'abord en copier les parties correspondantes dans la CPU, puis les exécuter.

Si l’on ne dispose pas encore du matériel, on utilisera le simulateur.



Vous devez préalablement avoir chargé le programme utilisateur dans la CPU.

Procédez de la manière suivante :

1. Ouvrez la fenêtre "Partenaires accessibles" ou la vue du projet en ligne.

2. Sélectionnez

· Le programme S7 dans la fenêtre du projet dans la vue du projet en ligne ou

· Le partenaire "MPI=..." dans la fenêtre "Partenaires accessibles".

3. Choisissez la commande Système cible > Visualiser/forcer des variables.

L'application de visualisation et de forçage des variables démarre et la fenêtre de la table des variables s'ouvre.

Vous pouvez également ouvrir cette fenêtre dans la vue du projet hors ligne, en ouvrant une table des variables (VAT) dans le classeur des blocs, par exemple par double-clic.

Insérer et éditer une table des variables

Procédez de la manière suivante :

1. Dans la fenêtre du projet (**vue du projet hors ligne**), sélectionnez le classeur des blocs dans lequel vous voulez insérer une table des variables (VAT).

2. Choisissez la commande Insertion > Bloc S7 > Table des variables (VAT).

3. Ouvrez l'objet VAT. L'application de visualisation et de forçage des variables sera alors démarrée. Vous pouvez y :

· éditer cette table des variables.

· créer d'autres tables de variables, les ouvrir et les éditer,

· établir une liaison en ligne avec la CPU, à partir du programme utilisateur à tester.

Type de variable dans un automate

Entrées **E (I)** (lecture dans la Mémoire Image d'Entrées MIE)

E y.x désigne une entrée, y est le numéro de voies (0 à 127), x sa position (0 à 8).

- EB y désigne un octet d'entrées.

- EW y désigne un mot d'entrées (16 bits).

- ED y désigne un double mot d'entrées (32 bits).

Les même termes précédé d'un P accèdent directement à la périphérie.

Sorties **A (Q)** (sortie dans la Mémoire Image de Sorties MIS)

- A y.x désigne une sortie y est le numéro de voies (0 à 127), x sa position (0 à 8).

- AB y désigne un octet de sorties.

- AW y désigne un mot de sorties (16 bits).

- AD y désigne un double mot de sorties (32 bits).

Les mêmes termes précédés d'un P accèdent directement à la périphérie

Mémentos **M** (lecture dans la mémoire interne)

- M y.x désigne un bit de mémoire y est le numéro d'octets (0 à 127), x sa position (0 à 8).

- MB y désigne un octet de mémoire.

- MW y désigne un mot de mémoire (16 bits).

- MD y désigne un double mot de mémoire (32 bits).

ATTENTION A LA NUMEROTATION.

MD 10 comprend les mots MW 10 et MW 12 soit les octets MB 10, MB 11, MB12, MB13. Les doubles mots sont donc adressés de 4 en 4 dans la mémoire, les mots de 2 en 2.Exemple :

MD 10 = AABB CCDD en hexadécimal.

alors en hexadécimal :

MW10 = AABB

MW12 = CCDD

MB 10 = AA

MB 11 = BB

MB 12 = CC

MB 13 = DD

Si MB 10 = 1000 0011 en binaire

Les Bits M 10.2 à M 10.6 valent 0, les bits M10.0, M10.1 et M10.7 valent 1.

Ce type de numérotation s'applique à tous les types de variables (A.., E..,P..., DB..,L..).

Les séries d'exercices

Serie1

Exercice 1

1. Pour le RDP de la figure1 indiquer :

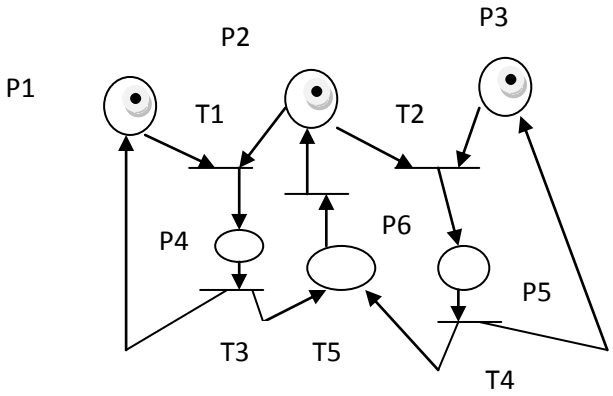


Figure1

- Les places d'entrée et de sortie des transitions.
- Le marquage initial du RDP.
- Les transitions franchissables.
- Le graphe de marquage accessible de réseau.

2. Ecrire le programme correspondant, figure1 en langage contact pour l'automate S7-300:
3. Pour le RDP de la figure 2 :

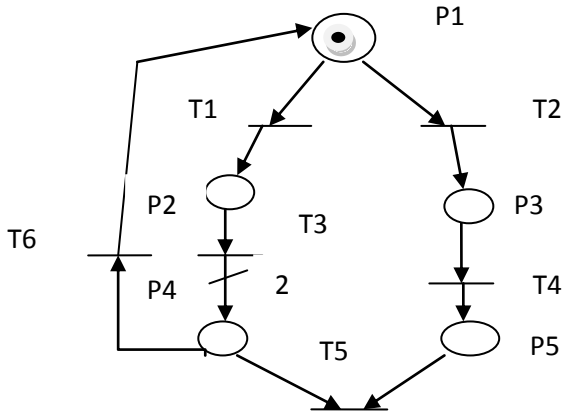


Figure2

En utilisant l'équation fondamentale quel sera le marquage après le franchissement des séquences possibles parmi les séquences suivantes :

S1= T1T3T6T6T2, S2= T1T3T2T4T3T4, S3= T1T3T6T2T4T5 et S5= T1T3T6T6T2T1T3T6T2T4.

Exercice 2

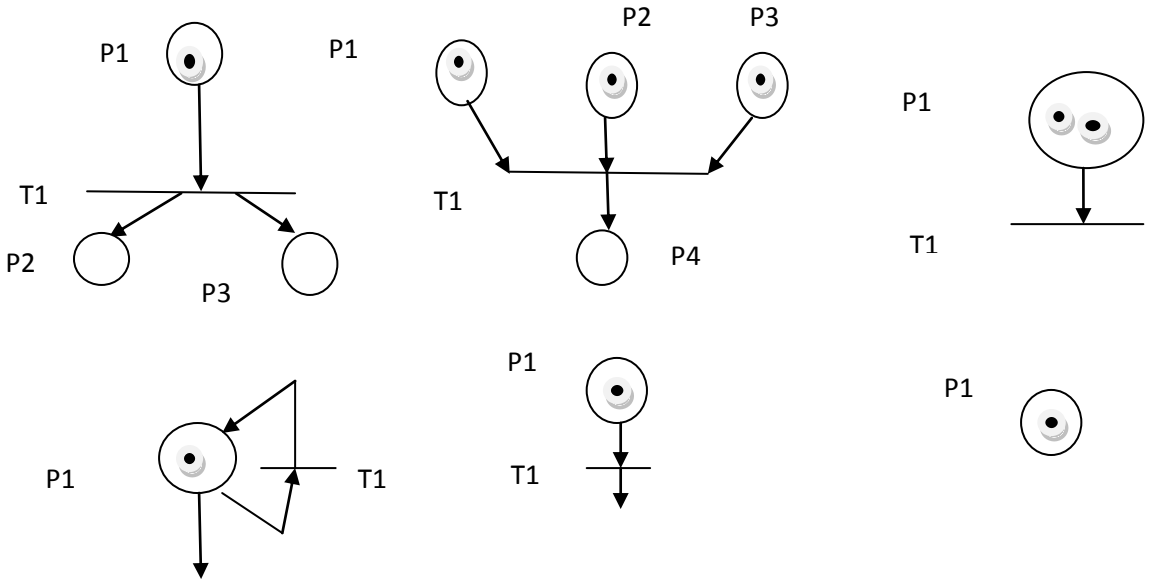
1. Les réseaux de la figure ci-dessous sont ils des réseaux de pétri?

2. Pour ceux qui sont des réseaux de pétri indiquer :

✓ Les transitions qui sont validées,

✓ Les marquages après franchissement,

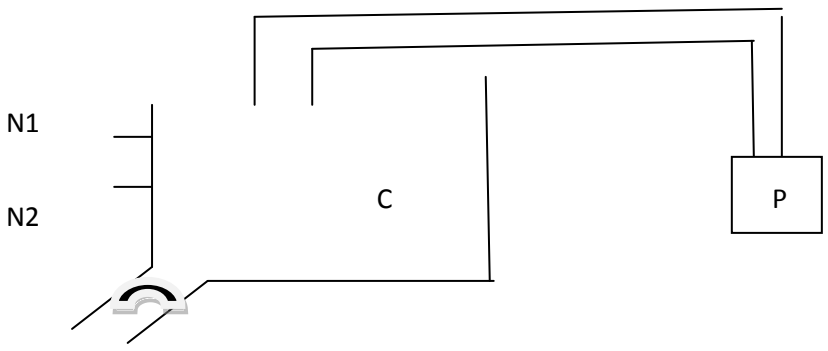
✓ Les transitions qui sont encore validées après franchissement.



Exercice 3

Donnees:

P: Pompe



C: Cuve alimentant un réseau de distribution dont la demande est variable.

- L'utilisateur veut maintenir le niveau du liquide N a un niveau N 2<N <N1.
- La pompe est activée si le niveau N devient inferieur à N2.
- La pompe est désactivée lorsque le niveau dépasse N1.

Condition initiale : Cuve pleine, pompe en arrêt.

Questions :

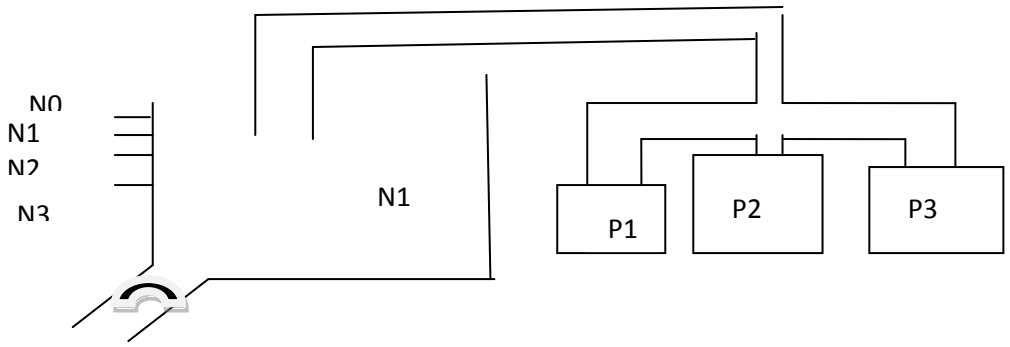
- Etablir le réseau de pétri de ce système.
- Quelle est la caractéristique que doit satisfaire la pompe pour assurer un bon fonctionnement du système.

Exercice 4 :

Données:

P1, p2, p3: trios Pompes.

C : Cuve alimentant un réseau de distribution dont la demande est variable.

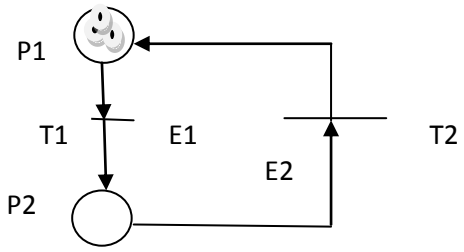


- ✓ Le débit de l'ensemble des trois pompes est supérieur au débit maximal de la demande.
- ✓ L'utilisateur veut maintenir le niveau du liquide N a un niveau N>N1et cela en actionnant :
- ✓ Une des trois pompes (P1) lorsque N <N1.

- ✓ Deux pompes (P1, P2) lorsque N <N2.
- ✓ Les trois pompes lorsque N <N3.

Exercice 5

Considérons le réseau de pétri de la figure suivante :



- Quelles sont les avances synchroniques maximales relatives aux transitions T1 et T2.
- Représenter le graphe des marquages accessibles du réseau.

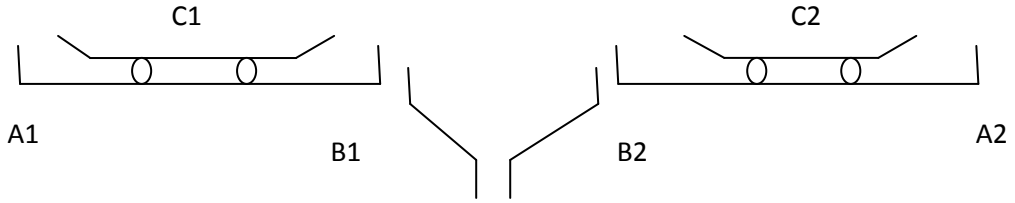
Serie2

Exercice 01

Données :

Les deux chariots C1 et C2se déplacent entre les points A et B.

- L'ordre de mise en service du système est donné sur un bouton DC (direct current).



- Le chargement des deux chariots dans les postes A est manuel.

- Au poste Ai(i1 ou 2), des qu'un wagon est charge, on lui donne l'ordre de départ sur un bouton FCi.
- L'arrivée du chariot Ci aux points Ai et Bi est détectée par des capteurs de fin de courses Ai et Bi respectivement.
- Le déchargement (dans le poste B) des chariots s'effectue automatiquement C1 et C2 (dans l'ordre).

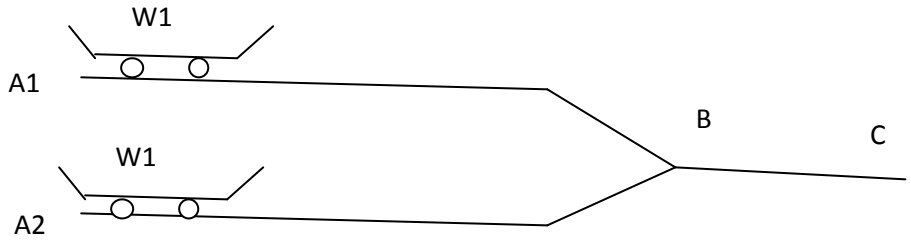
Condition initiale :

Initialement les deux chariots se trouvent en A(A1 et A2).

Question

Etablir le GRAFCET de ce système.

Exercice 02



L'ordre de mise en service du système est donné sur un bouton DC (direct current).

Les deux wagons W1 et W2 se font charger en (A1 et A2), le décharge en C.

La partie commune BC ne peut pas être occupe par les deux chariots.

Au poste A1 comme au poste A2, des qu'un wagon est charge, on lui donne l'ordre de départ sur un bouton FC.

L'arrivée du wagon Wi aux points A1, A2, B et C est détectée par des capteurs de fin de course A1, A2, B et C respectivement.

Condition initiale :

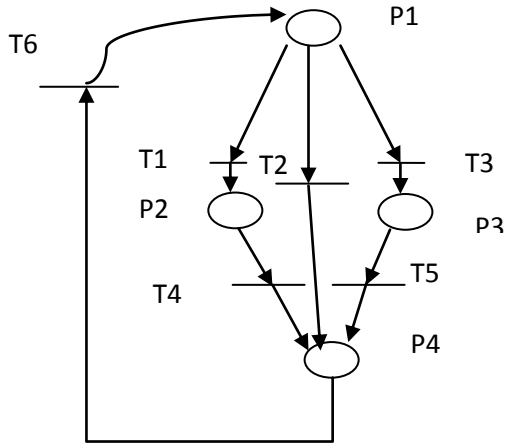
Initialement les deux chariots se trouvent en A (A1 et A2).

- **Question**
- Etablir le GRAFCET de ce système.

Exercice 03

Considérons le réseau de pétri ci-joint.

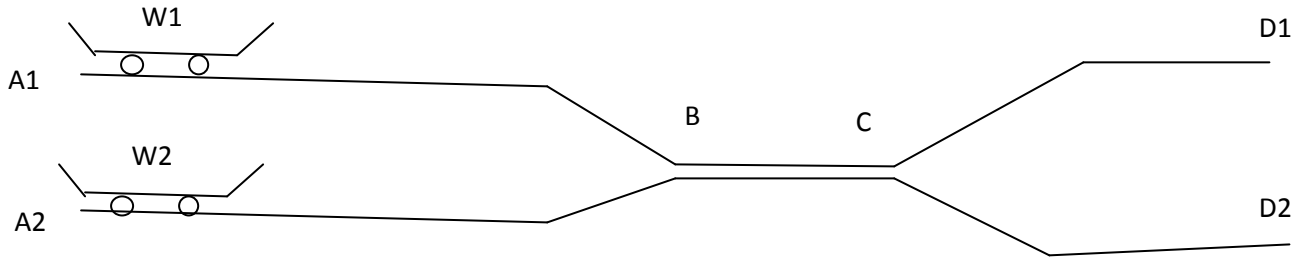
- Etablir le GRAFCET de ce système.



Exercice 4:

Données :

- Le chargement des deux chariots dans les postes A est manuel.
- Sur un ordre M les deux wagons se font charger avant de se déplacer vers les postes de déchargement D1 et D2.(W1 en D1 et W2 en D2).



Un wagon ne peut s'engager sur la voie commune que si elle est vide.

On considère que les wagons n'arrivent jamais en même temps en B et C.

Les wagons reviennent aux postes de changement par le même chemin en prenant les mêmes précautions que lors du trajet daller.

Condition initiale :

Initialement les wagons W1 et W2 sont respectivement en A1 et A2.

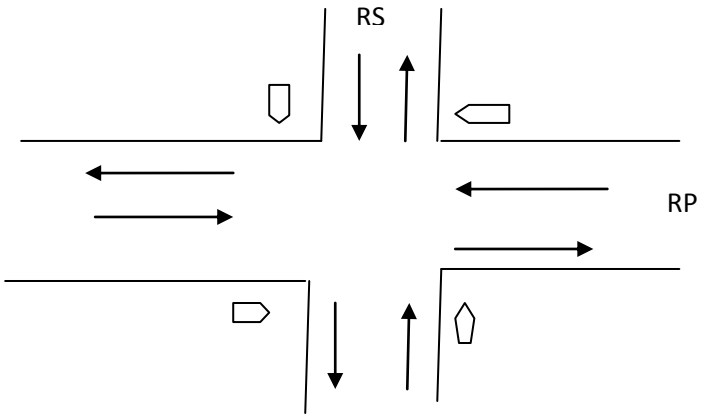
Question :

Etablir le réseau de Pétri et le GRAFCET de ce système.

Exercice 5

Soit le système de commande de feux de signalisation a un carrefour entre une voie principale et une voie secondaire.

Pour chaque voie et chaque sens de circulation, le carrefour est protégé par des feux tri couleurs.



Les feux de la voie principale et de la voie secondaire sont commandes respectivement par les signaux de RP et RS pour le rouge, OP et OS pour l'orange, ainsi que VP et Vs pour le vert.

Les feux verts de la voie principale restant allument pendant une durée de %mn, alors que les feux verts de la voie principale restant allument que pendant une durée égale à 2 mn, on a donc :

Vp: 5mn, OS, OP: 30s, VS: 2mn.

Question:

Donnée le modèle GRAFCET correspondant.

Les références

1. M.pinot &al, Du Grafcet aux automates programmables, collection L.P édit Foucher. Paris 1986.
2. G.Boujat & J.P. Pesty, Automatismes , collection AGATI. Serie Bac pro. Edit Dunod. Paris 1986.
3. F. Degoulange & al , Automatismes, classe de première et terminale E, F.BT. formation permanente. Edit Dunod. Paris 1983.
4. Paker Pneumatic &Telepneumatic, constituants pneumatiques, catalogue technique. Juin 1995.