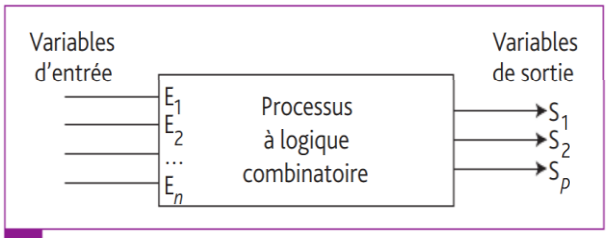
FICHE 44 : Outils de description du comportement d'un processus

I. Processus à logique combinatoire ou séquentielle ?

Un processus est dit à **logique combinatoire si les variables de sorties ne dépendent que des variables d'entrées** [document 22].



22 Structure d'un processus à logique combinatoire.

Le but étant de décrire le comportement du processus, il faut donc définir les fonctions logiques reliant les sorties aux entrées.

$$S_1 = f_1(E_1, E_2, \ldots, E_n)$$
$$S_2 = f_2(E_1, E_2, \ldots, E_n)$$
$$\ldots\ldots\ldots$$
$$S_p = f_p(E_1, E_2, \ldots, E_n)$$

Les fonctions f_1, f_2, …, f_p sont appelées fonctions logiques car elles manipulent des variables logiques binaires (0 ou 1). Ces fonctions permettent de décrire le comportement global du système étudié.

La description du comportement d'un système combinatoire peut être faite soit par :

- Une table de vérité ;
- Une fonction logique avec les opérateurs logiques de base ;
- Un logigramme.

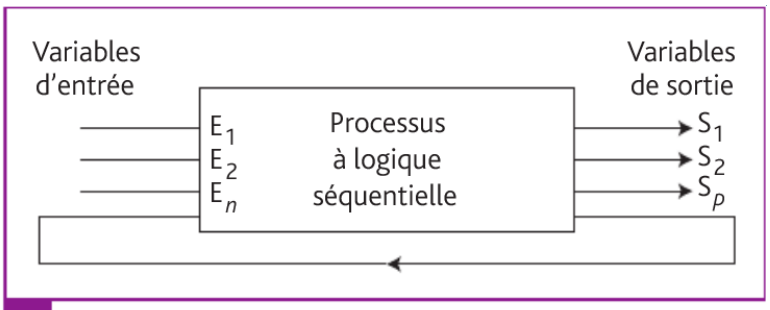
Une table de vérité est un tableau donnant l'état des variables de sorties d'un système logique en fonction de l'ensemble des états possibles des variables d'entrées. Exemple : un système à deux variables d'entrées (a et b) et deux variables de sorties (X et Y).

Entrées		Sorties	
a	b	X	Y
0	0	1	1
0	1	1	1
1	0	0	1
1	1	1	0

Un processus est dit à **logique séquentielle lorsque la ou les variables de sorties dépendent de la combinaison des variables d'entrées mais aussi de l'état précédent des sorties et/ou de la variable temps** [document 23].

Une même cause (même combinaison des entrées E_1, E_2, …, E_n) peut produire des effets différents (états différents des sorties S_1, S_2, … S_p). L'effet peut persister si la cause disparaît.

$S_i = f(E_1, E_2, \ldots, E_n, S_1, S_2, \ldots, S_p, t)$



23 Structure d'un processus à logique séquentielle.

La description du comportement d'un processus à logique séquentielle peut être faite soit par :

- Un algorithme (ou algorigramme) ;
- Un graphe d'état (fourni à titre d'information).

Ressources

II. Démarche de description du comportement d'un processus

Étape 1 : définir la frontière d'étude du processus étudié.

Étape 2 : recenser les variables d'entrées et de sorties et leur associer des noms de variables binaires (les noms associés aux variables sont parfois appelés mnémoniques).

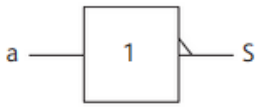
Étape 3 : identifier si le comportement du processus relève de la logique combinatoire ou séquentielle.

Étape 4 : traduire le comportement observé ou le comportement souhaité à l'aide d'un des outils les plus appropriés.

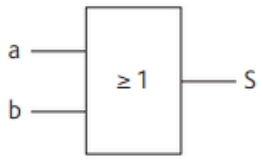
Étape 5 : tester le fonctionnement par simulation par exemple.

III. Les opérateurs logiques de base

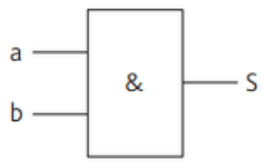
a. La fonction NON

Symbole	Équation logique	Table de vérité
	$S = \overline{a}$	a | S 0 | 1 1 | 0
Cette fonction complémente le niveau logique présent sur son entrée.		

b. La fonction OU inclusif

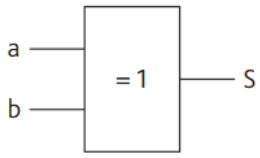
Symbole	Équation logique	Table de vérité
	$S = a + b$	a | b | S 0 | 0 | 0 0 | 1 | 1 1 | 0 | 1 1 | 1 | 1
Cette fonction présente un niveau logique haut sur sa sortie dès qu'au moins l'une de ses entrées est au niveau logique haut.		

c. La fonction ET

Symbole	Équation logique	Table de vérité
	$S = a \cdot b$	a | b | S 0 | 0 | 0 0 | 1 | 0 1 | 0 | 0 1 | 1 | 1
Cette fonction positionne sa sortie au niveau logique haut si toutes ses entrées sont au niveau haut.		

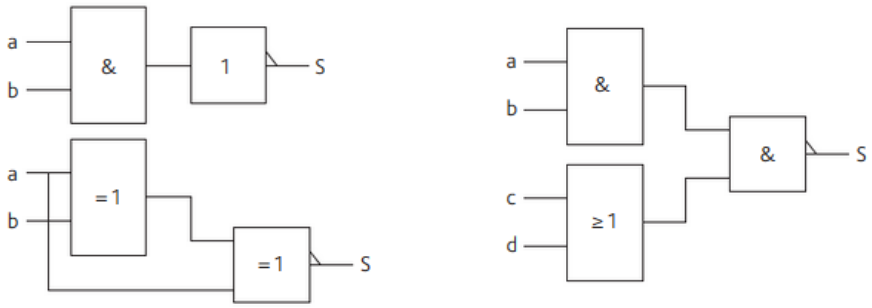
Ressources

d. La fonction OU Exclusif

Symbole	Équation logique	Table de vérité
	$S = a \oplus b$ ou $S = \bar{a} \cdot b + a \cdot \bar{b}$	a b S 0 0 **0** 0 1 **1** 1 0 **1** 1 1 **0**
Cette fonction présente sur sa sortie un niveau logique haut si ses entrées sont à un niveau logique différent.		

IV. Association d'opérateurs logiques de base

Il est possible d'associer des opérateurs logiques de base en les connectant en « cascade ». Un opérateur logique, dont l'une des entrées est reliée à la sortie d'un autre opérateur, se voit appliquer sur son entrée le résultat obtenu sur la sortie de l'opérateur logique précédent. On combine chaque nouvelle valeur des entrées dans l'équation de celui-ci pour obtenir l'équation de sortie [document 24].



24 Exemple d'association d'opérateurs logiques de base.

V. Détermination de fonctions logiques à partir d'une table de vérité

En supposant que, suite à une étude d'un cahier des charges, nous obtenons la table de vérité suivante où a et b sont les variables d'entrées et S la variable de sortie :

a	b	S
0	0	0
0	1	1
1	0	0
1	1	0

Nous nous apercevons que la variable de sortie S vaut 1 quand a = 0 et b = 1.
Si nous codons S = 1 par S, a = 0 par a et b = 1 par b, nous pouvons écrire :

$$S = \overline{a}.b$$

Dans cette expression, le point signifie "et".

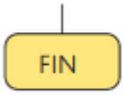
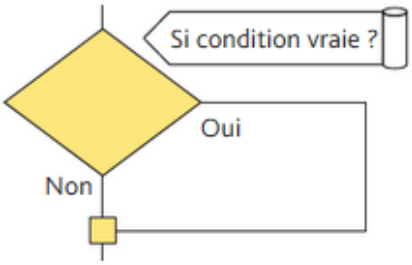
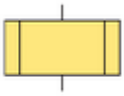
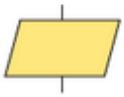
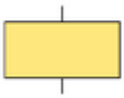
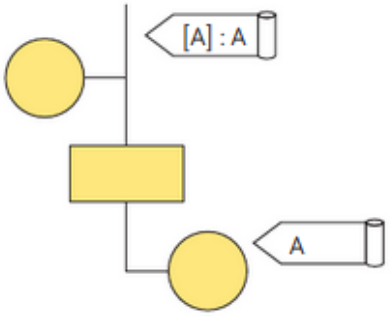
Ressources

VI. Description du comportement séquentiel par l'outil algorithme

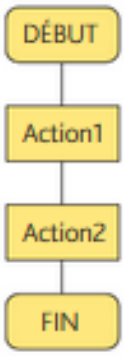
Un algorithme est l'ensemble de règles opératoires ordonnant à un microprocesseur d'exécuter dans un ordre déterminé un nombre d'opérations élémentaires.

Un algorigramme (appelé aussi ordinogramme) est une représentation graphique de l'algorithme utilisant des symboles normalisés.

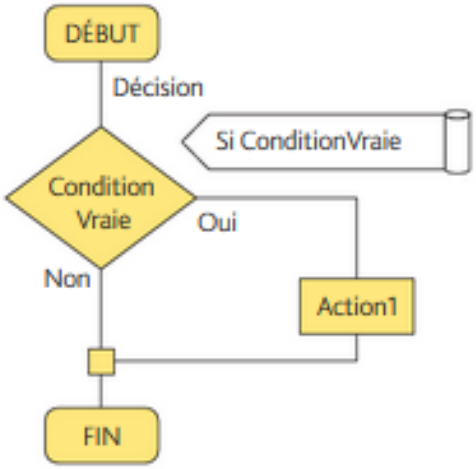
a. Constitution de l'outil de description algorigramme

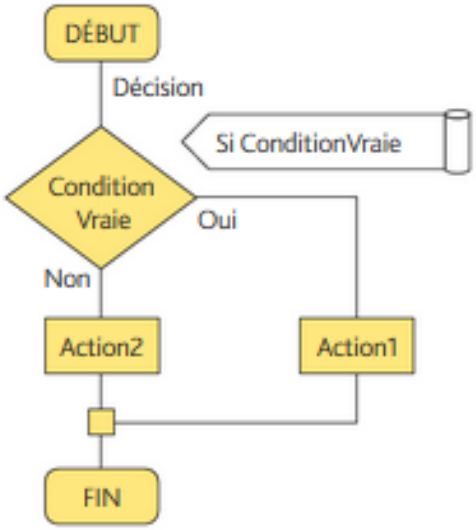
Symbole	Désignation du symbole
	Début de l'algorigramme
	Fin de l'algorigramme
	Test ou branchement conditionnel
	Appel à un sous-programme
	Symbole d'acquisition et ou de mise à jour des entrées/sorties
	Bloc de traitement de données
	Saut de séquence
	Commentaires

b. Constitution de l'outil de description algorigramme

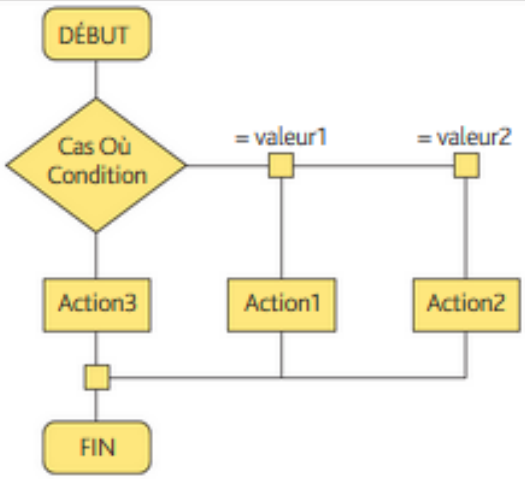
Algorigramme	Algorithme
	**Début** Action1 Action2 **Fin**
On exécute successivement une suite d'actions dans l'ordre de leur énoncé.	

Structures alternatives

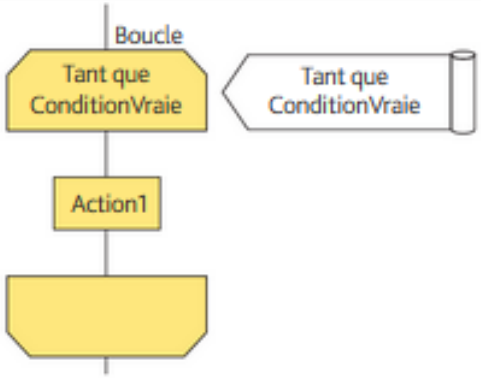
Algorigramme	Algorithme
	**Début** **Si** ConditionVraie **Alors** Action1 **Fin Si** **Fin**
Cette structure offre le choix de réaliser ou non une séquence d'action(s).	

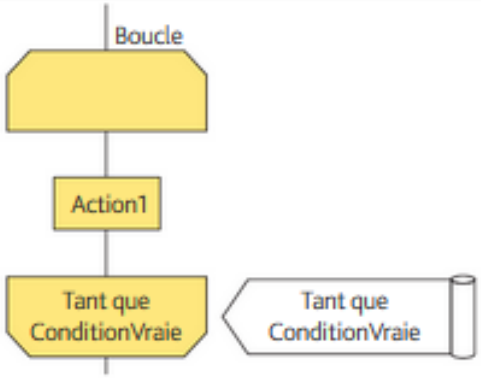
Algorigramme	Algorithme
	**Début** **Si** ConditionVraie **Alors** Action1 **Sinon** Action2 **Fin Si** **Fin**
Cette structure offre le choix entre deux séquences s'excluant mutuellement.	

Ressources

Algorigramme	Algorithme
	**Début** **Cas Où** Condition = valeur1 **Faire** Action1 Condition = valeur2 **Faire** Action2 **Autrement Faire** Action3 **Fin Cas Où** **Fin**
Cette structure offre le choix entre plusieurs séquences s'excluant mutuellement.	

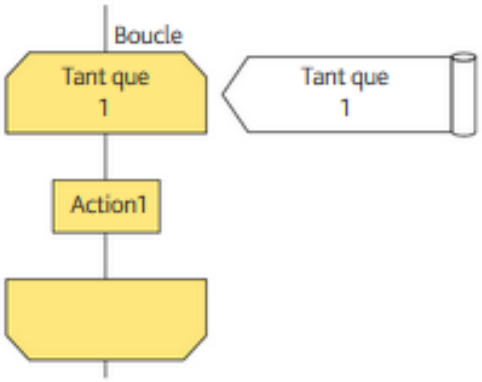
Structures itératives

Algorigramme	Algorithme
	**Tant que** ConditionVraie **Faire** Action1 **Fin Tant Que**
On teste d'abord la condition, la séquence est exécutée tant que la condition est vraie.	

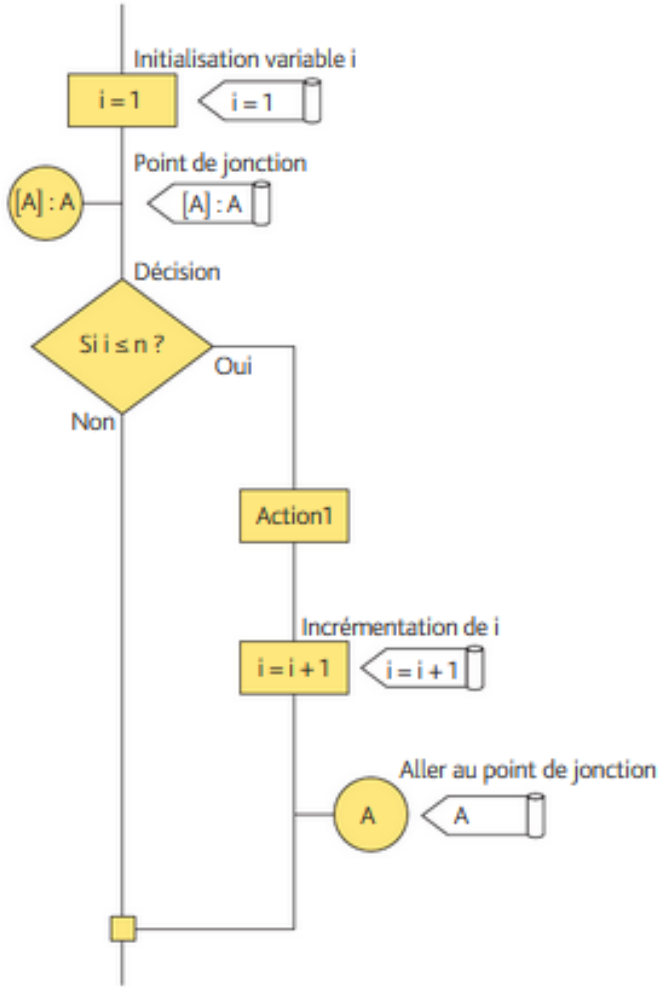
Algorigramme	Algorithme
	**Faire** Action1 **Tant que** ConditionVraie **Fin Tant Que**
L'action est exécutée au moins une fois, elle est répétée tant que la condition est vraie.	

Ressources

La structure **Tant que** … **Faire** … **Fin Tant Que** est très utilisée, notamment lorsque l'on veut faire une boucle infinie. On utilise alors :

Algorigramme	Algorithme
	**Tant que** 1 **Faire** Action1 **Fin Tant Que**

Structure à reprise de séquence

Algorigramme	Algorithme
	**Pour** i **allant** de 1 **à** n **avec pas de** 1 **Faire** Action1 **Fin Pour**
L'action ou la suite d'actions est exécutée un nombre de fois prédéfini.	

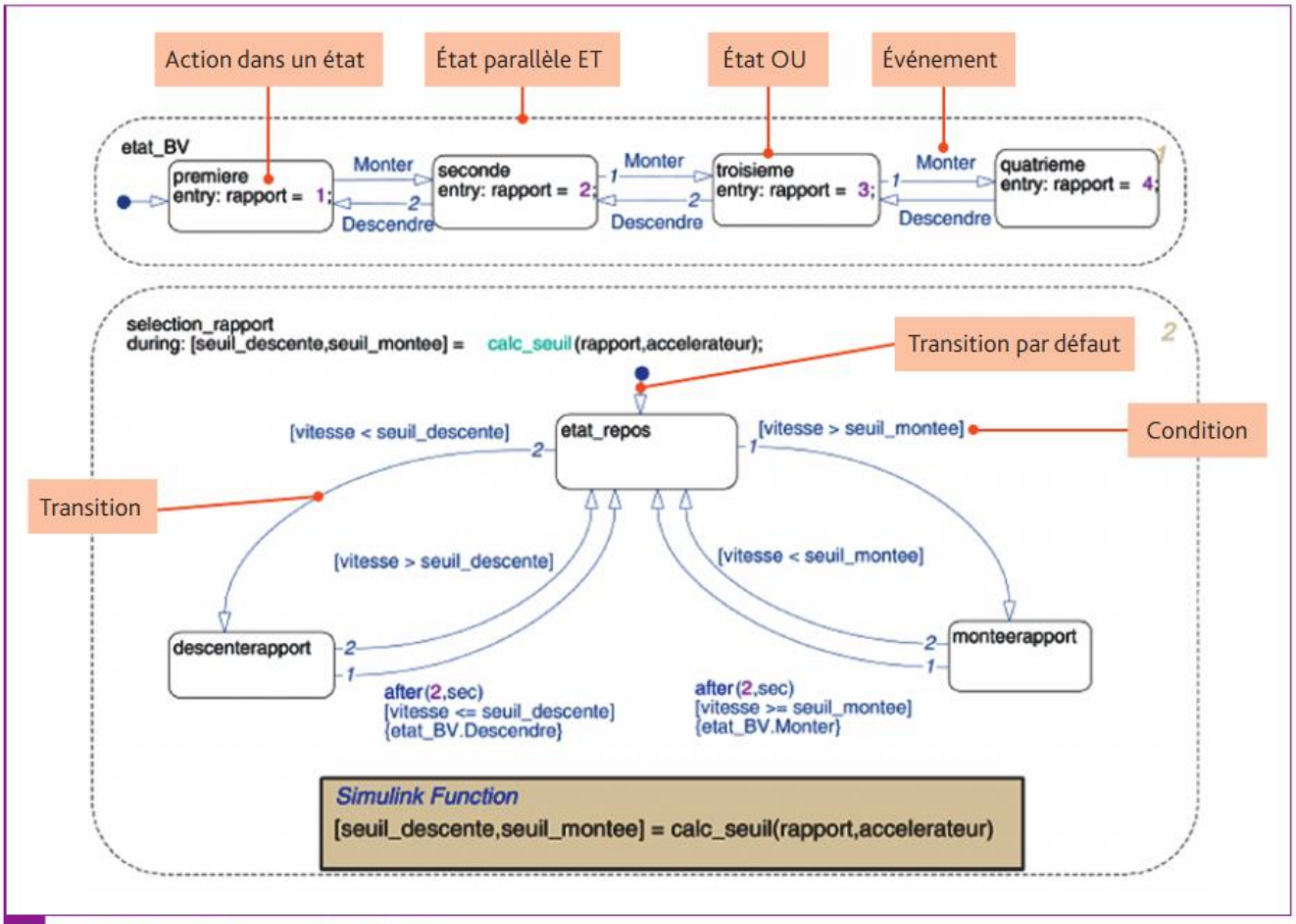
Ressources

VII. Description du comportement d'un processus par l'outil graphe d'états

Le **graphe d'états**, appelé aussi **diagramme états-transitions**, est l'un des outils permettant de décrire le comportement séquentiel d'un processus.

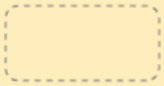
a. Constitution de l'outil graphe d'états

L'exemple ci-dessous montre les différents constituants d'un graphe d'états [document 25].



25 Constituants de graphe d'états.

b. Descriptif des constituants d'un graphe d'états

Symbole graphique	Dénomination	Descriptif
	État OU	Les états « OU » représentent des états de fonctionnement de manière mutuellement exclusive. Deux ou plusieurs états OU de même niveau hiérarchique ne peuvent être actifs simultanément.
	État ET	Les états « ET » représentent des états de fonctionnement totalement indépendants. Deux ou plusieurs états ET de même niveau hiérarchique peuvent être actifs simultanément. Ces états sont représentés graphiquement par un rectangle en traits pointillés.

Ressources

Symbole graphique	Dénomination	Descriptif
	Transitions	Les transitions sont représentées par des flèches orientées et permettent de décrire les évolutions du processus d'un état source vers un état destination.
[vitesse > seuil_descente]	Condition associée à une transition	Il s'agit de définir la condition pour passer d'un état source à un état destination. La condition est une fonction logique dont le résultat est binaire (vrai ou faux).
	Transition par défaut	Cette transition indique l'état (ou super-état) qui doit être actif à l'état initial. Elle ne peut apparaître que sur des états OU.
Nom_etat entry: actions1; during: actions2; exit: actions3;	Actions dans un état	Il s'agit de définir les actions à effectuer lorsque l'état est actif. On définit les trois types d'actions ci-dessous. • Action à l'activation de l'état : pour spécifier ce type d'action, la syntaxe est *entry* : *actions1*. • Action durant l'état : pour spécifier ce type d'action, la syntaxe est *during* : *actions2*. • Action à la désactivation de l'état : pour spécifier ce type d'action, la syntaxe est *exit* : *actions3*.

Remarque : lorsqu'un état est composé de sous-états, on parle de super-état, de macro-état ou d'état composite.

c. Règles d'évolution d'un diagramme états-transitions

Lorsque dans un diagramme états-transitions seuls des états OU sont présents, il est nécessaire de vérifier :
• qu'il y a toujours au moins un état destination (Condition 1) ;
• qu'il n'existe qu'un seul état destination (Condition 2).

Condition 1 : il doit être complet ou non ambigu.

Ceci signifie que le comportement du processus est toujours défini. À chaque évolution des variables d'entrées (conditions ou événements), et quel que soit l'état dans lequel se trouve le processus, il est nécessaire de pouvoir connaître l'état suivant. On peut traduire cette propriété sous forme d'équation booléenne en écrivant que le OU logique de toutes les conditions associées aux transitions partant d'un état quelconque est toujours vrai. Soit C1, C2, …, Ci ,…, Cn ces conditions, alors :

$$\sum_{i=1}^{i=n} C_i = 1$$

Condition 2 : il doit être non contradictoire.

Ceci signifie qu'à tout changement des variables d'entrées (condition ou événement), une seule transition est possible. Si plus d'une transition à sa condition associée est vraie, le diagramme est dit contradictoire (par exemple, deux actions contradictoires sont simultanément possibles). On peut traduire cette propriété en écrivant que le OU logique de tous les ET logiques de deux conditions associées aux transitions partant d'un état quelconque est toujours faux.

$$\sum_{i=1}^{i=n} \prod_{j=i+1}^{j=n} C_j = 0$$

Ressources