

#### D.3.8.2.1 *D'étermination des états requis*

L'auteur d'un diagramme LDS dispose généralement d'une certaine latitude pour définir les états d'un processus. Il peut avoir besoin d'une stratégie qui lui permette d'identifier les états du processus et cette stratégie peut être informelle ou formelle. Un jugement sûr (que seule une longue expérience permet d'acquiescer) est nécessaire si l'on veut établir des diagrammes LDS tels que l'identification d'un trop grand nombre d'états ne les complique pas inutilement ou qu'un nombre artificiellement réduit d'états ne les empêche d'exploiter les avantages qu'offre le LDS. Avant que l'auteur ne commence à établir le diagramme, certaines étapes préliminaires (étudiées au § D.3.2) doivent être achevées, par exemple:

- la structuration du système en blocs fonctionnels;
- la représentation d'un ou plusieurs processus LDS par bloc;
- le choix des signaux d'entrée et de sortie;
- l'utilisation des données dans le processus.

Tous les facteurs ci-dessus exercent un effet important dans la détermination des états de chaque processus. L'effet qu'exerce le choix des signaux d'entrée sur le nombre d'états d'un diagramme LDS est illustré par les deux exemples de la figure D-3.8.11.



Ayant appliqué une stratégie pour l'identification des états d'un processus, l'auteur d'un diagramme LDS estimera peut-être qu'un trop grand nombre d'états ont été utilisés. Le nombre des états est important car la dimension et la complexité d'un diagramme LDS sont souvent étroitement liées à ce nombre. Il existe certes des moyens qui permettent de réduire le nombre des états mais le fait qu'un diagramme LDS soit complexe n'est pas, en soi, une raison justifiant sa modification; en effet, la complexité du diagramme peut

être simplement due à la complexité inhérente au processus défini. Le choix de l'ensemble d'états doit généralement offrir le plus de clarté possible à la séquence d'interactions entre le processus et son environnement. Ce n'est d'ordinaire pas en minimisant le nombre d'états que l'on obtient cette clarté. Le nombre de séquences indépendantes traitées par un processus exerce un effet multiplicateur sur le nombre d'états. Il est donc souhaitable que les séquences indépendantes des processus séparés soient traitées de façon à réduire le nombre d'états et à obtenir une plus grande clarté.

On peut réduire le nombre des états en effectuant la séparation des fonctions communes ou la fusion des états ou encore en recourant au concept de service. Certaines structures de données peuvent également conduire à une réduction du nombre des états. Pour d'autres représentations, l'emploi des macros peut être avantageux.

### D.3.8.2.3 *Séparation des fonctions communes*

Lors de la mise en place d'un diagramme LDS, on peut constater que la définition d'un aspect particulier et répétitif d'un processus nécessite la représentation d'états répétitifs. La figure D-3.8.12 représente une partie d'un diagramme LDS définissant un processus de signalisation de ligne et illustrant la condition selon laquelle une tonalité de signalisation de ligne doit être présente pendant une certaine durée avant que l'on considère que le signal de ligne a été détecté.

Pour spécifier cet aspect, il convient d'insérer un état intermédiaire entre les états `aucun_signal_de_ligne_n'est_détecté` et `conversation`. Supposons que, dans un diagramme complet, une telle fonction commune doit être répétée à chaque point où le signal est détecté. Une autre solution consiste à définir un processus séparé qui est responsable du contrôle de la tonalité de signalisation de ligne et de la détection des signaux sur la base du temps de reconnaissance spécifiée. L'existence de ce nouveau processus permettrait de représenter le diagramme de la figure D-3.8.12 de la manière indiquée dans la figure D-3.8.13. (Dans un contexte donné, les figures peuvent être rendues équivalentes moyennant l'introduction d'un nouveau signal `signal_de_ligne_valide`.)

Il convient de noter la légère différence entre le processus de la figure D-3.8.12 et les deux processus de la figure D-3.8.13.

Le processus de la figure D-3.8.12 commence à compter dès réception de T1 alors que le processus de traitement des appels [processus b) de la figure D-3.8.13] commence à compter dès réception de `signal_de_ligne_valide` qui est engendré et émis à la réception de T1 par le processus a) de la figure D-3.8.13. Il s'ensuit que, dans le second cas, le comptage d'attente après T1 + temps de génération, d'émission et de réception du signal. En outre, d'autres signaux peuvent avoir été mis dans la file d'attente pendant le temps nécessaire à la génération et à l'émission de `signal_de_ligne_valide`.

Une seconde particularité est que cette séparation d'une sous-fonction ne serait pas possible si le signal que doit recevoir la sous-fonction devait être reçu également par la fonction principale, un signal étant toujours acheminé vers un seul processus. Si, dans l'exemple, le même signal `S_off` devait être reçu dans un autre état où il n'est pas besoin de le valider pour le temps T1, il n'aurait pas été possible de séparer la sous-fonction de validation en un autre processus.

Les solutions qui font appel à des processus distincts sont généralement utiles lorsque les signaux doivent être traités indépendamment des états dans le processus principal. Dans ce cas, les opérations qui précèdent et qui suivent les processus peuvent traiter les séquences détaillées et décharger un processus principal de tous les détails. Ceci engendre souvent une modularité utile permettant, par exemple, d'isoler les caractéristiques particulières des systèmes de signalisation, du processus principal plus axé sur le service.

Une autre manière de traiter le problème consiste à appliquer le concept de service, expliqué au § D.5.3.

Une solution différente du problème consisterait à employer la notation MACRO, comme indiquée à la figure D-3.3.1. Dans ce cas, on obtient pour le diagramme la compacité voulue sans modifier en rien la sémantique du

diagramme original. Il est en outre possible d'appeler la MACRO à partir de plusieurs états si la logique du processus l'exige.

**Figure D-3.8.12, p. 2**

**Figure D-3.8.13, p. 3**

#### D.3.8.2.4 *Fusion des états*

Si, dans un diagramme LDS, la destination future de deux états est la même, on peut, indépendamment de leur évolution antérieure, effectuer la fusion de ces deux états en un seul, sans affecter la logique du diagramme.

La figure D-3.8.14 représente une partie d'un diagramme LDS comportant deux états dont le <<passé>> est différent mais dont le <<futur>> est identique. Dans la figure D-3.8.15, les deux états ont été combinés en un seul. Il s'agit là d'un exemple relativement simple dans lequel la réduction de la complexité est peu importante, mais cette même technique peut être utilisée pour obtenir une plus grande simplification. La sémantique du LDS ne prévoit pas une décision consécutive à un état pour déterminer le <<passé>> du processus antérieurement à cet état (c'est-à-dire de déterminer, pour cet exemple, si A6 ou B4 a été émis), à moins que cette information n'ait été explicitement stockée avant l'entrée dans l'état. A noter, que l'attribution d'un nom aux données d'une entrée a pour effet de mettre la valeur en mémoire.

Un état doit représenter une situation logique du processus et il n'est donc pas souhaitable d'effectuer la fusion de situations logiques différentes en un seul état.

Des précautions sont à prendre si un diagramme fusionné doit être modifié ultérieurement. L'utilisateur devrait rechercher si la modification envisagée a ou non le même effet sur les diverses branches initiales.

**Figures D-3.8.14 et D-3.8.15, C | TE-A-C | TE, p. 4**

### D.3.8.3 *Entrées*

Le présent § D.3.8.3 a pour objet d'expliquer la notion d'entrée ainsi que l'utilisation des entrées dans des diagrammes LDS ne faisant pas appel à la notion de mise en réserve. La notion de mise en réserve et l'utilisation de cette notion en m | me temps que la notion d'entrée font l'objet du § D.3.8.4.

#### D.3.8.3.1 *Considérations générales*

Un symbole d'entrée attaché à un état signifie que, si le signal dont le nom figure dans le symbole d'entrée arrive pendant que le processus est dans cet état, il faut interpréter la transition qui suit le symbole d'entrée. Lorsqu'un signal a déclenché l'interprétation d'une transition, ce signal n'existe plus et on dit qu'il a été absorbé.

Un signal peut être accompagné de données associées. Par exemple, un signal portant le nom <<chiffre>> sert non seulement à déclencher l'exécution d'une transition par le processus de réception mais encore à véhiculer la valeur du chiffre (0 à 9), ces données pouvant être utilisées par le processus récepteur.

En LDS/PR, l'instruction INPUT contient une liste d'identificateurs de signaux. Les valeurs contenues dans les signaux sont indiquées à l'aide d'identificateurs de variables. Les identificateurs de variables doivent être du type indiqué dans la définition de signal, de sorte que leur position est très importante. Ces identificateurs de

variables sont placées entre parenthèses et séparées par des virgules (voir la figure D-3.8.16). Si une ou plusieurs valeurs de signal sont rejetées, les variables correspondantes font défaut, ce qui est indiqué par deux ou plusieurs virgules consécutives (figure D-3.8.17).



**Figure D-3.8.16 [T13.100] (à traiter comme tableau MEP), p. 6**

**Figure D-3.8.17 [T14.100] (à traiter comme tableau MEP), p. 7**

En LDS/GR, une entrée est représentée à l'aide d'un symbole d'entrée contenant la liste d'identificateurs de signaux et les identificateurs de variables correspondants pour les valeurs acheminées. Pour pouvoir être communiquées au processus, ces valeurs doivent être désignées nommément dans les symboles d'entrée, entre parenthèses.

On trouvera des exemples de réception de valeurs des entrées dans les figures D-3.8.18, D-3.8.19 et D-3.8.20.

Les données auxquelles un nom est assigné peuvent être utilisées par le processus récepteur quand l'entrée est interprétée.

La figure D-3.8.18 montre la réception du signal <<décroché>>. Le signal <<décroché>> est accompagné de données associées (numéro \_de \_l'abonné) avec pour valeur 9269. Le signal peut être reçu de trois manières, comme indiqué en a) et c) de la figure.

La figure D-3.8.19 montre comment envoyer et recevoir plusieurs valeurs en un seul signal. Chaque élément doit être séparé du suivant par une virgule. La figure D-3.8.20 c) montre comment ignorer les valeurs indésirables en laissant un blanc dans la liste de sorties.

A noter que, dans le signal de sortie, il est impossible d'écrire des expressions pour E1, E2 ou E3, alors que dans le signal d'entrée, il faut employer des variables pour recevoir les valeurs émises.

Dans le LDS, il n'est pas nécessaire de dessiner des symboles d'entrée pour représenter les signaux dont l'arrivée nécessiterait une transition nulle (c'est-à-dire une transition qui ne contient aucune action et qui ramène au même état). La convention admise est la suivante: pour tout signal qui n'est pas représenté dans un symbole d'entrée explicite à un état donné, il existe, dans cet état, un symbole d'entrée implicite et une transition nulle. Conformément à cette convention, les deux diagrammes représentés dans la figure D-3.8.21 sont logiquement équivalents et peuvent être indistinctement utilisés.

**Figure D-3.8.18, p. 8**

**Figure D-3.8.19, p. 9**

**Figure D-3.8.20, p. 10**

**Figure D-3.8.21, p. 11**

Lorsqu'un certain nombre d'entrées aboutissent à la même transition, tous les identificateurs de signaux qui s'y rapportent peuvent

être placés dans un même symbole d'entrée. Le LDS prévoit une notation abrégée pour représenter une entrée de tous les signaux (valable pour ce processus) non explicitement mentionnés dans cet état. La figure D-3.8.22 illustre cet aspect et les diagrammes qui y sont représentés sont logiquement équivalents. Si tous les identificateurs de signaux sont mentionnés, il faut les séparer par des virgules.

#### D.3.8.3.2 *Mécanisme implicite de mise en file d'attente*

Un ou plusieurs signaux peuvent être en attente d'absorption lorsqu'un processus parvient à un nouvel état. Cela signifie que les signaux doivent être mis en attente d'une certaine manière si l'on veut éviter qu'ils soient perdus. Lorsqu'un signal parvient au bloc de destination, il est placé dans la file d'attente d'entrée du processus de réception. La sémantique du LDS définit pour chaque processus un mécanisme théorique de mise en file d'attente selon lequel le mode de sélection des signaux pour l'absorption par le processus est fondé sur l'ordre d'arrivée des signaux dans ce processus. Lorsque le processus parvient à un état, il reçoit un seul signal en provenance de la file d'attente. Ceci signifie que si la file d'attente n'est pas vide, le processus absorbe le premier signal qui vient de la file d'attente en question. Si cette dernière est vide, le processus demeure en attente dans l'état jusqu'à l'arrivée à la file d'attente d'un signal qui est ensuite absorbé par le processus.

La figure D-3.8.23 utilise le concept de file d'attente pour expliquer le fonctionnement d'un processus LDS dans lequel les temps de transition sont différents de zéro. Il convient de noter les éléments suivants:

- le concept de mise en réserve n'est pas appliqué et les signaux sont absorbés dans l'ordre de leur arrivée;

— l'ordre de succession de l'arrivée des signaux est important. Si <<C>> était arrivé avant <<B>> dans la transition entre l'état 1 et l'état 2, la séquence des états aurait été 1, 2, 3 au lieu de 1, 2, 4;

**Figure D-3.8.22, p. 12**

- la file d'attente n'étant pas vide lorsque le processus arrive aux états 2 et 4, ce processus ne doit attendre ni dans l'un ni dans l'autre de ces états;

- il n'est pas possible d'attribuer la priorité à un signal. Un mécanisme spécial est prévu pour les communications entre services, afin que les signaux échangés entre service soient traités avant les autres signaux (§ D.5.3).

Si les temps de transition sont nuls, chaque signal sera absorbé au moment de son arrivée dans un processus, sauf si l'on a recours à la mise en réserve (voir le § D.3.8.4).

#### D.3.8.3.3 *Réception des signaux qui ne se présentent pas normalement*

Dans chacun des états, tous les signaux possibles doivent être indiqués implicitement ou explicitement. Des exceptions (signaux inattendus mais théoriquement possibles, signaux non définis ou logiquement faux à un endroit donné, etc.) peuvent se présenter dans presque tous les états. Normalement, l'auteur n'indique pas ces possibilités; il s'ensuit qu'un tel signal sera éliminé s'il se présente. Si toutefois l'auteur tient à faire figurer les exceptions dans son diagramme, il doit prévoir pour tous les états une entrée supplémentaire.

Une autre possibilité consiste à profiter des apparitions multiples d'un état portant le symbole <<tous>> (\*). Par exemple, si le signal A \_RACCROCHE peut être reçu dans tous les états et si les actions postérieures sont identiques, on peut recourir à la méthode indiquée à la figure D-3.8.24.

#### D.3.8.3.4 *Arrivée simultanée de signaux*

La Recommandation Z.100 (§ 2) prévoit que les signaux peuvent parvenir simultanément à un processus et précise qu'ils seront placés dans un ordre arbitraire.

Si un usager met au point un processus capable de recevoir des signaux simultanés, il doit veiller à ce que l'ordre d'arrivée ne bouleverse pas le fonctionnement escompté du processus.

Le LDS ne préconise pas de priorité parmi les signaux; ainsi, en cas d'arrivée simultanée de signaux, l'un d'eux est choisi arbitrairement. A noter cependant que les signaux pour communications entre services sont toujours traités les premiers (§ D.5.3).

Si plusieurs signaux sont disponibles au moment de l'entrée du processus dans un état, un seul signal est présenté au processus puis reconnu comme une entrée. Selon la sémantique du LDS, les autres signaux sont en fait retenus.



**Figure D-3.8.23, p. 13**

**Figure D-3.8.24, p. 14**

#### D.3.8.3.5 *Identification de l'émetteur*

Chaque signal est porteur de la valeur d'instance du processus (Pid) du processus émetteur. Lorsqu'un signal est absorbé, la valeur Pid du processus émetteur peut être obtenue au moyen de l'expression SENDER. On trouvera dans la figure D-3.8.25 un exemple d'emploi de cette variable.

**Figure D-3.8.25, p. 15**

#### D.3.8.4 *Mises en réserve*

Le concept de mise en réserve permet de différer l'absorption d'un signal jusqu'à ce qu'un ou plusieurs signaux ultérieurs aient été absorbés. Comme l'indique le § D.3.8.3, les signaux sont absorbés dans l'ordre dans lequel ils se présentent, sauf en cas d'emploi du concept de mise en réserve.

L'on peut faire appel au concept de mise en réserve afin de simplifier les processus lorsque l'ordre relatif d'arrivée de certains signaux n'a pas d'importance et que leur ordre effectif d'arrivée est indéterminé.

Dans chaque état, chaque signal est traité comme suit:

- il est représenté comme un symbole d'entrée ou,
- il est représenté comme un symbole de mise en réserve ou,
- il est, par convention, couvert par une entrée implicite aboutissant à une transition nulle implicite.

Le fonctionnement du mécanisme implicite de mise en file d'attente décrit dans le § D.3.8.3 s'applique également au concept de mise en réserve. A leur arrivée, les signaux sont placés dans la file d'attente et lorsque le processus atteint un état donné, les signaux qui se trouvent dans la file d'attente sont examinés un à un dans l'ordre de leur arrivée. Un signal couvert par un symbole d'entrée explicite ou implicite est absorbé et la transition qui s'y rapporte est exécutée. Un signal représenté dans un symbole de mise en réserve n'est pas absorbé et reste dans la file d'attente dans la même position séquentielle; le signal suivant de la file d'attente est considéré. Aucune transition ne suit un symbole de mise en réserve.

En LDS/PR, la construction de mise en réserve est exprimée à l'aide du mot-clé SAVE suivi d'une liste d'identificateurs de signaux. On trouvera un exemple simple d'énoncés de MISE EN RESERVE dans la figure D-3.8.26.

**Figure D-3.8.26 [T15.100] (a traiter comme tableau MEP), p. 16**

En LDS/GR, le concept de mise en réserve est représenté à l'aide du symbole de mise en réserve contenant les identificateurs de signaux.

Comme dans le cas des entrées, une <<notation avec astérisque>> peut servir à représenter la mise en réserve de tous les signaux (valides pour ce processus) qui ne sont pas explicitement mentionnés dans cet état.

La figure D-3.8.27 représente un exemple d'un processus LDS qui comporte un symbole de mise en réserve. Il convient de noter que les signaux S et R sont consommés dans l'ordre R, S, c'est-à-dire dans l'ordre inverse de leur réception. Un symbole de mise en réserve unique peut servir à mettre un signal en réserve tant que le processus se trouve dans l'état dans lequel le symbole apparaît; ce signal est mis en réserve pour la durée de la transition vers le prochain état. Dans le prochain état, le signal sera absorbé par l'intermédiaire d'une entrée explicite ou implicite (voir la figure D-3.8.27) sauf dans les cas suivants: lorsque le symbole de mise en réserve comportant le nom du signal est répété ou lorsque dans la file d'attente implicite, il existe avant lui, un autre signal de sauvegarde disponible pour absorption (voir la figure D-3.8.28).

**Figure D-3.8.27, p. 17**

**Figure D-3.8.28, p. 18**

Un signal mis en réserve n'est mis à disposition d'un processus que par l'intermédiaire d'un symbole d'entrée correspondant (explicite ou implicite). Aucune question relative à un signal mis en réserve ne peut être posée dans une décision avant la reconnaissance de ce signal comme une entrée; de même les données qui lui sont associées ne sont pas disponibles.

Dans un état où plusieurs signaux doivent être mis en réserve, on peut attribuer un symbole de mise en réserve à chaque signal ou on peut les représenter tous à l'intérieur du même symbole de mise en réserve. Si plusieurs signaux doivent être mis en réserve, la sémantique du symbole de mise en réserve exige que l'ordre de leur arrivée soit préservé.

Un troisième exemple de l'utilisation de la notion de mise en réserve est donné dans la figure D-3.8.29 et la figure D-3.8.30 décrit le fonctionnement du mécanisme de formation de la file d'attente.

L'utilisation du symbole de mise en réserve peut simplifier les diagrammes. Ainsi, en mettant un signal en réserve, l'on peut éviter de le recevoir et de devoir mettre en mémoire ses données associées jusqu'à l'état suivant.

Bien que le symbole de mise en réserve puisse être utilisé à chaque niveau de description, il y aurait peut-être lieu, au niveau inférieur, de décrire le mécanisme effectif qui permet cette mise en réserve.

Sans un minimum de précautions dans l'emploi de la mise en réserve, la file d'attente des signaux mis en réserve peut augmenter considérablement ou garder des signaux en mémoire trop longtemps, de sorte qu'un signal ancien peut être absorbé lorsqu'un signal récent est demandé.

Le LDS ne prévoit pas de limite à la longueur de la file d'attente, ce qui peut poser des problèmes de mise en oeuvre.

**Figure D-3.8.29, p. 19**

**Figure D-3.8.30, p. 20**

Les conditions de validation permettent une réception conditionnelle de signaux fondée sur la condition de validation spécifiée. Le signal est reçu et la transition interprétée si la condition est vraie. En cas de condition fausse, le signal est mis en réserve et le processus ne change pas d'état jusqu'à ce qu'un autre signal arrive ou que la condition devienne vraie. L'exemple de la figure D-3.8.31 illustre ceci. Lorsque P1 passe à l'état S1, la condition (c'est-à-dire de savoir si `IMPORT (X,P2)` est égal à 10) est évaluée. Si elle est vraie, le signal B peut être reçu. Sinon, le signal B est mis en réserve. Dans cet exemple, A arrive et provoque une transition vers S2. Pendant la transition, X passe à la valeur 11 et P2 exporte sa nouvelle valeur; alors, la condition liée au signal B dans l'état S2 est vraie. Étant donné que B est le premier signal de la file d'attente, la transition qui suit est exécutée et le processus prend fin à l'état S3.

Certaines caractéristiques des conditions de validation sont importantes:

- 1) la condition de validation est testée lorsque le processus arrive à un état; il est alors continuellement contrôlé tandis que le processus reste dans cet état. Ainsi, si la valeur exportée de X avait passé de 9 à 11 puis à 12 pendant la transition qui faisait suite à la réception de A, le processus serait resté à S2;
- 2) les conditions de validation peuvent être fondées sur des variables locales et/ou toute construction de langage qui peut être comprise dans une expression (par exemple, `IMPORT (IMPORTATION)`, `VIEW (VUE)`, `NOW (MAINTENANT)`);
- 3) alors qu'il est possible d'employer plus d'une condition par état, l'emploi de plus d'une condition de validation pour le même signal n'est pas autorisé. Ainsi, la condition indiquée dans la figure D-3.8.32 n'est pas autorisée. Si un signal donné exige des conditions multiples, il est possible de les combiner en une expression booléenne ainsi que le montre la figure D-3.8.33.

On peut évaluer les conditions de validation plusieurs fois et dans un ordre quelconque, de sorte que l'utilisateur doit être vigilant si les expressions ont des effets secondaires réciproques (par exemple `IMPORT` et `SENDER` combinés).

Il faut noter en outre que le signal spécifié dans la condition de validation ne peut influencer la valeur booléenne de la condition car ses valeurs transportées ne sont pas affectées avant l'absorption du signal. À titre d'exemple, les énoncés:

```
INPUT x(A) PROVIDED (A=5);... INPUT y PROVIDED(SENDER)=pid1;
```

priment à confusion car les valeurs de A et de `SENDER` dans les conditions correspondent à la situation telle qu'elle était avant l'absorption du signal.

Les signaux continus ont les mêmes propriétés fondamentales que la condition de validation, excepté le fait qu'ils ne sont accompagnés d'aucun signal. Ainsi, en l'absence de signaux dans la file d'attente susceptibles de provoquer une transition à leur entrée dans l'état, les signaux continus sont contrôlés; si l'un d'entre eux est vrai, la transition qui le suit est exécutée. L'exemple de la figure D-3.8.34 en donne l'illustration. À l'origine, le processus se trouve à l'état S1 et la valeur exportée de X est 9. En arrivant, le signal A provoque la transition vers l'état S2. Pendant la transition, X prend la valeur 11. Vu qu'aucun autre signal ne se trouve dans la file d'attente, la transition vers l'état S3 s'accomplit.

L'on trouvera ci-dessous certaines caractéristiques importantes des signaux continus:

- 1) de même que pour les conditions de validation, la valeur de la condition n'est contrôlée qu'à l'arrivée à un état;
- 2) les signaux continus multiples sont autorisés pour chaque état. Lorsque plusieurs signaux continus sont liés à un état, le signal continu ayant le rang de priorité le plus élevé (le numéro le plus bas) sera traité le premier. Deux signaux continus ne peuvent avoir le même rang de priorité. Le signal continu est toujours moins prioritaire que tout autre signal. Ceci est de nouveau dû au système sous-jacent du LDS: toutefois, la représentation à l'aide de modèles des signaux continus en LDS (emploi des signaux émis pendant l'exportation de variables), permet le recours à des priorités pour les signaux continus: ceci est d'ailleurs nécessaire afin d'éviter toute ambiguïté en cas de présence de plusieurs signaux continus. La figure D-3.8.35 en donne une illustration. Le processus commence à l'état S1 et ses variables locales X et Y ont respectivement les valeurs 10 et 11. Étant donné que les deux signaux continus sont vrais, celui qui a la plus haute priorité (rang de priorité le plus bas) est choisi et la transition vers l'état S2 s'accomplit. En S2, la condition de Y n'est plus vraie, et bien que



la priorité du signal continu de X soit inférieure à celle de Y, la transition qui le suit est exécutée et le processus parvient à l'état S3;

3) lorsque la transition d'un signal continu a une suite, l'expression SENDER (ÉMETTEUR) retourne la même valeur de SELF.

**Figure D-3.8.31, p. 21**

**Figure D-3.8.32, p. 22**

**Figure D-3.8.33, p. 23**

**Figure D-3.8.34, p. 24**

**Figure D-3.8.35, p. 25**

#### D.3.8.6 *Sorties*

Une sortie est l'émission d'un signal d'un processus vers un autre ou vers lui-même. Etant donné que le contrôle de la réception et de l'absorption du signal est associé au processus de réception (voir le § D.3.8.3), les règles sémantiques se rapportant directement aux symboles de sorties sont relativement simples. Du point de vue du processus d'émission, une sortie peut souvent être considérée comme une action instantanée qui, une fois achevée, n'a aucun autre effet direct sur le processus d'émission, lequel ne sera pas directement conscient du sort du signal.

Une action de sortie représente l'émission d'un signal et l'association de valeurs s'il en existe. Il est possible d'associer des valeurs à un signal de sortie en les plaçant entre parenthèses ou en mettant des expressions ayant des valeurs entre parenthèses (voir la figure D-3.8.37).

En LDS/PR, une action de sortie est représentée à l'aide du mot-clé OUTPUT (sortie) suivi d'une liste d'identificateurs de signaux. Une liste de paramètres réels mis entre parenthèses peut être associée à chaque identificateur de signal. S'il n'existe pas en fait de paramètre réel dans la sortie correspondant à une sorte dans la définition du signal, la variable correspondante dans l'entrée de réception aura la valeur <<indéfinie>>.

L'instance de processus de destination doit être exprimée dans l'instruction de sortie par le mot-clé TO (vers) suivi d'une expression d'instance de processus. Si l'instance de processus de destination peut être déterminée uniquement par le contexte, la clause TO peut être omise. Une condition d'adressage supplémentaire peut être fournie dans l'énoncé de sortie à l'aide du mot-clé VIA suivi d'une liste d'acheminement de signaux et d'identificateurs de canaux.

La figure D-3.8.36 donne quelques exemples valables d'énoncés de sortie.

**Figure D-3.8.36 [T16.100] (à traiter comme tableau MEP), p. 26**

En LDS/GR, une sortie est représentée à l'aide d'un symbole de sortie contenant la spécification de signaux, de paramètres réels et, en option, de destination et/ou de construction VIA.

Chaque sortie doit être dirigée vers une instance de processus donnée. Etant donné qu'il est généralement impossible de connaître l'instance de processus de tout processus au moment de l'élaboration d'une spécification, la méthode normale pour diriger les signaux consiste à employer une variable ou une expression dans le mot-clé TO (VERS). On trouvera dans les figures D-3.8.38, D-3.8.39 et D-3.8.40 des exemples. Dans la figure D-3.8.38, le paramètre de processus <<out\_to>> prend la valeur d'une instance de processus lors de la création du processus. Le rôle de <<out\_to>> au sein du processus est d'assurer la liaison entre le processus en question et le processus auquel il est connecté. Il convient de veiller lors de la conception du système, à ce que le type de processus indiqué par <<out\_to>> puisse recevoir les signaux qui sont émis. Dans la figure D-3.8.39, l'expression prédéfinie SENDER sert à renvoyer un signal au processus qui a émis le signal reçu peu avant. Dans la figure D-3.8.40, le signal est envoyé vers le descendant le plus récent du processus.



**Figure D-3.8.38 [T17.100] (à traiter comme tableau MEP), p. 28**

**Figure D-3.8.39, p. 29**

**Figure D-3.8.40, p. 30**

#### D.3.8.7 *T | che*

Dans une transition, une *t | che* sert à représenter à l'aide d'un texte informel des opérations sur des variables ou une opération spéciale.

En LDS/PR, une *t | che* est représentée par le mot-clé TASK (T | CHE) suivi d'une liste d'instructions ou de textes informels séparés par des virgules et se terminant par un point-virgule. Une instruction d'une *t | che* peut consister simplement en une affectation. Un texte informel consiste en une phrase délimitée par des apostrophes. On trouvera dans la figure D-3.8.41 des exemples de *t | ches* valables en LDS/PR.

**Figure D-3.8.41 [T18.100] (à traiter comme tableau MEP), p. 31**

En LDS/GR, une t | che se compose d'un symbole de t | che contenant la liste des instructions ou des textes informels.

Les usagers du LDS peuvent parfois éprouver de la difficulté à décider si un aspect du système défini doit | tre représenté par une t | che ou un processus distinct. Considérons l'exemple du processus représenté dans la figure D-3.8.42: l'action <<connecter-trajet-de-commutation>> doit-elle | tre représentée par une t | che ou par un processus distinct? Si l'on n'a pas identifié un processus distinct de commande de trajet de commutation, il serait indiqué d'utiliser le symbole t | che [voir la figure D-3.8.42 a)]. Si on a identifié un processus distinct de commande de trajet de commutation, on doit utiliser les signaux qui communiquent avec le processus de commande [voir la figure D-3.8.42 b)].

**Figure D-3.8.42, p. 32**

#### D.3.8.8 *Décisions*

Une décision est une action qui se produit au cours d'une transition et qui correspond à une question concernant la valeur d'une expression au moment de l'exécution de l'action; le processus a le choix entre deux ou plusieurs trajets, ce choix étant déterminé par la réponse consécutive à la décision. Les auteurs des diagrammes LDS doivent veiller à ce que les processus soient définis de manière qu'ils ne puissent tenter d'exécuter des décisions pour lesquelles des réponses (ou les données) ne sont pas disponibles; de telles décisions rendraient le diagramme tout à fait incorrect et entraîneraient une confusion considérable.

La question à laquelle correspond une décision peut | tre une expression ou un texte informel. Les réponses apportées par une décision sont représentées par une ou plusieurs valeurs possibles obtenues par l'évaluation de l'expression de la question ou par un ou plusieurs textes informels. Si la question ou l'une des réponses est informelle, toute la décision est informelle. Des réponses différentes sont séparées par des virgules. Les valeurs sont représentées par des expressions constantes, par des expressions constantes ayant un opérateur comme préfixe ou par des gammes dont les limites supérieures et inférieures sont des expressions constantes. Les valeurs de réponse doivent | tre du m | me type que l'expression contenue dans la question.



Il est possible d'indiquer explicitement certaines réponses et de grouper toutes les autres réponses possibles en employant le mot-clé ELSE (AUTRE).

En LDS/PR, la d'écision est représentée par le mot-clé DECISION suivi par la spécification de la question et la liste des réponses possibles, chacune étant associée à la transition correspondante. Les réponses sont indiquées entre parenthèses. L'ensemble des transitions sortantes est délimité par le mot-clé ENDDDECISION (FIN DE DECISION) placé à la fin (voir la figure D-3.8.43).

**Figure D-3.8.43 [T19.100] (à traiter comme tableau MEP), p. 33**

On trouvera certains exemples de d'écisions dans la figure D-3.8.44.

**Figure D-3.8.44 [T20.100] (à traiter comme tableau MEP), p. 34**

Toutes les transitions se terminent par le mot-clé ENDDDECISION (FIN DE DECISION). Les d'écisions qui ne se terminent pas par un énoncé terminal (c'est-à-dire jonction, état suivant, arrêt) continuent dans l'énoncé qui suit le mot ENDDDECISION, comme indiqué dans les deux branches équivalentes de la figure D-3.8.45.

**Figure D-3.8.45, p. 35**

L'instruction de décision peut en outre servir à former la structure IF-THEN (SI-ALORS), la structure DO-WHILE (FAIRE-PENDANT) et la structure LOOP-UNTIL (BOUCLE-JUSQU'A) comme en programmation structurée.

En LDS/GR, une décision est représentée à l'aide d'un symbole de décision contenant le texte de la question. Le symbole doit avoir deux branches ou plus associées aux réponses correspondantes. Chaque réponse doit être placée à la droite ou au sommet de la branche correspondante ou au-dessus de la branche qui interrompt la ligne de liaison. En LDS/GR, les parenthèses servant à délimiter les réponses sont facultatives mais il est suggéré de les utiliser pour éviter tout malentendu.

On trouvera certains exemples de décisions en LDS/GR dans la figure D-3.8.46.

**Figure D-3.8.46, p. 36**

Si une réponse ramène à la décision de la m | me transition, il convient d'exécuter certaines actions qui influencent la question ayant trait à la décision. Toutefois, m | me si cette règle est appliquée, des boucles infinies peuvent apparaître, comme indiquée dans la figure D-3.8.47. Il faut donc toujours faire attention lorsque des réponses se réfèrent à une décision de la m | me transition.

**Figure D-3.8.47, p. 37**

Des décisions peuvent être prises à l'aide de toute valeur existant dans le processus et notamment:

- des valeurs reçues par une entrée;
- des valeurs transmises en tant que paramètres effectifs au moment de la création du processus;
- des valeurs partagées.

L'expression qui figure dans la question peut comprendre des constantes de l'un quelconque des types de valeurs susmentionnés.

#### D.3.8.9 *Branchements et connecteurs*

Les branchements permettent de transférer la commande d'un point à un autre d'un corps de processus (ainsi qu'à l'intérieur d'un corps de procédure ou d'un corps de service).

En LDS/PR, les branchements sont équivalents à des énoncés <<GO TO>>. Des étiquettes sont utilisées comme points d'introduction associées aux instructions, comme indiquée dans la figure D-3.8.48. A l'intérieur d'un corps de processus (ou d'un corps de procédure), il est impossible de transférer la commande (et par conséquent les étiquettes associées) au type instructions indiquée dans la figure D-3.8.49. Les étiquettes demeurent toujours localisées dans un processus; il est donc impossible de transférer la commande d'un processus à un autre à l'aide d'un branchement.

**Figure D-3.8.48, p. 38**

**Figure D-3.8.49 [T21.100] (à traiter comme tableau MEP), p. 39**

En LDS/GR, les branchements correspondent aux connecteurs (de sortie et d'entrée). Elles peuvent être utilisées pour subdiviser les programmes, en cas de manque de place, ou pour éviter le croisement de lignes de liaison qui enlèverait de leur clarté aux diagrammes. En outre, il est généralement préférable, lorsque l'on trace un diagramme en LDS, que la liaison se dirige du haut au bas de la page.

En GR, toute ligne de liaison peut être interrompue par une paire de connecteurs associés; on admet alors que la liaison se dirige du connecteur de sortie au connecteur d'entrée. Chaque symbole de connecteur contient un nom, associé aux connecteurs portant le même nom. Il existe un seul connecteur d'entrée pour chaque nom mais il peut y avoir un ou plusieurs connecteurs de sortie.

En GR, il est souhaitable que la page de référence se rapportant au connecteur d'entrée appropriée soit spécifiée pour le connecteur de sortie et que la ou les références de pages relatives aux connecteurs de sortie appropriés devraient être données pour le connecteur d'entrée (voir l'exemple de la figure D-3.8.50).

**Figure D-3.8.50, p. 40**

### D.3.9 *Procédures*

En LDS, les procédures sont similaires aux procédures du CHILL et d'autres langages de programmation. Elles visent à

- a) permettre de structurer un processus en plusieurs niveaux de précision;
- b) conserver la compacité des spécifications en permettant de représenter comme un seul élément un assemblage complexe d'éléments qui peuvent être considérés isolément;
- c) permettre de définir et d'utiliser à plusieurs reprises les assemblages d'éléments utilisés.

Une définition de procédure ne peut exister que dans une définition de processus, dans une définition de service ou une définition de procédure et, par conséquent, une procédure n'est visible que pour le processus ou la procédure dans lesquels elle est définie.

Une définition de procédure se compose des éléments suivants (dont certains sont facultatifs):

- nom de la procédure;
- paramètres formels de procédure: liste de noms de variables associées à leur sorte. Ces paramètres servent à transférer l'information de la procédure ou à partir de celle-ci au moment de l'appel. Les paramètres de procédure peuvent être passés par valeur (paramètre IN) ou par référence (paramètre IN/OUT). Si un paramètre est passé par valeur, la spécification du paramètre formel définit une variable locale à la procédure; s'il est passé par référence, la spécification définit un synonyme pour la variable;
- définitions de procédure: procédures qui peuvent être appelées tout comme la procédure proprement dite;
- définitions de données: spécification de types de données locales à la procédure;
- définitions de variables: variables locales à la procédure;
- corps de procédure: spécification du comportement effectif de la procédure pour ce qui est des états et des actions (comme pour le corps de processus).

On trouvera dans la figure D-3.9.1 un exemple partiel de définition de procédure en LDS/PR (les mots-clés du langage sont écrits en majuscules). A noter que les paramètres formels sans attribut explicite ont un attribut implicite IN (var5 dans la figure).

**Figure D-3.9.1 [T22.100] (à traiter comme tableau MEP), p. 41**

#### D.3.9.1 *Corps de procédure*

Le corps de procédure présente de grandes similitudes avec le corps de processus; toutefois les différences sont les suivantes:

- la procédure termine son interprétation avec une indication <<retour>> au lieu d'une indication <<arrêt>>. En LDS/PR, l'énoncé de retour est représenté par le mot-clé RETURN;
- en LDS/GR, le symbole de début de procédure est légèrement différent du symbole de début de processus.

Les symboles de début et de retour de procédure sont indiqués dans le résumé du LDS/GR.

Une procédure peut utiliser la construction avec branchements mais seulement pour se référer à une étiquette interne. Un branchement peut ne pas être utilisé ou être utilisé pour accéder à une procédure de l'extérieur ou quitter celle-ci.

En LDS/GR, une définition de procédure est représentée par un diagramme de procédure très similaire au diagramme de processus. Le diagramme de procédure comprend les éléments suivants:

- un symbole de cadre facultatif: symbole de forme rectangulaire contenant tous les autres symboles;
- l'en-tête de procédure: le mot-clé PROCEDURE suivi du nom de la procédure et de la spécification des paramètres formels de procédure. Généralement, l'en-tête de procédure est placé dans l'angle supérieur gauche du cadre ou, s'il n'y a pas de cadre, dans l'angle supérieur gauche du support sur lequel le diagramme est tracé;
- une numérotation de pages facultatives (à placer dans l'angle supérieur droit);
- des symboles de texte: dans le cas d'un diagramme de procédure, un symbole de texte peut être utilisé pour contenir la spécification de définition de paramètres formels, de données et de variables;
- références de procédure: symboles de procédure, contenant chacun un nom de procédure représentant une procédure locale définie séparément;



- des diagrammes de procédure: permettant de définir directement les procédures locales;
- la zone graphique de procédure: spécification du comportement de la procédure pour ce qui est du début, des états, des entrées, des sorties, des tâches... et des arcs orientés.

Dans la figure D-3.9.2, on trouvera un exemple de définition de procédure en LDS/GR. La procédure portant la référence <<TERM<sub>u</sub>(emP)>> dans l'exemple est locale à la procédure d'appel.

Comme indiqué dans le cas des diagrammes de procédure (§ D.3.8), si une seule page ne suffit pas à contenir un diagramme de procédure, celui-ci peut être représenté sur plusieurs pages, avec répétition du symbole de cadre à la suite de l'entête et du numéro de page.



Les appels de procédure peuvent se produire chaque fois qu'une tâche est autorisée dans un graphe de processus ou de procédure. En un sens, une procédure peut être interprétée comme une tâche, avec les exceptions suivantes:

- 1) une procédure peut contenir des états et, si tel est le cas, recevoir des signaux;
- 2) une procédure peut émettre des signaux. L'instance de processus d'origine est celle qui a appelé la procédure.

Lorsqu'une procédure est appelée, son environnement est créé et l'interprétation de la procédure commence. Elle se poursuit jusqu'à ce que l'on ait atteint l'indication RETURN (retour). Pendant l'interprétation de la procédure, tous les signaux adressés au processus sont soit mis en réserve implicitement soit traités explicitement par la procédure. La procédure n'a pas sa propre file d'attente mais utilise celle du processus qui l'a appelée.

En LDS/PR, un appel de procédure est représenté par le mot-clé CALL suivi de l'identificateur de procédure et de la liste de paramètres réels mise entre parenthèses. Si un paramètre n'est pas donné, il convient de l'indiquer par deux virgules consécutives. Dans ce cas, le paramètre formel correspondant a la valeur <<indéfinie>>. A noter aussi que la déclaration de IN ou IN/OUT est faite dans la définition de procédure, de sorte qu'elle ne doit pas être répétée par l'énoncé d'appel. On trouvera certains exemples d'appel en LDS/PR dans la figure D-3.9.3.

**Figure D-3.9.3 [T23.100] (à traiter comme tableau MEP), p. 43**

En LDS/GR, un appel de procédure est représenté à l'aide d'un symbole d'appel de procédure contenant le nom de procédure et la liste des paramètres réels mise entre parenthèses. On trouvera un exemple d'appel en LDS/GR dans la figure D-3.9.2.

