5i’

FASCICULE X.2

Annexe D à la Recommandation Z.100

DIRECTIVES POUR LES USAGERS DU LDS

Blanc

MONTAGE: PAGE 2 = PAGE BLANCHE

ANNEXE D A LA RECOMMANDATION Z.100

DIRECTIVES POUR LES USAGERS DU LDS

TABLE DES MATIÈRES

D.1 *Préface*

Le langage de spécification et de description du CCITT (LDS) a tout d'abord fait l'objet des Recommandations Z.101 à Z.103 (Tome VI.4 du Livre orange, 1976) puis, sous une forme développée, des Recommandations Z.101 à Z.104 (Livre jaune, 1980) qui ont été complétées et regroupées en 1984 dans les Recommandations Z.100 à Z.104 (Livre rouge). Au cours de la période d'études 1985-1988, le langage a été encore développé et harmonisé; les Recommandations existantes ont été fondues en une seule et une définition mathématique a été ajoutée.

Des directives sont indispensables aux utilisateurs pour faciliter l'utilisation du LDS dans ses applications à une large gamme de systèmes de télécommunication. Ces directives ont pour but d'aider les utilisateurs à comprendre la Recommandation concernant le LDS et son application à différents secteurs.

L'emploi du LDS est largement répandu au sein du CCITT et des organisations qui en sont membres; en outre, la gamme des applications de ce langage ne cesse de se développer.Les présentes directives sont établies à l'intention de ceux qui envisagent d'utiliser ou utilisent déjà le LDS; elles complètent la Recommandation sur le LDS en y ajoutant des conseils judicieux et des exemples utiles. Il y aura certes quelques chevauchements entre les directives et la Recommandation; cela semble d'ailleurs souhaitable, si l'on veut que les directives soient autonomes et faciles à consulter. C'est néanmoins la Recommandation qui constitue le document de base.

D.2 *Introduction*

D.2.1 *Aperçu général du LDS*

Le LDS peut servir à spécifier le fonctionnement que l'on attend d'un système et à décrire le fonctionnement effectif d'un système. Il a été conçu pour spécifier et décrire le comportement des systèmes de commutation qui interviennent dans les télécommunications, mais peut également être utilisé dans une gamme d'applications plus large. De fait, le LDS convient particulièrement bien à tous les systèmes où il est possible de représenter correctement un comportement à l'aide de machines à états finis étendues (§ D.2.1.1) et où l'on s'intéresse spécialement aux phénomènes d'interaction.

Le LDS peut également servir de point de départ à des méthodes de documentation permettant de représenter intégralement la spécification ou la description d'un système. Dans ce contexte, la signification de la spécification et de la description est liée à leur emploi dans le cycle de vie d'un système. Chacune décrit les propriétés fonctionnelles d'un système d'une façon abstraite. La description comprend généralement certains aspects liés à la conception (par exemple, traitement des erreurs); elle est généralement plus complète en ce qui concerne les détails fonctionnels. Chacune doit concorder avec le modèle concret du système. Elles servent donc toutes deux de spécifications avant la mise en oeuvre du système, et de documentation (descriptions) après cette même mise en oeuvre.

Le LDS peut servir à représenter à divers niveaux de détail les propriétés fonctionnelles d'un système, d'une fonction ou d'une facilité, qu'il s'agisse de leurs spécifications ou de leurs descriptions. Les propriétés fonctionnelles désignent certaines propriétés structurelles (diagramme d'interaction de blocs) ainsi que le comportement. Par «comportement», on entend la manière dont un système réagit à des signaux reçus (entrées), c'est-à-dire les actions qu'il exécute, par exemple, envoi de signaux (sorties), formulation de questions (aux fins de décision) et exécution de tâches.

Les spécifications peuvent être très générales quand une Administration souhaite étudier les possibilités de mise à jour d'un système en introduisant de nouvelles caractéristiques, de nouveaux services, de nouvelles techniques, etc., tout en laissant au fournisseur la possibilité d'offrir de très nombreuses solutions pratiques. Des spécifications de ce genre ne donneront généralement que peu de détails. A l'autre extrémité, il y a les spécifications par lesquelles une Administration demande le remplacement ou l'extension d'un central existant. Dans ce cas, les détails devront probablement être plus poussés, les spécifications des interfaces devant être très détaillées.

Une spécification et une description peuvent être identiques. Il est toujours préférable de concevoir les nouvelles réalisations à partir de la spécification, afin d'en garantir le respect.

D'une manière générale, ce sont les fournisseurs qui rédigent les descriptions pour donner suite à une spécification (ou pour décrire des systèmes que le fournisseur veut mettre sur le marché). Une description sera généralement plus détaillée que la spécification puisqu'il s'agit de rendre compte du comportement détaillé du système.

Il est à noter également que le LDS permet de décrire un système de manière plus ou moins formelle.

Premièrement, il est possible de décrire un système au moyen de constructions LDS associées au langage naturel. La description ainsi obtenue permet le transfert de l'information à un lecteur qui connaît le contexte, mais pas à une machine. Les contrôles pouvant être effectués automatiquement sont très limités.

Deuxièmement, il est possible d'associer aux constructions LDS des énoncés formels constitués d'éléments de types définis et d'opérateurs sur ces éléments. Les propriétés de ces éléments ne sont pas spécifiées; exemple: <<Connecter A-B>>, où A et B sont du type abonné et <<Connecter>> est une opération autorisée pour ce type. La spécification ainsi obtenue permet le transfert de l'information aux lecteurs qui connaissent la signification des opérateurs utilisés. Une machine peut comprendre la description jusqu'à un certain niveau et peut procéder à certains contrôles; elle ne peut pas effectuer des contrôles complets ni <<mettre en oeuvre>> le système car les propriétés des opérateurs sont inconnues.

Troisièmement, il est possible également d'indiquer toutes les propriétés de tous les opérateurs. Dans ce cas, la description est entièrement formelle; une machine peut effectuer tous les contrôles et, en principe, mettre en oeuvre les systèmes décrits.

Selon l'objectif visé, les descriptions peuvent être adaptées aux besoins des usagers au moyen de différents niveaux de formalisme. Naturellement, plus la description est formelle, plus un être humain aura de la difficulté à la lire.

Dans le texte qui suit, le terme spécification sera utilisé à la fois pour la représentation nécessaire et pour celle des comportements réels.

D.2.2.1 *Le LDS fondé sur un modèle de machine à états finis étendue*

En cas d'emploi du LDS, le système à spécifier est représenté par un certain nombre de machines abstraites interconnectées. Une spécification complète comporte obligatoirement:

1) la définition de la structure du système en ce qui concerne les machines et leurs interconnexions,

2) le comportement dynamique de chaque machine, ses interactions avec les autres machines et avec l'environnement, et

3) les opérations sur les données associées aux interactions.

On décrit le comportement dynamique au moyen de modèles qui définissent les mécanismes de fonctionnement des machines abstraites ainsi que la communication entre les machines. La machine abstraite qu'emploie le LDS est une extension de la machine déterministe à états finis (FSM). La FSM est dotée d'une mémoire d'états finis internes et fonctionne avec un ensemble discret et fini d'entrées et de sorties. Pour chaque combinaison d'une entrée et d'un état, la mémoire définit une sortie ainsi que l'état suivant. On considère habituellement que la durée de transition entre deux états est nulle.

L'une des limites de la FSM est la suivante: toutes les données à mémoriser doivent être représentées sous la forme d'états explicites. Il est possible de représenter la plupart des systèmes de cette façon, mais ce n'est pas toujours pratique. On peut être appelé à mémoriser un grand nombre de valeurs importantes pour le comportement futur mais qui ne contribuent pas beaucoup à la compréhension globale du système. Cette information ne doit pas faire partie de l'espace des états explicites; en effet, ceci compliquerait la présentation. Il est possible pour ce genre d'applications d'étendre la FSM en la dotant d'une mémoire auxiliaire et d'une capacité de fonctionnement auxiliaire sur cette mémoire. Cette mémoire auxiliaire peut emmagasiner, par exemple, des informations concernant des adresses et des numéros d'ordre.

Les Recommandations relatives au LDS définissent deux opérations auxiliaires qu'il est possible d'inclure dans les transitions de la machine à états finis étendue (EFSM), à savoir les décisions et les tâches. Les <<décisions>> vérifient des param`etres associ´es aux entr´ees et aux donn´ees contenues dans la m´emoire auxiliaire lorsque ces donn´ees sont importantes pour le s´equencement de la machine principale. Les <<t | ches>> ex´ecutent des fonctions telles que le comptage, des op´erations sur la m´emoire auxiliaire et la manipulation de param`etres d'entr´ee et de sortie.

En LDS, des signaux repr´esentent les interactions entre machines, c'est-`a-dire que les EFSM re,coivent des signaux comme entr´ees et produisent des signaux comme sorties. Les signaux se composent d'un seul identificateur de signal et facultativement d'un ensemble de param`etres. Le LDS pr´evoit la possibilit´e d'un temps de transition diff´erent de z´ero, et d´efinit un m´ecanisme th´eorique de mise en file d'attente <<premier entr´e, premier sorti>> pour les signaux qui parviennent à une machine en train d'ex´ecuter une transition. Les signaux sont trait´es à tour de r | le, dans leur ordre d'arriv´ee.

D.2.2 *Formes syntaxiques du LDS*

Le LDS est un langage qui se pr´esente sous deux formes diff´erentes, fond´ees toutes deux sur le m | me mod`ele s´emantique. L'une est appel´ee LDS/GR (LDS graphical representation) et repose sur un ensemble de symboles graphiques normalis´es. L'autre s'appelle LDS/PR (LDS textual phrase representation) et repose sur des instructions analogues à un langage de programmation. L'une et l'autre repr´esentent les m | mes concepts du LDS.

Un langage graphique pr´esente l'avantage de montrer clairement la structure d'un syst`eme et de permettre à des | tres humains de visualiser facilement le flux de contr | le. La repr´esentation textuelle de phrases convient mieux à l'utilisation par des machines.

En tant qu'outil de conception, le LDS devrait | tre pr´esent´e sous une forme permettant à l'utilisateur d'exprimer ses id´ees clairement et avec concision. Le LDS/GR, qui permet de le faire, correspond davantage à la repr´esentation traditionnelle des machines à ´etats finis ´etendues.

Le LDS/GR est la forme originale du LDS. Il a ´et´e con,cu entre 1973 et 1976 a ´et´e publi´e pour la premi`ere fois dans la version de 1976 des Recommandations de la s´erie Z.100.

Le LDS/GR a ´et´e ´etabli sur la base de langages graphiques ´elabor´es par diff´erentes organisations pour leurs propres utilisations.

La repr´esentation textuelle de phrases du LDS, c'est-`a-dire le LDS/PR, a ´et´e con,cu pendant la p´eriode d'´etudes 1977-1980 mais il a fallu y apporter certaines am´eliorations avant qu'elle puisse faire l'objet d'une Recommandation. Ces am´eliorations ont ´et´e faites au cours de la p´eriode d'´etudes suivante et, d`es 1984, le LDS/PR est devenu l'une des syntaxes concr`etes recommand´ees du LDS.

Dans un premier temps, le LDS/PR devait | tre utilis´e comme un moyen ais´e d'introduire des documents en LDS dans une machine, ce qui ´etait trop difficile avec le GR (en effet, cela n´ecessitait l'intervention d'´equipements p´eriph´eriques graphiques). C'est pour cette raison que l'on a insist´e sur une mise en mati`ere de terminaux graphiques (capacit´es accrues et r´eduction des co | ts) ont fait que le GR est d´esormais susceptible d' | tre introduit en machine. Cela ne diminue en rien l'importance et l'utilisation du PR car certains utilisateurs le trouvent plus à leur convenance, particuli`erement ceux qui travaillent avec des langages de programmation.

Du fait de cette ´evolution, la corr´elation entre le GR et le PR est moins ´etroite; cependant, il est encore possible de mettre en correspondance sans difficult´e l'une de ces repr´esentations avec l'autre, bien que chacune d'elles ait ses propres particularit´es. A premi`ere vue, le PR ressemble fortement à un langage de programmation (voir la figure D-2.2.1).

Figure D-2.2.1 [T1.100] (a traiter comme tableau MEP, p.

En fait, tout d´epend de ce qui caract´erise un texte du point de vue du langage de programmation.

Si nous admettons qu'un programme est d´efini comme une <<information interpr´etable par une machine>>, non seulement les PR mais aussi les GR sont des <<programmes>>.

Il existe cependant certaines diff´erences entre une sp´ecification en LDS et un programme r´eel. Tout d'abord, il n'est pas indispensable qu'une sp´ecification en LDS puisse | tre ex´ecut´ee par une machine (bien que cela ne soit pas interdit); ce qui est essentiel, c’est sa capacit´e d’acheminer des renseignements pr´ecis d’un | tre humain à un autre | tre humain.

Si nous consid´erons une sp´ecification en LDS comme un programme, ce qui peut | tre tenu pour une <<sp´ecification en LDS erron´ee>> (en raison d’un texte informel incomplet) pourrait | tre parfaitement valable si elle ´etait consid´er´ee comme une repr´esentation des caract´eristiques fonctionnelles d’un syst`eme.

Une autre diff´erence r´eside dans le <<style>> d’une sp´ecification en LDS, si on la compare à la repr´esentation usuelle d’un programme.

Le LDS ayant pour but de faciliter la communication entre | tres humains, on s’est efforc´e de pr´eserver la possibilit´e de diff´erentes pr´esentations, afin que la pr´esentation en LDS puisse servir à orienter le lecteur vers certains aspects consid´er´es comme plus importants que d’autres. Cela est naturellement sans importance pour un programme, qui est cens´e
| tre interpr´et´e par une machine. La machine n’insiste pas sur un quelconque aspect particulier du programme, mais doit traiter de la m | me mani`ere son ensemble; en outre, elle n’essaie pas de <<comprendre>> le programme.

Etant donn´e ses similitudes avec un programme, le PR a la pr´ef´erence de certains programmeurs qui utiliseront probablement aussi le CHILL pour la mise en oeuvre des besoins. On serait donc probablement tent´e de rechercher une correspondance point par point entre le PR et le CHILL, afin que les besoins exprim´es en PR puissent | tre automatiquement transform´es en code CHILL. L’inverse est ´egalement int´eressant car cela permettrait la d´erivation d’une sp´ecification en PR à partir d’un programme en CHILL.

On trouvera au § D.9 des exemples de possibilit´es de mettre en correspondance le LDS avec le CHILL.

D.2.3 *Applicabilit´e du LDS*

La figure D-2.3.1 donne un ´eventail de possibilit´es d’emploi du LDS aux fins d’achat ou de founiture de syst`emes de commutation pour les t´el´ecommunications.

Dans cette figure, les rectangles repr´esentent des groupes fonctionnels typiques, dont les noms pr´ecis, qui peuvent varier d’une organisation à l’autre, ayant des activit´es repr´esentatives de plusieurs Administrations ou de plusieurs fabricants. Chacune des fl`eches (lignes de liaison) repr´esente la circulation d’une s´erie de documents entre un groupe fonctionnel et un autre; le LDS peut alors | tre employ´e en tant que partie de chacune de ces s´eries de documents. La figure, donn´ee seulement à titre d’illustration, n’est ni d´efinitive ni compl`ete.

Les domaines d’application sont ceux effectivement mod´elis´es par des machines à ´etats finis ´etendues, communiquant entre elles, par exemple: commutation t´el´ephonique t´elex ou de donn´ees, syst`emes de signalisation (par exemple, syst`eme de signalisation n^{o} 7), interfonctionnement des syst`emes de signalisation, protocoles pour donn´ees et interfaces d’usagers (LHM).

Si l’on consid`ere plus sp´ecialement les syst`emes de commutation SPC, on peut citer comme exemple de fonctions pouvant | tre sp´ecifi´ees ou d´ecrites à l’aide du LDS: le traitement des appels (acheminement, signalisation, comptage, etc.), la maintenance, le traitement des d´erangements (par exemple, alarmes, rel`eve automatique des d´erangements, configuration des syst`emes, essais p´eriodiques, etc.), la commande des syst`emes (par exemple, protection contre les surcharges), et les interfaces homme-machine. Le § D.10 donne des exemples d’application du LDS.

La sp´ecification des protocoles où intervient le LDS fait l’objet des Recommandations de la s´erie X du CCITT.

Figure D-2.3.1, p.

D.3 *Concepts de base du LDS*

D.3.1 *Syst`eme*

Comme on a pu le voir plus haut, le LDS repr´esente des syst`emes à l'aide de mod`eles. Ainsi ce qui est d´efini par une sp´ecification ou une description en LDS constitue le syst`eme. Un syst`eme LDS peut donc repr´esenter à l'aide de mod`eles une partie d'un syst`eme (ou d'un central) t´el´ephonique, ou un r´eseau complet de syst`emes t´el´ephoniques ou des parties d'un grand nombre de centraux t´el´ephoniques (par exemple, les dispositifs de contr | le de circuit aux deux extr´emit´es d'un circuit). Il importe de souligner que le syst`eme LDS contient, d'un point de vue LDS, tout ce que la sp´ecification ou la description tente de d´efinir. L'environnement ne rel`eve pas de la sp´ecificationet n'est pas d´efini en LDS.

Des canaux relient le syst`eme à l'environnement. Th´eoriquement, il suffit d'un seul canal bidirectionnel pour connecter le syst`eme à l'environnement. Dans la pratique, on d´efinit g´en´eralement des canaux pour chaque jonction logique avec l'environnement.

Chaque syst`eme se compose d'un certain nombre de blocs reli´es par des canaux. Chaque bloc du syst`eme est ind´ependant de tous les autres. Chaque bloc peut contenir un ou plusieurs processus d´ecrivant le comportement du bloc. Seule l'´emission de signaux dans les canaux permet la communication entre des processus plac´es dans deux blocs diff´erents. Pour subdiviser le syst`eme en blocs, on peut prendre des crit`eres tels que les suivants: d´efinir des parties d'une dimension facilitant la compr´ehension, cr´eer une correspondance avec la division effective entre le logiciel et le mat´eriel, suivre les subdivisions fonctionnelles naturelles, r´eduire au minimum les interactions, etc.

Pour de grands syst`emes LDS, il existe certaines constructions en LDS qui permettent de sp´ecifier les sous-structures des parties d'un syst`eme, de sorte qu'en partant d'un aper,cu g´en´eral du syst`eme, on peut donner toujours plus de d´etails. Dans ce cas, on peut dire que le syst`eme est repr´esent´e à diff´erents niveaux de d´etail. Ces constructions sont d´ecrites au § D.4.

Au premier niveau de pr´ecision, la sp´ecification en LDS d'un syst`eme d´ecrit la structure du syst`eme et comprend les points suivants, expliqu´es dans les paragraphes qui suivent:

— nom du syst`eme;

— d´efinitions de signaux: sp´ecification des types de signaux ´echang´es entre les blocs du syst`eme ou entre les blocs et l'environnement. Cela comprend la sp´ecification des types de valeurs achemin´es par les signaux (liste de sortes);

— d´efinitions de listes de signaux: sp´ecificationdes identificateurs groupant plusieurs signaux et/ou autres listes de signaux. De tels identificateurs peuvent servir à ´economiser de l'espace et à donner une sp´ecification plus claire;

— d´efinitions de canaux: sp´ecification des canaux reliant les blocs du syst`eme les uns aux autres et à l'environnement. Une d´efinition de canal comprend la sp´ecification des identificateurs des signaux achemin´es par ce canal;

— d´efinitions de donn´ees: sp´ecification de nouveaux types de syntypes et de g´en´erateurs d´efinis par l'utilisateur et visibles dans tous les blocs;

— d´efinitions de blocs: sp´ecification des blocs dans lesquels le syst`eme est subdivis´e;

— d´efinitions de macros: des directives concernant l'utilisation des macros sont donn´ees au § D.5.1.

Selon la Recommandation concernant le LDS, il existe des types de donn´ees pr´ed´efinis qui peuvent | tre utilis´es par chaque syst`eme. Ils ne n´ecessitent pas de d´efinition et peuvent | tre utilis´es au moyen de leurs noms pr´ed´efinis, c'est-`a-dire:INTEGER, REAL, CHARACTER, STRING, CHARSTRING, BOOLEAN, PID, TIME, DURATION. Les types de donn´ees pr´ed´efinis sont visibles à tous les niveaux de la d´efinition du syst`eme; ils peuvent | tre consid´er´es comme d´efinis implicitement dans une <<biblioth`eque>> de syst`eme accessible en tout point de la sp´ecification.

Les noms de sorte utilis´es dans les d´efinitions de signaux au niveau de syst`eme doivent | tre introduits par des d´efinitions de type partielles visibles au niveau du syst`eme, c'est-`a-dire des types de donn´ees pr´ed´efinis ou des newtypes d´efinis par l'utilisateur ou encore des syntypes d´efinis à ce niveau.

On trouvera des explications compl´ementaires sur l’utilisation des types de donn´ees aux § D.3.10 et D.6.

Le LDS/PR utilis´e pour une d´efinition de syst`eme consiste en un ensemble d’instructions se terminant par <<;>> (point-virgule). La d´efinition d’une structure de syst`eme commence par l’instruction <<SYSTEM nom;>> et se termine par l’instruction <<ENDSYSTEM nom;>>. Le nom de l’instruction terminale est facultatif mais s’il est donn´e, il doit | tre le m | me que le nom donn´e apr`es le mot cl´e SYSTEM. Il est sugg´er´e de placer toujours le nom dans l’instruction terminale, car cela am´eliore la lisibilit´e du document.

Le sch´ema du LDS/PR d’une d´efinition de structure de syst`eme est repr´esent´e dans la figure D-3.1.1.

Figure D-3.1.1 [T2.100] (a traiter comme tableau MEP, p.

Pour permettre d’obtenir une repr´esentation plus claire et plus simple de la structure du syst`eme ou permettre une sp´ecificationdescendante de syst`eme, le LDS contient un m´ecanisme g´en´eral de r´ef´erence. A ce niveau, le m´ecanisme de r´ef´erence peut
| tre appliqu´e aux d´efinitions de bloc. Cette caract´eristiquedu langage permet à l’utilisateur de sp´ecifier pr´ecis´ement le nom de bloc à l’int´erieur de la d´efinition de la structure de syst`eme; la d´efinition de bloc proprement dite peut | tre donn´ee s´epar´ement (voir la figure D-3.1.2).

Figure D-3.1.2 [T3.100] (a traiter comme tableau MEP, p.

Le m´ecanisme de r´ef´erence est particuli`erement utilis´e dans le LDS/GR parce que la plupart des diagrammes doivent tenir sur une seule page et que la place manque souvent pour des sp´ecifications graphiques embo | t´ees.

On trouvera des exemples de LDS/GR pour une d´efinition de syst`eme au § D.3.6.

D.3.2 *Blocs*

Les processus peuvent communiquer entre eux à l’int´erieur d’un bloc, au moyen de signaux, de valeurs partag´ees. Ainsi le bloc ne constitue pas seulement un m´ecanisme pratique pour le regroupement de processus, mais encore une limite à la visibilit´e des donn´ees. C’est pourquoi il convient, lors de la d´efinition de blocs, de s’assurer du caract`ere raisonnable et fonctionnel du regroupement de processus au sein d’un bloc. Dans la plupart des cas, il est utile de fractionner dans un premier temps le syst`eme (ou le bloc) en unit´es fonctionnelles, puis de d´efinir les processus à int´egrer dans le bloc.

A l’int´erieur d’un bloc, il est possible (a titre d’option) de d´efinir les trajets de communication entre les processus ou enter ceux-ci et l’environnement du bloc (c’est-`a-dire la fronti`ere du bloc). Ces trajets de communications sont appel´es acheminement de signaux.

Pour un syst`eme LDS de grandes dimensions, il est possible de d´ecrire la sous-structure d’un bloc en fonction d’autres blocs et de canaux, comme si le bloc ´etait un syst`eme en soi. Ce m´ecanisme est expliqu´e au § D.4.

La d´efinition de la structure d’un bloc peut comporter les points suivants:

— nom de bloc;

— d´efinitions de signaux: sp´ecification des types de signaux ´echang´es à l’int´erieur du bloc. Cela comprend la sp´ecification des types de valeurs achemin´es par les signaux (liste de sortes);

— d´efinitions de listes de signaux: sp´ecifications ou identificateurs correspondant à des listes de signaux et/ou à d’autres identificateurs de listes de signaux. Ces identificateurs, groupant plusieurs signaux, peuvent | tre utilis´es pour ´economiser de l’espace et obtenir une sp´ecification plus claire;

— d´efinitions d’acheminement de signaux: sp´ecifications des trajets de communication reliant les processus du bloc les uns aux autres et à l’environnement du bloc. Une d´efinition d’acheminement de signaux comprend la sp´ecification des identificateurs de signaux transport´es sur cet acheminement de signaux;

— connexions de canaux vers des acheminements: sp´ecifications des connexions entre les canaux ext´erieurs au bloc et les acheminements de signaux internes du bloc;

— d´efinitions de processus: sp´ecification des types de processus d´ecrivant le comportement du bloc. Si le bloc n'est pas d´ecrit en fonction de sa sous-structure, il faut qu'il y ait au moins une d´efinition de type de processus à l'int´erieur du bloc. Pour la d´efinition du processus, un m´ecanisme de r´ef´erence est fourni comme cela est indiqu´e, dans le cas des blocs, au § D.3.1;

— d´efinitions des donn´ees: sp´ecification de newtypes, de syntypes et de g´en´erateurrs d´efinis par l'usager et visibles dans tous les processus d´efinis du bloc et/ou dans la structure-structure du bloc;

— d´efinitions de macros: des directives pour l'utilisation de macros sont donn´ees au § D.5.1.

S'il existe une sous-structure à l'int´erieur du bloc, certains des points ci-dessus sont facultatifs (voir le § D.4 pour les explications concernant la structure).

Les types suivants sont visibles dans un bloc:

— des types de donn´ees pr´ed´efinis;

— des types de donn´ees d´efinis par l'usager, d´efinis dans le bloc proprement dit;

— des types de donn´ees d´efinis par l'usager visibles dans le bloc ascendant (en cas de subdivision des blocs).

Dans le LDS/PR, les mots cl´e BLOCK et ENDBLOCK servent à circonscrire une d´efinition de bloc. On trouvera des exemples du LDS/GR pour une d´efinition de bloc au § D.3.6.

D.3.3 *Canaux*

Les canaux sont les moyens de communication entre diff´erents blocs du syst`eme ou entre des blocs et leur environnement. Un canal peut relier un bloc à un autre ou un bloc à l'environnement dans une direction (canal unidirectionnel) ou dans les deux directions (canal bidirectionnel). Normalement, un canal est une entit´e fonctionnelle qui peut | tre utilis´ee pour d´esigner des chemins de communication sp´ecifiques. En fait, par la subdivision des canaux (d´ecrite au § D.4.5), il est possible de sp´ecifier formellement le comportement de chaque canal.

Pour chaque direction indiqu´ee ou chaque chemin de communication, la sp´ecificationdu canal contient une liste de tous les identificateurs de signaux que le canal peut acheminer dans cette direction. Cette liste de signaux offre le moyen de garantir que chaque signal envoy´e par un processus à une extr´emit´e du canal puisse | tre re,cu par le processus du bloc qui se trouve à l'autre extr´emit´e du canal. Ainsi, la sp´ecification de canal devient partie int´egrante de la sp´ecificationd'interface pour chaque bloc. Dans de grands projets int´eressant de nombreuses personnes, un accord d`es l'origine sur les signaux d'un canal et sur la sp´ecification de ces signaux r´eduit sensiblement la probabilit´e que deux processus ne pourront communiquer l'un avec l'autre comme pr´evu.

La d´efinition d'un canal comporte les ´el´ements suivants:

— nom du canal;

— un ou deux chemins de communication: un chemin de communication sp´ecifie l'origine et la destination d'une liste de signaux. Des identificateurs de bloc ou le mot cl´e <<ENV>> (environnement) peuvent | tre utilis´es dans ce contexte;

— une ou deux listes de signaux: une liste des signaux transport´es dans cette direction doit | tre sp´ecifi´ee pour chaque chemin de communication. Cette liste peut comporter des identificateurs de signaux ainsi que des identificateurs d'autres listes;

— une d´efinition facultative de sous-structure de canal (ou une r´ef´erence à une telle d´efinition): voir le § D.4.5.

Dans le LDS/PR, une d´efinition de canal est comprise entre les deux mots cl´esCHANNEL et ENDCHANNEL. Les mots cl´es FROM et TO servent à d´esigner les chemins de communication et le mot cl´e WITH les listes de signaux. On trouvera dans la figure D-3.3.1 un exemple de d´efinition de canal en LDS/PR.

Dans le LDS/GR, une d´efinition de canal est repr´esent´ee à l'aide d'une ligne reliant les deux parties mises en jeu dans la communication. Le nom du canal doit | tre plus proche de la ligne que tout autre symbole. Les trajets sont repr´esent´es au moyen de fl`eches et les listes de signaux doivent | tre plac´es entre crochets, comme indiqu´e dans les exemples de la figure D-3.3.2. Pour ´eviter une confusion entre les canaux et les acheminements de signaux, les fl`eches ne doivent pas | tre plac´ees en l'un des deux points terminaux de la ligne (voir le § D.3.5).

Dans un canal bidirectionnel, chacune des deux listes de signaux doit | tre aussi proche que possible de la fl`eche correspondante.

Figure D-3.3.1 [T4.100] (a traiter comme tableau MEP, p.

Figure D-3.3.2, p.

D.3.4 *Signaux*

Les signaux peuvent | tre d´efinis au niveau du syst`eme, au niveau du bloc ou dans la partie interne de la d´efinition de processus. Les signaux d´efinis à un certain niveau peuvent | tre utilis´es à ce niveau ou ´egalement aux niveaux inf´erieurs; cependant, pour simplifier chaque niveau, il est sugg´er´e de d´efinir les signaux de mani`ere aussi circonscrite que possible. Les signaux d´efinis dans le cadre d'une d´efinition de processus peuvent | tre ´echang´es entre des instances de m | me type de processus (§ D.3.8) ou entre services d'un processus (§ D.5.3).

La d´efinition d'un signal comprend les points suivants:

— nom du signal;

— liste de sortes (facultative): repr´esente la liste des types de valeurs achemin´ees par ce signal;

— affinage du signal (facultatif): voir le § D.4.7.

En LDS/PR, une d´efinition de signal est sp´ecifi´ee par le mot cl´e SIGNAL. Plusieurs signaux (n'ayant pas fait l'objet d'un affinage) peuvent
| tre d´efinis à l'int´erieur de la construction, donnant ainsi le nom du signal et la liste de types. Des exemples de d´efinitions de signaux en LDS/PR sont donn´es dans la figure D-3.4.1.

Figure D-3.4.1 [T5.100] (a traiter comme tableau MEP, p.

En LDS/GR, on sp´ecifie une d´efinition de signal en ins´erant des ´enonc´es lin´eaires dans un symbole de texte comme indiqu´e dans la figure D-3.4.2.

Figure D-3.4.2, p.

D.3.5 *Acheminements de signaux*

Des acheminements de signaux servent à repr´esenter des chemins de communication similaires aux canaux. Ils peuvent | tre utilis´es au niveau des blocs et à celui des processus. Comme les canaux, les acheminements de signaux peuvent | tre unidirectionnels ou bidirectionnels mais ils ne peuvent | tre subdivis´es.

Au niveau des blocs, ils repr´esentent un moyen de communication entre les processus d'un bloc ou entre les processus et l'environnement de ce bloc, c'est-`a-dire un canal menant à ce bloc ou venant de celui-ci.

Au niveau du processus, les acheminements de signaux peuvent | tre utilis´es lorsque le processus est subdivis´e en services (voir le § D.5.3). Dans ce cas, ils relient les services les uns aux autres ou avec les acheminements de signaux du processus.

Si un signal est remis à un acheminement de signaux aboutissant à une fronti`ere de bloc, il passe dans le canal reli´e à l'acheminement de signaux. Lorsqu'un signal parvient au bloc, à partir d'un canal reli´e à un ou plusieurs acheminements de signaux, ce signal est plac´e dans l'acheminement de signaux qui est capable de la transf´erer.

En LDS/PR, la d´efinition d'un acheminement de signal commence par le mot cl´e SIGNALROUTE. La syntaxe des chemins de communication et la liste des signaux sont les m | mes que dans le cas des canaux.

En LDS/GR, la seule diff´erence entre un acheminement de signaux et un canal est que, pour les acheminements de signaux, des fl`eches doivent
| tre plac´ees aux points terminaux des lignes; pr`es de chaque fl`eche, doit se trouver une liste de signaux appropri´es. On trouvera des exemples d'acheminements de signaux dans les figures D-3.6.3 et D-3.6.5 des paragraphes qui suivent.

D.3.6 *Diagrammes de syst`eme et de bloc*

En LDS/GR, une d´efinition de syst`eme est repr´esent´ee au moyen d'un ensemble de diagrammes. La structure d'un syst`eme en canaux et en blocs, est repr´esent´ee à l'aide d'un diagramme de syst`eme.

Le diagramme de syst`eme se compose des ´el´ements suivants:

— le symbole de cadre: symbole de forme rectangulaire contenant tous les autres symboles. Il repr´esente la fronti`ere du syst`eme; l'environnement du syst`eme se trouve en dehors de ce cadre;

— l'en-t | te du syst`eme: mot cl´e SYSTEM suivi du nom du syst`eme (plac´e dans l'angle sup´erieur gauche);

— une num´erotation de page facultative (plac´ee dans l'angle sup´erieur droit du cadre);

— des symboles de texte: un tel symbole peut contenir des ´enonc´es lin´eaires. Il sert g´en´eralement à pr´esenter dans le diagramme des d´efinitions de signaux, des listes de signaux et les donn´ees;

— la zone d'interaction de blocs qui comprend la sp´ecification des blocs du syst`eme, les canaux et les listes de signaux achemin´es par les canaux;

— des diagrammes de macros: les directives concernant l'utilisation des macros sont donn´ees au § D.5.1.

Dans le diagramme de syst`eme, la sp´ecification d'un bloc peut
| tre l'un des deux points suivants:

— une r´ef´erence de bloc: symbole de bloc contenant uniquement le nom de bloc;

— un diagramme de bloc: cadre contenant la sp´ecificationde la structure de bloc en fonction de ses processus et interactions. Si un bloc est subdivis´e en sous-blocs, la sp´ecification de la sous-structure ou la r´ef´erence à celle-ci doit | tre plac´ee à l'int´erieur du cadre de bloc (§ D.4.3).

On trouvera dans le r´esum´e concernant le LDS/GR la forme des symboles utilis´es dans le syst`eme et celle des diagrammes de bloc. En ce qui concerne les d´efinitions de symboles, il convient de se r´ef´erer au § D.7.1.4.

La figure D-3.6.1 donne un exemple d'un diagramme de syst`eme pour le syst`eme <<s>>. Dans cet exemple, le syst`eme s'est divis´e en deux blocs B1 et B2 reli´es l'un à l'autre et à l'environnement par les canaux C1, C2, C3 et C4. Pour les blocs B1 et B2, on ne donne dans cet exemple qu'une r´ef´erence. Des explications compl´ementaires sur les canaux et les symboles de listes de signaux sont donn´es dans les paragraphes qui suivent:

Le m | me exemple en LDS/PR est repr´esent´e dans la figure D-3.6.2.

Figure D-3.6.1, p.

Figure D-3.6.2 [T6.100] (a traiter comme tableau MEP, p.

La structure d'un bloc en ce qui concerne les processus et les acheminements de signaux est repr´esent´ee en LDS/GR à l'aide d'un diagramme de bloc.

Le diagramme de bloc se compose des ´el´ements suivants:

— le symbole cadre: symbole de forme rectangulaire contenant tous les autres symboles. Il repr´esente les fronti`eres du bloc: au-del`a du cadre commence l'environnement du bloc;

— l'en-t | te du bloc: le mot d´e BLOCK suivi du nom de bloc (plac´e dans l'angle sup´erieur gauche du cadre);

— une num´erotation de page facultative (plac´ee dans l'angle sup´erieur droit du cadre);

— des symboles de texte: ces symboles peuvent englober des ´enonc´es lin´eaires. Ils sont g´en´eralement utilis´es pour pr´esenter les d´efinitions de signaux, de listes de signaux et de donn´ees;

— la zone d'interaction de processus: cette zone comprend la sp´ecificationdes processus du bloc et ´eventuellement des acheminements de signaux et des listes de signaux transport´es par les acheminements de signaux. Dans cette zone, il est ´egalement possible de repr´esenter les processus qui cr´eentd'autres processus; cette caract´eristique est d´ecrite au § D.3.8.1;

— des identificateurs de canaux: si le diagramme fait appara | tre des acheminements de signaux aboutissant à l'environnement du bloc ou venant de celui-ci, les identificateurs des canaux reli´es à ces acheminements de signaux doivent | tre sp´ecifi´es en dehors du cadre, de fa,con correspondante aux lignes des acheminements de signaux;

— des diagrammes de macros: on trouvera au § D.5.1 des directives sur l'utilisation des macros.

Dans le diagramme de bloc, la sp´ecification d'un processus peut
| tre:

— soit une r´ef´erence à un processus: symbole de processus contenant le nom de processus et, en option, la sp´ecification d'instances de processus. Une telle sp´ecification comprend deux couples de nombres entiers s´epar´es par une virgule et plac´es entre parenth`eses (voir le § D.3.8);

— soit un diagramme de processus: cadre contenant un graphique de symboles reli´es d´ecrivant le comportement du processus en ce qui concerne les ´etats, les entr´ees, les sorties, les actions, etc. (voir le § D.3.8). Si le processus est subdivis´e en services, le diagramme de processus contient la zone d'interaction de service (§ D.5.3).

Dans la figure D-3.6.3, on trouvera un exemple de diagramme de bloc pour le bloc <<B1>> pr´esent´e dans l'exemple de la figure D-3.6.1. Ce bloc B1 est d´ecrit du point de vue des deux processus P1 et P2, reli´es par les acheminements de signaux R1, R2, R3, R4 et R5. Pour les processus P1 et P2, on se borne à donner des r´ef´erences dans cet exemple. Les identificateurs de canaux C1, C2 et C3 sont ´egalement sp´ecifi´es en dehors du cadre.

Le m | me exemple est pr´esent´e en LDS/PR dans la figure D-3.6.4.

Comme cela a d´ej`a ´et´e indiqu´e pr´ec´edemment, des diagrammes de bloc peuvent | tre compris dans les diagrammes de syst`eme, en lieu et place de r´ef´erences. Dans la figure D-3.6.5 par exemple, les diagrammes des figures D-3.6.1 et D-3.6.3 sont r´eunis en un seul diagramme de syst`eme.

Une recommandation g´en´erale concernant l'´etablissement de diagrammes de ce genre, est qu'ils ne doivent pas | tre trop complexes afin de rester lisibles et qu'ils doivent tenir sur une seule page.

Figure D-3.6.3, p.

Figure D-3.6.4 [T7.100] (a traiter comme tableau MEP, p.

Figure D-3.6.5, p.

D.3.7 *Commentaires et extension de texte*

D.3.7.1 *Commentaires*

Il est possible d'ajouter des commentaires à une spécification en LDS pour aider le lecteur et préciser certains points. Deux types de commentaires adaptés au LDS/PR et au LDS/GR ont été introduits dans le LDS.

Le premier type, appelé <<note>> et utilisé particulièrement en LDS/PR. Il est délimité par <</*>> au début et par <<*/>> à la fin.

En LDS/PR, un tel commentaire peut s'insérer partout où se trouve un espace. Il ne doit pas contenir la séquence spéciale <<*/>>.

En LDS/GR, un tel commentaire peut se présenter partout où il existe un espace à l'intérieur des instructions linéaires.

On trouvera dans les figures D-3.7.1 et D-3.7.2 certains exemples de cette forme de commentaire, respectivement en LDS/PR et en LDS/GR.

Figure D-3.7.1 [T8.100] (à traiter comme tableau MEP, p.

Figure D-3.7.2, p.

La seconde forme de commentaire permet une mise en correspondance élément par élément entre le LDS/PR et le LDS/GR; elle convient mieux aux applications comportant des traductions automatiques.

En LDS/PR, un tel commentaire se compose du mot clé COMMENT suivi d'une chaîne de caractères; il peut être inséré comme une instruction (c'est-à-dire suivi de <<;>>) partout où une instruction de tâche peut être insérée. De plus, il peut être inséré avant le symbole <<;>> (;) à la fin de toute instruction.

En LDS/GR, cette forme de commentaire est représentée à l'aide d'un symbole de commentaire contenant le texte du commentaire. Le symbole de commentaire est un symbole de forme rectangulaires auquel le côté gauche ou droit fait défait. Ce symoble doit être étendu aussi bien horizontalement que verticalement, de manière à contenir tout le texte. Il peut être relié à un symbole quelconque en LDS/GR ou à une ligne de liaison. Pour indiquer la connexion il faut employer une ligne en traits discontinus. Si l'association entre le texte du commmentaire et le symbole du commentaire ne présente pas d'ambiguïté, le symbole de commentaire peut se présenter simplement sous la forme de crochets.

Dans les figures D-3.7.3 et D-3.7.4, on trouvera certains exemples de cette forme de commentaire, respectivement en LDS/PR et en LDS/GR.

Figure D-3.7.3 [T9.100] (à traiter comme tableau MEP, p.

Figure D-3.7.4, p.

D.3.7.2 *Extension de texte*

Normalement, le texte associé à un symbole graphique devrait être placé à l'intérieur de ce symbole. Cependant, cela n'est pas toujours possible ou pratique. Une autre solution consiste à placer le texte dans un symbole d'extension de texte rattach´e au symbole associ´e. Le symbole d'extension de texte est similaire au symbole de commentaire; la seule diff´erence est que la ligne le reliant est en trait continu et non en trait discontinu (voir la figure D-3.7.5).

Figure D-3.7.5, p.

Un caract`ere de soulignement (<< _>> peut | tre utilis´e à la fin d'une ligne de texte comme caract`ere de continuation. Dans ce cas, les espaces restant sur la m | me ligne ne sont pas consid´er´es comme faisant partie du texte. On trouvera au § D.3.13 des directives plus d´etaill´ees sur la syntaxe des noms.

D.3.8 *Processus*

Un processus est une machine à ´etats finis ´etendue qui d´efinit le comportement dynamique d'un syst`eme. Les processus sont fonci`erement dans un ´etat d'attente des signaux. A la r´eception d'un signal, le processus r´epond en accomplissant les actions pr´ecises qui correspondent à chaque type de signal pouvant | tre re,cu par le processus. Les processus contiennent de nombreux ´etats qui leur permettent d'accomplir diff´erentes actions en cas de r´eception d'un signal. Ces ´etats m´emorisent les actions pr´ec´edemment accomplies. Apr`es l'accomplissement de toutes les actions li´ees à la r´eception d'un signal donn´e, un nouvel ´etat est atteint et le processus se met en attente d'un autre signal.

Les processus peuvent soit exister au moment de la cr´eation du syst`eme, soit | tre cr´e´es suite à une demande de cr´eation ´emise par un autre processus. En outre, les processus peuvent durer ind´efiniment ou s'arr | ter du fait du d´eclenchement d'une action d'arr | t.

Une d´efinition de processus repr´esente la sp´ecification d'un type de processus; plusieurs instances du m | me type de processus peuvent
| tre cr´e´ees et exister simultan´ement;elles peuvent fonctionner ind´ependamment et concurremment. Une d´efinition de processus comprend les points suivants (dont certains sont facultatifs):

— nom de processus;

— une paire de nombre entiers. Le premier de ces nombres entiers sp´ecifiele nombre d'instances de processus cr´e´ees lors de la cr´eation du syst`eme; s'il est omis, il a une valeur par d´efaut de 1: le second entier sp´ecifie le nombre maximal d'instances de processus simultan´ees; s'il est omis, la valeur par d´efaut maximale est illimit´ee;

— param`etres formels: liste d'identificateurs de variables associ´es à leurs types qui est utilis´ee pour transmettre l'information au processus au moment de la cr´eation. Dans la demande de cr´eation de processus, une liste de param`etres r´eels peut | tre donn´ee à cet effet. Les valeurs des param`etres formels de processus cr´e´es au moment de la cr´eation du syst`eme sont ind´efinies;

— ensemble de signaux d'entr´ee valides: liste d'identificateurs de signaux d´efinissant les signaux que le processus peut recevoir;

— d´efinitions de signaux: sp´ecification des signaux qui peuvent
| tre ´echang´es entre instances du m | me processus ou entre services de ce processus (§ D.5.3);

— d´efinitions de proc´edures: sp´ecification des proc´edures qui peuvent | tre appel´eespar le processus. Une r´ef´erence de proc´edure peut | tre utilis´ee dans ce contexte (les proc´edures font l'objet du § D.3.9);

— d´efinition de donn´ees:sp´ecifications des nouveaux types, syntypes et g´en´erateurs d´efinis par l'usager et localis´es dans le processus;

— d´efinition de variables: d´eclarations des variables du processus. A titre facultatif, on peut indiquer qu'une variable peut | tre partag´ee avec plusieurs autres processus du m | me bloc (variable REVEALED) ou export´ee vers d'autres processus ainsi que vers d'autres blocs (variable EXPORTED). Pour chaque variable d´eclar´ee, il faut sp´ecifier l'identificateur de sa sorte. Une variable initiale peut facultativement | tre sp´ecifi´ee;

— d´efinitions de visibilit´e: d´eclarations d'identificateurs de variable qui peuvent servir à l'obtention des valeurs de variables appartenant à d'autres instances de processus. Pour chaque identificateur de variable, la sorte de variable doit

| tre sp´ecifi´ee;

— d´efinitions d'import: sp´ecification d'identificateurs de variables appartenant à d'autres processus et que le processus veut importer. Pour chaque identificateur, il convient de sp´ecifier la sorte de la variable;

— d´efinition de temporisateur (timer): fait l'objet du § D.3.11;

— d´efinitions de macros: on trouvera des directives sur l'utilisation des macros au § D.5.1;

— corps de processus: sp´ecification du comportement r´eel du processus exprim´e à l'aide d'´etats, entr´ees, sorties, t | ches, etc. Si le processus est divis´e en sous-parties (services), la d´efinition du processus comporte une section de d´ecomposition en services en lieu et place du corps du processus. On trouvera des directives à ce sujet au § D.5.3.

Des exemples et des explications concernant les donn´ees, les variables, les d´efinitions de visibilit´e et les d´efinitions d'import sont donn´es au § D.3.10.

On trouvera dans la figure D-3.8.1 un exemple partiel de d´efinition de processus en LDS/PR (les mots cl´es du langage sont ´ecrits en lettres majuscules).

Figure D-3.8.1 [T10.100] (a traiter comme tableau MEP, p.

Le corps de processus repr´esente le graphe r´eel de la machine à ´etats finis. Il consiste en une s´equence d'instructions ordonn´ees en LDS/PR et, en LDS/GR, c'est une s´equence de symboles reli´es par des arcs orient´es (similaires à un organigramme). La sp´ecification du corps de processus doit toujours commencer par l'indication START suivi d'un ensemble d'actions (transition). L'interpr´etation d'une instance de processus commence à la cr´eation de cette instance de processus.

Les actions qui peuvent | tre ex´ecut´ees dans une transition sont les suivantes:

— t | che: affectation de variable (ou texte informel);

— exportation: exportation de variable;

— initialisation (set): demande d'activation d'un temporisateur;

— r´einitialisation (reset): r´einitialisation d'un temporisateur;

— sortie: ´emission d'un signal en direction d'un autre processus;

— demande de cr´eation: cr´eation d'une instance d'un type de processus sp´ecifi´e;

— d´ecision: s´election d'un ensemble d'actions d´ependant d'une question;

— appel de proc´edure: demande d'interpr´etation d'un ensemble d'actions s´epar´e autonome (m | me usage que dans des langages de de programmation);

— branchement (join): sp´ecification d'un <<saut>> vers un autre ensemble d'actions.

Une transition peut se terminer par l'une des actions suivantes:

— nexstate (´etat suivant): sp´ecification de l'´etat dans lequel se trouvera l'instance de processus;

— arr | t: arr | t imm´ediat de l'instance de processus.

Apr`es la sp´ecification de l'action de d´epart et de la transition de d´epart facultative, le corps du processus comprend les d´efinitions de tous ses ´etats possibles. Chaque d´efinitin d'´etat commence par la sp´ecification des stimuli possibles, attendus par le processus dans cet ´etat. Les stimuli possibles sont les suivants:

— entr´ees: signaux qui peuvent | tre re,cus;

— mises en r´eserve: signaux qui doivent | tre mis en r´eserve pour traitement ult´erieur;

—   conditions de validation: décrites au § D.3.8.5;

—   signaux continus: décrits au § D.3.8.5.

Une transition doit être spécifiée en correspondance avec chaque stimulus sauf en ce qui concerne les mises en réserve. De telles transitions représentent la séquence d'actions que le processus exécutera si le stimulus apparaît.

Si un processus exécute une action d'arrêt et que des signaux émis mais non encore reçus se trouvent en suspens, ces signaux sont mis au rebut.

En LDS/GR, une définition de processus est représentée à l'aide du diagramme de processus. Un diagramme de processus se compose des éléments suivants:

—   un symbole de cadre: symbole de forme rectangulaire contenant tous les autres symboles. Si aucun acheminement du signal n'est rattaché au symbole de cadre, celui-ci peut être omis;

—   l'en-tête du processus: le mot clé PROCESSUS suivi de l'identificateur de processus, puis éventuellement de la spécification des paramètres formels. L'en-tête de processus est placé dans l'angle supérieur gauche du cadre;

—   une numérotation de page facultative (placée dans l'angle supérieur droit);

—   des symboles de texte: dans le cas d'un diagramme de processus, un symbole de texte peut être utilisé pour contenir des définitions de signal, de variable, de visibilité, d'importation, de données et de temporisateur (timer);

—   références de procédure: symbole de procédure contenant un nom de procédure représentant une procédure du processus qui est définie séparément;

—   diagrammes de procédure: un pour chaque procédure locale du processus qui n'est pas référenciée;

—   la zone de graphe de processus: spécification du comportement du processus en termes de départ, d'états, d'entrées, de sorties, de tâches, ... et d'arcs orientés. Si le processus est structuré en services, la zone du graphe de processus contient la spécification des services ou leurs références (voir le § D.5.3). Les symboles en GR utilisés pour le corps du processus sont indiqués dans le résumé du LDS/GR;

—   diagrammes de macro: on trouvera au § D.5.1 des directives sur l'utilisation des macros.

La figure D-3.8.2 donne un exemple de définition de processus en LDS/GR; on trouvera des explications et des exemples complémentaires sur les graphes de processus dans les paragraphes qui suivent.

Figure D-3.8.2, p.

Si un graphe de processus ne tient pas sur une seule page, le diagramme peut être présenté sur plusieurs pages; il faut noter:

—   qu'une numérotation de page contenant le numéro de la page et le nombre total de pages devrait être fournis, c'est-à-dire: 1 (9);

—   que le symbole de brachement (join) ou le symbole d'état suivant (nextstate) peut être utilisé pour représenter des connexions entre différentes parties du graphe du processus.

Un bon critère pour diviser un graphe de processus en plusieurs parties consiste à représenter une définition d'état sur chaque page. Si une définition d'état ne tient pas sur une seule page, il est possible d'utiliser le symbole de branchement pour représenter des connexions avec une partie du graphe qui se trouve sur une autre page.

D.3.8.1 *Création de processus*

Comme indiqué au paragraphe précédent, des processus (c'est-à-dire des instances de processus) peuvent être créés à la suite d'une demande explicite ou lors de la création du système.

La demande explicite de cr´eation ne peut | tre formul´ee que par un autre processus du m | me bloc que le processus à cr´eer; elle permet la sp´ecification de param`etres r´eels pour le transfert de l'information vers la nouvelle instance cr´e´ee. La figure D-3.8.3 donne un exemple de cr´eation en PR et en GR.

Figure D-3.8.3, p.

Si une ou plusieurs instances de processus d'un type de processus donn´e sont cr´e´ees au moment de la cr´eation du syst`eme, la d´efinition de ce (ou ces) processus ne doit pas avoir acc`es aux param`etres formels car ils seront ind´efinis. En cons´equence, chaque d´efinition de processus ayant des param`etres formels doit ´egalement faire le n´ecessaire pour interdire l'acc`es à ses param`etres formels avant que ceux-ci aient re,cu une valeur ou qu'ils comportent explicitement la sp´ecification d'aucune instance de processus au moment de la cr´eation du syst`eme (voir la figure D-3.8.4).

Figure D-3.8.1 [T11.100] (a traiter comme tableau MEP, p.

Lors de la cr´eation d'un processus, il est possible de d´eterminer des valeurs d'instance de processus à l'aide des expressions pr´ed´efinies suivantes:

(OFFSPRING) DESCENDANT: renvoie la valeur PId de la derni`ere instance cr´e´ee,

SELF: renvoie la valeur PId de l'instance de processus proprement dite,

(PARENT) ASCENDANT: renvoie la valeur PId de l'instance qui proc`ede à la cr´eation

(SENDER) EMETTEUR: renvoie la valeur PId du processus ´emettant le dernier signal utilis´e.

Lorsqu'un processus est cr´e´e par suite de la cr´eation du syst`eme, seule l'expression SELF renvoie une valeur PId; les expressions OFFSPRING et PARENT donnent la valeur NULL.

De telles valeurs rev | tent une grande importance lorsqu'il existe plusieurs instances de processus du m | me type de processus car elles constituent le seul moyen d'adresser sans ambigu"it´e les signaux aux instances. En fait, comme cela est mieux expliqu´e au § D.3.8.6, lorsqu'un processus ´emet un signal, il doit sp´ecifier l'instance de destination, à moins que celle-ci ne puisse | tre d´etermin´ee sans ambigu"it´e.

L'utilisateur doit veiller à ce que les instances de processus cr´e´ees puissent, au besoin, communiquer les unes avec les autres. Pour y parvenir, il faut souvent pr´evoir certains types de processus d'initialisation. Ces processus, cr´e´es en m | me temps que le syst`eme, devront cr´eer les autres processus et ´eventuellement communiquer les valeurs PId appropri´ees à tous les processus auxquels elles seront n´ecessaires.

Il convient de noter d'autres points importants concernant la cr´eation de processus:

1) apr`es la cr´eation du syst`eme, les processus ne peuvent | tre cr´e´es que par un autre processus du m | me bloc. Une possibilit´e de cr´eation d'un processus dans d'autres blocs consiste à avoir un processus sp´ecial dans chaque bloc; ce processus provoque la cr´eation d'un processus à la r´eception d'un signal provenant d'un processus d'un autre bloc;

2) apr`es leur cr´eation, les processus ont une dur´ee de vie qui leur est propre. Ils ne cessent d'exister que lorsqu'ils accomplissent une action d'arr | t pendant une transition. On peut construire des syst`emes qui autorisent les op´erations ext´erieures d'´elimination de processus en employant un signal sp´ecial d'´elimination. A la r´eception du signal d'´elimination, le processus accomplit une action d'arr | t.

La relation entre les processus cr´eateurs et les processus cr´e´es peut etˆre repr´esent´ee dans un diagramme de bloc à l'aide de symboles de lignes de cr´eation, comme indiqu´e dans la figure D-3.8.5.

Figure D-3.8.5, p.

D.3.8.2 *Etats*

Un état est une étape du processus pendant laquelle aucune action ne s'accomplit, mais où la file d'entrée contrôle l'arrivée des signaux entrants. Grâce à l'identificateur de signaux que contient le signal d'entrée, l'arrivée du signal fait sortir le processus de l'état et lui fait accomplir une succession donnée d'actions. Un signal qui est arrivé et a entraîné une transition a été <<absorbé>> et cesse d'exister.

En LDS/PR, un état est représenté par le mot clé STATE suivi du nom de l'état. La spécification de l'état se termine soit à l'énoncé d'état suivant soit à la fin du processus ou au moyen du mot clé explicite ENDSTATE (FIN D'ÉTAT). On peut utiliser un astéristique dans l'énoncé d'état au lieu du nom de l'état; il s'agit là d'une notation abrégée indiquant que les entrées pour les mises en réserve suivantes et les transitions correspondantes doivent être interprétées pour chaque état.

Le mot clé NEXTSTATE (ÉTAT SUIVANT), suivi du nom de l'état, est utilisé pour indiquer l'état qui suit. On peut employer un tiret dans l'énoncé d'état suivant au lieu du nom de l'état pour indiquer que l'état suivant est précisément l'état dont la transition actuelle tire son origine.

La figure D-3.8.6 donne un exemple partiel en LDS/PR.

Figure D-3.8.6 [T12.100] (à traiter comme tableau MEP, p.

En LDS/GR, un état est représenté à l'aide d'un symbole d'état contenant le nom d'état et est relié aux symboles d'entrée ou de mises en réserve. Le même symbole d'état, avec une flèche d'arrivée, sert à représenter l'énoncé état suivant.

Par commodité, pour simplifier le dessin ou faciliter la compréhension, le même état peut apparaître plusieurs fois dans un diagrmme en LDS. Si tel est le cas, le diagramme est considéré comme entièrement équivalent au diagramme qui résulterait, de la fusion de toutes les apparitions multiples du même état. Les figures D-3.8.7 et D-3.8.8 donnent des exemples de cette situation. Dans la figure D-3.8.7 b), on utilise un symbole d'état comme lien avec l'état principal portant le même nom, lorsqu'il est mentionné dans le symbole d'état suivant. Dans la figure D-3.8.8 un état est représenté par des symboles multiples n'ayant chacun qu'un sous-ensemble des entrées (ou des mises en réserve).

Les diagrammes a) et b) de la figure D-3.8.7 sont logiquement équivalents. Le diagramme a) contient une seule apparition de chaque état tandis que le diagramme b) utilise des apparitions multiples. Dans le diagramme b), l'état a une apparition principale, dans laquelle tous ses symboles d'entrées (et de mises en réserve) associés sont indiqués. Lorsque cet état peut être atteint à partir d'autres points du diagramme (par exemple le point de terminaison d'une transition), il est indiqué comme un état sans entrée ni mise en réserve associée. Un commentaire du symbole d'état suivant se référant au symbole d'état améliorera la clarté, notamment lorsqu'ils apparaissent sur des pages différentes.

La figure D-3.8.8 utilise les apparitions multiples d'un état pour constituer l'ensemble total d'entrées (et de mises en réserve). Chaque apparition de l'état est indiquée uniquement avec un sous-ensemble de ces entrées.

Figure D-3.8.7 a, p.

Figure D-3.8.7 b, p.

Figure D-3.8.8, p.

Cette approche a été appliquée avec succès lorsque des états avaient un nombre relativement grand d'entrées ou de mises en réserve mais présentaient le risque que le lecteur interprète de façon erronée le diagramme s'il n'était pas conscient de l'existence d'occurrences multiples. Pour éviter ce malentendu, les états n'indiquant qu'un sous-ensemble d'entrées/mises en réserve devraient être accompagnés d'un commentaire indiquant une référence à d'autres états avec leurs entrées et mises en réserve associées, comme indiqué dans la figure D-3.8.8.

Des apparitions multiples peuvent être utilisées pour attirer l'attention du lecteur sur certains aspects (par exemple la séquence normale des signaux traités), laissant d'autres aspects pour d'autres pages (par exemple le traitement de situations d'alarme).

Au cours d'une transition un processus ne sait pas explicitement quel signal d'entrée a occasionné la transition. Seul le contexte permet de le déduire (c'est-à-dire que cette transition ne peut se produire qu'en cas de réception d'un signal donné). Dans la figure D-3.8.9, la tâche T1 ne s'accomplit qu'en cas de réception de I1. Toutefois, la tâche T2 s'accomplit en cas de réception de I2 ou de I3. S'il importe que T2 sache quel signal d'entrée a été reçu, il est souhaitable de concevoir le processus comme l'illustre la figure D-3.8.10.

Figure D-3.8.9, p.

Figure D-3.8.10, p.