Assistance à la conception et à l'évaluation de l'architecture de conduite des systèmes de production complexes

Bruno Denis

Thèse
présentée à
l'Université de Nancy I

pour l'obtention du grade de
Docteur de l'Université de Nancy I
en Production Automatisée

par
Bruno DENIS

Assistance à la conception et à l'évaluation de l'architecture de conduite des systèmes de production complexes.

Soutenue publiquement le 17 février 1994 devant la commission d'examen composée de :

Directeur de thèse	P. Bourdet	Professeur à l'ENS de Cachan
Président	G. Morel	Professeur à l'Université de Nancy I
Rapporteurs	J.C. Gentina	Professeur à l'École Centrale de Lille (Directeur)
	G. Villermain-Lecolier	Professeur à l'Université de Reims
Examinateurs	P. Lhoste	Maître de conférences à l'Université de Nancy I
	J.L. Delcuvellerie	Ingénieur, société EURIWARE, docteur de l'INPL
	J.J. Lesage	Maître de conférences à l'IUFM de Créteil

Laboratoire Universitaire de Recherche en Production Automatisée
ENS de Cachan - 61 avenue du Président Wilson - 94235 Cachan Cedex

Assistance à la conception et à l'évaluation de l'architecture de conduite des systèmes de production complexes.

par
Bruno DENIS

Université de Nancy I
Ecole Doctorale :
Informatique, Automatique, Electrotechnique, Mathématiques
Département de Formation Doctorale :
Automatique et Production Automatisée

ENS de Cachan
Laboratoire Universitaire de Recherche
en Production Automatisée (LURPA - EA 1385)

pour l'obtention du grade de
Docteur de l'Université de Nancy I
en Production Automatisée

Soutenue publiquement le 17 février 1994

Directeur de thèse	P. Bourdet	Professeur à l'ENS de Cachan
Président	G. Morel	Professeur à l'Université de Nancy I
Rapporteurs	J.C. Gentina	Professeur à l'École Centrale de Lille (Directeur)
	G. Villermain-Lecolier	Professeur à l'Université de Reims
Examinateurs	P. Lhoste	Maître de conférences à l'Université de Nancy I
	J.L. Delcuvellerie	Ingénieur, société EURIWARE, docteur de l'INPL
	J.J. Lesage	Maître de conférences à l'IUFM de Créteil

à la mémoire de Gérard

à mes Parents

à Philippe et Christine

Avant-propos

Le travail présenté dans ce mémoire a été effectué au sein du Laboratoire Universitaire de Recherche en Production Automatisée (LURPA - EA 1385) de l'École Normale Supérieure de Cachan. Il a été dirigé par Monsieur Jean-Jacques LESAGE, Maître de conférences à l'IUFM de Créteil, sous la responsabilité scientifique de Monsieur Pierre BOURDET, Professeur à l'ENS de Cachan.

Que le Professeur Pierre BOURDET, trouve ici toute ma reconnaissance pour m'avoir accueilli dans le laboratoire qu'il dirige, et pour m'avoir accordé sa confiance.

Je tiens à remercier tout spécialement Monsieur Jean-Jacques LESAGE à la fois pour ses conseils avisés, sa disponibilité, et son soutien dans ce travail, mais également pour son enthousiasme à la recherche qu'il a su me communiquer.

Les rapports auprès de la formation doctorale ont été établis par Monsieur Jean-Claude GENTINA, Directeur de l'École Centrale de Lille, et Monsieur Gérard VILLERMAIN-LECOLIER, Professeur à l'Université de Reims. Je les remercie vivement d'avoir accepté cette tâche et leur sais gré pour les conseils qu'il m'ont apportés pour la rédaction de ce mémoire.

Je suis particulièrement reconnaissant à Monsieur Gérard MOREL, Professeur à l'Université de Nancy I, de me faire l'honneur de présider le jury.

Que Monsieur Pascal LHOSTE, Maître de conférences à l'Université de Nancy I accepte mes remerciements pour sa participation au jury en qualité d'examinateur.

Je remercie tout particulièrement Monsieur Jean-Luc DELCUVELLERIE, de la société EURIWARE, et les membres la commission SAP de l'EXERA qu'il anime, pour l'intérêt porté par le milieu industriel à ces travaux.

Je ne pourrais terminer sans remercier tous les membres du laboratoire pour leur soutien ainsi que leur contribution directe ou indirecte à ce mémoire. Je pense plus particulièrement à Guy, Jean-Marc, Christophe, Laurent et Loïc mais je n'oublie pas les autres.

Table des matières

Table des illustrations

Liste des abréviations

API	Automate Programme Industriel (NF C63-850)
CIM	Computer Integrated Manufacturing
CPU	Central Processing Unit
DEB	Document d'Expression des Besoins
ICAM	Integrated Computer Aided Manufacturing
IDEF 0	ICAM DEFefinition
MCD	Modèle Conceptuel des Données
MCT	Modèle Conceptuel des Traitements
MIPS	Million d'instructions par seconde
MOT	Modèle Organisationnel des Traitements
RdP	Réseau de Petri
SA	System Analysis
SADT	System Analysis and Design Technique
SART	System Analysis - Real Time
SNCC	Système Numérique de Contrôle Commande

Vocabulaire

Architecte matérielle

Description organique des constituants d'architecture et de leurs interconnexions. Dans l'architecture matérielle, les constituants d'architecture sont décrits en terme de référence à un matériel fourni par les constructeurs du marché de l'offre.

Cahier de charges fonctionnel

Document qui regroupe les spécifications fonctionnelles (résultat de l'analyse fonctionnelle) et les contraintes opérationnelles du système à concevoir.

Constituant d'architecture

Indifféremment équipement de traitement, équipement de communication, ou équipement de dialogue utilisé pour concevoir les architectures de conduite des systèmes de production

Contraintes opérationnelles

Ensemble de contraintes ne relevant pas des besoins fonctionnels mais qui sont fixées dès le début du projet soit pour s'adapter à un environnement existant, soit pour imposer une partie de la solution.

Document d'Expression des Besoins

Document élaboré en phase d'analyse des besoins et à partir du cahier des charges de consultation fourni lors d'un appel d'offre.

Ingénieriste

Individu ou par extension ensemble d'individus qui élabore une offre à partir d'un cahier des charges de consultation qui lui est proposé lors d'un appel d'offre

Notations

$C(t)$	ensemble des conjonctions associées au traitement procédural t
$D(t)$	ensemble des disjonctions associées au traitement procédural t
$d_e(e)$	débit de l'équipement de communication e
$d_{mini}(r_i)$	délai minimum pour transmettre un message dans l'équipement de communication r_i (unité : $\text{sec}onde$)
$D_{E_s}(e)$	ensemble des données affectées à l'équipement de stockage e
E_{API}	ensemble des équipements de type automate programmable industriel et de type commande numérique
E_{cal}	ensemble des équipements de type calculateur d'un modèle organique
E_{cma}	équipement de communication à accès maître-esclave
E_{com}	équipement de communication à accès aléatoire
$E_s(p)$	ensemble des équipements de stockage supportés par l'équipement de traitement p
$F_a(f)$	fréquence d'accès au flot de donnée f (unité : $\text{sec}onde^{-1}$)
$F(d)$	ensemble des flots accédant à la donnée d
$F(e)$	ensemble des flots de donnée qui circulent dans l'équipement de communication e
$F_f(e)$	sous-ensemble de $F(e)$ constitué des flots fonctionnels qui véhiculent les données
$F_t(e)$	sous-ensemble de $F(e)$ constitué des flots techniques périodiques issus des traitements de scrutation et de réponse
$F_T(t)$	fréquence d'exécution du traitement t (unité : $\text{sec}onde^{-1}$)
$M(d)$	ensemble des messages qui sont émis lors de la disjonction d

$n(e)$	nombre moyen d'instructions que devra exécuter l'équipement de traitement qui supporte l'équipement de stockage $e \in E_s$ (unité : $octet$)
$n_i(t)$	nombre moyen d'instructions élémentaires que devra exécuter un équipement de traitement pour que le traitement procédural t soit réalisé (unité : $octet$)
$n_{mot}(t)$	nombre moyen de kilo-mots que devrait exécuter un automate programmable industriel pour que le traitement de contrôle-commande t soit évalué une fois (unité : $octet$)
$P_{exe}(t)$	période d'exécution souhaitée pour le traitements de contrôle-commande t (unité : $seconde$)
$P_{sc}(t)$	période de scrutation du traitement technique t (unité : $seconde$)
$S_c(t)$	ensemble des stockages de données à lire à la suite d'une conjonction associée au traitement procédural t
$S_e(d)$	ensemble des stockages qui vont être effectués lors de la disjonction d
$T_{cc}(p)$	ensemble des traitements de contrôle-commande affectés dans p
T_{cycle}	temps de cycle d'un automate programmable industriel (unité : $seconde$)
$T_d(f)$	taille moyenne des données qui circulent dans le flot f (unité : $octet$)
T_{sc}	taille des messages techniques de scrutation et de réponse mis en œuvre dans les traitements techniques de scrutation (unité : $octet$)
T_{sce}	ensemble des couples de traitements techniques scrutation-réponse qui transmettent une demande d'écriture
T_{scl}	ensemble des couples de traitements techniques scrutation-réponse qui transmettent une demande de lecture
T_{scm}	ensemble des couples de traitements techniques scrutation-réponse qui transmettent un message
$t_{sc}(f)$	traitement technique de scrutation dont dépend $f \in F_t(e)$
$T_p(p)$	ensemble des traitements procéduraux affectés dans p
V	«vitesse d'exécution» de l'API en nombre d'instructions exécutées par unité de temps (unité : $octet/seconde$)

Introduction

La performance des systèmes automatisés de production est pour une part conditionnée par la performance de leur système de conduite. Le degré d'automatisation, de réactivité et de flexibilité exigé des systèmes de production allant croissant, le système de conduite pèse de plus en plus lourd dans leur conception - à la fois sur le plan technique et sur le plan économique.

La production scientifique de ces dernières années sur le thème de la conception de la conduite des systèmes de production est donc naturellement très importante. Cependant, force est de constater que les énergies de recherche sont à l'heure actuelle quasiment exclusivement consacrées à l'amélioration de la conception de la *partie logicielle* des systèmes de conduite. Leur *partie matérielle*, et notamment la conception de leur *architecture*, reste un terrain quasiment inexploré sur le plan scientifique, alors qu'elle est un enjeu industriel quotidien pour les ingénieristes.

Les travaux dont les résultats sont exposés dans ce mémoire portent sur la *conception de l'architecture de conduite des systèmes automatisés de production*. Il s'agit de travaux *prospectifs* dont l'objectif est de construire un *cadre méthodologique formel* de conception et d'évaluation d'architectures. En l'absence de travaux de recherche antérieurs ayant porté sur ce sujet, c'est *l'ensemble du processus de conception d'architectures qui a été abordé*. Nous avons donc volontairement privilégié le taux de couverture du sujet traité plutôt que le développement en profondeur d'un de ses aspects.

Pour exposer les résultats de nos travaux, nous avons scindé ce mémoire en deux parties.

La première partie est consacrée à l'exposé du cadre méthodologique retenu et comporte quatre chapitres dans lesquels sont successivement abordés :

- la problématique générale de la conception d'architectures de conduite, ainsi qu'une revue des travaux scientifiques dans le domaine,
- la présentation générale de la méthode proposée avec le point de vue des activités de conception, d'évaluation et de choix d'architecture,
- la présentation détaillée de la construction d'architecture avec cette fois le point de vue des différents modèles produits,

- la présentation détaillée de l'évaluation et du choix de l'architecture retenue avec encore une fois le point de vue des modèles d'évaluation.

La seconde partie du mémoire est destinée à préciser et à valider par l'exemple la méthode proposée. Pour ce faire, nous l'avons appliquée à la conception de l'architecture de conduite d'un poste de lancement d'un transfert libre. Le plan de cette étude suit de manière ordonnée la démarche que nous avons présenté dans la première partie.

Partie 1

Méthode proposée

1.1 Le problème et son positionnement

Ce travail présente un cadre méthodologique pour la *conception de l'architecture de conduite des systèmes de production complexes*. Les systèmes de production visés intègrent à la fois des équipements de l'informatique générale, de l'informatique industrielle, ainsi que des équipements de régulation et d'automatisme. On peut citer à titre d'exemple des systèmes très différents tels qu'un atelier automatisé d'assemblage de moteurs de véhicules automobiles ou une centrale de production d'électricité.

Ce cadre méthodologique doit être vu comme une *boîte à outils* dans laquelle le concepteur d'une architecture de conduite trouve à la fois des *outils d'assistance*, et un *fil conducteur* lui permettant d'orienter et de formaliser son travail. Compte tenu de la nature des systèmes envisagés et du point précis auquel on s'attache (leur architecture de conduite), les *ingénieristes* sont les utilisateurs naturels de cette boîte à outils.

1.1.1 La conception d'architecture

Lors de la conception d'architecture de conduite, l'ingénieriste est confronté à des problèmes de nature très différente. Il doit assurer à la fois :

- la gestion des relations avec le client,
- la gestion de la complexité du système,
- la gestion de la diversité des technologies et de l'évolution du marché.

Le client exprime ses besoins, généralement de manière très informelle, sous la forme d'un appel d'offre, à des sociétés d'ingénierie [1] (fig. 1). Celles-ci fournissent en retour une offre chiffrée répondant aux besoins exprimés. Pour élaborer cet avant-projet, l'ingénieriste doit entre autre concevoir une architecture de conduite qu'il lui sera difficile de remettre en cause dans la suite du projet. Pourtant cette architecture de conduite orientera de manière décisive la suite de la conception et de la réalisation du système. "L'architecte automaticien" doit donc affronter le paradoxe suivant :

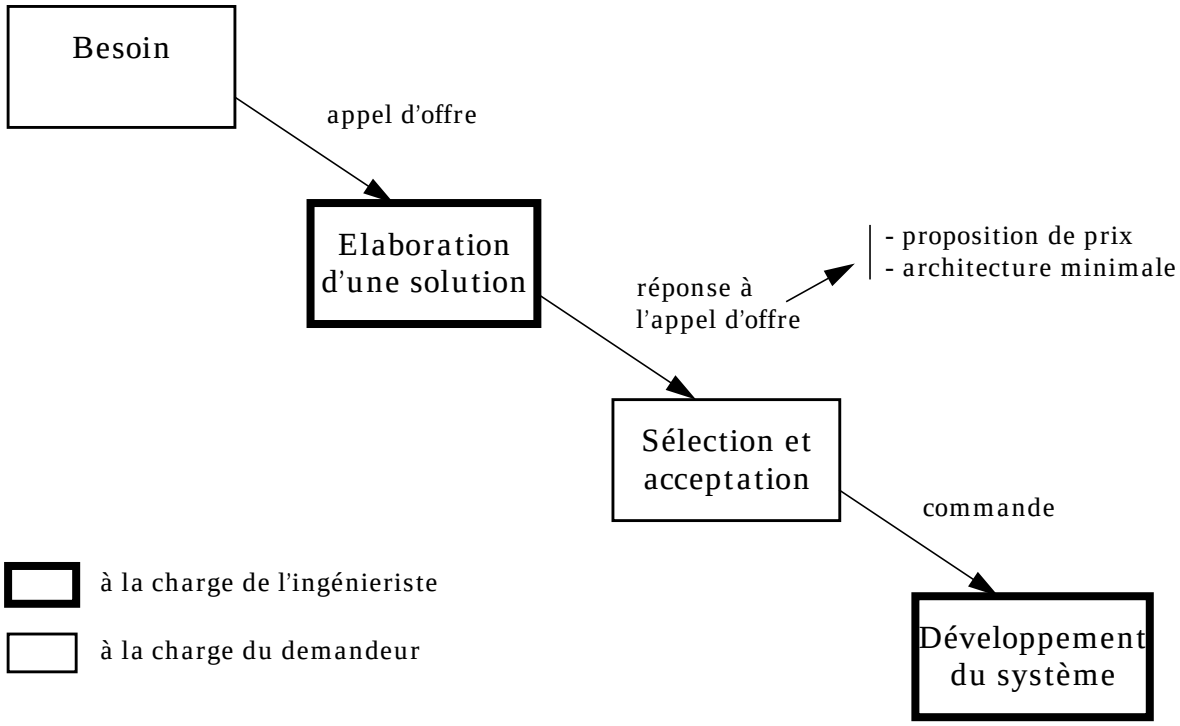


Figure 1 : les attributions de l'ingénieriste

- étudier et formaliser *suffisamment en détail* les besoins du client afin de lui proposer une solution *pertinente* (c'est-à-dire pressentir avant la conception une grande partie du travail de conception), avec des coûts calculés au plus juste dans l'espoir d'emporter le marché. ;
- étudier *suffisamment sommairement* l'appel d'offre pour ne pas engager trop de frais d'étude sur une affaire dont la société d'ingénierie n'a pas encore la charge contractuelle.

Seule l'expérience de l'ingénieriste permet de contourner ce paradoxe. Par la pratique de cas similaires déjà traités (l'expertise), il est capable de proposer une solution suffisamment fine, à l'issue d'une pré-étude rapide. Il n'en reste pas moins que des outils d'assistance à la conception et à l'évaluation de solutions d'architecture lui permettraient :

- de *rationaliser* et d'*homogénéiser* l'expertise à l'intérieur de la société d'ingénierie ;
- de *fixer* et de *partager* l'expertise capitalisée lors des études précédemment effectuées ;

- d'augmenter la *qualité* des architectures conçues, tout en faisant face à l'augmentation régulière de la complexité des systèmes de production. Cette augmentation de la complexité est due à l'augmentation de la partie automatisée des systèmes ainsi qu'à la volonté de les intégrer dans un système plus vaste. En conséquence de quoi, la conduite de ces systèmes croît en volume et en taux de connexion ;
- d'augmenter la *réactivité* de l'ingénieriste vis-à-vis de l'évolution rapide du marché de l'offre en équipements de régulation et d'automatisme. Les réseaux de terrain, par exemple, ouvrent de nouvelles perspectives pour les architectures de conduite. Les clients sont même parfois demandeurs de certaines solutions techniques comme les architectures décentralisées qui connaissent aujourd'hui un véritable essor [2]. L'ingénieriste est donc parfois confronté à des projets pour lesquels il ne peut pas compter sur ses expériences antérieures ;
- d'augmenter la *productivité* de la phase d'avant-projet à laquelle se situe la conception d'architectures.

L'ingénieriste est donc confronté à un environnement complexe et très évolutif. Des *outils de modélisation* et des *méthodes de conception* doivent lui fournir le moyen de gérer cette complexité tout en tendant vers une qualité toujours accrue des systèmes développés.

1.1.2 Domaine d'application

Pour tout système on peut définir une mission, c'est-à-dire son objectif principal parfois nommé but, finalité, ou projet [3] [4]. Une mission s'exprime en terme de transformation d'entrées-sorties, ce qui est particulièrement net dans le cas des systèmes de production industriels, où le moyen de production (machine, station, atelier, usine) transforme des matières premières pour en faire des produits élaborés. Le système de conduite n'est alors qu'un sous-système dans le système. Afin d'en préciser la frontière nous allons nous appuyer sur le modèle canonique de J.-L. Le Moigne et la décomposition d'un système en sous-systèmes de J. Mélèse.

J.-L. Le Moigne présente dans [5] un modèle canonique de tout système (fig. 2), et met en place un vocabulaire aujourd'hui couramment utilisé dans la communauté scientifique du génie automatique.

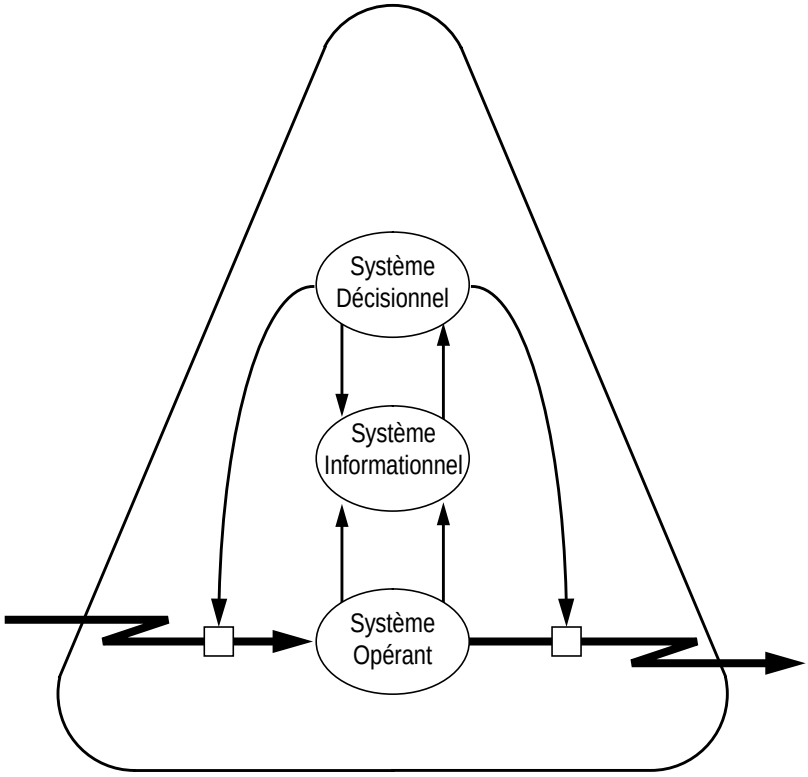


Figure 2 : le modèle canonique OID (d'après J.-L. Le Moigne [5])

J. Mélèse propose pour sa part une déclinaison de ce modèle canonique pour les systèmes de production, en mettant en évidence trois sous-systèmes principaux [6] (fig. 3) :

- le *sous-système technologique* qui est identifié, quelle que soit la nature du système, comme étant la partie qui effectue la transformation sur le flux caractéristique de la mission principale,
- le *sous-système de pilotage* qui guide, contrôle, régule les transformations technologiques,
- le *sous-système d'information* et de mesure qui capte, stocke, traite et distribue les informations en provenance du système technologique.

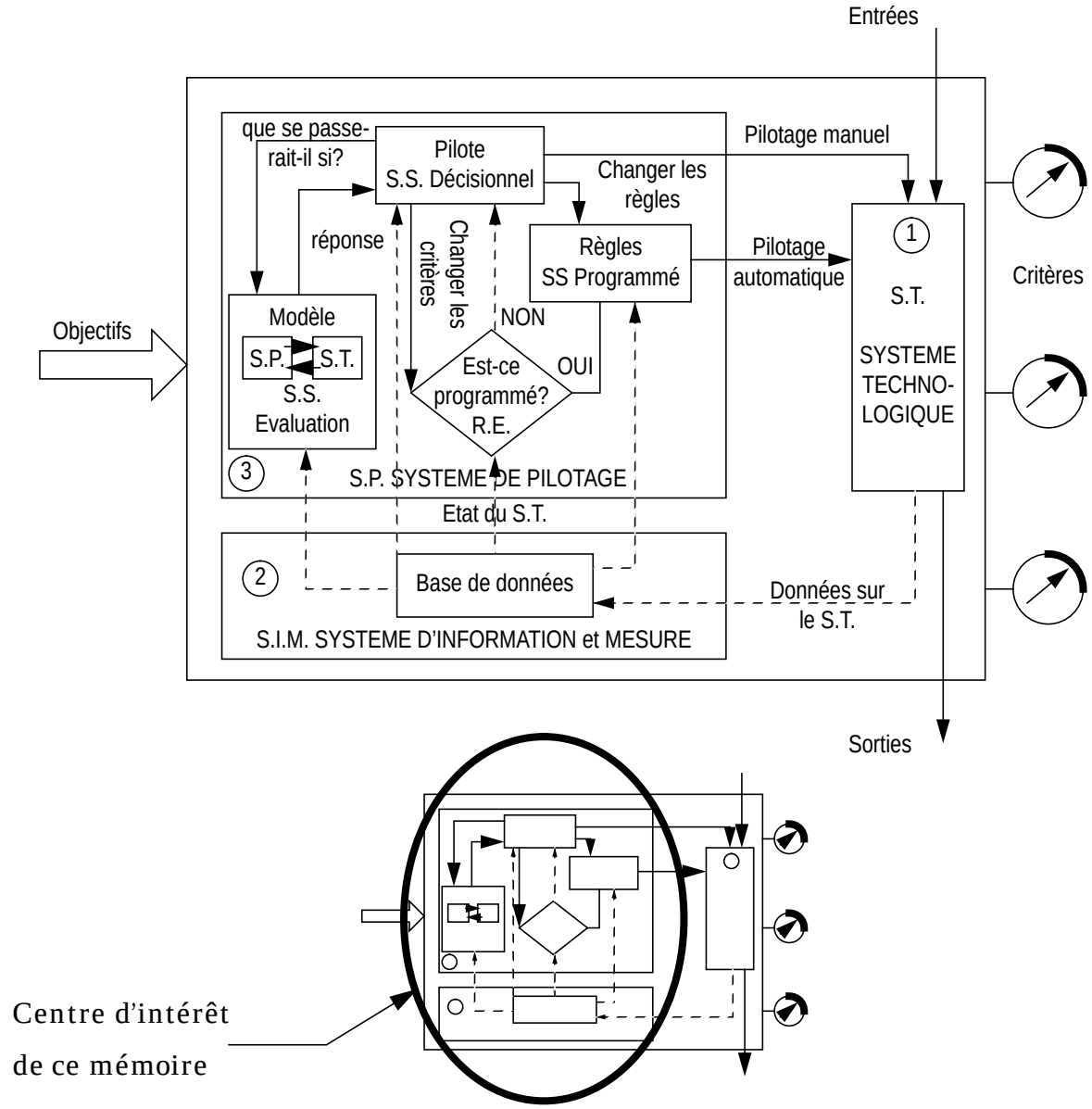


Figure 3 : structure d'un système de production (d'après J. Mélèse [6])

En étudiant un système de production complexe au travers du modèle de référence de J. Mélèse, on peut positionner le système dont nous voulons concevoir l'architecture, comme *englobant le sous-système d'information et mesure et le sous-système de pilotage*. Nous le nommerons dans la suite *(sous) système de conduite.*

A l'opposé de cette vue conceptuelle, le résultat de la conception d'une architecture de conduite est couramment représenté dans la littérature par une vue matérielle comme sur la figure 4.

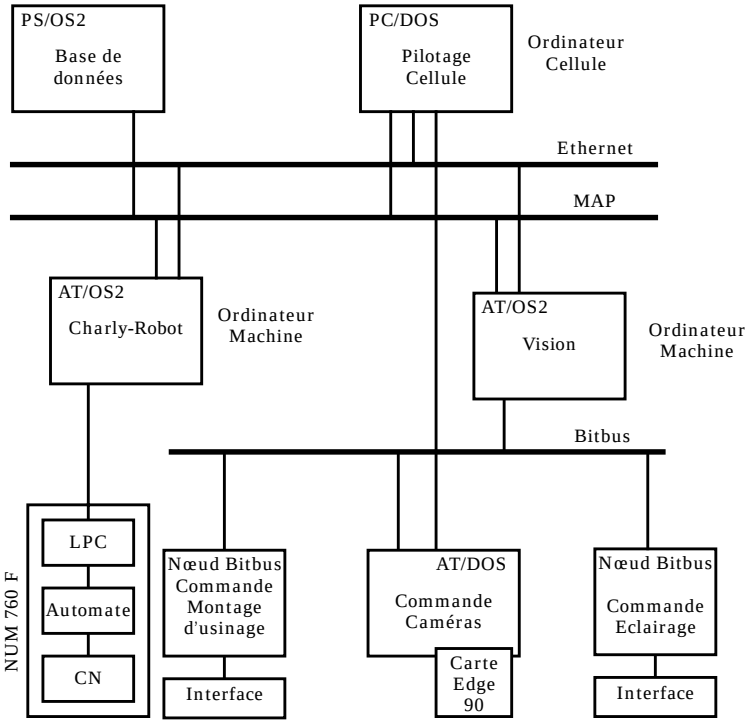


Figure 4 : un exemple de vue matérielle de l'architecture de conduite (extrait de [7])

L'écart existant entre ces deux vues de l'architecture de conduite illustre assez bien l'effort intellectuel nécessaire à l'ingénieriste pour en assurer la conception.

Les travaux décrits dans ce mémoire vont donc porter sur la conception de l'architecture des moyens matériels nécessaires au système de conduite, et ont pour ambition de contribuer à combler le fossé existant entre la vue conceptuelle d'une application et sa vue matérielle, résultat de la conception d'architecture.

1.1.3 Cycle de développement d'un système de production

Le développement d'un système est couramment décomposé en étapes, plus ou moins indépendantes et distinctes. Le domaine du génie logiciel y joua un rôle précurseur dans la standardisation et la normalisation. A titre d'exemple, la figure 5 présente un cycle de développement, appliqué aux systèmes microprogrammés, avec les normes associées [8]. Cet exemple, qui présente l'avantage de prendre en compte à la fois le logiciel et le matériel (cartes micro-programmées), reste très marqué par la culture informatique, et de ce fait s'applique bien aux systèmes dont l'architecture

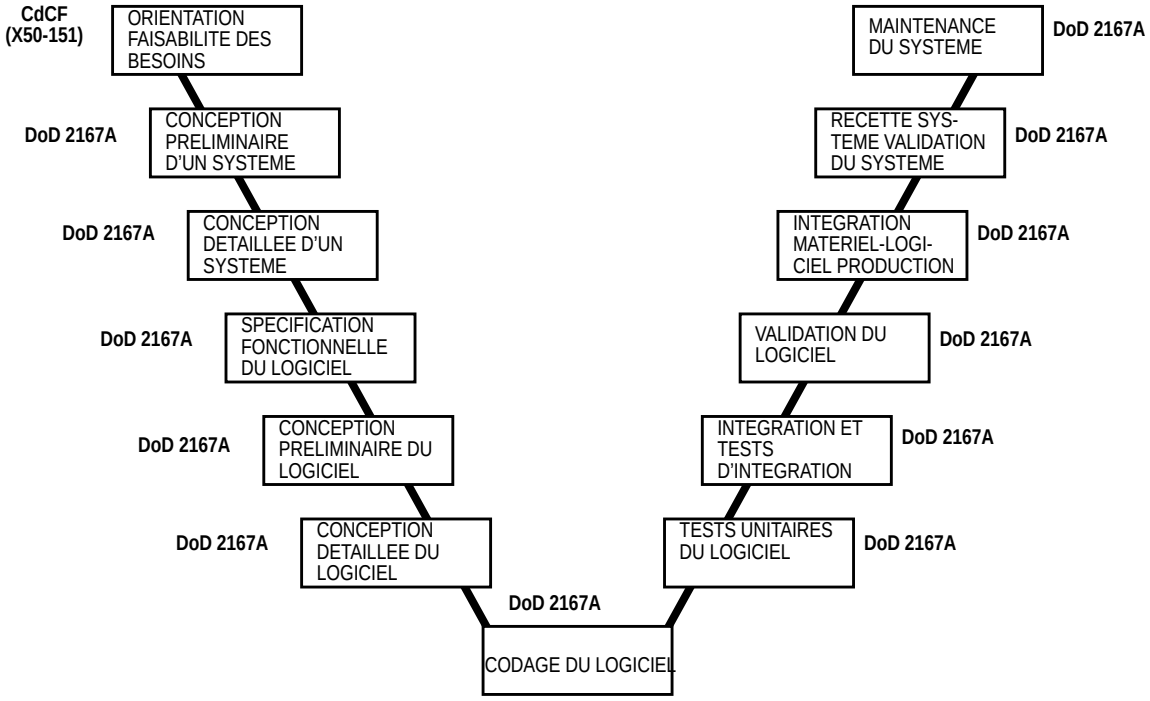


Figure 5 : cycle de développement d'un système (extrait de [8])

des équipements de commande est réduite, voire limitée à un seul équipement. La phase de conception de l'architecture de conduite n'y apparaît d'ailleurs même pas.

Des cycles de développement plus adaptés aux systèmes de production ont été élaborés. C'est par exemple le cas de la carte des activités de développement d'un système automatisé de production proposé par le CCGA[1] (fig. 6). On y trouve pris en compte à la fois les phases de développement du système opérant et de développement de sa commande. Celles-ci sont elles-mêmes décomposées en un développement de l'aspect logiciel et de l'aspect matériel. La conception de l'architecture de la conduite y est positionnée dans l'activité de «conception de la partie commande». Etant située en plein milieu du cycle de développement, elle ne peut encore une fois concerner que des architectures simples, car dans le cas contraire le choix de l'architecture est contractuel, donc nécessairement effectué en phase de «spécification et conception du SAP».

La conception de l'architecture des systèmes complexes que l'on considère dans notre étude prend donc naturellement sa place en amont des cycles de développement tra-

1. Centre Coopératif de Génie Automatique — ISMCM, 3 rue F. Hainaut, 93407 Saint Ouen cedex

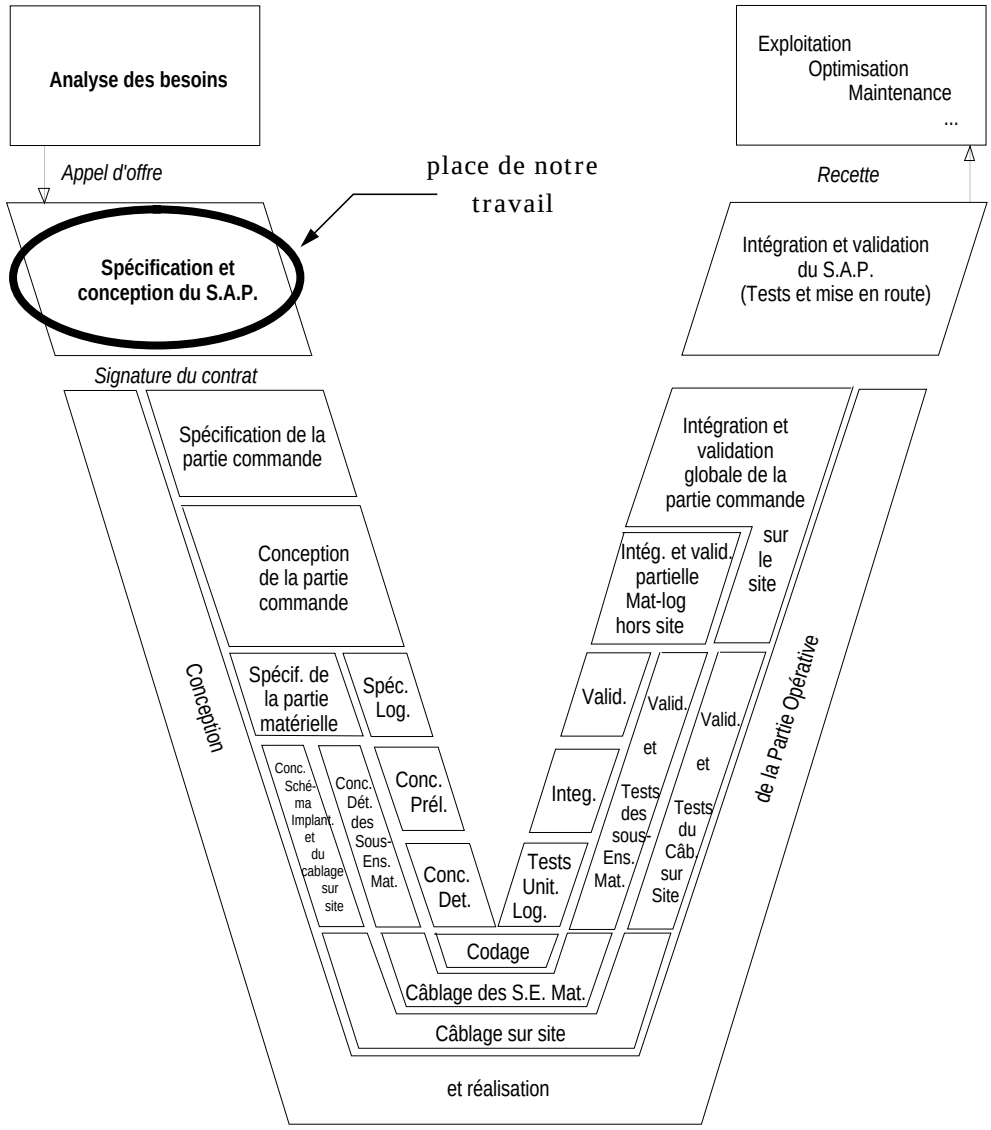


Figure 6 : carte des activités de développement d'un SAP [9]

ditionnels, c'est-à-dire en phase de «spécification et conception du SAP» si l'on reprend l'exemple du cycle de développement de la figure 6.

1.1.4 Les travaux sur la conception des systèmes de production et leur architecture de conduite

Ces dernières années de nombreux travaux ont contribué à l'essor du génie automatique dans le domaine de la structuration des systèmes de conduite. Certains d'entre eux ont tiré des enseignements du génie logiciel en utilisant et spécialisant ses outils aux besoins des SAP. La méthode SADT, par exemple, a été sujette à des extensions méthodologiques [10] [11] et temporelles [12] [13] pour l'étude des systèmes automa-

tisés de production. On peut également citer les environnements intégrés de production de logiciel [14] tels que PCTE (programme européen ESPRIT) [15] pour lequel le projet de norme Base-PTA est complémentaire dans le génie automatique [16].

D'autres travaux ont pour leur part contribué au développement d'outils de spécification, conception et simulation spécifiques au génie automatique et à la structuration de la commande. On peut citer à ce sujet les travaux ayant lieu :

- au Centre de Recherche en Automatique de Nancy (CRAN), au sein de l'Équipe Automatique et Commande Numérique (EACN) autour d'un ensemble intégré d'outils (ORCHIS-GRILLE-SPEX) [17] [18],
- au Laboratoire d'Automatique de Besançon (LAB) pour la conception des systèmes robotisés [19] [20] [21],
- au Laboratoire d'Automatique de Grenoble (LAG) au sein de l'équipe «conduite des systèmes de production» [22] [23],
- au Laboratoire d'Automatique et d'Informatique Industrielle de Lille (LAIL) autour des projets CASPAIM et CASPAIM2 [24] [25] [26] [27] [28],
- au Laboratoire d'Informatique de Robotique et de Micro-électronique de Montpellier (LIRMM) autour de la chaîne PIASTRE [29] [30], et du modèle ACSY [31] [32],
- au Laboratoire Universitaire de Recherche en Production Automatisée (LURPA) de l'ENS de Cachan, pour la formalisation des méthodes et des modèles de conception des systèmes de production [33] [34] [35].

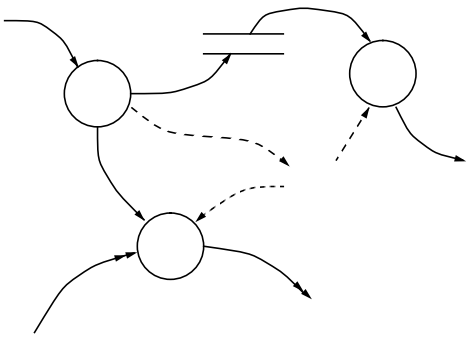
Si tous ces travaux contribuent à mieux structurer et outiller les activités de conception au moment où l'on a une vue conceptuelle du système, la prise en compte de l'implantation dans une architecture matérielle de conduite n'est généralement pas abordée. Lorsqu'elle est évoquée c'est seulement pour préciser sa place dans la démarche de conception proposée, et elle se trouve systématiquement placée en fin de démarche juste avant l'implantation logicielle. Cela reste parfaitement homogène avec le cycle de développement du CCGA présenté dans le paragraphe précédent. La conception de l'architecture des systèmes de grande ampleur se place donc en amont de ces travaux dont les résultats constituent des méthodes de développement de chaque sous-système issu de l'architecture.

D'autres travaux prennent plus spécifiquement en compte la notion d'architecture de conduite. Nous allons maintenant les analyser et mettre en évidence leurs idées fortes et leurs limites, afin de positionner l'apport de notre travail. Il s'agit :

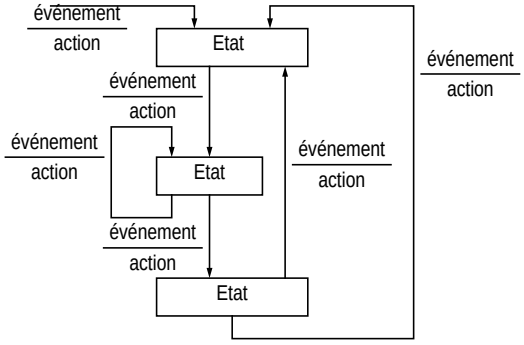
- de la méthode de développement des systèmes temps réel SART proposée par Paul T. Ward et Stephen J. Mellor [36],
- de la méthode de développement des systèmes temps réel SART proposée par Derek J. Hatley et Imtiaz A. Pirbhai [37],
- de la méthode de spécification et conception des systèmes électroniques MCSE proposée par J.P. Calvez [38] [39],
- des travaux de doctorat de J.L. Delcuvellerie [40].

1.1.4.1 SART selon Ward et Mellor

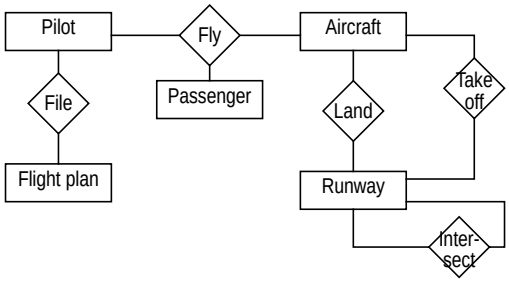
Dans leurs travaux, Ward et Mellor se sont intéressés au développement des systèmes temps réel embarqués. Leur approche comporte deux grandes étapes. La première consiste en l'élaboration d'un modèle de spécification appelé le *modèle essentiel*. La deuxième consiste en l'élaboration d'un modèle appelé le *modèle d'implantation* qui est la projection du modèle essentiel dans un environnement technologique. Le modèle essentiel (fig. 7) est basé sur l'analyse structurée (SA) de DeMarco [41], avec des enrichissements portant sur les flots de donnée qui peuvent être soit continus soit discrets, et sur l'apparition de flots de contrôle et de transformations de contrôle. Les transformations de contrôle peuvent ainsi contrôler l'activation, la désactivation et l'inhibition des transformations de donnée. De plus, les transformations de donnée, les transformations de contrôle et les données sont respectivement décrites par du pseudocode, des automates à état, et des diagrammes entité-relation. Une fois le modèle de spécification élaboré, l'étape suivante s'attache à le traduire dans le modèle d'implantation, c'est-à-dire à identifier les processeurs qui effectueront les transformations de donnée. Il s'agit d'une réorganisation des transformations comme le montre la figure 8. Aucun élément de choix n'est proposé pour créer l'architecture matérielle des processeurs qui supporteront l'application. Par contre, une fois l'architecture posée, quelques conseils généraux relatifs à la réorganisation des transformations sont prodigués. Mais ces conseils ne guident pas l'analyste dans son travail, ils lui présentent simplement les principales difficultés qu'il rencontrera.



Transformations de données et de contrôle



Description d'une transformation de contrôle



Description des données

```
Do Until (ENDFILE)
      Readsequential (PLANE) from (PLANES)
      if PALNE.PLANETYPE = INPUT.PLANETYPE
      then
            Do Until (ENDASSOCIATONS)
                  Findassociated (LANDING) for (PLANE)
                  if LANDING.WEATHERCONDITION =
                  INPUT.WEATHERCONDITION
                  then
                        Findassociated (AIRPORT) for (LANDING)
                        if AIRPORT.AIRPORTCLASSIFICATION =
                        INPUT.AIRPORTCLASSIFICATION
                        then
                              put PLANEIDNUMBER, OWNERSNAME
                              AIRPORTNAME, TIME into LANDINGREPORT
Write (LANDINGREPORT)
```

Description d'une transformation de donnée

Figure 7 : modèle essentiel de SART selon Ward et Mellor

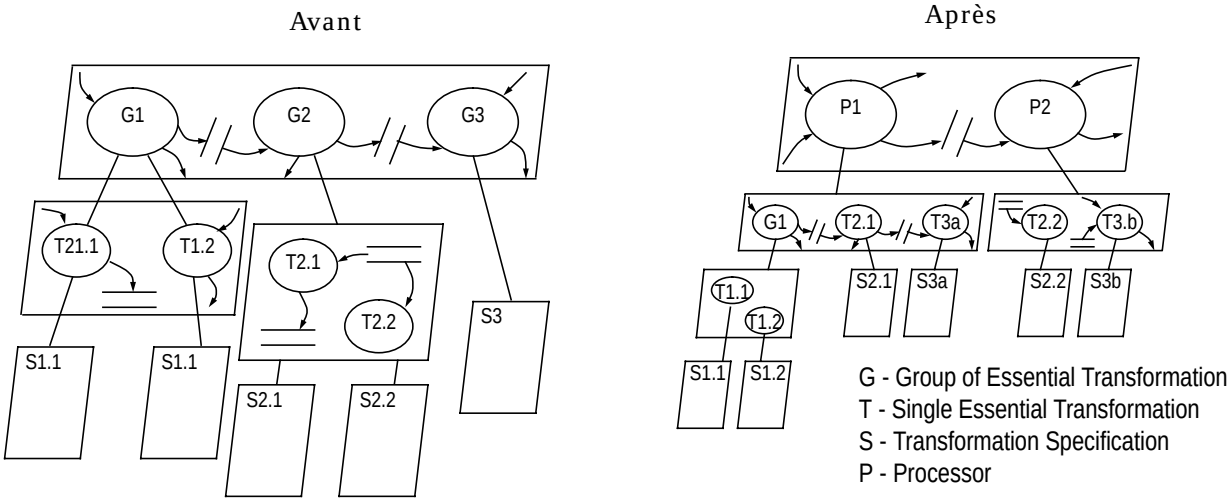


Figure 8 : réorganisation des transformations dans des processeurs

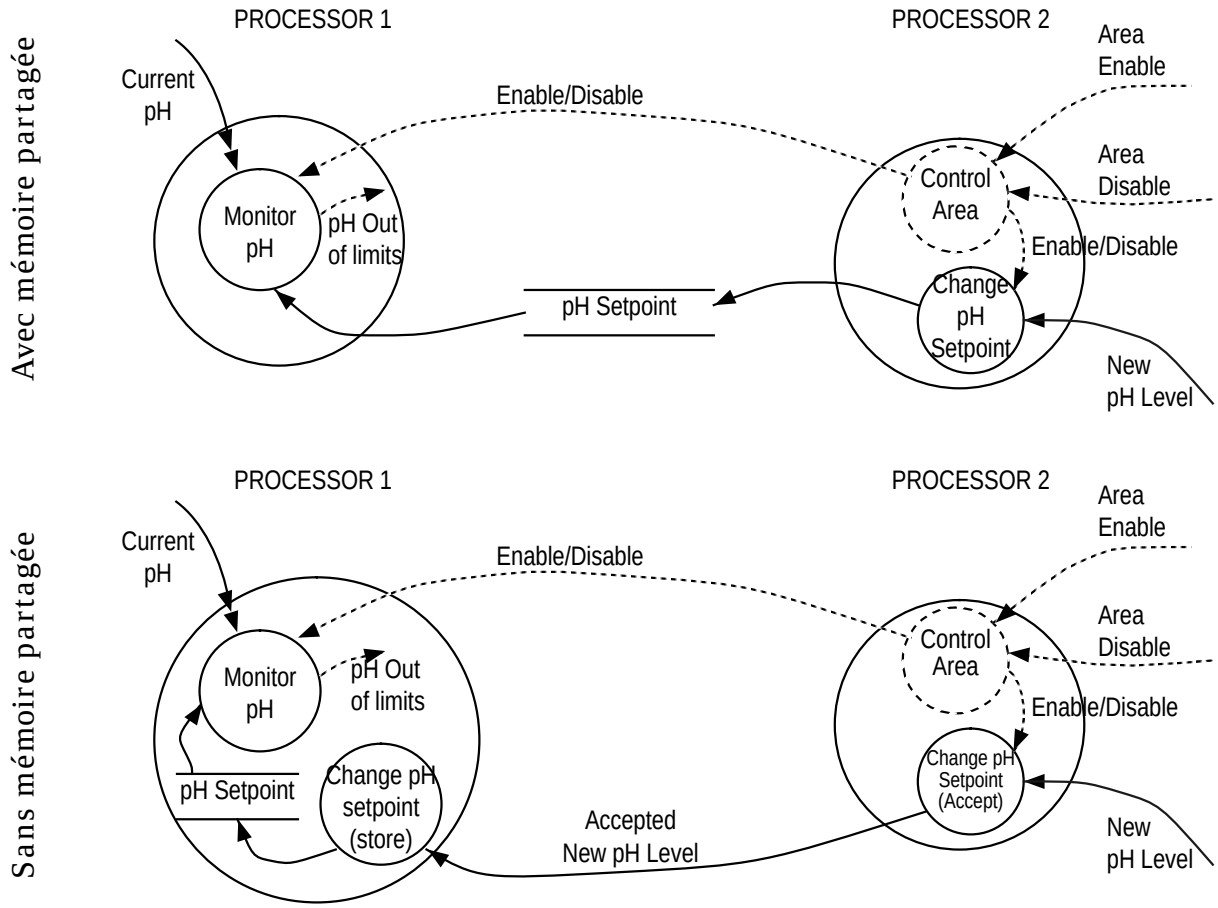


Figure 9 : problème de l'allocation du modèle essentiel dans le modèle d'implantation

La figure 9 montre ainsi que le même modèle essentiel peut se traduire en deux modèles d'implantation, selon que l'on dispose, ou pas, de mémoire partagée entre les processeurs. On remarquera sur cet exemple l'apparition d'une nouvelle transformation «Change pH setpoint (store)» et d'un nouveau flot de donnée «Accept New pH Setpoint» dans le cas où l'on n'utilise pas de mémoire partagée. Cette transformation et ce flot de donnée ne sont pas fonctionnels mais le résultat d'un choix technique d'implantation. Malheureusement, les auteurs ne donnent pas de cadre méthodologique pour élaborer et choisir des solutions d'implantation, mais se contentent de proscrire ce type de solutions.

On pourra également regretter dans cette méthode le manque de pertinence dans le choix du formalisme graphique de représentation de l'implantation, car c'est le même que celui du modèle essentiel. Cependant, une fois cette phase d'allocation des transformations dans des processeurs effectuée, les auteurs proposent une méthode très détaillée pour assurer la conception des programmes pour chaque processeur.

En conclusion, si la phase de conception de l'architecture est abordée dans cette méthode, elle n'a de sens que pour des architectures simples (un à trois processeurs) fortement connectées (tout processeur peut échanger des données avec tout autre processeur, comme dans un rack avec bus intégré) car les notions de canaux de communication ou de réseaux de communication entre équipements de commande ne sont pas évoquées.

1.1.4.2 SART selon Hatley et Pirbhai

De la même façon, les travaux de Hatley et Pirbhai portent sur le développement des systèmes embarqués temps réel (à l'origine les systèmes d'avionique). L'approche comporte deux grandes étapes. La première consiste en l'élaboration d'un modèle de spécification appelé le *modèle des besoins*. La seconde consiste en l'élaboration d'un modèle appelé le *modèle d'architecture* qui montre les entités physiques qui constituent les systèmes, qui définit les flots d'information entre ces entités physiques, et qui précise les canaux sur lesquels passe l'information.

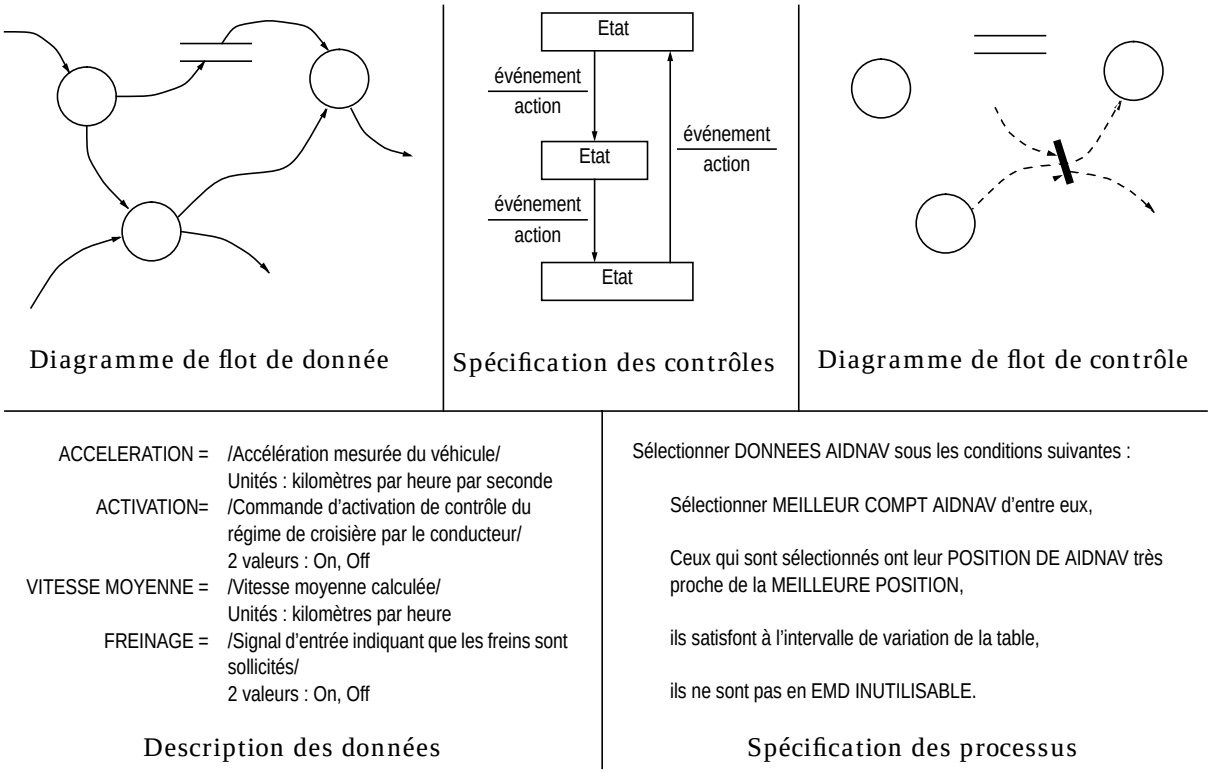


Figure 10 : modèle des besoins de SART selon Hatley et Pirbhai

Le modèle des besoins (fig. 10) est construit sur une base de SA, avec des enrichissements concernant les flots de contrôle. Ces derniers convergent vers une «barre», porte d'entrée à un modèle de spécification des contrôles établi selon le formalisme des automates finis. Lorsque le modèle des besoins est établi, l'analyste se porte sur l'élaboration du modèle d'architecture. A cette fin, les auteurs proposent un modèle de référence appelé *canevas* (fig. 11) du modèle d'architecture, dont le rôle est de

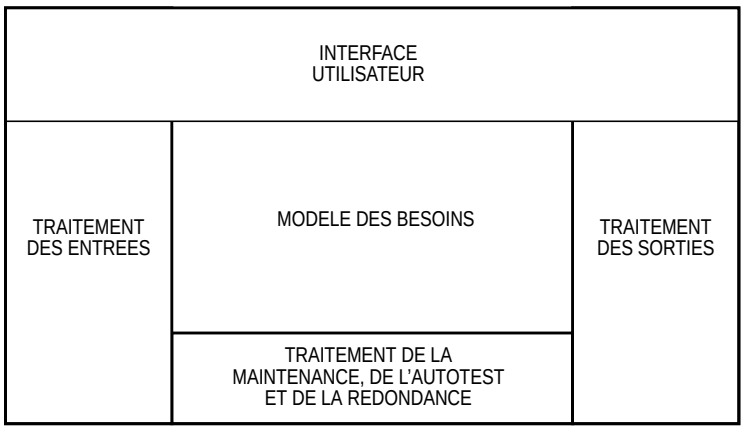


Figure 11 : canevas du modèle d'architecture

mettre en évidence les modules d'architecture inhérents aux systèmes temps réel embarqués. On met ainsi en évidence des modules relatifs au traitement des entrées, au traitement des sorties, à l'interface opérateur, et à la gestion du traitement de la maintenance, des auto-tests et de la redondance.

Afin d'assister l'intégration du modèle des besoins dans une architecture, Hatley et Pirbhai proposent une méthode à deux phases. Dans un premier temps, on ne s'intéresse qu'à l'allocation des processus et des flots du modèle des besoins dans des modules d'architecture. On obtient ainsi le *diagramme de flot d'architecture* qui est une représentation de la configuration physique du système. Les modules d'architecture ont leur propre symbole de représentation, et la communication entre eux est représentée par le symbole d'arc de flot d'information. A ce niveau de la méthode, les flots représentés entre les modules ne donnent qu'une vue logique des échanges inter-modules. Dans un deuxième temps, on réalise le *diagramme d'interconnexion d'architecture* où cette fois-ci les canaux de communication entre modules apparais-

sent selon la représentation graphique rapportée figure 12. La figure 13 présente un exemple de chaque diagramme d'architecture à titre d'illustration.

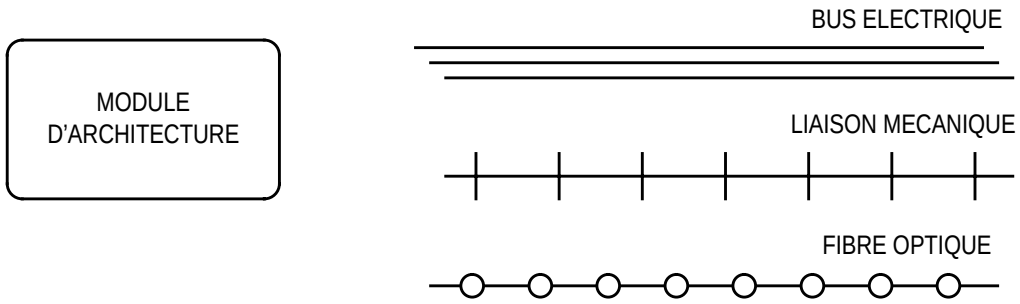


Figure 12 : Symboles graphiques du diagramme d'interconnexion d'architecture

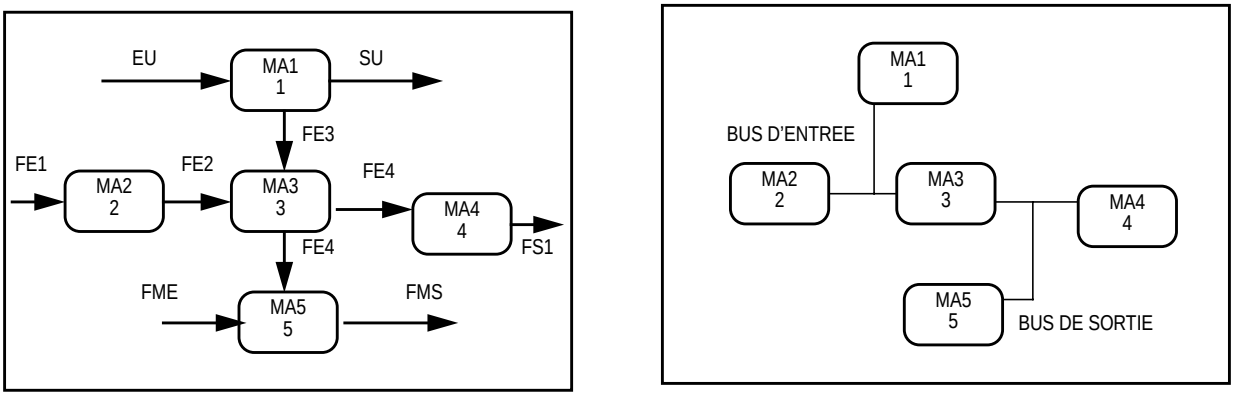


Diagramme de flot d'architecture

Diagramme d'interconnexion d'architecture

Figure 13 : les modèles de l'architecture selon Hatley et Pirbhai

La prise en compte de l'architecture dans la méthode SART de Hatley et Pirbhai est donc un peu plus poussée qu'elle ne l'est dans la version Ward et Mellor. D'une part un formalisme graphique clair et spécifique a été mis au point prenant en compte à la fois les processeurs (modules d'architecture) et les canaux de communication. D'autre part, la présence d'un modèle de référence (canevas) permet aux concepteurs de systèmes embarqués d'adapter et de compléter facilement leur modèle des besoins. Bien entendu, l'utilisation du canevas en dehors du contexte pour lequel il a été conçu est inefficace, mais il s'agit d'un corollaire à toute utilisation d'un modèle de référence. Malgré ces points positifs pour le sujet qui nous préoccupe, la conception de l'architecture est le point ultime dans le déroulement de cette méthode. De

plus, il n'est pas expliqué comment obtenir la spécification de chaque module d'architecture à partir du modèle des besoins et du modèle d'architecture réunis.

1.1.4.3 La méthode MCSE

A l'instar des méthodes SART, la méthode MCSE de J.P. Calvez insère la conception d'architecture dans une démarche globale. Les systèmes visés intègrent à la fois les techniques de l'électronique et de l'informatique. Les exemples d'application cités par l'auteur sont : un système de commande d'un centrifugeur, la gestion technique centralisée d'un grand bâtiment, la surveillance de chaînes d'ancrage pour plates-formes pétrolières.

MCSE appréhende les systèmes selon trois points de vue complémentaires (fig. 14) : une vue fonctionnelle qui est décrite par un *modèle fonctionnel* au formalisme spécifique, une vue temporelle qui décrit le comportement des constituants fonctionnels à l'aide d'un formalisme spécifique dérivé des automates finis appelé *modèle comportemental*, et une vue matérielle qui exprime les constituants matériels et leurs interconnexions nécessaires pour le fonctionnement du système (*modèle d'exécution*). L'intégration du modèle fonctionnel dans les constituants du modèle d'exécution est particulièrement bien formalisée, et fait l'objet d'un modèle spécifique appelé *modèle d'allocation*, qui assure la mise en correspondance des éléments des deux modèles selon les règles d'association présentées dans le tableau 1. On remarquera que le

Tableau 1 : correspondance entre le modèle fonctionnel et le modèle d'exécution

Eléments du modèle fonctionnel		Eléments du modèle d'exécution
Structure fonctionnelle partielle (SF)	->	processeur (P)
variable (VE)	->	mémoire (M)
événement (EV)	->	signalisation (S)
port (PT)	->	nœud de communication (N)
lien: fonction <-->EV,VE,PT	->	lien: processeur <-->S,M,N

modèle d'allocation associe un lien fonctionnel à un lien processeur, ce qui nécessite une grande connectivité entre les processeurs (chaque couple de processeurs devant échanger des données doit être connecté par un lien processeur), interdisant l'utilisation d'un répéteur, d'un pont ou d'une passerelle de communication. De ce fait, le

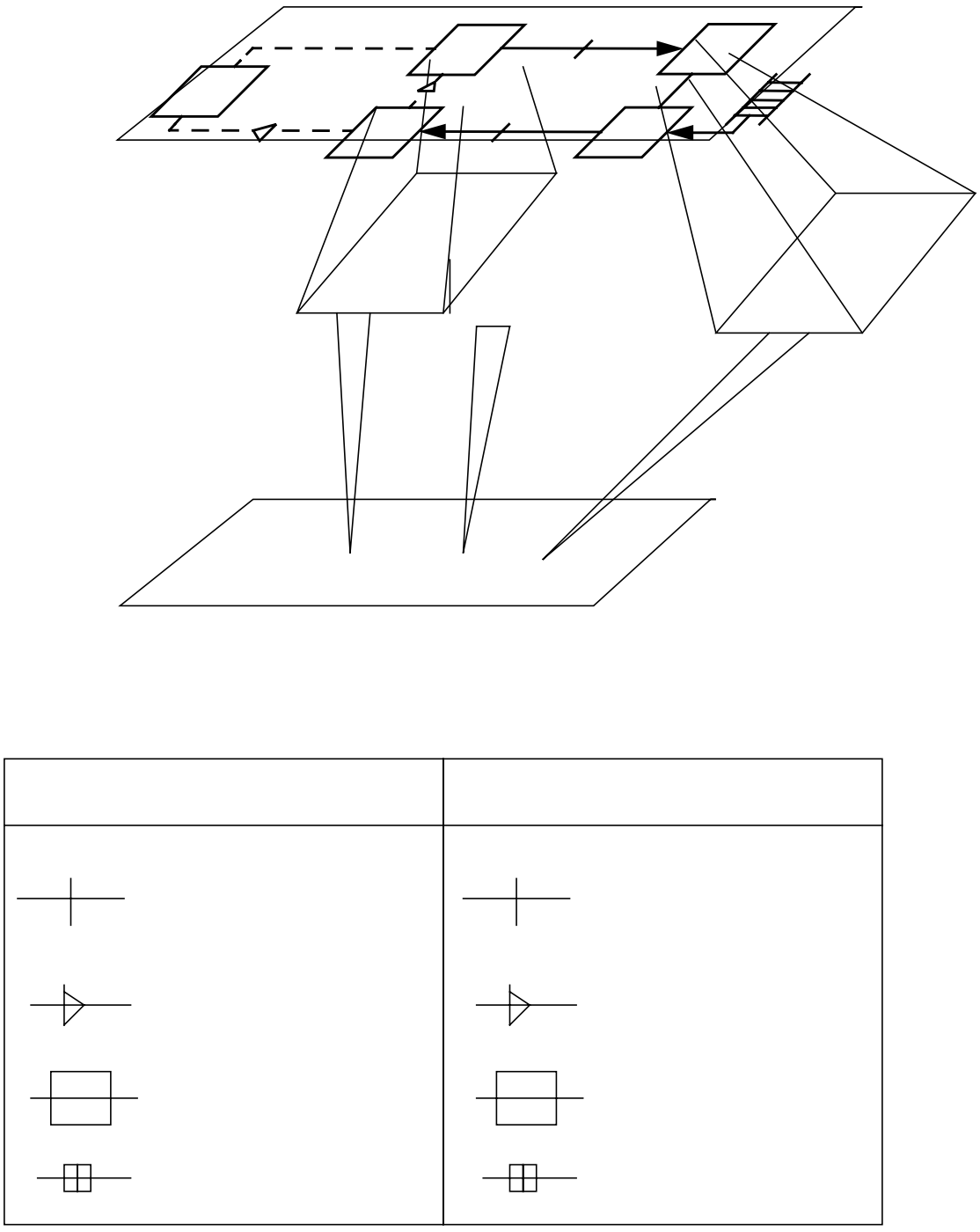


Figure 14 : le modèle à trois dimensions de la méthodologie MCSE

problème de l'apparition de nouvelles fonctions introduites par la répartition du modèle fonctionnel dans le modèle d'exécution ne se pose pas. Par contre, lors de l'implantation d'une partie du modèle fonctionnel dans un processeur, la méthode précise comment de nouvelles fonctions apparaissent selon la présence ou non d'un exécutif temps réel dans le processeur. Malheureusement cela échappe de la conception de l'architecture proprement dite.

Il reste un point fort de la méthode MSCE à aborder, celui de la validation de l'architecture grâce à deux règles. La première, de nature quantitative, vérifie que le taux d'occupation maximum des processeurs reste inférieur à un, en utilisant des données prévisionnelles estimées. La seconde, de type check-list, propose de vérifier la compatibilité avec les durées d'exécution maximales, les dates de fin, et les temps de réaction sur événement externe. On regrettera cependant que les formalismes graphiques soient tous spécifiques, même pour les modèles fonctionnels ou dynamiques pour lesquels plusieurs formalismes sont déjà largement répandus. De plus l'utilisation d'un formalisme quasi identique pour la structure fonctionnelle et la structure d'exécution (fig. 14) peut conduire l'analyste à des confusions (de même que pour la méthode SART de Ward et Mellor)

En conclusion, si la méthode MCSE met en valeur la conception d'architecture dans la démarche, c'est également sans guide méthodologique et au prix d'une grande spécificité, c'est-à-dire pour des architectures de conduite très spécifiques.

1.1.4.4 Les travaux de doctorat de J.L. Delcuvellerie

Les travaux de doctorat de J.L. Delcuvellerie ont une place particulière dans les méthodes de développement qui viennent d'être présentées car ils sont exclusivement consacrés aux architectures de conduite des systèmes de production et ont les mêmes préoccupations industrielles que dans ce mémoire.

Dans le cadre de sa thèse de doctorat, J.L. Delcuvellerie propose une revue des connaissances manipulées par l'ingénieriste lors de la conception des architectures de «systèmes informatisés d'automatisation». L'aspect spécification de ces systèmes y est vu au travers d'un modèle générique. Ce dernier est basé sur la notion de service pour assurer la structuration fonctionnelle, et sur la notion d'entité pour modéliser le processus de production (fig. 15). L'aspect organique du système de conduite, appelé *architecture opérationnelle,* est ensuite décrit dans un modèle constitué de dispositifs de traitement et de systèmes de communication (réseau opérationnel et liaison opérationnelle) (fig. 16). L'implication d'une structure fonctionnelle dans une architecture est présentée comme une relation entre *une* structure fonctionnelle et *une à n* architectures opérationnelles.

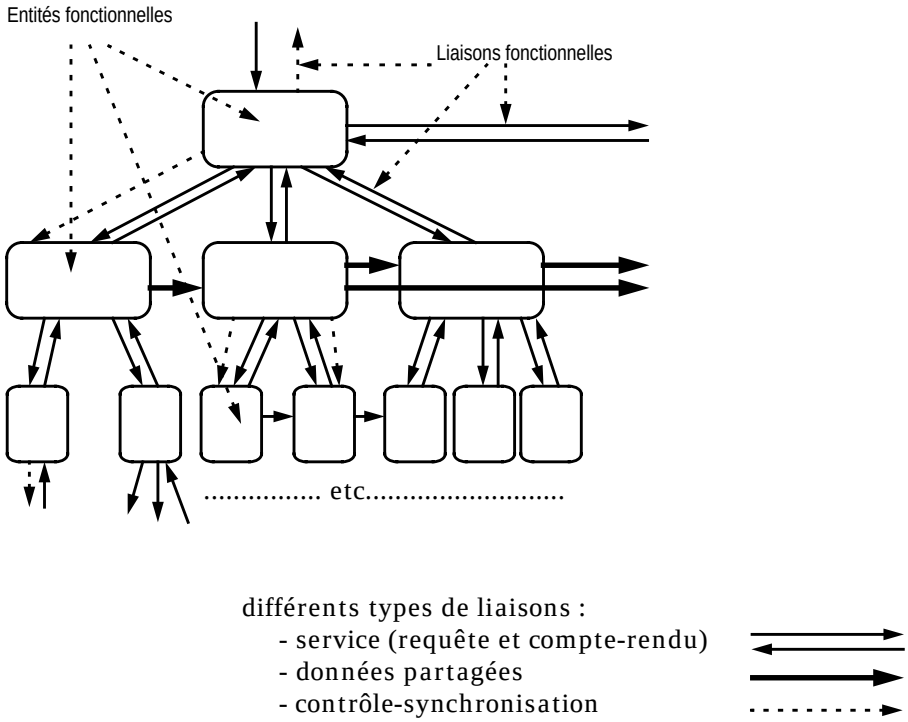


Figure 15 : les éléments du modèle générique de structuration fonctionnelle

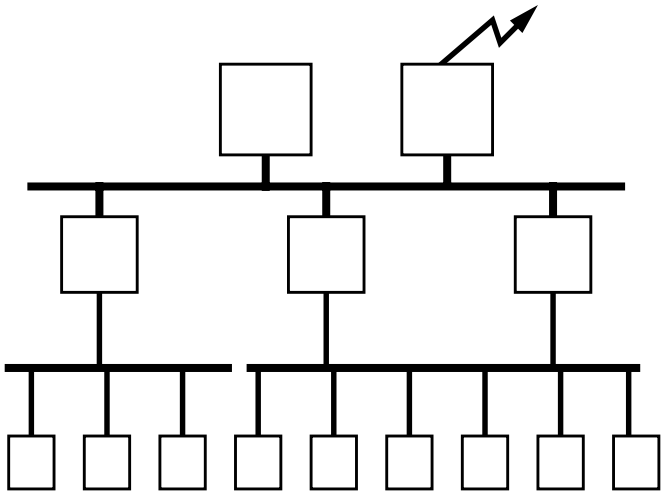


Figure 16 : l'architecture opérationnelle d'un système automatisé de production

Bien que très élémentaire, le modèle générique de la structuration fonctionnelle fournit des modèles d'une granularité jugée suffisante. Il a de surcroît permis le développement d'un outil d'analyse qualitative de la robustesse des architectures aux défaillances. Cet outil permet, à partir d'un défaut simulé sur un équipement de

l'architecture opérationnelle, de visualiser sur la structure fonctionnelle les entités en défaut (soit directement soit par propagation du défaut initial).

Cependant, les fonctions techniques[1] issues des choix de conception de l'architecture ne sont pas étudiées, alors que les architectures envisagées dans ces travaux nécessitent généralement des fonctions de routage de données, de partage de ressources telle qu'une base de données, etc...

1.1.4.5 Conclusions sur la revue bibliographique

Les travaux évoqués ci-dessus proposent systématiquement une démarche globale de conception du système, incluant à la fois la conception de la conduite du système lui-même et celle de son architecture. D'où :

- une confusion entre les concepts propres à la conception de l'architecture et ceux de la conception du système,
- une absence de guide de la conception d'architecture,
- une limitation du domaine d'application, car une méthode est toujours destinée à résoudre une gamme précise de problèmes.

La conception d'architecture est vue comme un point de passage obligé dont les méthodes tiennent compte, mais pour lequel elles ne procurent aucune assistance à la conception. D'autre part les fonctions techniques résultant d'un choix d'architecture ne sont pas prises en compte, ou bien elles sont éludées par préconisation de solution d'architecture ne nécessitant pas de fonction technique. Quant aux évaluations des solutions d'architecture, seul J.L. Delcuvellerie propose un outil d'analyse qualitatif de la robustesse aux défaillances.

L'ingénieriste se trouve donc face à des méthodes dont il ne peut pas extraire uniquement la partie conception de l'architecture. S'il souhaite utiliser l'une de ces méthodes en conception d'architecture de conduite, il devra adopter, puis appliquer, la méthode choisie *dans son ensemble.*

---

1. Le terme «fonction technique» est ici employé dans son acception retenue dans les documents normatifs consacrés à l'analyse de la valeur [42]. Les «fonctions techniques» sont alors à opposer aux «fonctions de service».

1.1.5 L'apport de ce travail

Ce mémoire va s'attacher à développer les *concepts intrinsèques à la conception d'architecture*, indépendamment de la méthode de spécification utilisée en amont par l'ingénieriste. On y proposera :

- un cadre méthodologique pour guider l'analyste spécifiquement durant l'activité de conception d'architecture,
- des modèles et des méthodes qui ne préjugent pas de la spécification amont, ni de la méthode de développement aval du système, car il ne s'agit pas de proposer une nouvelle méthode de développement des systèmes de production,
- une représentation formelle évaluable des architectures conçues,
- des techniques fortes d'intégration favorisant l'insertion de la conception de l'architecture dans toutes les méthodes de développement telles qu'elles sont proposées dans les travaux précédemment cités, pour assurer une *portabilité* au sens large du terme :
  - portabilité du guide de conception d'architecture dans les méthodes de développement de systèmes, grâce aux concepts de *méthode intégrée*,
  - portabilité à différents domaines d'application par l'intermédiaire de *bibliothèques métier*,
  - portabilité au sens informatique, c'est-à-dire interconnexion aux ateliers de génie automatique existants grâce à la *métamodélisation*.

1.2 Présentation générale de la méthode

Ce chapitre a pour but de donner un aperçu global de la démarche que nous proposons pour concevoir les architectures de conduite des systèmes de production. Le lecteur y trouvera un positionnement de notre travail au sein de la conception globale d'un système, une description de ses entrées/sorties ainsi que de ses principales activités. Le chapitre 1.3 «Construction d'une solution d'architecture» et le chapitre 1.4 «Evaluation et choix d'architecture» reprendront chacune de ces activités en détaillant les modèles qui y sont construits.

Notre propos sera illustré en permanence par des schémas dont les vertus de synthèse et de concision sont largement reconnues. Nous utiliserons à cet effet des actigrammes SADT pour deux raisons essentielles :

- une parfaite adéquation avec les objectifs de ce chapitre,
- une utilisation suffisamment répandue pour que nous puissions considérer SADT comme faisant partie de la culture du «génie automaticien».

Un avertissement au lecteur doit cependant être formulé sur l'emploi des actigrammes qui jouent ici un simple rôle d'illustration. Ils seront utilisés au fil du texte et donc extraits du modèle SADT qui leur donne une cohérence globale. Ce modèle est reproduit in extenso dans l'Annexe A.

1.2.1 Les objectifs

Partant de l'idée qu'il faut étudier la conception de l'architecture indépendamment des méthodes de développement des systèmes de production, la figure 17 montre le contexte dans lequel s'organise la conception de l'architecture autour du travail de l'ingénieriste.

Une fois les spécifications du système constituées, selon les habitudes de la société d'ingénierie et avec ses formalismes, la première étape consiste en l'extraction de toutes les informations contenues dans le cahier des charges fonctionnel et nécessaires à la conception de l'architecture. Cette opération effectuée, dans une deuxième

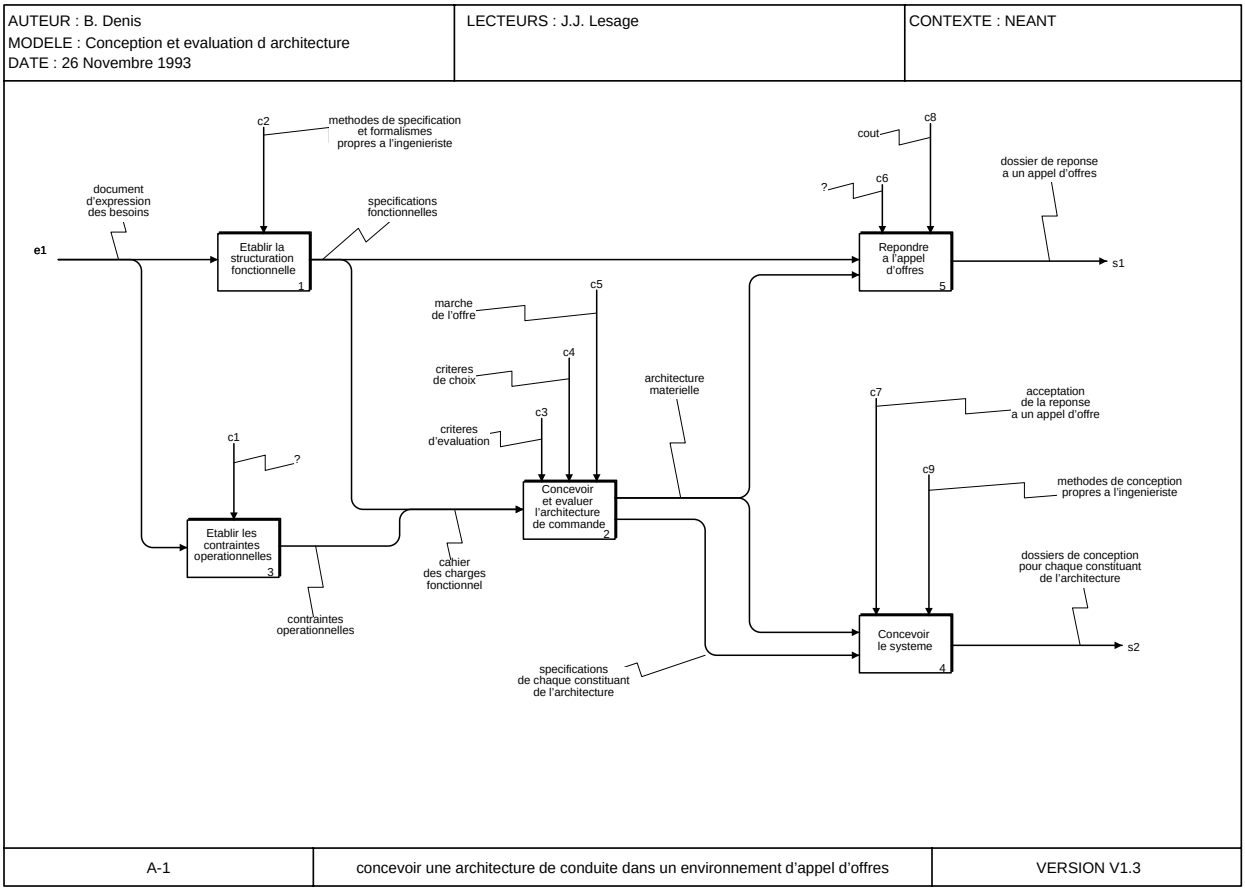


Figure 17 : position de l'apport des travaux par rapport au travail de l'ingénieriste

étape, l'architecte automaticien[1] peut travailler de manière autonome sur le projet en suivant la démarche, et en utilisant les outils propres à la conception d'architecture que nous lui proposons. La troisième et dernière étape consiste à rapatrier les résultats du travail de l'architecte pour que l'ingénieriste puisse dérouler la suite du projet. Les spécifications de chaque constituant matériel, établies à partir des spécifications de l'ensemble du système et de la solution d'architecture retenue, lui permettent alors de continuer le cycle de développement du projet.

La démarche que nous proposons s'effectue par conséquent à partir des spécifications du système établies selon les méthodes de l'ingénieriste et des critères d'évaluation et de choix propre à l'architecte. On voit par là que, non seulement l'ingénieriste est libre dans sa démarche de spécification, mais également que

1. L'ingénieriste et l'architecte sont souvent dans la pratique une seule et même personne, ou une seule et même équipe projet. La distinction faite ici, a pour but de mettre en valeur le rôle joué par les protagonistes dans les différentes étapes du développement d'un système.

l'architecte reste maître de ses critères d'évaluation et de ses critères de choix des solutions d'architecture. En fin de déroulement de notre démarche, l'architecte est capable de fournir à l'ingénieriste à la fois une architecture matérielle, comme celle représentée figure 4, et une spécification pour développer la commande de chaque constituant de l'architecture matérielle (fig. 18).

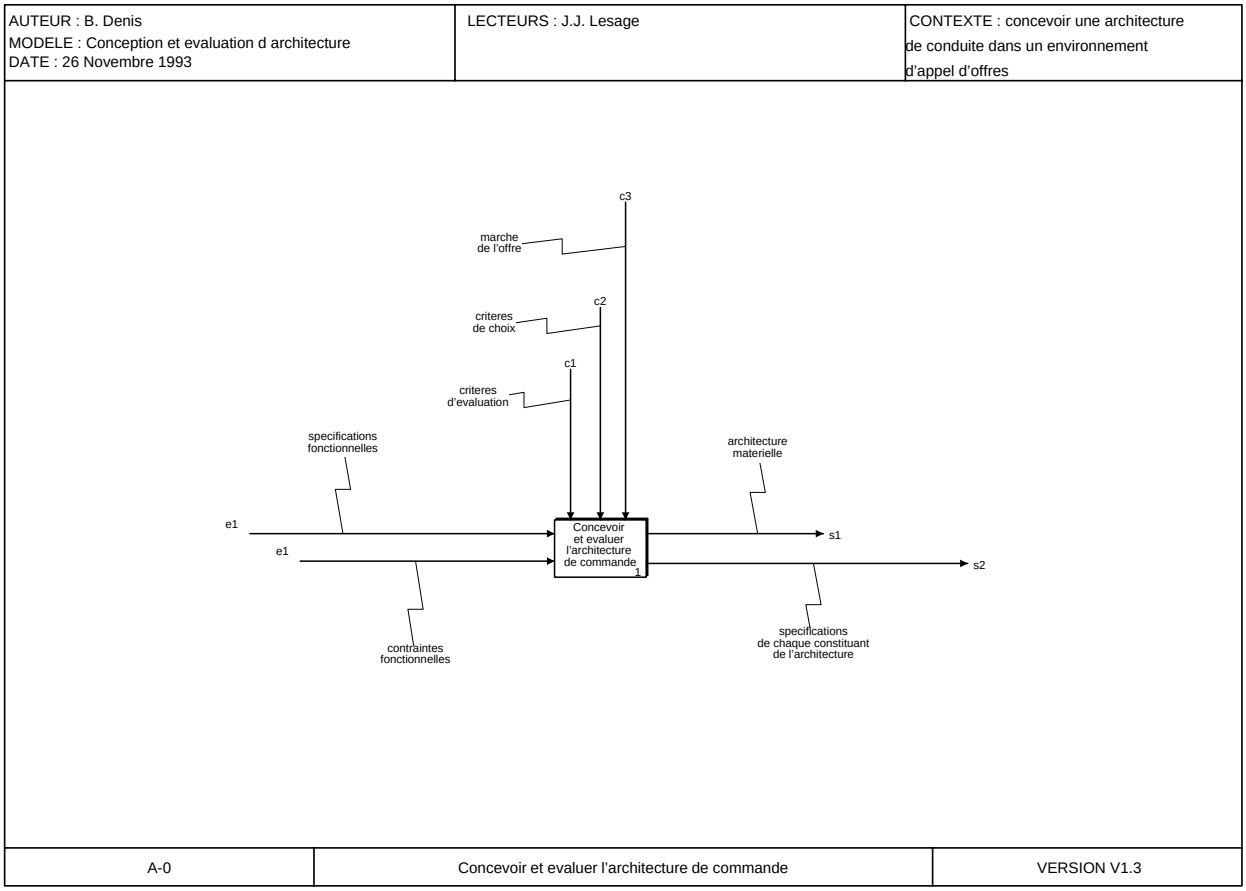


Figure 18 : les entrées-sorties de notre travail

1.2.2 Description générale de la démarche

Le cœur de l'architecture d'un système de production c'est ce qui permet de répondre à la question *«qui fait quoi ?»* ou plutôt *«qui fait quoi et en s'organisant comment ?»*. Cela nous a amené à développer trois techniques de modélisation respectivement aptes à réaliser des modèles selon trois points de vue différents de l'architecture. Le *quoi* sera décrit dans un modèle dit *modèle d'implantation* où l'on retrouve entre autre tous les traitements issus des besoins fonctionnels qui ont pour vocation d'être exécutés par un constituant de l'architecture matérielle. Le *qui* sera décrit dans un

modèle dit *modèle organique* où l'on retrouve interconnectés entre eux les constituants de l'architecture tels que les automates programmables, les calculateurs industriels, ou encore les réseaux d'atelier. Le «qui fait quoi et en s'organisant comment ?», montrant l'organisation entre les éléments du modèle d'implantation et les éléments du modèle organique sera décrit dans un modèle dit *modèle d'affectation*.

Les activités «Elaborer un modèle d'implantation», «Elaborer un modèle organique» et «Elaborer un modèle d'affectation» produisent respectivement le modèle d'implantation, le modèle organique et le modèle d'affectation. Ces trois modèles représentent la première partie de notre contribution à la formalisation de l'architecture, et les activités d'élaboration de ces modèles jouent un rôle central dans la démarche. Autour de ce cœur, deux activités de la conception d'architecture ont également fait l'objet de travaux rapportés dans ce mémoire : «Evaluer une solution d'architecture» qui constitue la seconde partie de notre contribution à la formalisation de l'architecture, et «Choisir une solution d'architecture» préalablement évaluée. La figure 19 présente l'ensemble de ces activités et les interconnexions entre leurs entrées-sorties.

1.2.3 Description du processus de construction d'architecture

1.2.3.1 Elaborer un modèle d'implantation

Elaborer un modèle d'implantation consiste à extraire les fonctions et les données qui nécessitent une implantation dans l'architecture matérielle de conduite, ainsi que toute information relative à l'implantation ou à la synchronisation des fonctions les unes par rapport aux autres, et ce, à partir des modèles établis par l'ingénieriste lors de la spécification du système.

Le modèle produit en sortie de l'activité de conception du modèle d'implantation regroupe un ensemble de traitements nécessaires à l'exécution des fonctions de la spécification, et un ensemble de stockages nécessaires à la mémorisation des données. La synchronisation des traitements y est décrite en terme de règles de déclenchement, ou par une modélisation plus fine de la cinématique de chaque traitement.

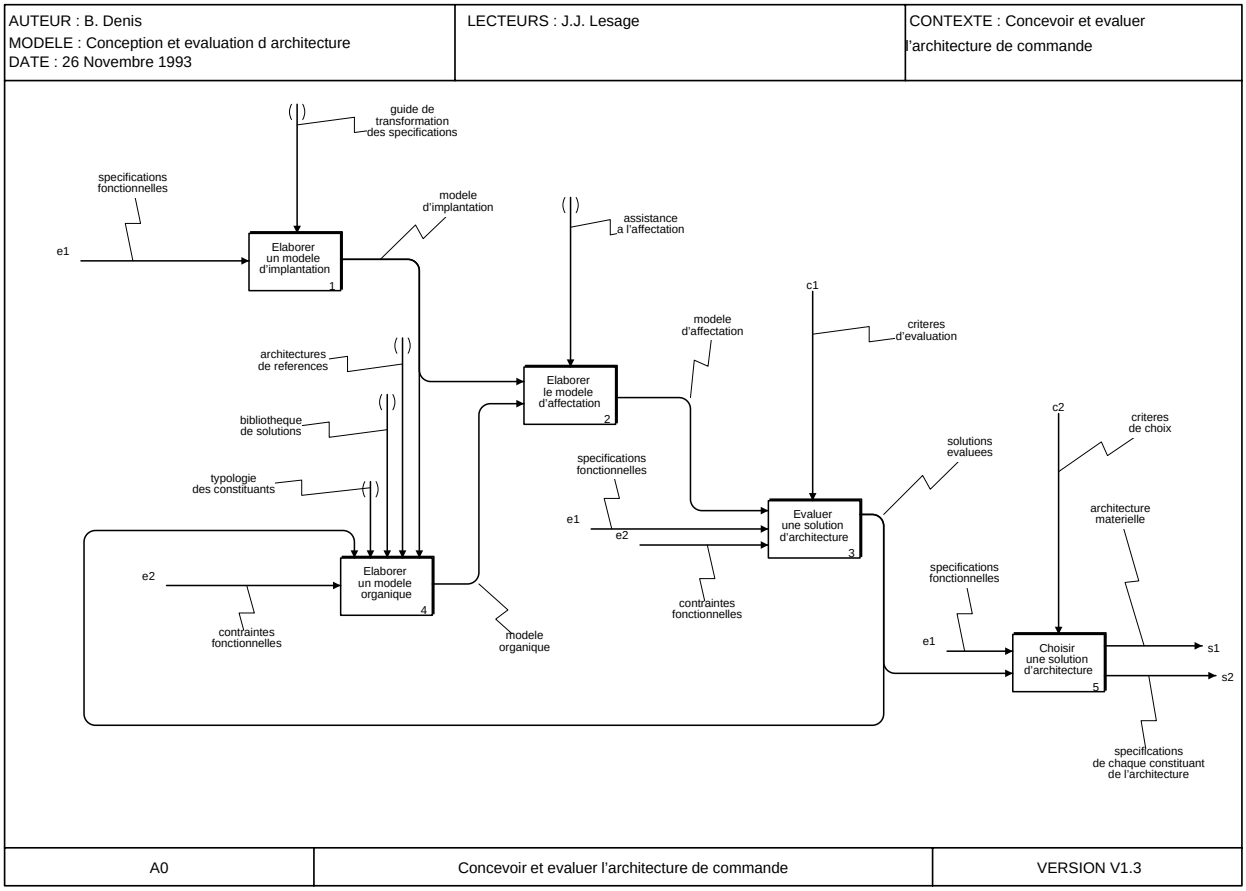


Figure 19 : vue d'ensemble de la démarche

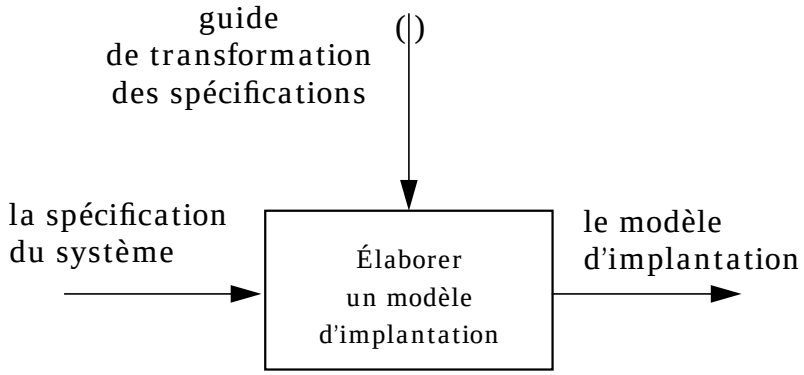


Figure 20 : les entrées-sorties de l'activité «Elaborer un modèle d'implantation»

Le modèle d'implantation doit assurer les fonctions suivantes :

- regrouper et intégrer en un seul et unique formalisme les seules informations pertinentes pour la conception d'architecture. Par exemple, toute fonction exprimée dans la spécification du système de conduite, devant être supportée

par un composant logiciel doit être représentée dans le modèle d'implantation. De même, toute donnée devant être stockée ou devant circuler devra y être représentée ;

- préciser à quels moments les fonctions de conduite doivent être déclenchées. Par exemple, si la spécification décrit une fonction qui consomme des données afin d'en produire de nouvelles, le modèle d'implantation devra préciser quelles conjonctions d'événements sont nécessaires à l'accomplissement de la fonction (c'est-à-dire *«quand»* la fonction doit-elle être réalisée ?) ;
- préciser si les données exprimées dans la spécification sont fugaces ou pérennes. En d'autres termes, leur durée de vie est-elle limitée au temps de transfert dans le canal de communication qui relie la fonction productrice à la fonction consommatrice, ou est-il nécessaire de prévoir un espace de stockage assurant la pérennité de l'information ?

Le modèle d'implantation est alors construit *comme si l'architecture était réduite à un seul et unique constituant* (un calculateur industriel par exemple). Il constitue à ce titre un modèle transitoire entre les modèles de spécification du système, élaborés sans aucune référence à l'architecture, et le modèle d'affectation qui prend en compte à la fois les contraintes dues à l'exécution dans des constituants microprogrammés, et les contraintes dues à la répartition dans plusieurs constituants.

D'autre part, si la spécification du système est complète, le modèle d'implantation est unique car il ne dépend en rien des futurs choix d'architecture. Il devient de ce fait, la matière d'œuvre de l'architecte. Cependant, l'hypothèse de complétude de la spécification vis-à-vis du travail de l'architecte, est rarement vérifiée. Le modèle d'implantation est donc également nécessaire pour compléter, si besoin est, la description du système.

Afin d'illustrer notre propos, la figure 21 présente une vue d'un exemple de modèle d'implantation dont la syntaxe et les règles de construction sont développées dans la division 1.3.1 «Le modèle d'implantation».

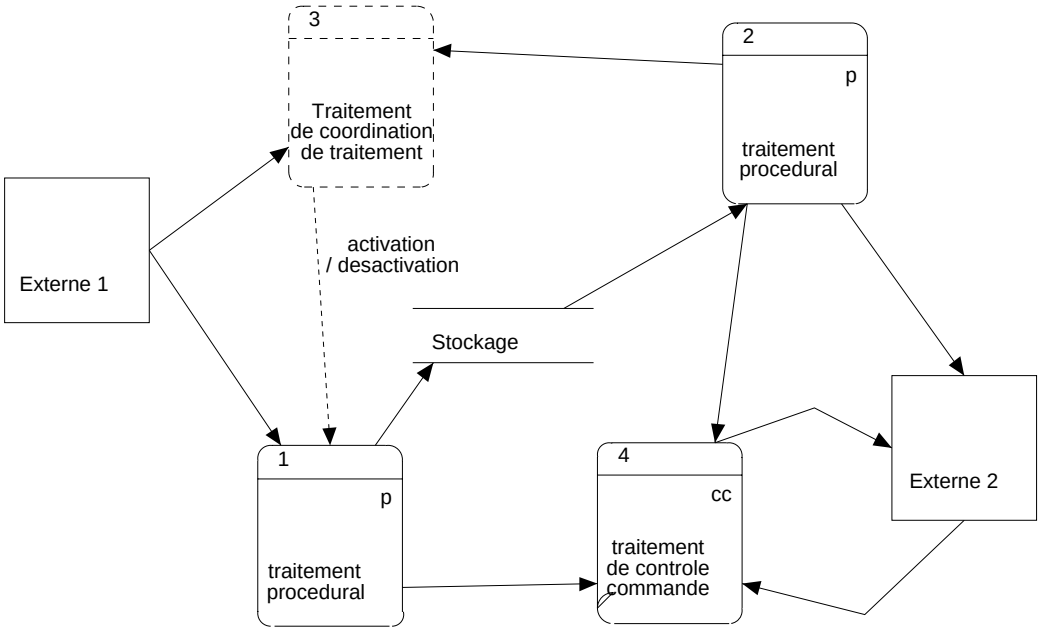


Figure 21 : exemple de modèle d'implantation

1.2.3.2 Elaborer un modèle organique

Elaborer un modèle organique consiste à produire une représentation physique de l'architecture matérielle de la conduite que l'architecte pense à même de répondre au problème.

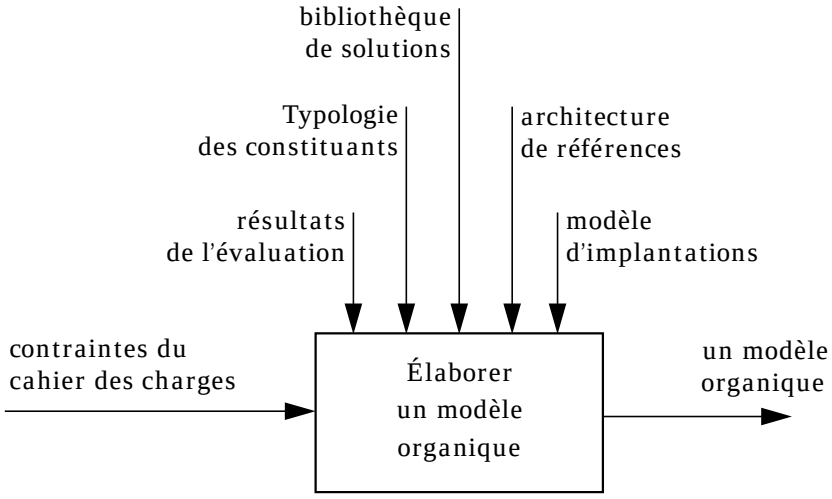


Figure 22 : les entrées-sorties de l'activité «Elaborer un modèle organique»

Les éléments d'entrée de l'activité de «Elaborer un modèle organique» sont très disparates. Le plan d'implantation du système, fournissant la répartition géographique de l'installation, donne par exemple un premier ensemble d'informations. L'architecte doit également prendre en compte les contraintes d'emploi de matériels spécifiques, comme par exemple lors de l'utilisation d'un robot d'assemblage qui ne peut être commandé que par sa baie spécifique. Le cahier des charges peut également imposer tout ou partie de l'architecture de conduite du système. La satisfaction des contraintes effectuée, l'architecte doit compter sur son expérience pour imaginer une solution d'implantation. Il peut toutefois recourir à des aides méthodologiques comme l'emploi de modèles de référence sectoriels [43] [44] ou utiliser une bibliothèque de solutions.

Le modèle produit en sortie de l'activité «Elaborer un modèle organique» est un réseau d'équipements de contrôle-commande, d'informatique industrielle et d'informatique de gestion. On y trouve l'ensemble des équipements de traitement de l'information de l'architecture de conduite (automates programmables industriels, calculateurs industriels, calculateurs de supervision…) ainsi que leurs interconnections assurées par des équipements de communication (liaison point-à-point, réseau maître-esclave, réseau à accès aléatoire…) (fig. 22).

Le cœur de notre travail d'assistance à la conception d'architecture devant être indépendant des méthodes de travail des ingénieristes, il doit l'être également du marché de l'offre des équipements de contrôle-commande. Pour ce faire, nous avons retenu des objets génériques de modélisation issus d'une typologie des équipements de contrôle-commande. Cela nous permet de ne manipuler, lors de la constitution du modèle organique, que des représentants d'une classe d'équipements et non pas directement les équipements proposés par les constructeurs dans chacune des classes.

Le modèle organique est donc construit à partir d'équipements types choisis dans une bibliothèque. De plus, la constitution de bibliothèques d'équipements permet d'atteindre un des objectifs de ce mémoire, à savoir la portabilité à différents domaines d'application. Lorsqu'on souhaite étendre le modèle organique à un nouveau domaine, un expert de la spécialité doit faire un premier travail de synthèse pour

faire émerger une typologie des équipements qui lui sont d'intérêt, puis construire un modèle de chaque type d'équipement identifié. On trouvera un exemple de cette démarche appliqué aux équipements des systèmes de production manufacturiers dans la division 1.3.2 «Le modèle organique».

Le modèle organique n'est donc pas exactement l'architecture matérielle telle qu'elle est souvent présentée dans la littérature (fig. 4). La figure 23 présente un exemple de modèle organique dont la syntaxe et les règles de construction sont développées dans la division 1.3.2 «Le modèle organique».

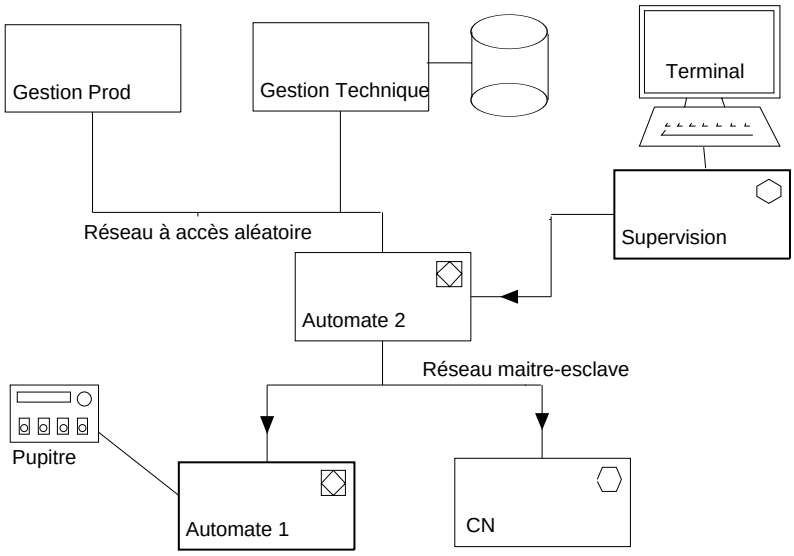


Figure 23 : exemple de modèle organique

1.2.3.3 Elaborer un modèle d'affectation

Elaborer un modèle d'affectation consiste, lorsque le modèle d'implantation est élaboré et qu'un premier jet du modèle organique est mis en place, à projeter le modèle d'implantation sur le modèle organique.

Pour ce faire, les traitements vont être affectés à des équipements de traitement pour assurer leur exécution, de même les stockages vont trouver des mémoires pour accueillir leurs données, et les flots de données vont se voir affecter un trajet à travers les équipements de communication.

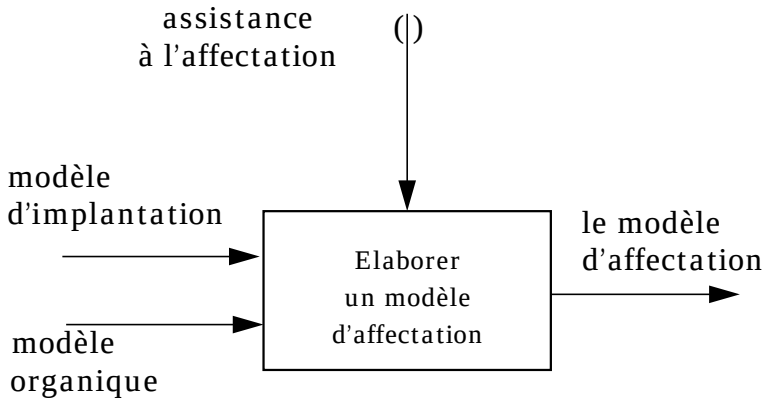


Figure 24 : les entrées-sorties de l'activité «Elaborer un modèle d'affectation»

Pour identifier ce trajet que doit suivre un flot de données il faut savoir où se situe sa source et sa destination. Il est par conséquent naturel de commencer la démarche d'affectation par les traitements et les stockages qui sont sources et destinations des données. Chaque traitement et chaque stockage est dans un premier temps affecté à un équipement de traitement et de stockage. Dans un deuxième temps, les flots doivent trouver leur trajet dans le modèle organique. A ce niveau de la démarche, nous proposons une assistance à l'architecte par la mise en évidence de l'existence ou de la

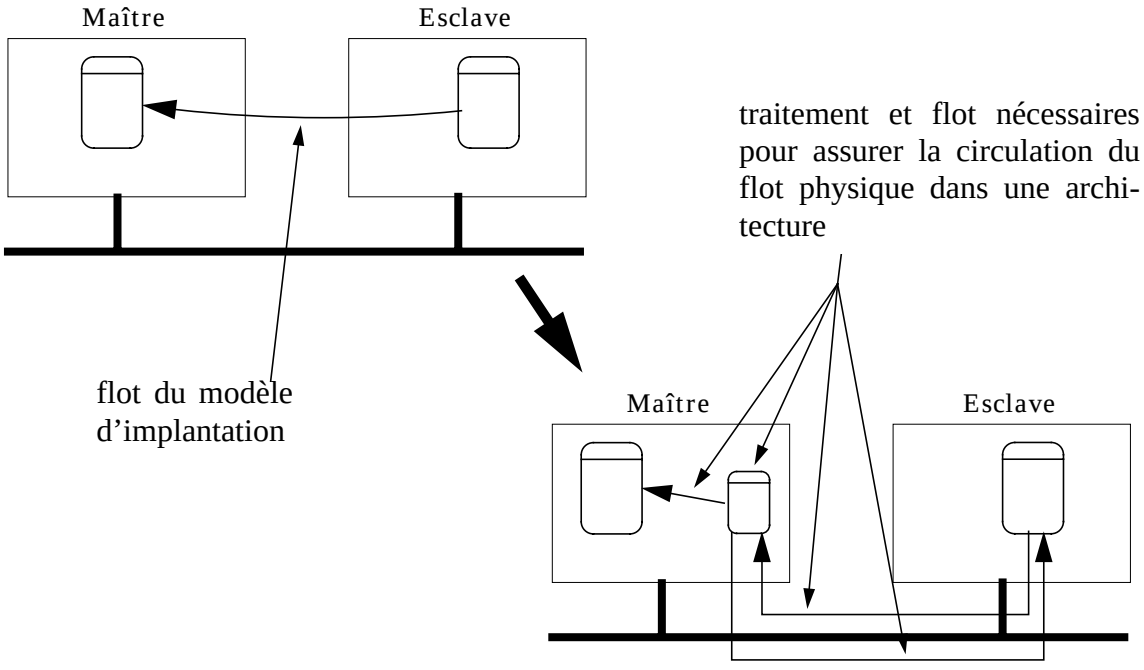


Figure 25 : exemple d'organisation nécessaire pour assurer la circulation d'un flot

non existence d'un trajet. Cette assistance sera présentée dans la division 1.3.3 «Le modèle d'affectation».

Le modèle d'affectation a également pour objectif d'identifier les traitements techniques et les flots techniques, c'est-à-dire les éléments non fonctionnels, donc n'apparaissant pas dans le modèle d'implantation, mais dont la nécessité apparaît lors de l'affectation dans le modèle organique. L'exemple de la figure 25 montre comment un flot du modèle d'implantation peut se transformer en un ensemble de flots techniques et un traitement technique, lorsque son traitement émetteur et son traitement destinataire sont répartis dans deux équipements distincts communicant entre eux par un réseau de type maître esclave.

L'apparition de ces traitements et flots techniques a une grande incidence sur les performances globales d'une architecture, mais ils sont rarement pris en compte dans la conception d'architecture car d'un niveau de détail très fin[1]. Nous proposons dans la division 1.3.3 «Le modèle d'affectation» une détermination systématique des traitements et flots techniques pour la constitution du modèle d'affectation.

Une fois élaboré, le modèle d'implantation fournit à l'architecte une vue complète des traitements qui seront exécutés dans une structure d'équipements donnée. En ce sens, il personnifie plus sûrement une architecture de conduite que le modèle organique. Un modèle d'implantation *est une solution d'architecture* qu'il convient alors d'évaluer.

1.2.4 Description du processus de sélection d'architecture

1.2.4.1 Evaluer une solution d'architecture

Evaluer une solution d'architecture consiste à déterminer des indications qualitatives et quantitatives de la performance de cette architecture, à partir de son modèle d'affectation et de critères d'évaluation propres à l'architecte.

1. paradoxe de l'ingénieriste qui doit étudier suffisamment en détail les besoins du client afin de lui proposer une solution pertinente, tout en les étudiant suffisamment sommairement pour ne pas engager trop de frais d'étude sur une affaire dont la société d'ingénierie n'a pas encore la charge contractuelle.

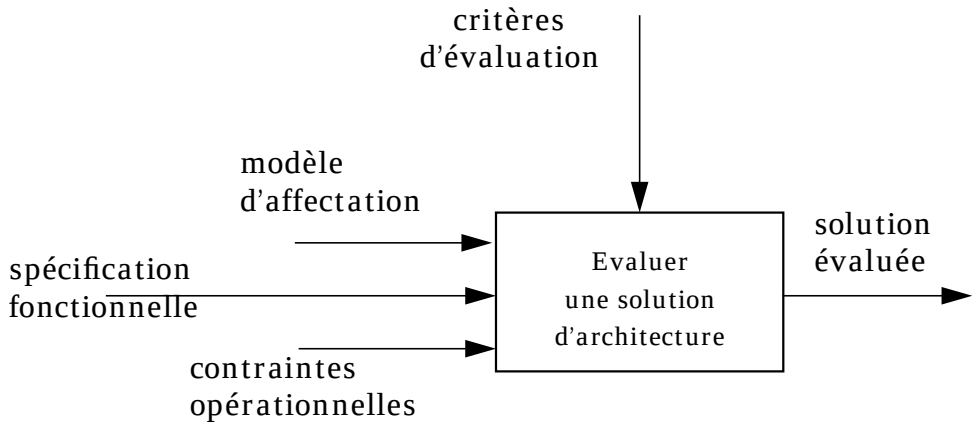


Figure 26 : les entrées-sorties de l'activité «Evaluer une solution d'architecture»

Il s'agit en fait dans cette phase de la conception de vérifier l'adéquation entre une solution d'architecture et les contraintes de performance contenues dans le cahier des charges, de dimensionner les constituants du modèle organique et d'effectuer un choix entre différentes solutions.

Pour donner à l'architecte les moyens d'appréhender les performances a priori de son architecture, nous construirons un *modèle évaluable* de son modèle d'affectation basé sur une description de la dynamique de l'architecture. A partir de ce nouvel apport à la formalisation de la conception d'architecture, vérifier l'adéquation entre une solution et le cahier des charges consiste à utiliser des métriques sur le modèle dynamique de la solution, ou à simuler son comportement avec des sollicitations sous lesquelles le cahier des charges impose des performances précises. Dès l'issue de cette phase de vérification, une solution pourra être rejetée car ne répondant pas aux exigences du cahier des charges. L'architecte devra alors élaborer un nouveau modèle organique puis un nouveau modèle d'affectation propre à être évalué à son tour dans l'espoir de répondre aux exigences de cahier des charges.

L'activité d'évaluation d'une solution a également pour objectif de dimensionner les constituants du modèle organique. Le modèle d'affectation fournit pour chaque constituant la liste exhaustive des éléments qui s'y retrouvent (éléments provenant du modèle d'implantation, et des contraintes techniques dues au choix d'architecture effectué). Par exemple, l'ensemble des traitements affectés à un calculateur permettra de dimensionner sa puissance de calcul requise, ou encore l'ensemble des flots de

communication permettra de dimensionner son débit requis. La mise en évidence d'un constituant «hors norme», c'est-à-dire nécessitant des performances dépassant celles des constituants du marché, conduira à un refus de la solution puis à un remaniement des modèles organique et d'affectation.

Lorsqu'une solution est conforme aux attentes du cahier des charges, et réalisable avec des constituants matériels du marché de l'offre, elle est validée. Pour pouvoir choisir entre différentes solutions validées, l'architecte est amené à appliquer ses propres critères d'évaluation.

La construction d'un modèle dynamique évaluable est également une manière de ne pas imposer les critères d'évaluation qui restent du domaine de l'expert architecte et qui restent très conditionnés par le secteur applicatif de chaque système. L'architecte peut donc à loisir utiliser le modèle dynamique de sa solution pour en extraire des indicateurs qu'il juge pertinents pour le système conçu.

1.2.4.2 Choisir un solution d'architecture

Choisir une solution d'architecture consiste à sélectionner une solution définitive pour le système et déterminer «de quel matériel de chez quel constructeur sera constituée l'architecture ?», et ce, à partir des solutions d'architecture préalablement évaluées, des critères de choix propres à l'architecte, et de l'état de l'offre en matériel à un instant donné. Cette activité consiste également à construire des modèles de spécification pour chaque constituant de l'architecture retenue, et ce, à partir des spécifications du système complet et de la solution d'architecture retenue.

Pour répondre à un appel d'offre et a fortiori pour commencer le développement du système, l'ingénieriste doit disposer d'une architecture matérielle. Le cœur de la conception d'architecture que nous avons voulu indépendant du marché de l'offre des équipements de contrôle-commande, nous conduit à effectuer les choix technologiques durant cette dernière activité de la conception d'architecture. Construire un modèle d'architecture matérielle, c'est faire des choix technologiques là où le modèle organique avait fait des choix techniques. Par exemple, là où dans le modèle organique l'architecte avait choisi un automate programmable industriel, une architecture matérielle devra proposer l'automate de référence X de chez le constructeur Y. Le

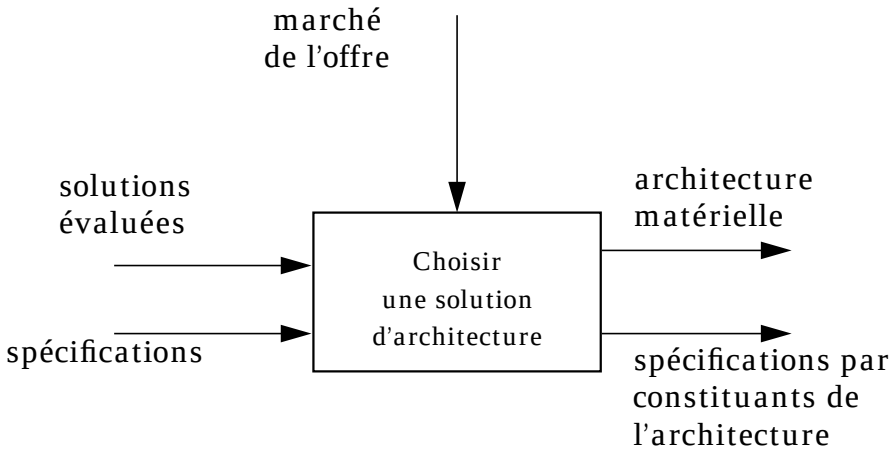


Figure 27 : les entrées-sorties de l'activité «Choisir une solution d'architecture»

modèle d'architecture matérielle est donc l'aboutissement de notre démarche, c'est également le modèle le plus représenté dans la littérature, bien que l'on ignore systématiquement la démarche permettant de l'élaborer. Cette fois-ci, la figure 4 est un exemple typique de modèle d'architecture matérielle.

Pour assurer la suite du développement du système, l'ingénieriste va scinder son étude selon le découpage du modèle organique de l'architecture. Chaque constituant de traitement du modèle va suivre un cycle de conception et de réalisation largement indépendant des autres. Pour ce faire, l'équipe projet à laquelle revient la charge du développement d'un des constituants, doit se voir remettre les spécifications propres de ce constituant. Toujours pour assurer une indépendance entre les méthodes de travail des ingénieristes et la démarche de conception de l'architecture que nous proposons, nous montrerons dans la division 1.4.4 «Les spécifications de chaque constituant» comment construire les modèles de spécification de chaque constituant selon le formalisme de l'ingénieriste, et ce, à partir de notre modèle de solution d'architecture et des spécifications du système.

1.2.5 Déroulement de la démarche

Nous terminerons ce chapitre de présentation générale de la démarche par un éclairage cinématique, en examinant le déroulement chronologique des activités de conception (fig. 28).

L'ordre dans lequel se déroulent les activités de conception de l'architecture, se déduit assez naturellement de l'enchaînement causal de la production des différents modèles :

- la construction du modèle de l'architecture matérielle produit au sein de l'activité «Choisir une solution d'architecture» nécessite la connaissance du ou des modèles de solutions évaluées,
- la construction du ou des modèles de solutions évaluées produit au sein de l'activité «Evaluer une solution d'architecture» nécessite la connaissance du modèle d'affectation,
- la construction du modèle d'affectation produit au sein de l'activité «Elaborer le modèle d'affectation» nécessite la connaissance du modèle d'implantation et du modèle organique,
- et enfin, la construction du modèle organique produit au sein de l'activité «Elaborer le modèle organique» nécessite la connaissance du modèle d'implantation.

La figure 28 est donc le reflet de ces relations d'antériorité. Elle présente, selon le formalisme des organigrammes de traitement, l'enchaînement des activités. On y remarquera deux boucles. La première de nature impérative, consiste à produire des modèles organiques tant que le modèle de solution d'architecture correspondant ne respecte pas les contraintes du cahier des charges, c'est-à-dire tant qu'une solution viable n'a pas été trouvée. La seconde boucle de nature optionnelle, consiste à rechercher une nouvelle solution d'architecture même lorsqu'une solution viable à déjà été trouvée, et ce dans le but de disposer de plusieurs solutions viables au moment du choix définitif afin de sélectionner la plus satisfaisante.

1.2.6 Conclusion du chapitre 1.2

Nous avons dans un premier temps examiné sous un angle fonctionnel la démarche que nous proposons pour concevoir l'architecture de conduite des systèmes de production. Après l'avoir replacée dans le contexte de travail de l'ingénieriste, nous avons examiné l'objectif et les entrées-sorties de chaque activité de la démarche proposée. Puis, dans un second temps, nous avons présenté la séquentialité de cette démarche.

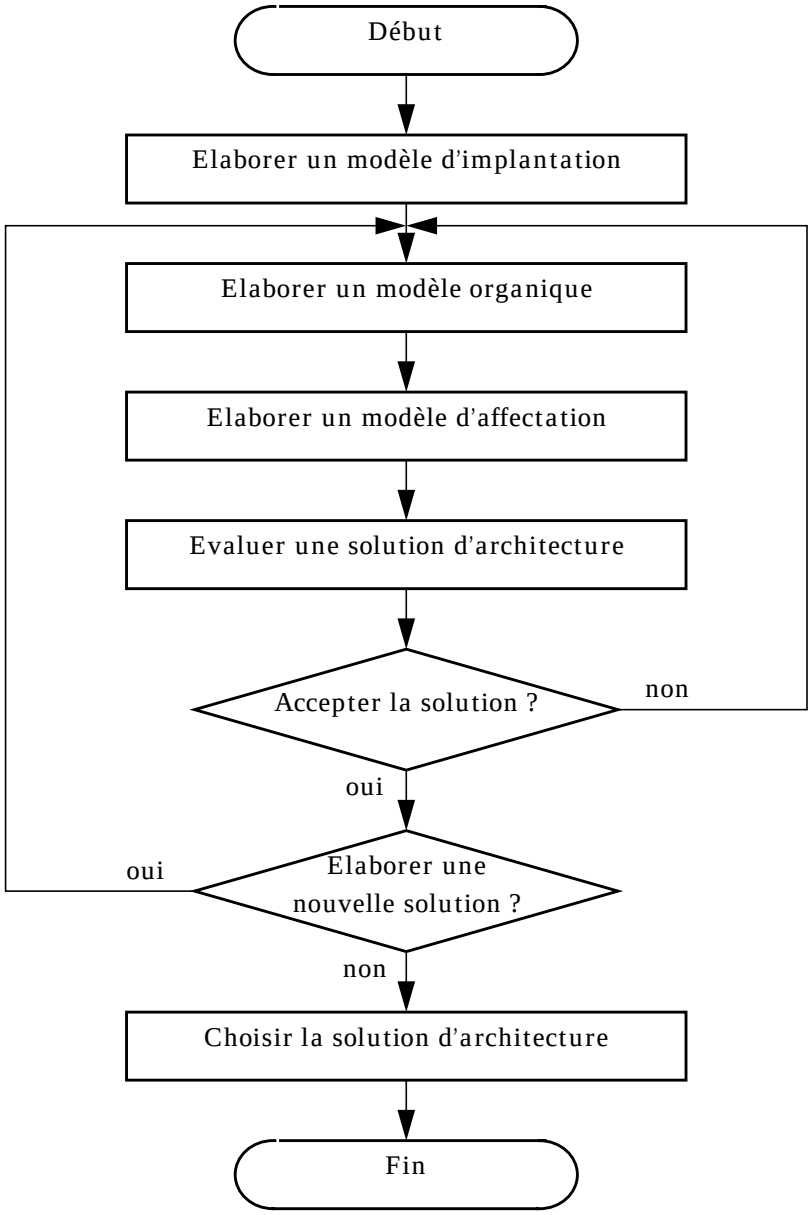


Figure 28 : déroulement chronologique des activités de la démarche

Les principaux points forts de la conception d'architectures de conduite, que nous nous attacherons à développer dans la suite de ce mémoire, nous semblent être :

- une indépendance totale par rapport aux méthodes de conception de systèmes, favorisant une intégration aisée dans les différentes démarches des ingénieristes,
- une conceptualisation et une formalisation importante du processus de conception d'architectures, jusqu'à présent essentiellement perçu comme "l'art empirique de l'expert architecte".

1.3 Construction d'une solution d'architecture

Le chapitre 1.2 «Présentation générale de la méthode» a présenté la démarche que nous proposons en adoptant le point de vue des activités qui la constituent. On a pu remarquer que chaque activité a pour objectif la production d'un modèle (toujours dans le but d'améliorer la formalisation de la conception d'architecture). Dans ce chapitre, on se propose maintenant de détailler les phases de notre démarche qui permettent de construire des solutions d'architecture prêtes à être évaluées avec cette fois *le point de vue des modèles constitués*. Tous les modèles vont être étudiés dans l'ordre chronologique de leur construction. Nous décrirons leur syntaxe et leur sémantique, ainsi leur processus d'élaboration. D'une manière générale nous avons cherché à utiliser des techniques de modélisation existantes et déjà éprouvées dans des domaines scientifiques identifiés plutôt que d'en proposer de nouvelles, même si nous avons parfois été amenés à les enrichir.

1.3.1 Le modèle d'implantation

1.3.1.1 Formalisme du modèle

Le choix du formalisme retenu a été conduit en considérant la nature des entités que l'on souhaite décrire. Le modèle d'implantation doit représenter l'ensemble des traitements nécessaires à l'exécution des fonctions de conduite, les stockages et les flots de données, ainsi que les règles de synchronisation entre traitements. Le modèle conceptuel des traitements (MCT) de la méthode merise [45] était un formalisme potentiel car on y retrouve des traitements et leurs synchronisations. Un formalisme à base de flots de données lui a été préféré pour deux raisons : une plus grande diffusion dans le domaine du génie automatique, et de nombreux travaux ayant déjà porté sur des extensions à ce formalisme [46] [47] [48].

Le modèle d'implantation est, comme le montre la figure 29, un réseau d'interconnexion de traitements, reliés par les données qui circulent. Le modèle se construit à partir de quatre primitives graphiques : des *traitements*, des *stockages*, des *flots*, et des entités *externes*.

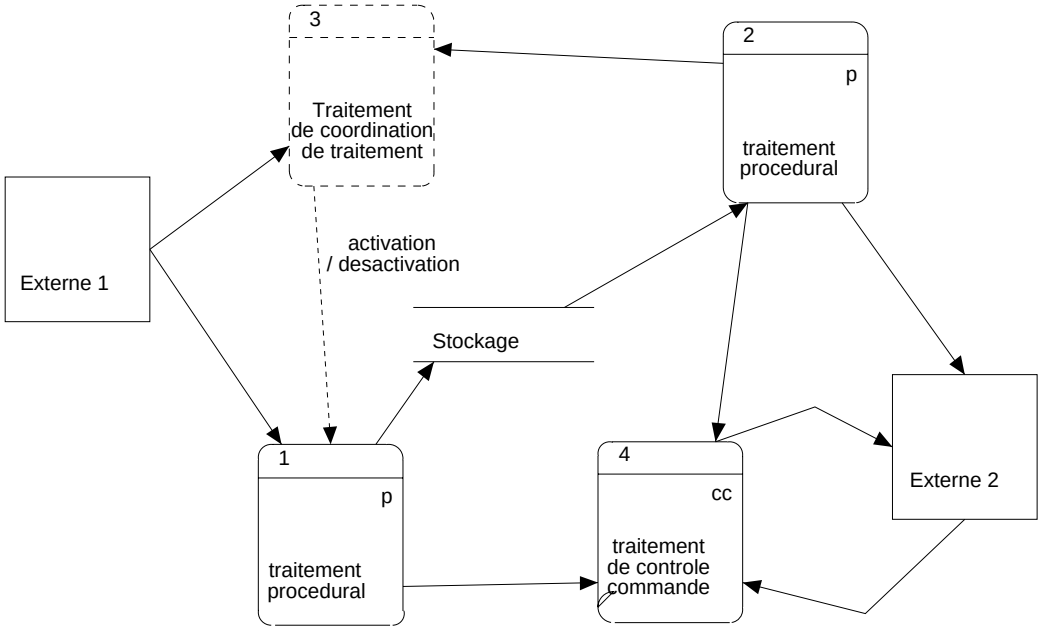


Figure 29 : exemple de modèle d'implantation

Les traitements sont séparés en trois classes distinctes. Elles correspondent à des mises en œuvre différentes au niveau de leur exécution par un constituant microprogrammé. On distingue ainsi (fig. 30) :

- les *traitements procéduraux*, qui sont caractérisés par un début et une fin, entre lesquels des séquences en nombre fini s'enchaînent de manière continue,
- les *traitements de contrôle/commande*, qui nécessitent une exécution permanente, et qui sont construits autour d'un automate fini produisant des sorties en fonction de l'apparition d'entrées et de son état interne,
- les *traitements de coordination* de traitements, qui contrôlent les autres traitements du modèle d'implantation par le biais d'activations et de désactivations.

Les stockages de données permettent la mémorisation de tout type de donnée, indépendamment de sa taille et de son contenu. Ils doivent être vus par les traitements comme des services qui leurs offrent la possibilité de stocker (écrire) et de déstocker (lire) des données.

Les entités externes indiquent la provenance ou la destination des flots traités par le système.

Les flots permettent de relier les traitements entre eux et de relier les traitements avec les stockages de données. Ils sont séparés en trois classes distinctes correspondant à la nature des informations qu'ils véhiculent. On distingue ainsi (fig. 30) :

- les *flots de donnée*, qui relient les traitements aux stockages et inversement,
- les *flots de message*, qui sont porteurs à la fois d'une donnée et d'un événement ; ils relient les traitements entre eux et permettent à un traitement émetteur de déclencher l'exécution d'un traitement récepteur par l'événement contenu dans le flot, tout en lui transmettant une donnée,
- les *flots de coordination*, qui sont issus des traitements de coordination de traitement, permettent l'activation et la désactivation des traitements procéduraux et des traitements de contrôle-commande.

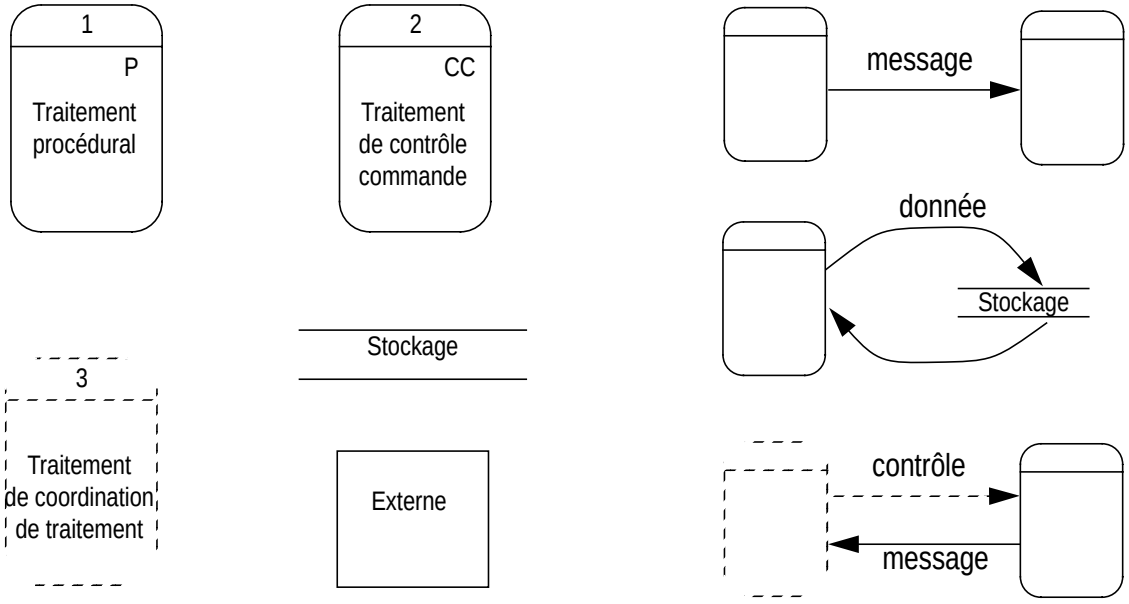


Figure 30 : primitives graphiques de représentation du modèle d'implantation

Selon leur nature, les traitements s'exécutent conformément à des règles différentes.

1.3.1.1.1 Les traitements procéduraux

Les traitements procéduraux se comportent en transformateurs de données. Le début de leur exécution est conditionné par l'apparition d'une conjonction d'événements véhiculés dans les messages, qui marque le début du traitement. En fin de traitement les données produites sont regroupées au sein d'une disjonction. Selon les

alternatives rencontrées durant le traitement, une et une seule des disjonctions possibles sera émise**.** Les stockages de données ne contribuent pas à la synchronisation des traitements. Ce sont les traitements qui lisent ou écrivent des données dans les stockages selon leur propre initiative. La description d'un traitement procédural doit donc être complétée par le renseignement du triplet $(C(t), D(t), S_c(t))$ pour décrire la cinématique, avec :

- $C(t)$ l'ensemble des conjonctions associées au traitement t,
- $D(t)$ l'ensemble des disjonctions associées au traitement t,
- $S_c(t)$ l'ensemble des stockages de données à lire à la suite d'une conjonction associée au traitement t,

et où $\forall d \in D(t)$, d est le couple $(S_e(d), M(d))$ dans lequel :

- $S_e(d)$ est l'ensemble des stockages qui vont être effectués lors de la disjonction d,
- $M(d)$ est l'ensemble de messages qui sont émis lors de la disjonction d.

Un traitement procédural vérifie alors les propriétés suivantes :

- le traitement t est déclenché lorsque l'équation booléenne (1) est vérifiée, c'est-à-dire lorsqu'il existe une conjonction pour laquelle tous les messages associés sont émis,

$$\sum_{c_i \in C(t)} \left(\prod_{m \in c_i} m \right) = 1 \qquad (1)$$

- $\forall d \in D(t)$, lorsque la disjonction se produit, une donnée est émise dans tous les stockages de $S_e(d)$, puis tous les messages de $M(d)$ sont émis.

La figure 31 propose un exemple de traitement procédural. Il est déclenché lorsque l'on a :

$$\begin{aligned} &(message_1 \cdot message_2) + \\ &(message_2 \cdot message_3) + \\ &(message_1 \cdot message_3) = 1 \end{aligned} \qquad (2)$$

En fin d'exécution on pourra :

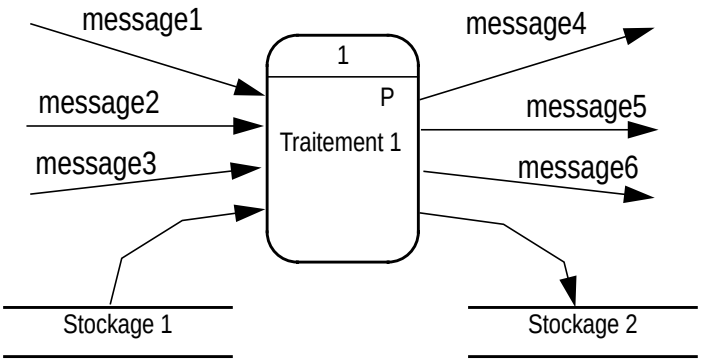


Figure 31 : exemple de représentation graphique d'un traitement procédural

- soit stocker un résultat dans le stockage 2 et émettre les messages $message_4$ et $message_5$,
- soit émettre les messages $message_5$ et $message_6$.

En d'autres termes, la description graphique de la figure 31 doit être complétée par le triplet $\left(\{c_1,c_2, c_3\}, \{d_1,d_2\}, \{stockage_1\} \right)$ avec

- $c_1 = \{message_1,mesage_2\}$
- $c_2 = \{message_2,mesage_3\}$
- $c_3 = \{message_1,mesage_3\}$
- d_1 est le couple $(\{stockage_2\},\{message_4,message_5\})$
- d_2 est le couple $(\varnothing,\{message_5,message_6\})$

1.3.1.1.2 Les traitements de contrôle-commande

Les traitements de contrôle-commande étant construits autour d'un automate fini, la description de leur comportement est une description dynamique. Toutefois, il est fréquent qu'un traitement de contrôle-commande ait un début et une fin d'exécution. C'est le cas lorsqu'il faut conduire un processus d'un état à un autre au vu de l'état de ses capteurs. Le traitement de contrôle-commande se déclenche alors à l'apparition d'une information élémentaire «début de contrôle-commande» véhiculée par un message noté m_d, et dès que le traitement a fini de conduire le processus vers son état final, il émet l'information «fin de contrôle-commande» sous la forme d'un message noté m_f. La description d'un traitement de contrôle-commande doit donc être complétée par le renseignement du couple (m_d, m_f). Les messages m_d et m_f peu-

vent être le message nul, c'est-à-dire pas de message. C'est le cas par exemple d'un traitement qui contrôle et commande en permanence un processus (pas de début, ni de fin d'exécution du traitement). Les entrées-sorties du processus contrôlé, pourront être regroupés dans 2 flots entre le traitement et l'entité externe désignant tout ou partie du processus (fig. 32).

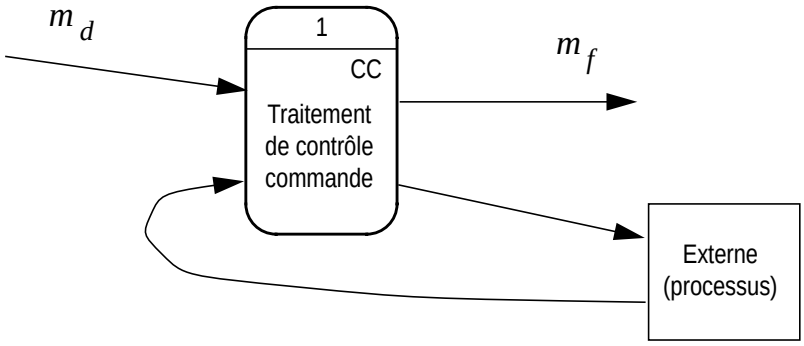


Figure 32 : exemple de représentation graphique d'un traitement de contrôle-commande

1.3.1.1.3 Les traitements de coordination

Les traitements de coordination ont un comportement dynamique, c'est-à-dire qu'à tout moment leur réaction dépend à la fois des sollicitations auxquelles ils sont soumis et de leur état interne.

Il est important de faire deux remarques générales sur le modèle d'implantation :

- il n'y a pas de description hiérarchique des traitements. Chaque traitement représenté correspond à une future unité d'exécution tangible. Le modèle d'implantation est donc un modèle *à plat*. Cette caractéristique serait certes un handicap si le modèle d'implantation avait pour vocation de servir de support de spécification pour l'ingénieriste. Or, son seul rôle est d'accueillir les éléments pertinents d'une spécification déjà élaborée. En revanche un modèle à plat a l'avantage de montrer à l'architecte l'ensemble des traitements à réaliser.
- il n'y a pas de regroupement graphique des flots entre eux. Chaque flot est représenté par une et une seule flèche avec une source unique et une destination unique. Ce choix ne va pas non plus dans le sens de la lisibilité du modèle, mais a pour but de bien identifier séparément les flots, car ils nécessiteront tous

un support physique. Cela a l'avantage une fois encore de montrer à l'architecte l'ensemble des flots qui auront à être implantés dans l'architecture.

Le modèle d'implantation n'a pas pour vocation d'être conçu globalement. Nous venons de voir que les options prises dans son formalisme mettent en évidence les éléments qu'il faudra implanter dans l'architecture, mais rendent le modèle difficilement élaborable. L'activité d'élaboration d'un modèle d'implantation est plutôt une activité de traduction directe d'un modèle en un «modèle équivalent», tout comme par exemple dans [49], qui propose des règles de transformation de diagrammes IDEF0 en réseaux de Petri colorés.

1.3.1.2 Intégration du modèle d'implantation et des modèles de spécification

Nous avons déjà évoqué dans les deux premiers chapitres de ce mémoire notre soucis de *portabilité* de nos travaux, que ce soit au niveau des concepts introduits, de la méthode proposée ou de la sémantique des modèles. Afin de satisfaire cet objectif, nous allons maintenant proposer un méta-modèle du modèle d'implantation et illustrer ses liens sémantiques avec les modèles de spécification, par exemple avec SART.

La méta-modélisation est un concept fort d'intégration entre modèles ayant déjà fait l'objet de nombreux travaux au L.U.R.P.A. [50] [51].

Nous rappellerons que modéliser une méthode d'élaboration de modèles c'est :

- modéliser chacun des modèles mis en jeu (dans notre cas, les modèles de spécification, et le modèle d'implantation),
- modéliser les techniques de construction de modèle (sémantiques associées aux modèles).

La figure 33 illustre le procédé de formalisation de la méthode de construction du modèle d'implantation.

Nous appellerons méta-modèle de la méthode de traduction des modèles de spécification en modèle d'implantation, la description formelle, ainsi constituée. Pour modéli-

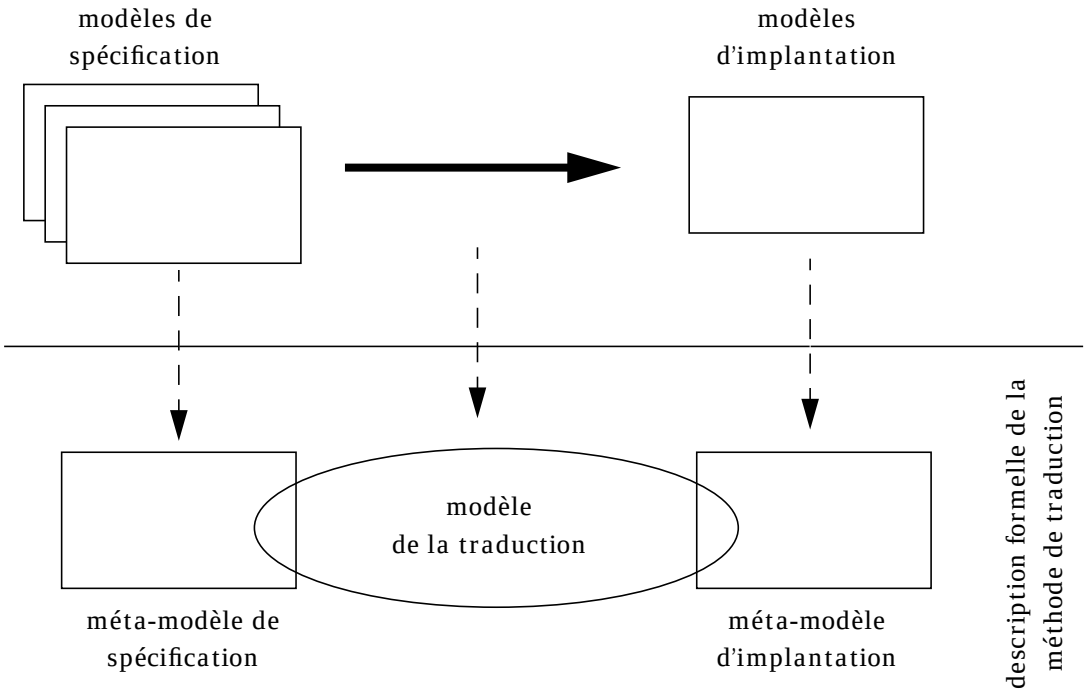


Figure 33 : principe de la formalisation de la construction d'un modèle d'implantation

ser la méthode d'élaboration du modèle d'implantation (méta-modéliser), nous avons retenu le formalisme entité-relation, pour les raisons suivantes :

- il est bien adapté à la représentation de la sémantique des techniques de modélisation et pas seulement de leur syntaxe,
- il permet de représenter à la fois simplement et formellement les procédures de passage entre modèles (souvent exprimées en termes d'associations entre entités des modèles),
- il donne un point de vue «système d'information» de la méthode considérée. C'est le point de vue privilégié de l'intégration.

La figure 34 présente le méta-modèle du modèle d'implantation, expression de la sémantique contenue dans le modèle d'implantation. Il a été élaboré avec pour objectif d'assurer l'intégration avec des modèles de spécification.

Pour poursuivre la présentation de notre approche, il faut convenir de l'utilisation d'un modèle pour la spécification. Cela permettra de mettre en évidence comment

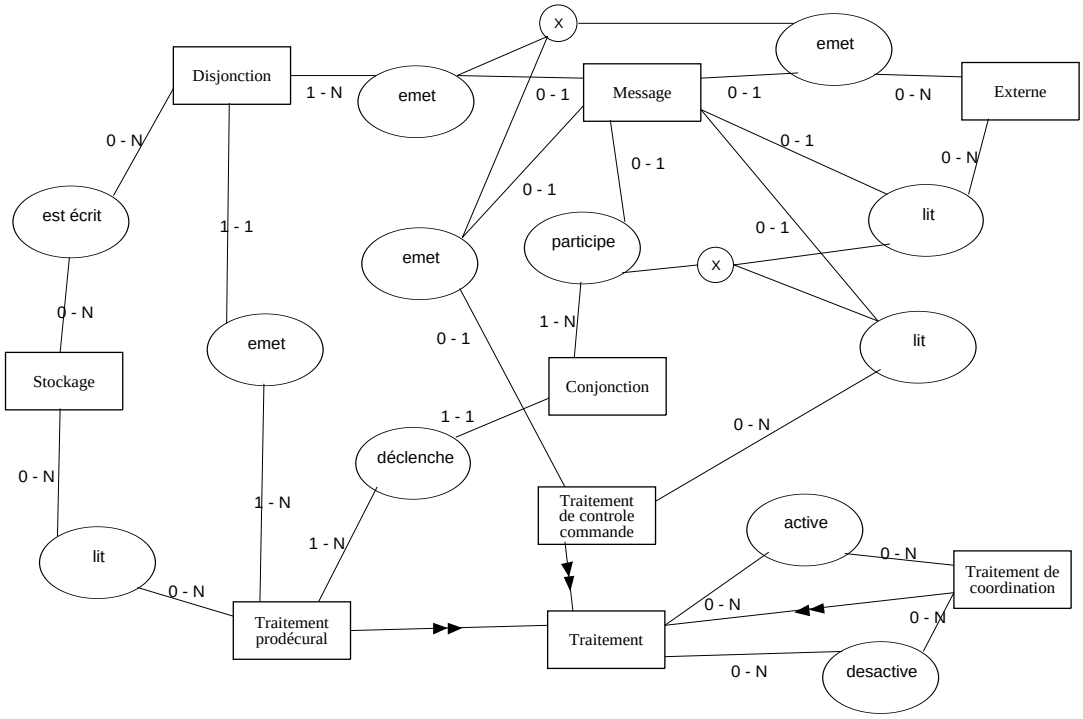


Figure 34 : méta-modèle du modèle d'implantation

une intégration est possible, à partir d'une spécification indépendante des modèles utilisés pour la conception d'architecture. Nous avons choisi la méthode SART de Hatley et Pirbhai pour élaborer les spécifications du système de production. Ce choix ne doit cependant pas être vu comme un restriction des méthodes pouvant être prises en compte dans notre démarche, il doit seulement être interprété comme un choix nécessaire à l'illustration de nos travaux. Toutefois ce choix, se veut représentatif des modèles de spécification couramment utilisés, dans la mesure où il aborde à la fois les aspects fonctionnels, les aspects dynamiques et les aspects de structuration des données du système de production spécifié. C'est de plus celui qui a été retenu dans l'exemple traité dans la deuxième partie de ce mémoire. La figure 35 présente le méta-modèle de spécification, expression de la sémantique nécessaire à l'intégration contenue dans le modèle SART.

La formalisation de l'intégration entre le modèle de spécification et le modèle d'implantation se traduit alors par des associations entre entités des deux méta-modèles (fig. 36).

Le méta-modèle proposé permet ainsi :

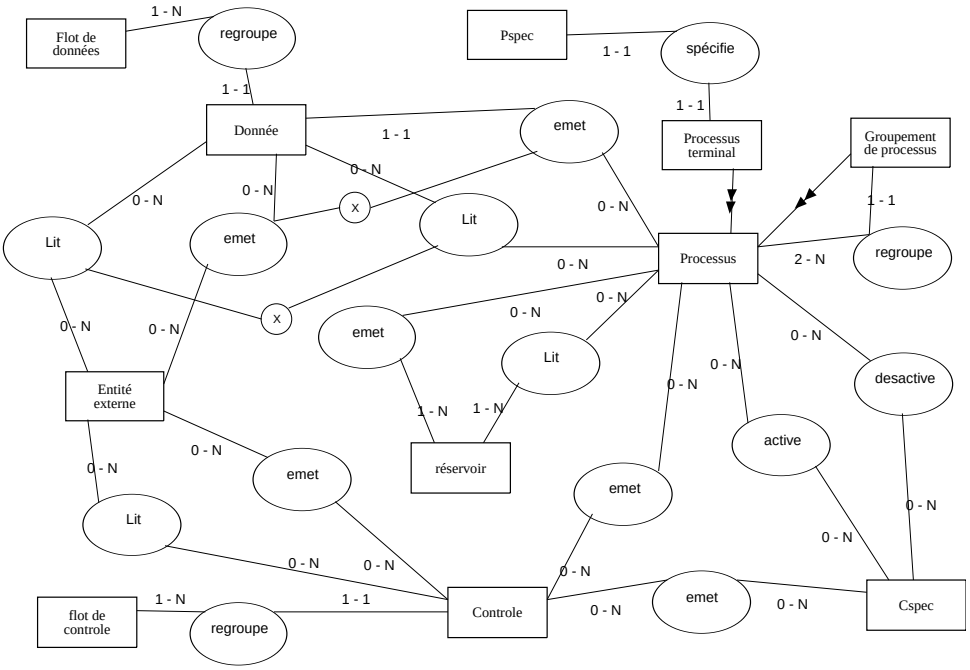


Figure 35 : méta-modèle partiel de SART

- de fixer un savoir faire de l'architecte vis à vis de l'extraction des données pertinentes pour la conception d'architecture dans un modèle de spécification,
- de garantir la cohérence du modèle d'implantation par rapport au modèle de spécification,
- d'automatiser, si besoin est, une grande partie de la construction du modèle d'implantation,
- d'assister l'architecte pour l'élaboration des aspects non automatisables de la construction, comme la détermination des conjonctions qui ne peuvent être déduites du modèle de spécification, mais pour lesquelles on est capable de proposer la liste des messages pouvant y participer.

Le modèle d'implantation permet donc d'exprimer ce que doit réaliser le sous-système de conduite. Son formalisme, sa sémantique et son processus de construction à partir des spécifications du système de production, ont été développées, notamment grâce à la méta-modélisation.

Nous allons maintenant définir le modèle organique qui exprime quant à lui la nature matérielle des équipements de conduite, de dialogue homme/machine et de communication, ainsi que leur inter-connexions.

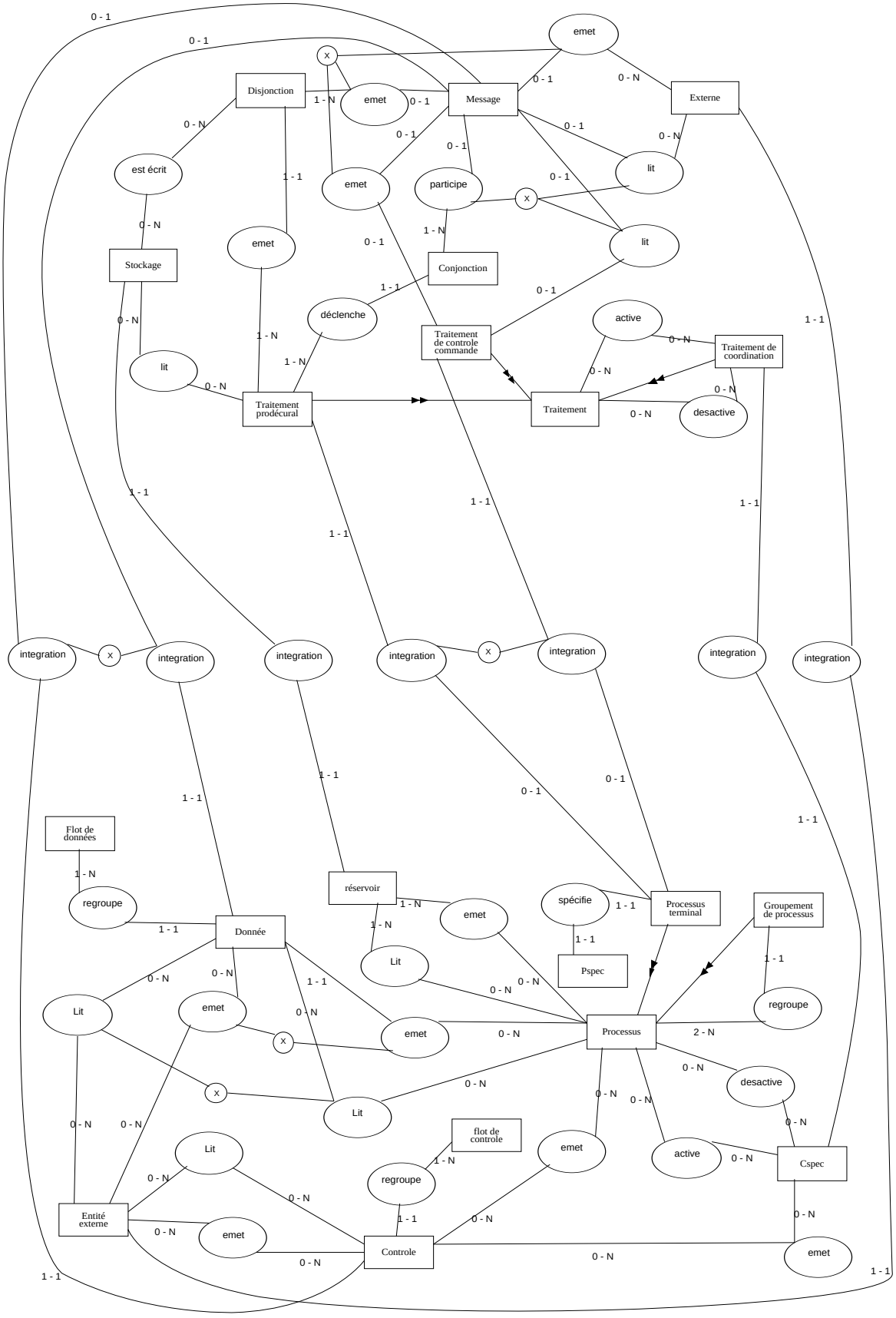


Figure 36 : méta-modèle de l'intégration entre le modèle de spécification et le modèle d'implantation

1.3.2 Le modèle organique

Pour le modèle organique, l'absence de formalisme établi permettant de représenter la structure organique d'une architecture de conduite, nous à conduit à adopter une représentation proche des usages de représentation du modèle matériel. Toutefois, la norme française NF E 04-203-3 intitulée «Régulation, mesure et automatisme des processus industriels : représentation symbolique», nous propose pour quelque organes de traitements une représentation graphique rapportée dans le tableau 2.

Tableau 2 : représentation graphique des équipements selon NF E 04-203-3

Dénomination	Symbole
Instrument individuel	
Calculateur de processus	
Calculateur de supervision	
Automate	
Système d'affichage ou de commande en temps partagé	

Le modèle organique est, comme le montre la figure 37, une interconnexion d'équipements de traitement, d'équipements de stockage et d'équipements de dialogue, reliés entre eux par des media de communication. Le modèle est construit à partir de quatre éléments graphiques de base : les *équipements de traitement*, les *équipements de stockage*, les *équipements de dialogue*, et les *media de communication*.

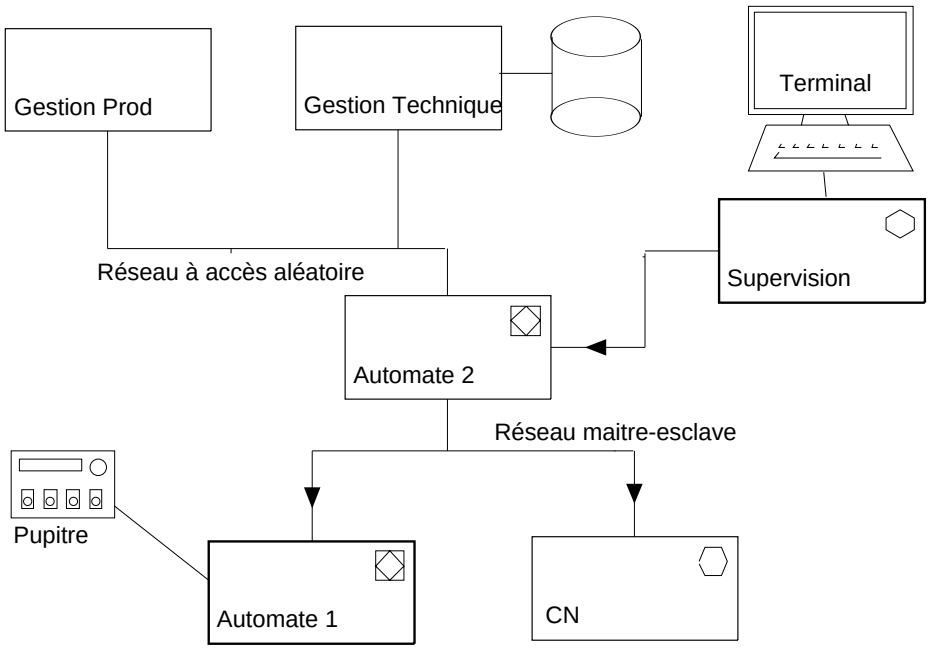


Figure 37 : exemple de modèle organique

Les équipements sont dont séparés en classes distinctes correspondant aux quatre principales fonctions à assurer aux sein de la future architecture matérielle :

- exécuter des traitements,
- stocker des données,
- faire circuler des données (routage),
- dialoguer avec des opérateurs.

Quel que soit le domaine d'application du système de production dont on conçoit l'architecture, on retrouve ces quatre fonctions énumérées. Par contre, les types d'équipements aptes à réaliser chacune de ces fonctions sont très liés au domaine cible. La liste des équipements types que nous proposons n'est ni exhaustive, ni générale. Elle se veut représentative des systèmes de production industriels, et à dominante manufacturiers. Cette typologie d'équipements doit être élaborée par un expert auquel nous nous sommes momentanément et sans doute imparfaitement substitué.

Parmi les équipements de traitement, nous distinguons ainsi :

- les Automates Programmables Industriels (API),

- les Systèmes Numériques de Contrôle/Commande (SNCC),
- les Calculateurs que l'on subdivise en calculateurs de processus (informatique embarquée), en calculateurs de supervision (édition de journaux, de bilans, calculs scientifiques), et en calculateurs banalisés.

Parmi les équipements de stockage, on ne distinguera pas les systèmes de gestion de fichiers utilisant un stockage de masse comme un disque dur, des systèmes de gestion de base de données plus complexes. Dans les deux cas, l'équipement de stockage sera géré par un équipement de traitement qui assurera les accès aux données. L'utilisation d'un équipement de stockage traduit la volonté de l'architecte d'utiliser un système (logiciel et mémoire de masse) dédié à la gestion d'un volume important de données.

Parmi les équipements de dialogue avec l'opérateur on distingue ainsi :

- les terminaux alphanumériques,
- les terminaux graphiques,
- les pupitres.

Parmi les media de communication on distingue :

- les media bi-connexions que l'on subdivise en media informatique (liaison série, liaison parallèle, ...), à accès maître-esclave (ou half duplex, c'est-à-dire qu'un seul des deux interlocuteurs à l'initiative des communications), en media informatique à accès aléatoire (ou full duplex, c'est-à-dire que les deux interlocuteurs peuvent prendre l'initiative de la communication) et en media «fils à fils» comme entre un pupitre de boutons et de voyants et un API, ou entre deux API que l'on relie avec quelques entrées-sorties ;
- les media multi-connexions que l'on subdivise en réseaux de capteurs, en réseaux d'automates à accès maître-esclave, et les réseaux locaux à accès aléatoire.

La figure 38 présente les symboles utilisés dans le modèle organique.

Le modèle organique possède des primitives imposant à l'architecte de poser des solutions potentielles d'architecture en terme de *classes d'équipements* ayant des

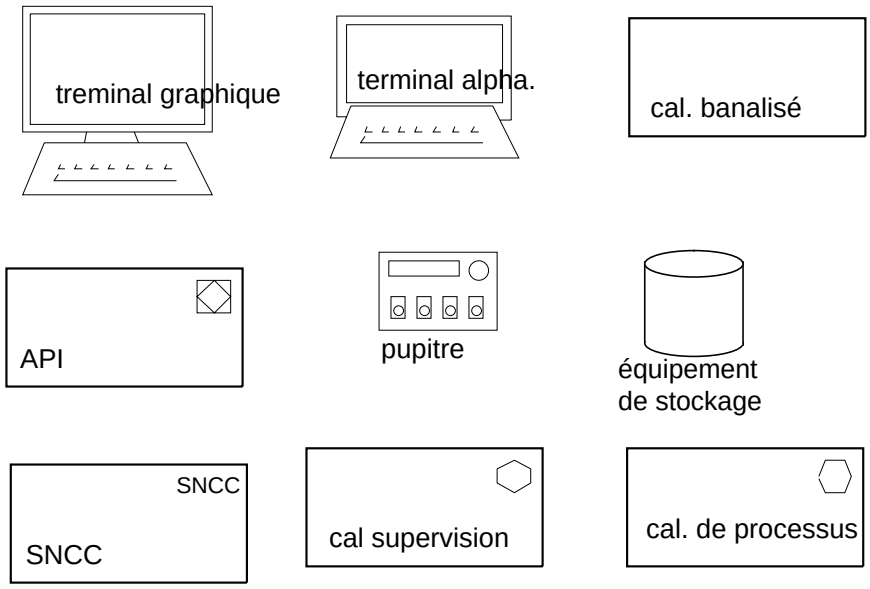


Figure 38 : symboles pour la représentation du modèle organique

capacités techniques identifiées et non en terme de *composants technologiques* du marché de l'offre. Cette démarche nous parait en effet particulièrement apte à supporter les évolutions importantes des technologies de commande et à s'adapter à des systèmes cibles très variés. De plus, nous allons montrer dans le chapitre qui suit que pour choisir, donc dimensionner, les constituants matériels d'une architecture de conduite, la constitution - et l'évaluation - du modèle d'affectation est impérative, notamment pour tenir compte des fonctions et données techniques inhérentes à des choix d'affectation.

1.3.3 Le modèle d'affectation

Le modèle d'affectation permet d'associer les entités du modèle d'implantation avec les éléments du modèle organique, ou en d'autres termes de décider de la répartition des traitements et données du système de conduite dans les équipements de son architecture. L'association commence par les traitements du modèle d'implantation. Pour chacun d'entre eux, on détermine «qui» doit assurer leur exécution. Les équipements de traitement sont les seuls candidats à l'association. Il y a cependant deux cas de figure :

- le traitement considéré est affecté à un et un seul équipement de traitement,
- le traitement est affecté à au moins deux équipements de traitement.

Dans le premier cas il suffit de prendre acte de l'affectation, dans le second cas un ensemble de traitements et de flots techniques vont apparaître sur le modèle d'implantation.

A chaque instance t_i du traitement t affecté dans plusieurs équipements, on associe en amont un *traitement filtre* par lequel passent tous les messages à destination de t_i. Ce traitement filtre a pour but de laisser circuler, ou de stopper, les messages à destination de t_i. A l'origine de chaque message à destination de t un nouveau traitement est associé afin de diffuser le message à tous les traitements t_i. A l'opposé, à la destination des messages et des données originaires de t, un *traitement collecteur* fait son apparition pour regrouper les flots en provenance des t_i. Pour sélectionner le traitement adéquat, un traitement de coordination assure la sélection du traitement par l'activation ou la désactivation des différents traitements filtre. La figure 39 présente un exemple de transformation du modèle d'implantation. Les traitements filtre sont affectés dans les mêmes équipements de traitement que le traitement pour lequel ils assurent la fonction de filtrage des messages, tandis que les traitements diffuseur sont implantés dans les mêmes équipements de traitement que le traitement émetteur du message associé, et que les traitements collecteur sont implantés dans les mêmes équipements de traitement que le traitement destinataire du message associé ou que le stockage destinataire.

Dans un deuxième temps, les stockages doivent être affectés à un équipement du modèle organique. Chaque stockage peut être accueilli soit par un équipement de traitement (au sein de ses ressources de stockage propre, telle que sa mémoire), soit par un équipement spécifique de stockage.

Dans un troisième temps, on s'intéresse aux flots qui circulent entre les traitements et entre traitements et stockages. Tout flot situé entre deux éléments (traitement ou stockage) affectés dans le même équipement, est considéré comme étant pris en charge par l'équipement de traitement lui-même (par exemple avec des techniques de mémoire partagée, de pipe entre processus, ...). Les flots qui doivent retenir notre attention sont ceux qui doivent circuler d'un équipement de traitement à un autre.

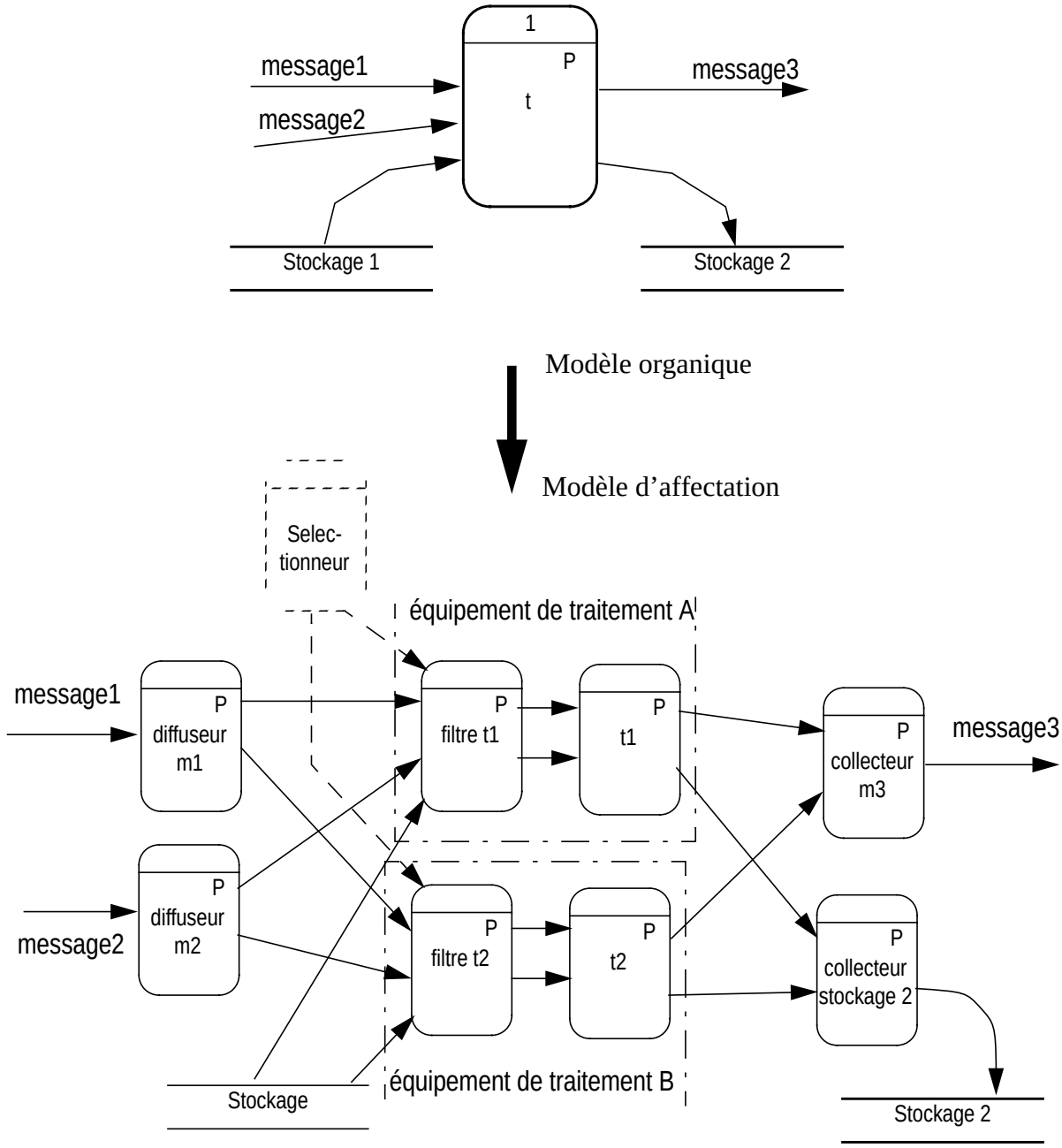


Figure 39 : exemple d'affectation d'un traitement dans deux équipements de communication

Afin d'assister l'architecte dans sa démarche de recherche du trajet que doit suivre un flot, nous lui proposons une technique basée sur la théorie des graphes. L'idée consiste à transformer selon une stratégie adéquate le modèle organique en un graphe. La recherche d'un trajet dans le modèle organique pour un flot donné se traduit par la recherche d'un chemin dans le graphe associé.

La stratégie de construction du graphe est la suivante :

- à chaque équipement de traitement et chaque équipement de dialogue est associé un sommet du graphe,
- à chaque media multi-connexions à accès aléatoire est associé un sommet du graphe,
- pour tout media bi-connexions entre deux équipements de traitement ou entre un équipement de traitement et un équipement de dialogue, on place une arête entre les deux sommets correspondants,
- pour tout media multi-connexions à accès maître-esclave une arête est placée entre le sommet associé à l'équipement maître, et chaque sommet correspondant aux équipements esclaves,
- pour toute multi-connexion à accès aléatoire, une arête est placée entre le sommet associé au media, et chaque sommet correspondant aux équipements connectés.

La figure 40 illustre cette stratégie de construction du graphe associé.

Assister l'architecte pour trouver le trajet d'un flot dont on connaît l'équipement de traitement source et l'équipement de traitement destination, consiste donc à identifier les deux sommets associés aux deux équipements de traitement mis en jeu, puis à chercher dans le graphe un chemin propre[1]. Trois cas peuvent alors se présenter :

- Aucun chemin n'existe entre les deux sommets, c'est-à-dire aucun trajet n'est possible pour faire circuler le flot entre les deux traitements. L'architecte doit remettre en cause l'affectation de ces traitements ou même éventuellement remettre en cause son modèle organique.
- Un chemin propre unique est identifié entre les deux sommets. L'assistance est totale, pas besoin d'une décision de l'architecte pour déterminer le trajet entre les deux équipements de traitement mis en jeu,
- Plusieurs chemins propres sont identifiés entre les deux sommets. C'est le cas général pour les architectures de grande envergure ou fortement interconnectées. Dans ce cas c'est à l'architecte de faire le choix du «meilleur» chemin. Deux assistances peuvent alors l'aider dans son choix : l'identification sur le graphe

1. Un chemin propre est un chemin tel qu'en le parcourant, on ne rencontre pas deux fois le même sommet

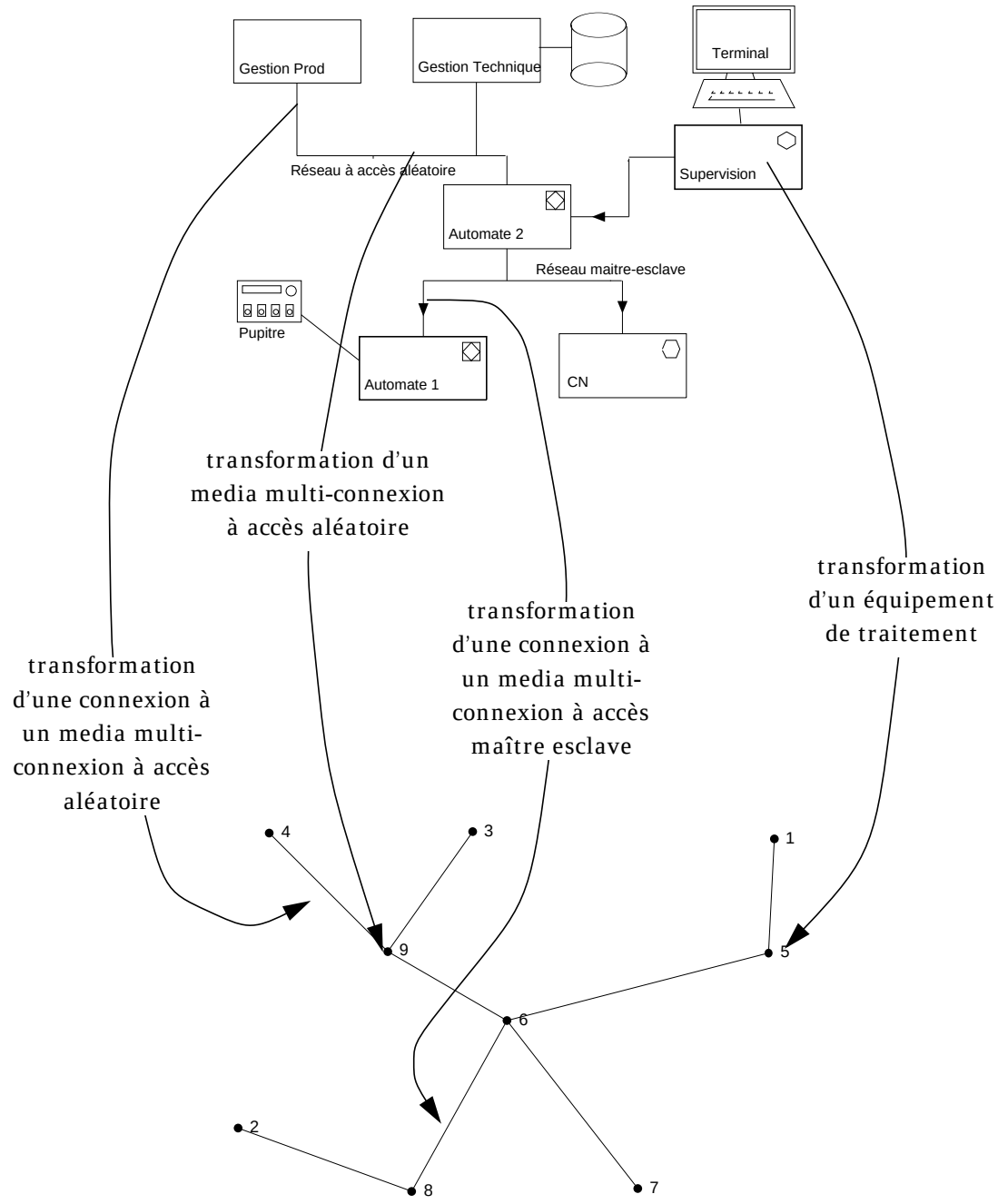


Figure 40 : exemple de transformation d'un modèle organique en graphe équivalent

de tous les chemins propres existants, et la détermination du chemin propre le plus «court».

La notion de chemin propre le plus court est à définir par l'architecte lui-même selon ses critères d'évaluation d'architecture. A type d'exemple, on peut choisir l'échelle des temps de transmission comme métrique de la longueur d'un chemin. La longueur d'une arête ou plus exactement son poids, sera inversement proportionnel à la

vitesse de transmission. Rechercher le chemin le plus court, c'est-à-dire celui dont la somme des poids de ses arêtes est minimum, c'est alors rechercher le trajet le plus rapide (temps de transmission minimum).

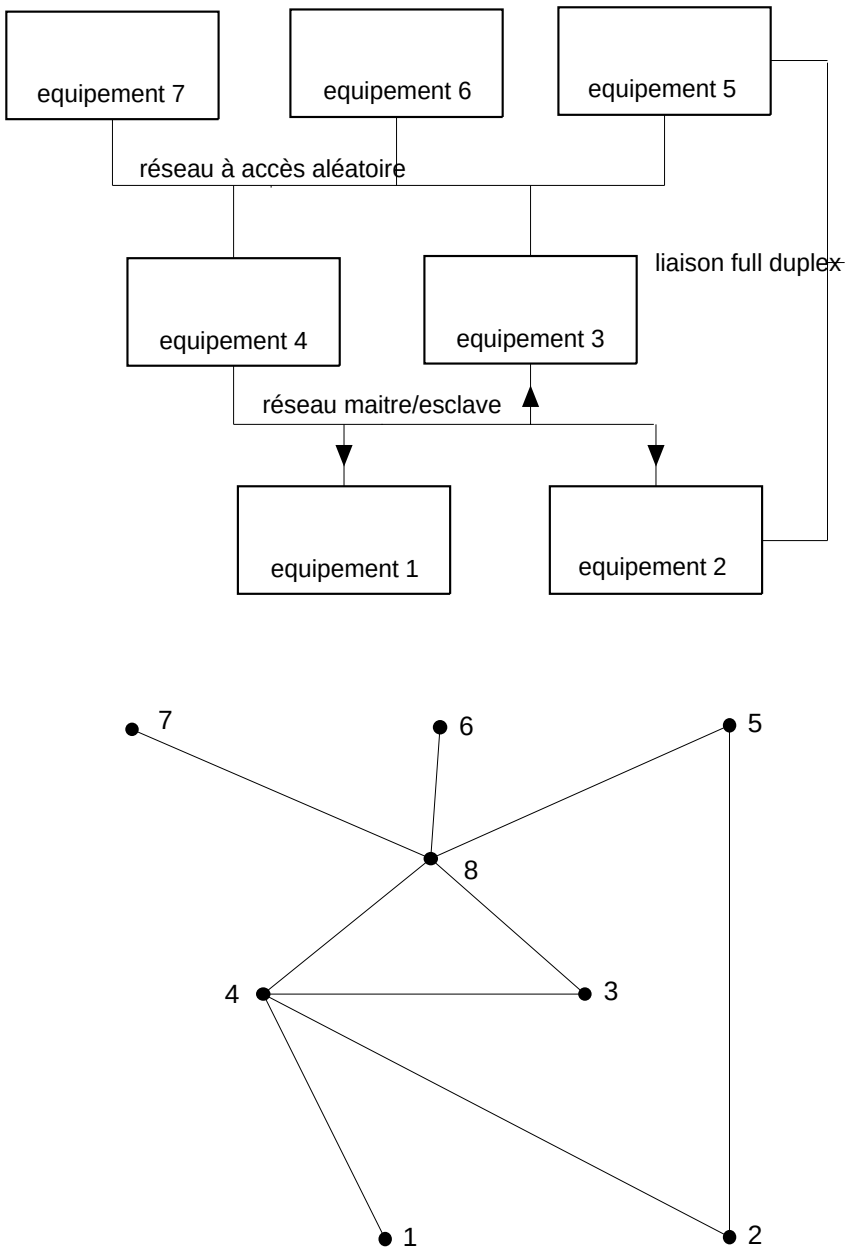


Figure 41 : exemple de modèle organique fortement interconnecté avec son graphe associé

La figure 41 présente une architecture très interconnectée, dans laquelle la transmission d'un message de l'équipement de traitement 7 (somment 7 du graphe) vers l'équipement de traitement 2 (somment 2 du graphe) peut suivre plusieurs trajets.

Cela se traduit sur le graphe associé par l'existence de trois chemins propres qui sont proposés à l'architecte pour qu'il effectue son choix :

- $chemin_1 = ([7,8], [8,4], [4,2])$
- $chemin_2 = ([7,8], [8,3], [3,4], [4,2])$
- $chemin_2 = ([7,8], [8,5], [5,2])$

Par contre, si l'architecte décide de choisir systématiquement le trajet propre le plus court selon un critère de son choix, il lui suffit d'affecter des poids aux arêtes en relation avec le critère d'évaluation qu'il a choisi. Par exemple, en retenant comme critère de choix d'un trajet, la plus grande vitesse de transmission, l'architecte va associer un poids identique p_1 aux arêtes $[7,8]$, $[8,3]$, $[8,4]$, et $[8,5]$ car il correspond au même équipement de communication[1], il va associer le poids p_2 à l'arête $[4,2]$, et il va associer le poids p_3 à l'arête $[5,2]$, avec :

$$p_1 = \frac{1}{2 \cdot vitesse_1}; p_2 = \frac{1}{vitesse_2}; p_3 = \frac{1}{vitesse_3}$$

Soit L_i la longueur du $chemin_i$, dans notre exemple on a :

$$L_1 = 2p_1 + p_2 \quad (3)$$

$$L_2 = 2p_1 + 2p_2 \quad (4)$$

$$L_3 = 2p_1 + p_3 \quad (5)$$

Le chemin $chemin_i$ est le chemin minimum si et seulement si la relation (6) est vérifiée

$$L_i = min(L_1, L_2, L_3) \quad (6)$$

Une fois que l'architecte a à sa disposition le trajet des flots, il faut identifier les contraintes techniques à mettre en place pour assurer leur circulation. A ce niveau d'avancement dans la démarche, l'architecture est suffisamment formalisée pour permettre une détermination systématique des traitements techniques.

---

1. La stratégie d'affectation nous ayant conduit à associer un sommet aux réseaux à accès aléatoire, une arête telle que [7,8] ne représente que la moitié du trajet effectué par un flot dans le réseau. Le poids que l'on souhaiterait affecter au réseau lui-même devra être corrigé d'un facteur 2 avant d'être affecté aux arêtes représentant les demi-trajets.

Dans un premier temps nous proposons de construire un chemin réduit pour lequel les sommets seront des équipements de traitement et les arêtes seront supportées par des équipements de communication. Le but de cette opération est le filtrage des sommets associés aux équipements de communication. Sur ce graphe réduit on mettra également en évidence les méthodes d'accès afin de savoir quels sont les sommets qui ont l'initiative dans les communications. Pour deux équipements de traitement connectés par un réseau à accès aléatoire ou une liaison full duplex, l'initiative des communications est double, alors que pour deux équipements connectés par un réseau maître-esclave ou une liaison half duplex, un seul d'entre eux à l'initiative des communications. Le chemin $([7,8],[8,3],[3,4],[4,2])$ de la figure 41 admet pour chemin réduit le chemin de la figure 42.

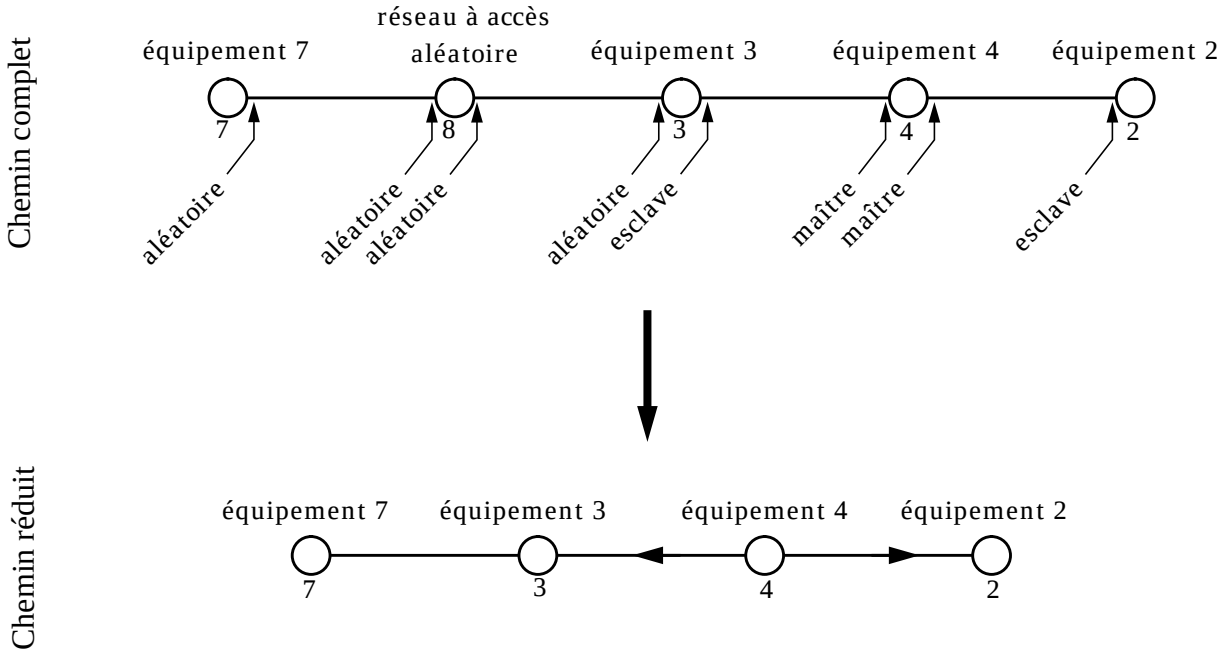


Figure 42 : exemple de chemin réduit pour un flot

Dans un deuxième temps, à partir du chemin réduit que doit suivre le flot, et en fonction de la nature de l'émetteur et du destinataire (traitement ou stockage), les traitements techniques et les flots techniques, sont construits de proche en proche en suivant les arêtes du chemin. On débute la détermination en plaçant l'émetteur dans l'équipement du premier sommet du chemin et l'on fait comme si le destinataire devait être implanté dans le sommet 2. Le tableau 3 permet d'identifier les traitements techniques et les flots techniques induits par le franchissement de l'arête

[1,2]. Lorsque cette opération est terminée, on déporte le destinataire dans le sommet 3, et l'on recommence l'opération d'identification à l'aide du tableau 3. D'une manière générale, lorsque les traitements techniques et les flots techniques ont été identifiés du sommet 1 jusqu'au sommet i, on déporte le destinataire dans le sommet i+1, et l'on détermine avec le tableau 3 (fig. 43).

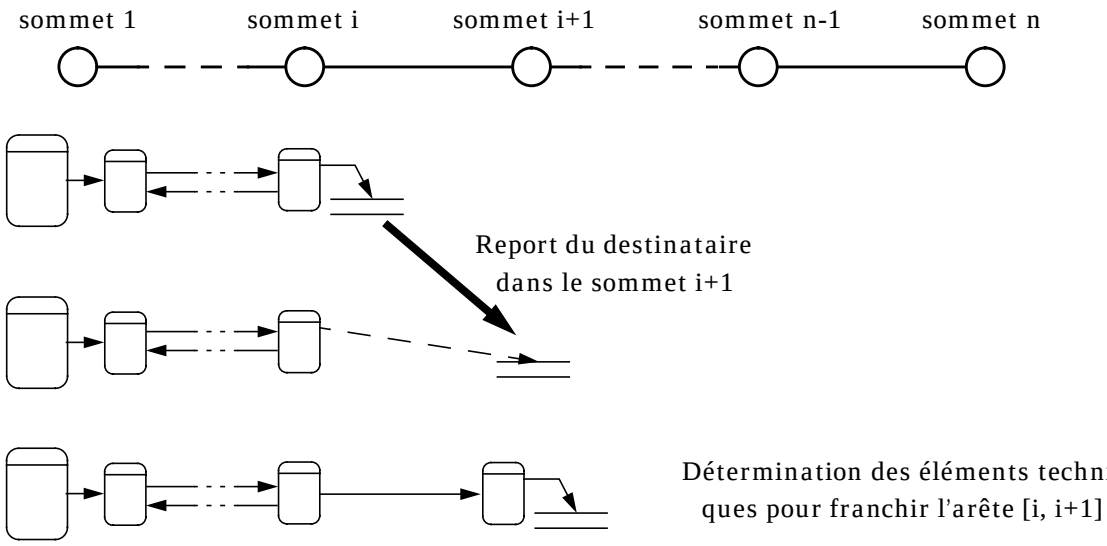


Figure 43 : processus de détermination des éléments techniques

Pour chaque arête, les besoins en traitement technique diffèrent en fonction de la nature de la méthode d'accès au media support de l'arête. Deux familles d'arêtes sont à prendre en considération. Celle pour laquelle les deux sommets peuvent indifféremment avoir accès au media, et celle pour laquelle un seul des deux sommets a l'initiative des communications, c'est la méthode d'accès maître-esclave. Le tableau 3 fait l'inventaire des problèmes et présente les solutions associées.

Tableau 3 : les traitements techniques induits des choix d'affectation

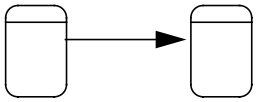
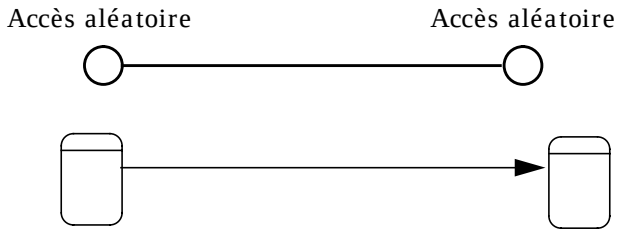
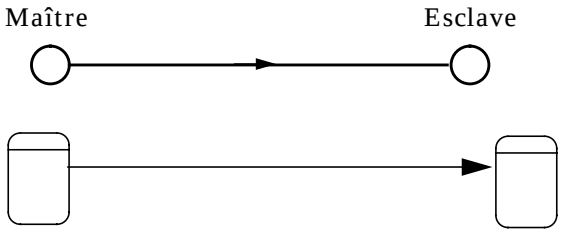
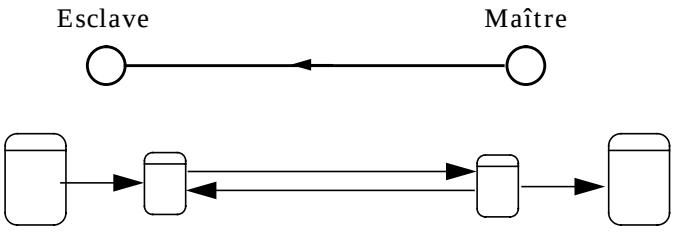
<table>
<tr><th>Emetteur-destinataire</th><th>Eléments techniques nécessaires</th></tr>
<tr><td rowspan="3"> 
L'émetteur et le destinataire sont tous les deux des traitements</td><td>La transmission d'un message entre deux traitements répartis sur deux équipements connectés par un media à accès aléatoire (ou full duplex) ne nécessite pas l'addition d'éléments techniques.
  </td></tr>
<tr><td>La transmission d'un message entre deux traitements, depuis un équipement maître vers un équipement esclave, ne nécessite pas l'addition d'éléments techniques.
  </td></tr>
<tr><td>La transmission d'un message entre deux traitements, depuis un équipement esclave vers un équipement maître, nécessite l'addition d'un traitement de scrutation dans le maître, et d'un traitement de réponse dans l'esclave.
  </td></tr>
</table>

Tableau 3 : les traitements techniques induits des choix d'affectation

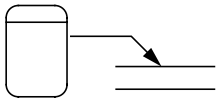
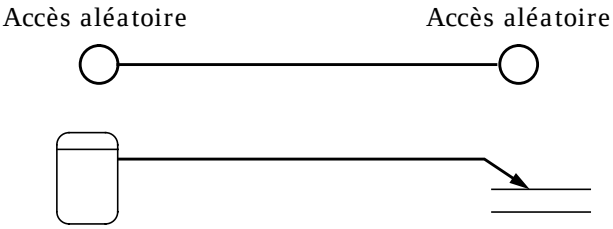
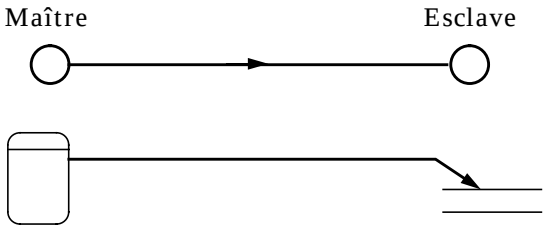
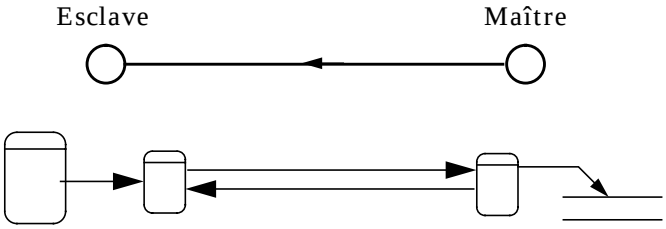
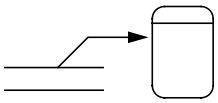
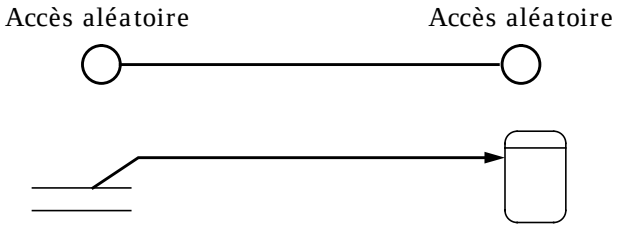
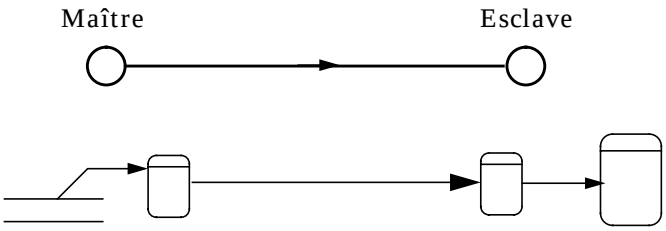
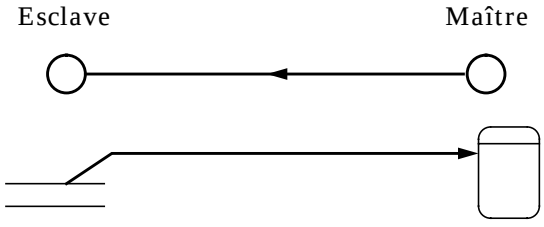
Emetteur-destinataire	Eléments techniques nécessaires
 L'émetteur est un traitement et le destinataire est un stockage de donnée	La transmission d'une donnée entre un traitement et un stockage répartis sur deux équipements connectés par un media à accès aléatoire (ou full duplex) ne nécessite pas l'addition d'éléments techniques. 
	La transmission d'une donnée, depuis un traitement affecté dans un équipement maître vers un stockage affecté dans un équipement esclave, ne nécessite pas l'addition d'éléments techniques. 
	La transmission d'une donnée, depuis un traitement affecté dans un équipement esclave vers un stockage affecté dans un équipement maître, nécessite l'addition d'un traitement de scrutation dans le maître, et d'un traitement de réponse dans l'esclave. 

Tableau 3 : les traitements techniques induits des choix d'affectation

<table>
<tr><th>Emetteur-destinataire</th><th>Eléments techniques nécessaires</th></tr>
<tr><td rowspan="3"> 
L'émetteur est un stockage de donnée, et le destinataire est un traitement.</td><td>La transmission d'une donnée entre un stockage et un traitement répartis sur deux équipements connectés par un media à accès aléatoire (ou full duplex) ne nécessite pas l'addition d'éléments techniques.
  </td></tr>
<tr><td>La transmission d'une donnée, depuis un traitement affecté dans un équipement maître vers un stockage affecté dans un équipement esclave, nécessite l'addition d'un traitement de scrutation dans le maître, et d'un traitement de réponse dans l'esclave.
  </td></tr>
<tr><td>La transmission d'une donnée, depuis un stockage affecté dans un équipement esclave vers un traitement affecté dans un équipement maître, ne nécessite pas l'addition d'éléments techniques.
  </td></tr>
</table>

On peut dégager quelques points importants des pages qui viennent d'être consacrées à la détermination des traitements techniques et des flots techniques :

- un grand nombre de traitements techniques apparaissent dans une architecture de conduite répartie, il est donc important de les prendre en compte,

- malgré leur nombre, les traitements techniques restent simples et donc peu consommateurs de la ressource CPU de leurs équipements de traitement, en revanche ils génèrent un trafic de communication important qui semble non négligeable dans la conception de l'architecture,
- l'augmentation de trafic dû aux flots techniques est particulièrement importante au niveau des réseaux maître-esclave, or il sont généralement utilisés au niveau des communications entre automates programmables industriels, là où justement les temps de réponse imposés sont les plus sévères.

Même si l'étude des traitements techniques peut paraître longue et fastidieuse à l'architecte, elle est un point de passage important pour prendre en compte toutes les incidences d'un choix de répartition dans une architecture de conduite. Le caractère systématique de notre étude, la rend automatisable donc transparente pour un architecte utilisateur.

1.4 Evaluation et choix d'architecture

Le chapitre 1.3 «Construction d'une solution d'architecture» a décrit les activités conduisant à l'élaboration de solutions d'architecture non évaluées qui se présentent sous la forme de modèle d'affectation. Nous nous proposons dans ce chapitre de montrer l'intérêt d'avoir formalisé le processus d'élaboration de solutions lors de la phase d'évaluation des architectures.

Deux niveaux d'évaluation sont proposés dans ce chapitre : un premier basé sur des données statistiques fournies par l'architecte, et le second basé sur une description du comportement dynamique du modèle d'affectation.

1.4.1 Evaluation statistique

Pour le premier niveau d'évaluation, élaborer un modèle c'est essentiellement enrichir les modèles d'implantation et d'affectation par l'ajout de caractéristiques aux entités de ces modèles. Relativement au modèle d'implantation, l'architecte doit renseigner :

- pour chaque traitement procédural t, le nombre moyen d'instructions élémentaires noté $n_i(t)$ que devra exécuter un équipement de traitement pour que le traitement procédural soit réalisé, ainsi que la fréquence $F_T(t)$ d'exécution du traitement,
- pour chaque traitement de contrôle-commande t, le nombre moyen de kilo-mots noté $n_{mot}(t)$ que devra exécuter un automate programmable industriel pour que le traitement de contrôle-commande soit évalué une fois, ainsi que $P_{exe}(t)$, la période d'exécution souhaitée pour les traitements de contrôle-commande qui sont affectés dans un calculateur, et non pas dans un automate programmable industriel
- pour chaque flot f, la fréquence d'accès $F_a(f)$, et la taille moyenne des données qui y circulent $T_d(f)$.

Relativement au modèle d'affectation, l'architecte doit renseigner :

- pour chaque traitement technique de type scrutation t, la période de scrutation notée $P_{sc}(t)$.

Avec ces nouvelles informations, la fréquence et la taille des données qui circulent sur les flots techniques sont déduites du modèle d'affectation. Un traitement qui relait la transmission d'un flot à travers un équipement de traitement, retransmet une donnée de taille $T_d(f)$ identique à la donnée retransmise, et à la même fréquence $F_a(f)$. De proche en proche un ensemble de flots du modèle d'affectation est ainsi caractérisé. Il ne reste plus qu'à déterminer la taille d'un message technique de scrutation. En effet, lorsqu'un traitement de scrutation interroge son interlocuteur, il transmet des informations dont la taille est indépendante de la taille de la donnée que l'on souhaite transmettre par cette technique de scrutation maître-esclave. Tous les traitements techniques de scrutation et de réponse à une scrutation sont caractérisés par une taille de message transmis unique noté T_{sc}, et fixée une fois pour toutes.

Pour être tout à fait complet vis à vis de la connaissance des caractéristiques de tout ce qui consomme du temps CPU et de tout ce qui circule dans les media, il faut encore caractériser la charge supplémentaire imposée à un processeur pour assurer un accès à un système de gestion de fichiers, ou de gestion de base de données. Nous appellerons $n(e)$, le nombre moyen d'instructions que devra exécuter l'équipement de traitement qui supporte l'équipement de stockage $e \in E_s$, pour assurer l'accès à une donnée, et ce, quelque soit sa taille. Cette information n'est pas facile à déterminer, elle devra être identifiée par l'architecte sur des architectures qu'il a déjà réalisée.

L'ensemble de ces informations collectées, est une partie du modèle d'évaluation. Il est dès à présent possible de proposer des indicateurs sur l'architecture en cours de conception. Il s'agit de la charge de chaque constituant de l'architecture organique.

1.4.1.1 La charge d'un calculateur

La charge prévisionnelle $C(p)$ d'un équipement de traitement de type calculateur $p \in E_{cal}$ est obtenue de la façon suivante.

- soit $T_p(p)$ l'ensemble des traitements procéduraux affectés dans p
- soit $T_{cc}(p)$ l'ensemble des traitements de contrôle-commande affectés dans p

- soit $E_s(p)$ l'ensemble des équipements de stockage supportés par l'équipement de traitement p
- soit $D_{E_s}(e)$ l'ensemble des données affectées à l'équipement de stockage e
- soit $F(d)$ l'ensemble des flots accédant à la donnée d

La charge due aux traitements procéduraux est exprimée dans la formule (7), la charge due aux traitements de contrôle-commande est exprimée dans la formule (8), la charge due aux équipements de stockage associés est exprimée dans la formule (9).

$$C_{T_p}(p) = \sum_{t_i \in T_p(p)} (n_i(t_i) \times F_T(t_i)) \quad (7)$$

$$C_{T_{cc}}(p) = \sum_{t_i \in T_{cc}(p)} (n_{mot}(t_i) \times P_{exe}(t_i)) \quad (8)$$

$$C_{E_s}(p) = \sum_{e_k \in E_s(p)} \left(n(e_k) \times \sum_{d_j \in D_{E_s}} \left(\sum_{f_i \in F(d_j)} F_a(f_i) \right) \right) \quad (9)$$

La charge totale d'un équipement de traitement de type calculateur, $C(p)$ est exprimée dans la formule (10), tandis que la formule (11) donne le pourcentage d'exécution du CPU consommé par les entités non fonctionnelles, c'est-à-dire les fonctions techniques.

$$C(p) = C_{T_p}(p) + C_{T_{cc}}(p) + C_{E_s}(p) \quad (10)$$

$$\frac{C_{E_s}(p)}{C_{T_p}(p) + C_{T_{cc}}(p) + C_{E_s}(p)} \times 100 \quad (11)$$

1.4.1.2 Le temps de cycle d'un API

Le temps de cycle prévisionnel T_{cycle} d'un équipement de traitement de type automate programmable industriel $p \in E_{API}$, est obtenu de la façon suivante.

- soit $T_{cc}(p)$ l'ensemble des traitements de contrôle-commande affectés dans p
- soit $T_p(p)$ l'ensemble des traitements procéduraux affectés dans p
- soit T_{cycle} le temps de cycle de l'API pour effectuer tous ces traitement, et V la «vitesse d'exécution» de l'API en nombre d'instructions exécutées par unité de temps.

La formule (12) exprime la charge due au traitement procéduraux dans l'API, et la relation (13) donne T_{cycle} en fonction de la vitesse de traitement V de l'API.

$$C_{fond} = \sum_{t_i \in T_p(p)} (n_i(t_i) \times F_T(t_i)) \quad (12)$$

$$T_{cycle}(V) = \frac{1}{V - C_{fond}} \times \left(\sum_{t_i \in T_{cc}(p)} n_{mot}(t_i) \right) \quad (13)$$

La relation (13) permet d'obtenir une valeur prévisionnelle du temps de cycle de l'API à partir du choix d'une gamme d'automate programmable industriel, donc de la connaissance de l'ordre de grandeur de sa vitesse d'exécution. Inversement, une contrainte de temps de réponse, donc la connaissance du temps de cycle souhaité pour l'API, permet d'obtenir la gamme dans laquelle il faudra choisir l'API. Bien entendu, la mise en évidence de contraintes de performances non disponibles sur le marché des API entraînera une remise en cause de l'architecture de conduite, en particulier de son modèle organique.

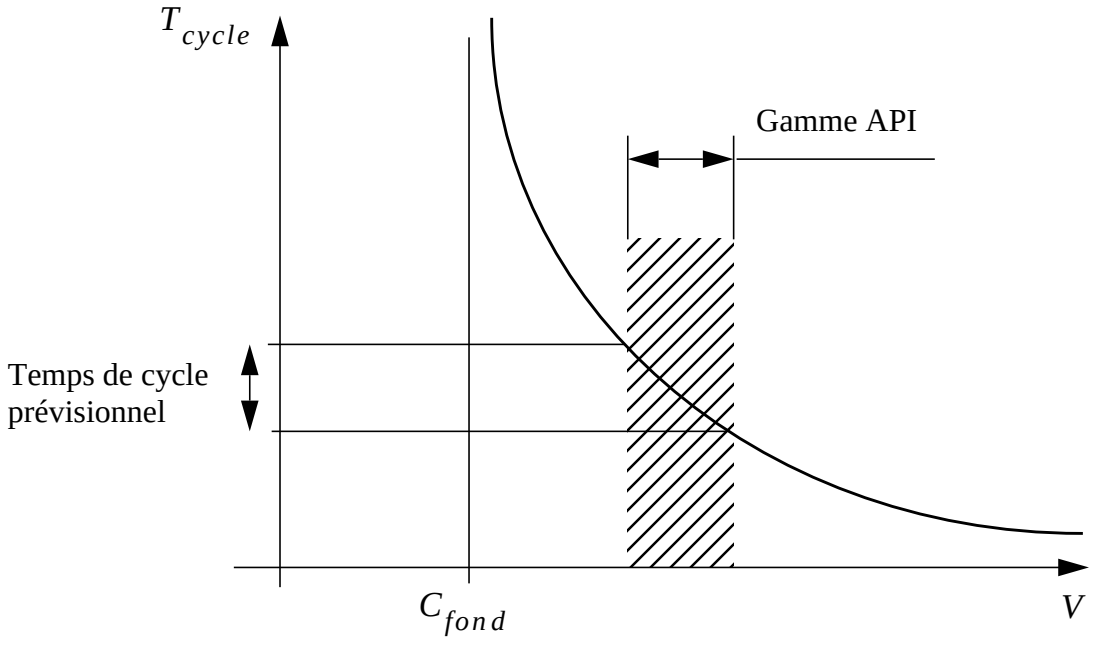


Figure 44 : détermination du temps de cycle d'un API à partir de sa vitesse de traitement

1.4.1.3 La charge d'un équipement de communication à accès aléatoire

La charge $C(e)$ d'un équipement de communication à accès aléatoire $e \in E_{com}$ est obtenue de la façon suivante.

- soit $F(e)$ l'ensemble des flots de donnée qui circulent dans $e \in E_{com}$

La relation (14) donne la charge de e. Comme pour les automates programmables industriels, la mise en évidence d'un taux de charge supérieur aux performances des équipements de communication disponibles sur le marché, entraînera une remise en cause de l'architecture de conduite au niveau du modèle organique.

$$C(e) = \sum_{f_i \in F(e)} F_a(f_i) \times T_d(f_i) \tag{14}$$

1.4.1.4 La charge d'un équipement de communication à accès maître-esclave

La charge $C(e)$ d'un équipement de communication à accès maître-esclave $e \in E_{cma}$ est obtenue de la façon suivante.

- soit $F(e)$ l'ensemble des flots de donnée qui circulent dans $e \in E_{cma}$,
- soit $F_f(e)$ le sous-ensemble de $F(e)$ constitué des flots fonctionnels qui véhiculent les données,
- soit $F_t(e)$ le sous-ensemble de $F(e)$ constitué des flots techniques périodiques issus des traitements de scrutation et de réponse,
- soit $f \in F_t(e)$ on notera $t_{sc}(f)$ le traitement technique de scrutation dont f dépend.

Les sous ensembles, $F_f(e)$ et $F_t(e)$ vérifient les propriétés (15) et (16).

$$F(e) = F_f(e) \cup F_t(e) \tag{15}$$

$$F_f(e) \cap F_t(e) = \varnothing \tag{16}$$

La charge totale $C(e)$ est exprimée dans la formule (17), tandis que (18) donne le pourcentage de communication technique par rapport à l'ensemble du trafic. Une fois encore, la mise en évidence d'un taux de charge supérieur aux performances des

équipements de communication disponibles sur le marché, entraînera une remise en cause de l'architecture de conduite au niveau du modèle organique.

$$C(e) = \sum_{f_i \in F_f(e)} (F_a(f_i) \times T_d(f_i)) + \sum_{f_i \in F_t(e)} \frac{T_{sc}}{P_{sc}(t_{sc}(f_i))} \quad (17)$$

$$\frac{\sum_{f_i \in F_t(e)} \frac{T_{sc}}{P_{sc}(t_{sc}(f_i))}}{\sum_{f_i \in F_f(e)} (F_a(f_i) \times T_d(f_i)) + \sum_{f_i \in F_t(e)} \frac{T_{sc}}{P_{sc}(t_{sc}(f_i))}} \times 100 \quad (18)$$

Nous avons proposé quelques indicateurs pour mettre en évidence l'intérêt de la formalisation du cœur de l'architecture en phase d'évaluation. Ces indicateurs ne sont pas exhaustifs et selon les besoins de chaque architecte, de nouveaux indicateurs peuvent être établis.

1.4.2 Evaluation dynamique du comportement du modèle d'affectation

Après avoir abordé un premier niveau d'évaluation statistique, examinons maintenant le second niveau d'évaluation. Pour cela, le choix du formalisme a été conduit par la nature des évaluations dont un architecte peut avoir besoin pour valider et choisir une solution. Les critères couramment évoqués par les professionnels [52] sont les *temps de réponse* et les *taux de charge* des équipements de traitement et les *débits* dans les équipements de communication. Afin de fournir des indications sur les temps de réponse, le modèle d'évaluation doit être un modèle du comportement dynamique de l'architecture. Notre choix c'est porté sur les *réseaux de Petri temporisés et colorés*, car la temporisation rend le modèle simulable, donc évaluable. La coloration permettra d'accroître la concision et la généricité des modèles élaborés.

Le *modèle évaluable* se construit à partir du modèle d'affectation qui traduit à la fois l'aspect fonctionnel de l'architecture issu des spécifications au travers du modèle d'implantation, et à la fois l'aspect organisation au travers des traitements techniques (traitements de scrutation, ...). L'originalité de ce modèle est de tenir compte de ces deux aspects. Dans de nombreux travaux, des descriptions de la dynamique d'une spécification de système ont été proposées [53] [54] [55], et dans d'autres tra-

vaux des modèles du comportement dynamique de l'architecture organique sont étudiés [56] [57] [58] [59]. Le modèle d'évaluation que nous proposons permet quant à lui de décrire le *comportement dynamique du modèle d'implantation projeté sur le modèle organique.*

La démarche proposée est la suivante :

- chaque traitement et chaque stockage du modèle d'affectation vont voir leur comportement dynamique décrit par un réseau de Petri générique dont l'évolution interne est conditionnée par l'apparition de messages, et subordonnée par des demandes de ressources (telles que la lecture d'une donnée dans un stockage, ou telles que la demande de temps CPU à un équipement de traitement support) dont l'évolution d'un état de ce RdP à un autre peut à son tour générer des envois de messages ;
- chaque équipement, qu'il soit de traitement ou de communication, va être décrit dynamiquement par un réseau de Petri générique le décrivant comme une ressource prête à répondre aux sollicitations des traitements et des stockages ;
- constitution du modèle évaluable par assemblage des différents réseaux de Petri génériques.

Les réseaux de Petri produits dans la suite de cette division ne se veulent pas être les modèles les plus complets, mais les plus pertinents pour l'étude de l'architecture. Il ont été élaborés pour refléter le comportement de toute une architecture, et non pour étudier précisément un type d'équipement de l'architecture organique, ou une partie de la spécification fonctionnelle du système. Le lecteur trouvera des modèles plus fins concernant par exemple les réseaux dans [60] ou concernant par exemple les calculateurs dans [61].

Les modèles que nous proposons ont été conçus de manière à être les plus génériques possibles. Ils constituent une bibliothèque que l'architecte peut compléter ou modifier, et dans laquelle il va venir chercher les éléments nécéssaires à la constitution de la représentation dynamique de son modèle d'implantation projeté sur son modèle organique.

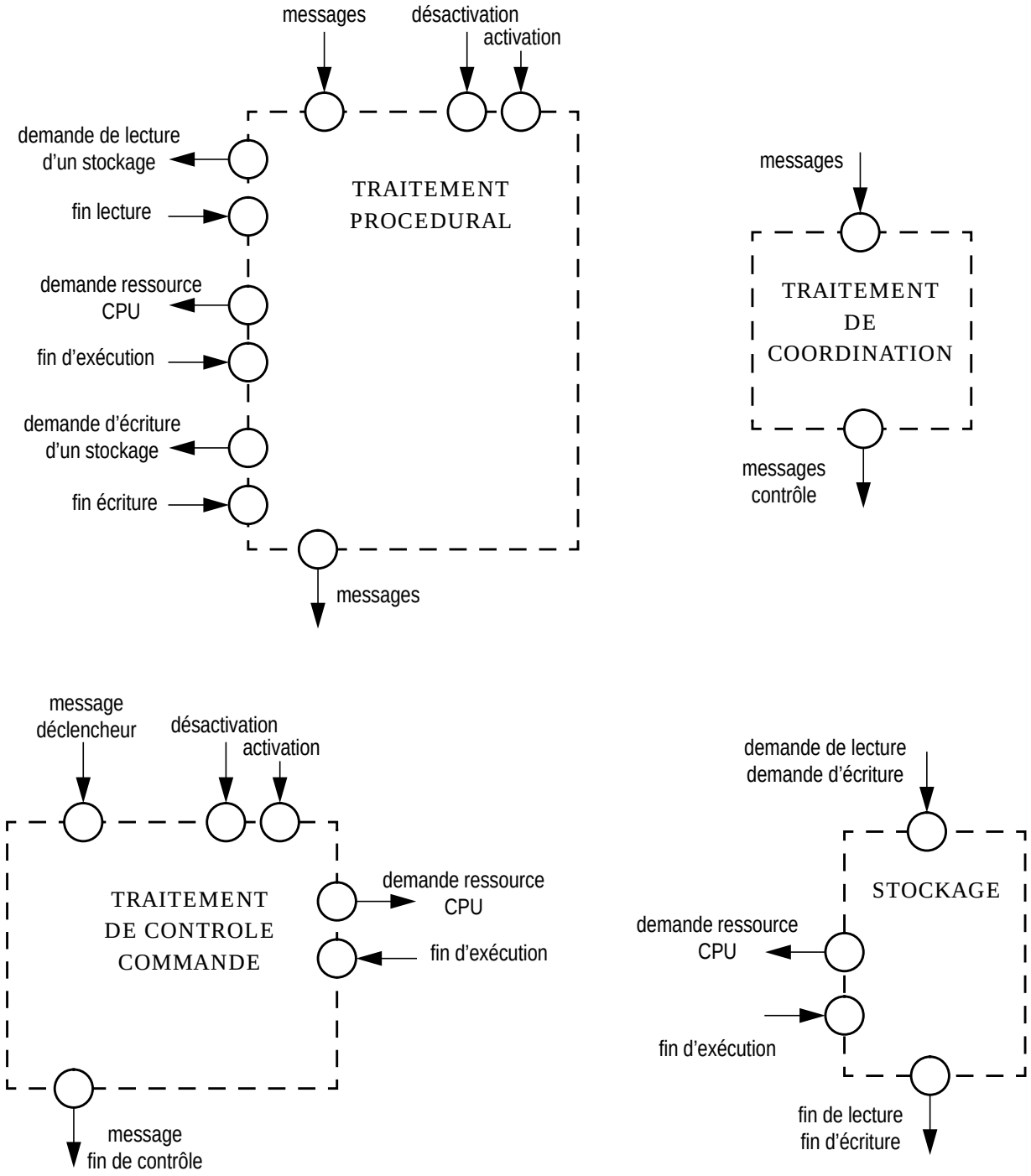


Figure 45 : les entrées-sorties des modèles dynamiques associés aux entités du modèle d'implantation

1.4.2.1 Les modèles génériques de comportement des éléments du modèle d'implantation

La figure 45 présente les entrées-sorties des modèles dynamiques génériques de comportement des entités du formalisme du modèle d'implantation.

1.4.2.1.1 Les traitements procéduraux

Le modèle des traitements procéduraux décrit tout $t_i \in T_p$, où T_p est l'ensemble des traitements procéduraux du modèle d'implantation (fig. 46). Il est constitué de deux parties relativement distinctes, l'une modélise la cinématique du traitement et l'autre modélise son état d'activation. Pour la partie cinématique, le modèle décrit le déclenchement d'une conjonction (P1, et T1), puis la lecture successive de toutes les données (P2, P3, P4, P5, P6, P7, T2, T3, T4, T5 et T6), puis une demande de ressource CPU (P28, P29 et T7), puis le déclenchement d'une disjonction (P8, T8 et T10), puis l'écriture de toutes les données associées à la disjonction (P9, P10, P11, P12, P13, P30, T11, T12 et T13), et enfin l'émission des messages associés à la disjonction (P14 et T14). Pour la partie décrivant l'état d'activation, le modèle propose pour chaque traitement, un état actif P22, un état pour une désactivation en cours P23, et un état désactivé P24. La gestion des demandes d'activation (P19, P20, P21, T17, T18 et T19) et des demandes de désactivation (P16, P17, P18, T15, T16 et T20) interdit toute accumulation des demandes. Examinons point par point les éléments du modèle inventorié ci-dessus.

• Déclenchement d'une conjonction

Lorsque le traitement t_i est déclenché suite à l'apparition d'une conjonction c_j d'événements, la transition T1 est franchie vis-à-vis de la couleur $\langle t_i, c_j \rangle$. Les marques de couleur $\langle m_i \rangle$, représentant les messages participant à la conjonction c_j, sont alors retirées de la place P1. Une marque $\langle t_i \rangle$ est alors placée dans P2, montrant que le traitement vient d'être déclenché.

• Lecture successive de toutes les données

Si le traitement t_i n'a pas besoin de lire des données avant de s'exécuter, alors la transition T4 est franchie, et la phase de lecture des données est terminée, sinon la transtion T5 est franchie et la séquence de lecture des données est prête à être déroulée. Une démarche de lecture de donnée dans un stockage se traduit par le dépot d'une marque $\langle t_i, s_j, l \rangle$ dans la place P4 ($\langle t_i, s_j, l \rangle$: le traitement t_i, demande au stockage s_j une lecture l de son contenu). La présence d'une marque $\langle t_i \rangle$ dans P7 indique que le traitement t_i a encore des stockages à lire, et le marquage de la place P3 indique, pour chaque traitement t_i, le prochain stockage s_j à lire ($\langle t_i, s_j \rangle$: le traitement t_i devra lire le stockage s_j). Lorsque qu'une demande de lecture est

satisfaite, une marque $\langle t_i, s_j, l \rangle$ se retrouve dans la place P5. S'il s'agit de la dernière donnée lue, alors la transition T6 est franchie, sinon la transition T3 est franchie en déposant une marque $\langle t_i \rangle$ dans la place P7 (le traitement t_i a encore des stockages à lire) et une marque $\langle t_i, s_{j+1} \rangle$ dans la place P3 (le prochain stockage à lire est s_{j+1}).

• Demande de ressource CPU

Lorsque toutes les données nécessaires au traitement t_i ont été lues, ce dernier effectue une demande de ressource CPU en plaçant une marque $\langle t_i \rangle$ dans la place P28. La fin d'exécution du traitement génère une marque $\langle t_i \rangle$ dans la place P29.

• Déclenchement d'une disjonction

Selon son déroulement un traitement t_i va émettre un certaine disjonction d_i. La fonction f_7 détermine la conjonction de fin d'exécution lors du franchissement de la transition T8. Une marque $\langle t_i, d_j \rangle$ dans la place P8 représente le traitement t_i en fin d'exécution, et prêt à émettre la disjonction d_i.

• Ecriture de toutes les données associées à la disjonction

L'écriture des données associées à une disjonction se déroule selon un principe similaire à la lecture des données d'une conjonction. Après avoir vérifié que la disjonction nécessite l'écriture de données (alternative entre les transitions T10 et T9), les demandes d'écriture dans les stockages sont émises successivement en plaçant une marque $\langle t_i, s_k, e \rangle$ dans la place P10. Il est à noter que le jeton $\langle t_i, s_k, e \rangle$ est banalisé vis-à-vis de la disjonction dont il est issu. En effet le stockage n'a pas à connaître plus que la référence du traitement émetteur, sur l'origine de la requête d'écriture qui lui est demandé. La place P10 assure la mémorisation de la disjonction courante pour chaque traitement. Lorsque toutes les données sont écrites, une marque $\langle t_i, d_j \rangle$ se trouve dans la place P13.

• Emission des messages associés à la disjonction

Le franchissement de la transition T14 assuse la transformation de la marque $\langle t_i, d_j \rangle$ dans la place P13 en un ensemble de marques $\langle m_k \rangle$ correspondant aux messages à émettre.

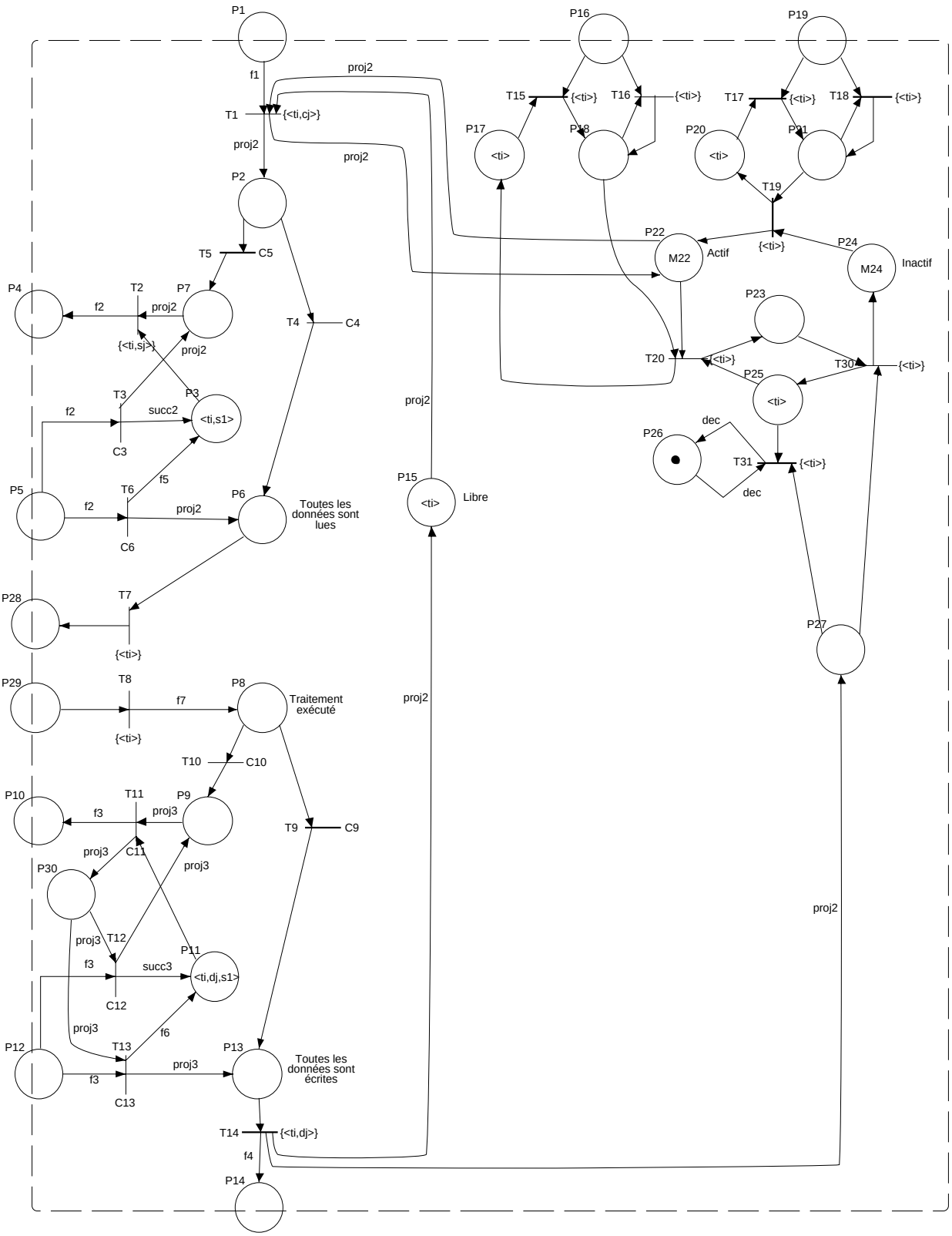


Figure 46 : modèle dynamique des traitements procéduraux

Tableau 4 : description du modèle dynamique des traitements procéduraux

Libellé	Description
f1	$f_1(\langle t_i, c_j\rangle) = \sum_{m_k \in C_j} \langle m_k\rangle$
f2	$f_2(\langle t_i, s_j\rangle) = \langle t_i, s_j, l\rangle$
f3	$f_3(\langle t_i, d_j, s_k\rangle) = \langle t_i, s_k, e\rangle$
f4	$f_4(\langle t_i, d_j\rangle) = \sum_{m_k \in M(d_j)} \langle m_k\rangle$
f5	$f_5(\langle t_i, s_j\rangle) = \langle t_i, s_1\rangle$
f6	$f_6(\langle t_i, d_j, s_k\rangle) = \langle t_i, d_j, s_1\rangle$
f7	f7 est la fonction qui tire une marque dans la population $\{\langle t_i, d_j\rangle \mid d_j \in D(t_i)\}$ avec une probabilité de tirage de la marque $\langle t_i, d_j\rangle$ égale à $P(\langle t_i, d_j\rangle)$ respectant la contrainte $\sum_{d_k \in D(t_i)} P(\langle t_i, d_k\rangle) = 1$
C3	$\{\langle t_i, s_j\rangle \mid s_j \in S_c(t_i), j \in [1, \text{card}(S_c(t_i))[\}$
C4	$\{\langle t_i\rangle \mid S_c(t_i) = \varnothing\}$
C5	$\{\langle t_i\rangle \mid S_c(t_i) \neq \varnothing\}$
C6	$\{\langle t_i, s_j\rangle \mid s_j \in S_c(t_i), j = \text{card}(S_c(t_i))\}$
C9	$\{\langle t_i, d_j\rangle \mid d_j \in D(t_i), S_e(d_j) = \varnothing\}$
C10	$\{\langle t_i, d_j\rangle \mid d_j \in D(t_i), S_e(d_j) \neq \varnothing\}$

Tableau 4 : description du modèle dynamique des traitements procéduraux

Libellé	Description
C11	$\{\langle t_i, d_j, s_k\rangle \mid d_j \in D(t_i), s_k \in S_e(d_j)\}$
C12	$\{\langle t_i, d_j, s_k\rangle \mid d_j \in D(t_i), s_k \in S_e(d_j), k \in [1, \mathrm{card}(S_e(d_j))[\}$
C13	$\{\langle t_i, d_j, s_k\rangle \mid d_j \in D(t_i), s_k \in S_e(d_j), k = \mathrm{card}(S_e(d_j))\}$
M3	$\sum_{t_i \in T_p} \langle t_i, s_1\rangle$
M11	$\sum_{t_i \in T_p} \sum_{d_j \in D(t_i)} \langle t_i, d_j, s_1\rangle$
M22, M24	$M22 + M24 = \sum_{t_i \in T_p} \langle t_i\rangle$
M15, M17, M20, M25	$\sum_{t_i \in T_p} \langle t_i\rangle$

1.4.2.1.2 Les traitements de contrôle-commande

Le modèle des traitements de contrôle-commande décrit tout $t_i \in T_{cc}$, où T_{cc} est

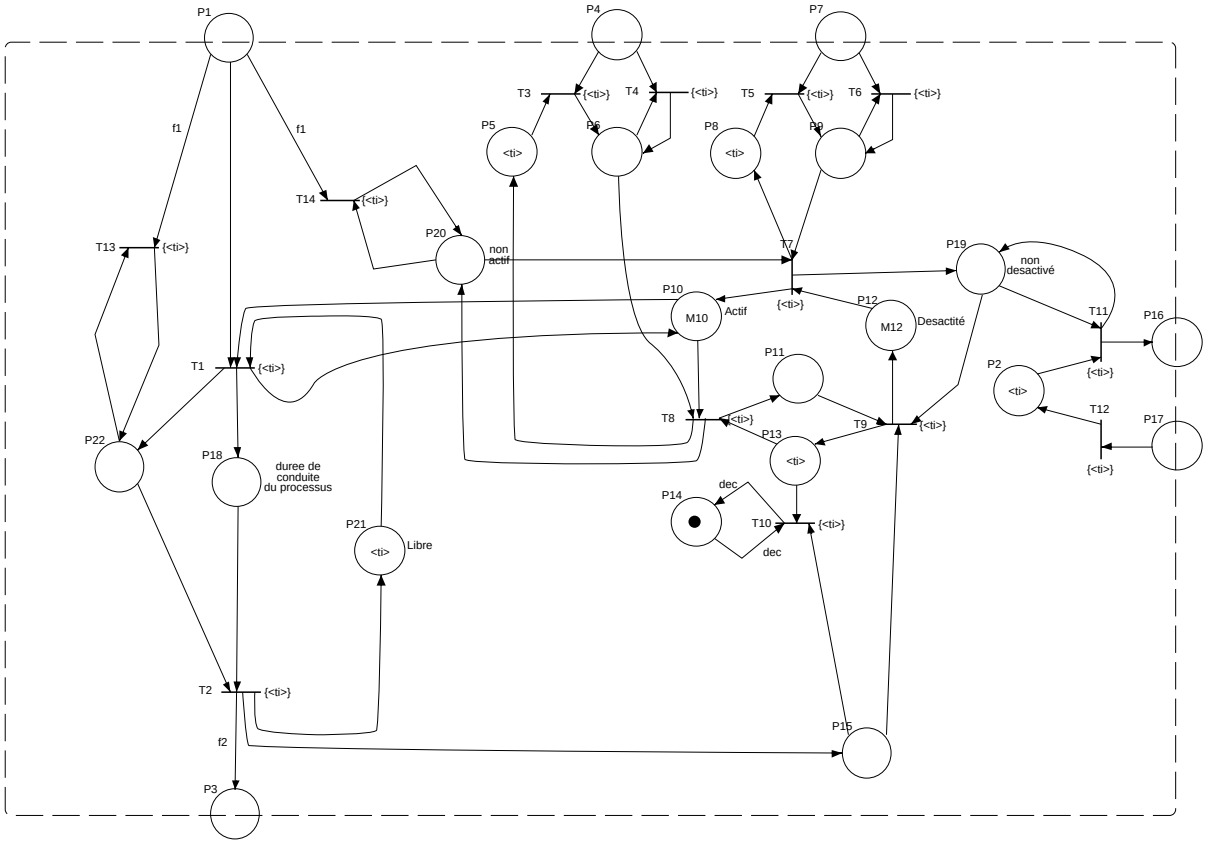


Figure 47 : modèle dynamique des traitements contrôle-commande

l'ensemble des traitements de contrôle-commande du modèle d'implantation. Il est organisé autour de la description de son état d'activation (P10, P11, P12, T7, T8 et T9), en amont duquel une gestion des demandes d'activation (P7, P8, P9, T5 et T6) et des demandes de désactivation (P4, P5, P6, T3 et T4) interdit toute accumulation des demandes. Lorsqu'ils sont actifs, les traitements de contrôle-commande peuvent conduire des processus s'ils en reçoivent l'ordre $m_t(t_i)$, et jusqu'à ce qu'ils aient terminé en émettant $m_f(t_i)$ (P1, P18, P3, T1 et T2). Une gestion du déclenchement en interdit toute accumulation (P20, P22, T13 et T14). Lorsqu'ils ne sont pas inactivés les traitements de contrôle-commande demandent en permanence une ressource CPU (P2, P16, P17, T11 et T12).

Tableau 5 : description du modèle dynamique des traitements contrôle-commande

Libellé	Description
f1	$f_1(\langle t_i \rangle) = m_t(t_i)$
f2	$f_2(\langle t_i \rangle) = m_f(t_i)$
M10, M12	$M10 + M12 = \sum_{t_i \in T_{cc}} \langle t_i \rangle$
M2, M13, M21	$\sum_{t_i \in T_{cc}} \langle t_i \rangle$

1.4.2.1.3 Les traitements de coordination de traitements

Le modèle des traitements de coordination de traitements décrit tout $t_i \in T_c$, où T_c

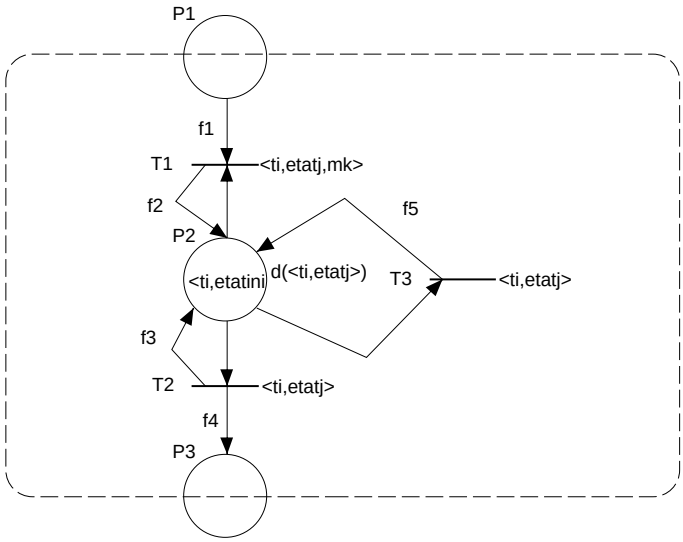


Figure 48 : modèle dynamique des traitements de coordination de traitements

est l'ensemble des traitements de coordination du modèle d'implantation. Ce modèle est un automate fini dont le modèle générique se réduit à une place P2 dont le marquage représente l'état de chaque traitement, T1 permet de faire évoluer les traitements vers un autre état à partir d'un événement extérieur, tandis que T3 permet de faire évoluer les traitements de manière interne. T2 émet les messages et les contrôles (activation et désactivation) à destination des autres traitements.

Tableau 6 : description du modèle dynamique des traitements de coordination

Libellé	Description
f1	$f_1(\langle t_i, etat_j, m_k \rangle) = \langle t_i, etat_j, m_k \rangle$
f2	$f_2(\langle t_i, etat_j, m_k \rangle) = \langle t_i, etat_j, m_l \rangle$
f3	$f_3(\langle t_i, etat_j \rangle) = \langle t_i, etat_k \rangle$

Tableau 6 : description du modèle dynamique des traitements de coordination

Libellé	Description
f4	$f_2(\langle t_i, etat_j \rangle) = m_f(t_i)$ $f_2(\langle t_i, etat_j \rangle) = \langle act, t_j \rangle$ $f_2(\langle t_i, etat_j \rangle) = \langle desact, t_j \rangle$
f5	$f_5(\langle t_i, etat_j \rangle) = \langle t_i, etat_j \rangle$
M2	$\langle t_i, etat_{initial} \rangle$

1.4.2.1.4 Les stockages

Le modèle des stockages décrit tout $s_i \in S$, où S est l'ensemble des stockages du modèle d'implantation (fig. 49). Il est organisé autour des deux places P6 et P7 qui représentent respectivement les données en cours d'accès par un traitement et celles qui ne le sont pas. En amont, les places P1, P2, P3 et les transitions T1, T2, T3 représentent une file FIFO (first in, first out) de taille $n(s_i)$ telle que décrite dans [62] [63]. Dès qu'une demande d'accès sort de la file (transition T3), une demande de ressource CPU est émise. Une couleur telle que $\langle s_i, t_j, l, e_k \rangle$ doit être interprétée comme une demande de lecture (l) traitée par le stockage s_i pour le traitement t_j, demande qui occupe l'emplacement e_k dans la file d'attente.

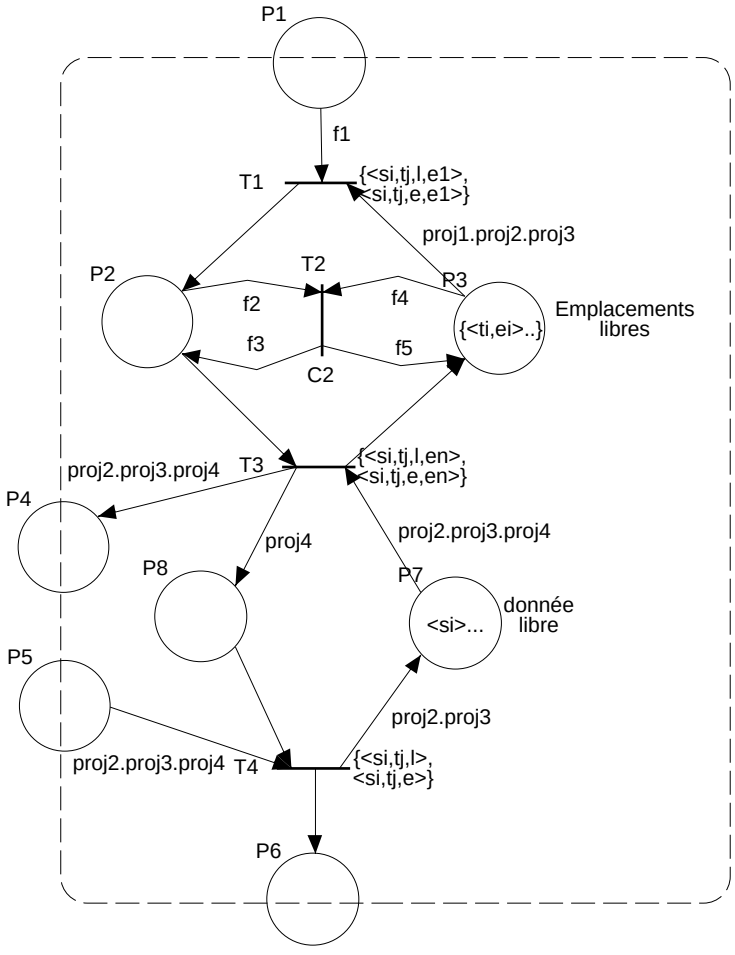


Figure 49 : modèle dynamique des stockages

Tableau 7 : description du modèle dynamique des stockages

Libellé	Description
f1	$f_1(\langle s_i, t_j, l, e_1\rangle) = \langle t_j, s_i, l\rangle$ $f_1(\langle s_i, t_j, e, e_1\rangle) = \langle t_j, s_i, e\rangle$
f2	id
f3	$f_3(\langle s_i, t_j, l, e_k\rangle) = \langle t_j, s_i, l, e_{k+1}\rangle$ $f_3(\langle s_i, t_j, e, e_k\rangle) = \langle t_j, s_i, e, e_{k+1}\rangle$
f4	$f_4(\langle s_i, t_j, l, e_k\rangle) = \langle s_i, e_{k+1}\rangle$ $f_4(\langle s_i, t_j, e, e_k\rangle) = \langle s_i, e_{k+1}\rangle$
f5	$f_5(\langle s_i, t_j, l, e_k\rangle) = \langle s_i, e_k\rangle$ $f_5(\langle s_i, t_j, e, e_k\rangle) = \langle s_i, e_k\rangle$
C2	$\{\langle s_i, t_j, l, e_k\rangle, \langle s_i, t_j, e, e_k\rangle, k \in [1, n(s_i)[\ \}$
M3	$\sum_{s_i \in S}\left(\sum_{j=1}^{n(s_i)} \langle s_i, e_j\rangle\right)$
M7	$\sum_{s_i \in S} \langle s_i\rangle$

1.4.2.2 Les modèles génériques de comportement des équipements

Après avoir présenté les modèles dynamiques des éléments du modèle d'implantation, nous allons proposer des modèles dynamiques des *constituants d'architecture* représentatifs : des calculateurs, des automates programmables industriels, des media à accès aléatoire et des media à accès maître-esclave. La figure 45 décrit les entrées-sorties de chaque module.

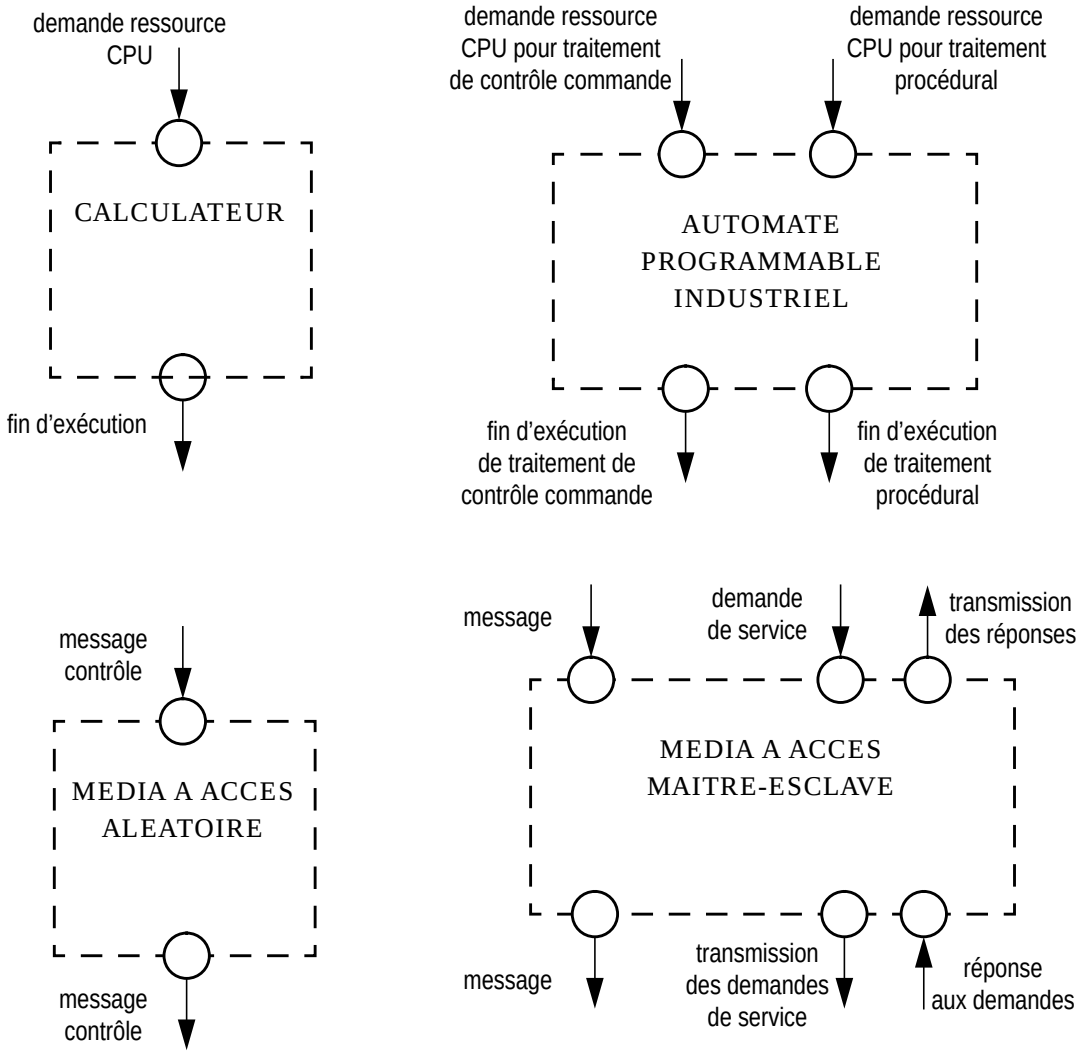


Figure 50 : les entrées-sorties des modèles dynamiques associés aux constituants du modèle organique

1.4.2.2.1 Les équipements de type calculateur

Le modèle des équipements de type calculateur décrit le comportement d'un calculateur multi-tâches où tous les traitements ont la même priorité (fig. 51). Chaque trai-

tement t_j y est exécuté par bloc de $n_{bloc}(p_i)$ instructions. La durée d'exécution d'un bloc sur un calculateur p_i est de $du(p_i)$. L'ensemble des traitements en cours d'exécution est modélisé par le marquage de P6, tandis que le marquage de P5 représente l'ensemble des traitements qui ne sont pas en cours d'exécution.

Tableau 8 : description du modèle dynamique des traitements de coordination

Libellé	Description
f1	$\forall t_j \in T_p(E_{cal}) \qquad f_1(\langle p_i, t_j\rangle) = \left(E\left(\frac{n_i(t_j)}{n_{bloc}(p_i)}\right)+1\right)\cdot\langle p_i, t_j\rangle$ $\forall t_j \in T_{cc}(E_{cal}) \qquad f_1(\langle p_i, t_j\rangle) = \left(E\left(\frac{n_{mot}(t_j)}{n_{bloc}(p_i)}\right)+1\right)\cdot\langle p_i, t_j\rangle$ (avec E qui représente la partie entière d'un réel)
f2	$f_2(\langle p_i, t_j\rangle) = \langle p_{i+1}, t_j\rangle \qquad modulo \qquad card(E_{cal})$
M7	$\sum_{p_i \in E_{cal}} \left(\sum_{t_j \in (T_p(E_{cal}) \cup T_{cc}(E_{cal}))} \langle p_i, t_j\rangle \right)$

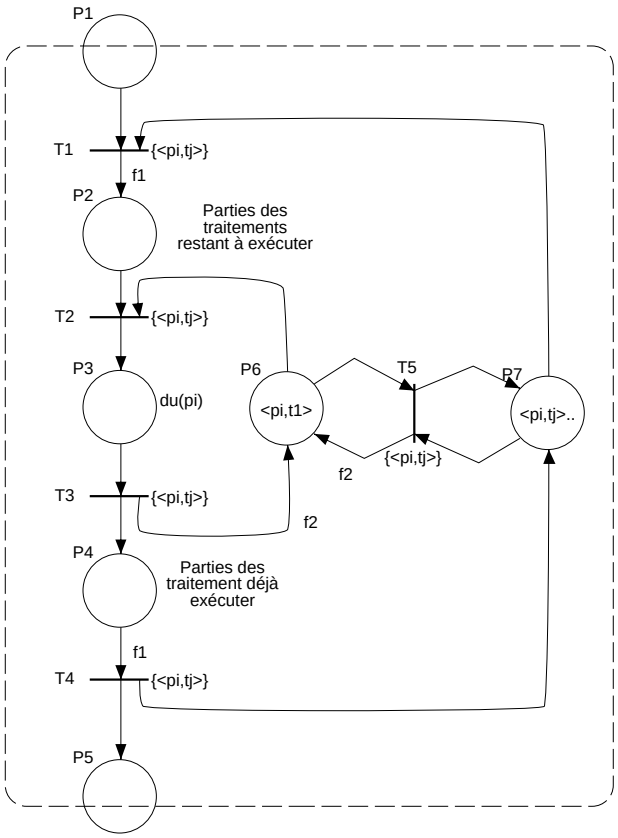


Figure 51 : modèle dynamique des équipements de type calculateur

1.4.2.2.2 Les équipements de type automate programmable industriel

Le modèle dynamique des équipements de type automate programmable industriel décrit le comportement d'un API $p_i \in E_{API}$ à temps de cycle variable, et permettant d'exécuter certains traitements de manière cyclique et d'autres en tâche de fond, dans la limite d'un temps de cycle maximum borné par $du_{cycle}(p_i)$ (fig. 52). Ces APIs ont un cycle de travail qui consiste à exécuter tous les traitements cycliques puis à exécuter les traitements de fond, s'il y en a durant ce cycle, jusqu'à ce que la durée maximum du cycle soit atteinte (P1, P2, P5, P9, T1, T3, T4, T5, T6). Les traitements de contrôle-commande sont exécutés en traitements cycliques, alors que les traitements procéduraux sont exécutés en traitements de fond.

Tableau 9 : description du modèle dynamique des APIs

Libellé	Description
C3	$\{\langle p_i, t_n\rangle \mid n = \text{card}(T_{cc}(E_{API}))\}$
C8	$\{\langle p_i, t_j\rangle \mid j \in [1, \text{card}(T_{cc}(E_{API}))[\}$
f1	$\forall t_j \in T_p(E_{API}) \qquad f_1(\langle p_i, t_j\rangle) = E\left(\frac{n_i(t_j)}{n_{bloc}(p_i)}\right) + 1$ (avec E qui représente la partie entière d'un réel)
f2	$f_2(\langle p_i, t_j\rangle) = \langle p_{i+1}, t_j\rangle \ \ modulo \ \ card(T_p(E_{API}))$
D12	$n_{mot}(t_j) / V(p_i)$
D19	$n_i(t_j) / V(p_i)$
M1, M16	$\sum_{p_i \in E_{API}} \langle p_i\rangle$
M22	$\sum_{p_i \in E_{API}} \langle p_i, t_1\rangle \qquad t_1 \in T_p(E_{API})$
M23, M24	$\sum_{p_i \in E_{API}} \left(\sum_{t_j \in T_p(E_{API})} \langle p_i, t_j\rangle \right)$

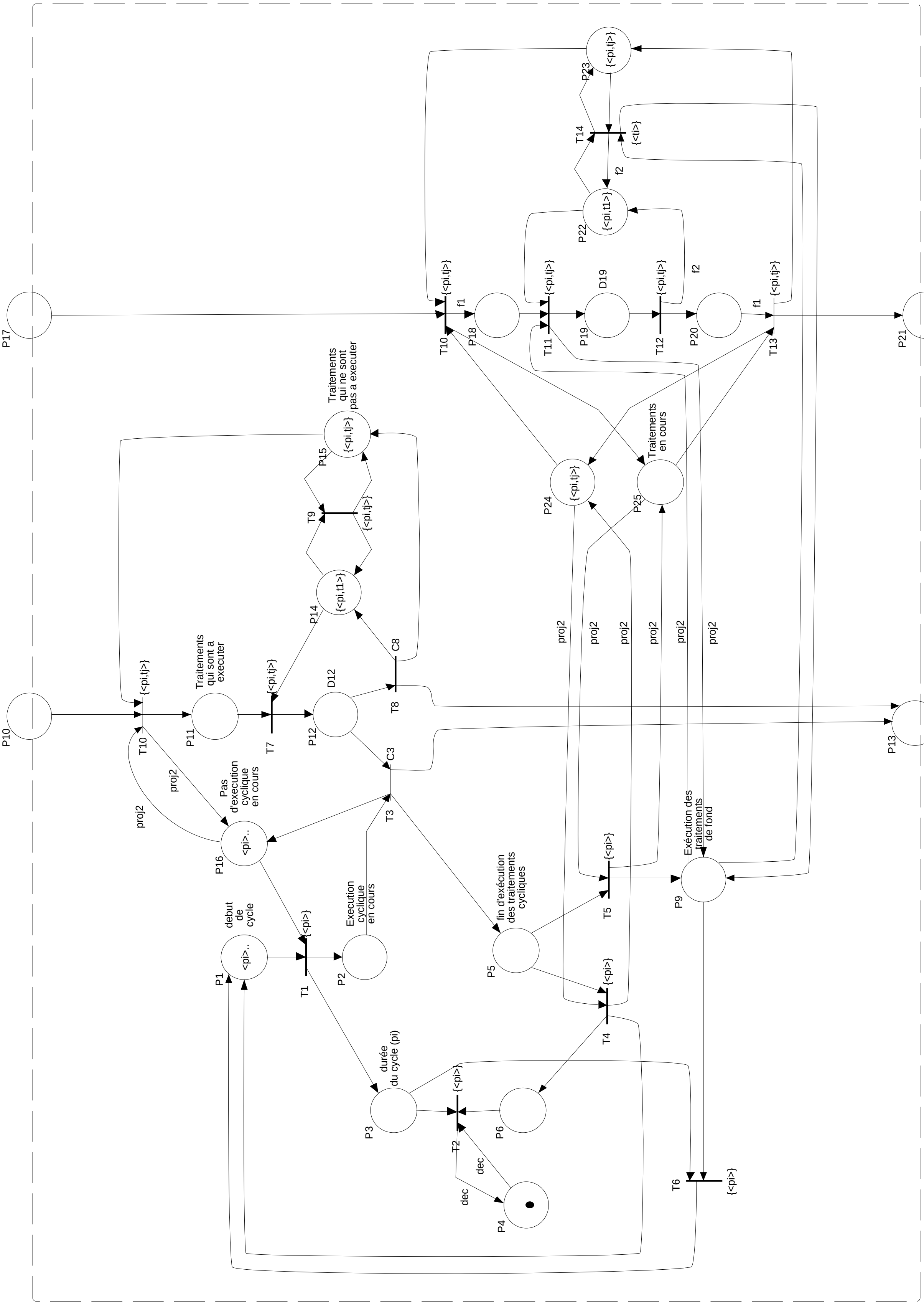


Figure 52 : modèle dynamique des équipements de type API

Un dispositif de surveillance (P3, P4, P6 et T2) assure le respect du temps de cycle borné. Toutes les demandes d'exécution de traitement de contrôle-commande faites avant le début d'un cycle automate sont exécutés durant ce cycle (P10, P11, P12, P13, P14, P15, P16). Puis, tant que la durée $du_{cycle}(p_i)$ ne s'est pas encore écoulée depuis le début du cycle, les demandes d'exécution des traitements procéduraux sont traitées de la même manière que dans les calculateurs (P17, P18, P19, P20, P21, P22, P23, T10, T11, T12 T13 et T14).

1.4.2.2.3 Les équipements de communication à accès aléatoire

Le modèle dynamique des équipements de communication de type multi-connexions à accès aléatoire ou de type bi-connexion full duplex, se réduit à une place représentant la ressource de chaque media (P4) et d'une autre place (P2) représentant le transfert d'information dans chaque media.

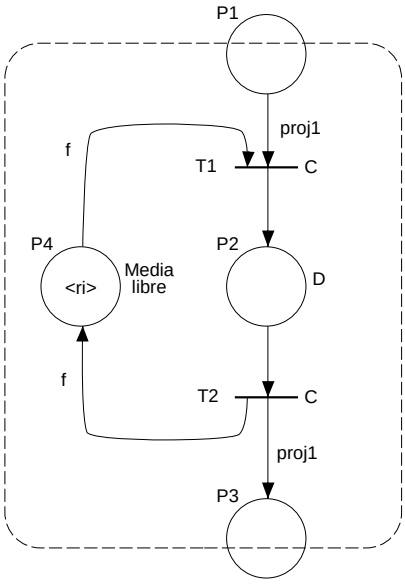


Figure 53 : modèle dynamique des équipements de communication de type accès aléatoire

Tableau 10 : description du modèle dynamique des équipements de communication de type accès aléatoire

Libellé	Description
C	$\{\langle r_i, t_j, s_k, e\rangle, \langle r_i, t_j, s_k, l\rangle, \langle r_i, s_j, t_k, e\rangle, \langle r_i, s_j, t_k, l\rangle$, $\langle r_i, m\rangle, \langle r_i, act, t_j\rangle, \langle r_i, desact, t_j\rangle\}$
f	$\forall c \qquad f(\langle c\rangle) = \langle r_i\rangle$
D	soit $d_{mini}(r_i)$ le délai minimum pour transmettre un message dans l'équipement de communication r_i - soit f le flot qui circule entre t_j et s_k alors $D\left(\langle r_i, t_j, s_k, e\rangle\right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ et $D\left(\langle r_i, t_j, s_k, l\rangle\right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ - soit f le flot qui circule entre s_j et t_k alors $D\left(\langle r_i, s_j, t_k, e\rangle\right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ et $D\left(\langle r_i, s_j, t_k, l\rangle\right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ - soit f le flot sur lequel circule le message m, alors $D(\langle r_i, m\rangle) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ - soit f le flot sur lequel circulent les contrôles de traitement, alors $D(\langle r_i, act, t_j\rangle) = d_{mini}(r_i)$ et $D(\langle r_i, desact, t_j\rangle) = d_{mini}(r_i)$

1.4.2.2.4 Les équipements de communication à accès maître-esclave

Le modèle dynamique des équipements de communication de type multi-connexions à accès maître-esclave ou de type bi-connexion half duplex, est construit autour de la ressource de chaque media (P5). Quel que soit la nature des informations échangées, suite à une question du maître, l'esclave interrogé utilise à son tour la ressource pour répondre. Dans le cas des messages, pour lesquels le destinataire ne donne pas explicitement d'acquitement, le modèle prend en compte lui-même le flux d'acquitement (P2, P3, P4, P5, T1, T2 et T3), alors que dans le cas où le flux d'acquitement de l'esclave est généré explicitement à l'extérieur de ce modèle, celui-ci décrit ce comportement avec P6, P7, P8, P9, P10, P11, T4, T5, T6 et T7

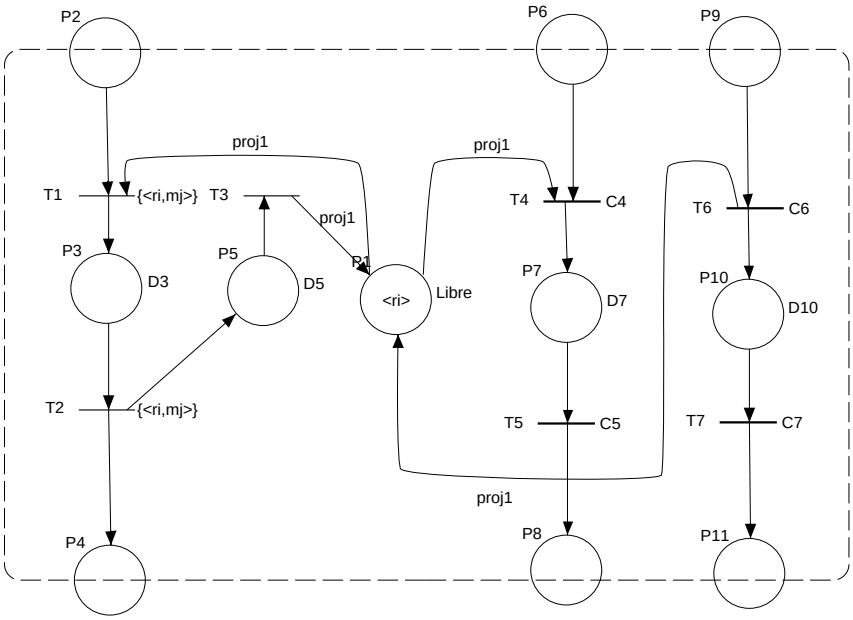


Figure 54 : modèle dynamique des équipements de communication de type maître-esclave

Tableau 11 : description du modèle dynamique des équipements de communication de type accès maître-esclave

Libellé	Description
C4, C5	$\{\langle r_i, t_j, s_k, e\rangle, \langle r_i, t_j, s_k, l\rangle, \langle r_i, s_j, t_k, e\rangle, \langle r_i, s_j, t_k, l\rangle,$ $\langle r_i, t_j, s_k, q\rangle, \langle r_i, t_j, m_k, q\rangle\}$
C6, C7	$\{\langle r_i, t_j, s_k, e\rangle, \langle r_i, t_j, s_k, l\rangle, \langle r_i, s_j, t_k, e\rangle, \langle r_i, s_j, t_k, l\rangle,$ $\langle r_i, t_j, s_k, non\rangle, \langle r_i, t_j, s_k, acq\rangle, \langle r_i, t_j, m_k, non\rangle,$ $\langle r_i, t_j, m_k\rangle\}$
D7, D10	- soit f le flot qui circule entre t_j et s_k alors $D\left(\langle r_i, t_j, s_k, e\rangle\right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ et $D\left(\langle r_i, t_j, s_k, l\rangle\right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ - soit f le flot qui circule entre s_j et t_k alors $D\left(\langle r_i, s_j, t_k, e\rangle\right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ et $D\left(\langle r_i, s_j, t_k, l\rangle\right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ - soit f le flot technique entre un traitement de scrutation et un traitement de réponse, alors $D\left(\langle r_i, t_j, s_k, q\rangle\right) = d_{mini}(r_i)$ et $D\left(\langle r_i, t_j, m_k, q\rangle\right) = d_{mini}(r_i)$
D7, D10	- soit f le flot qui circule entre t_j et s_k alors $D\left(\langle r_i, t_j, s_k, e\rangle\right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ et $D\left(\langle r_i, t_j, s_k, l\rangle\right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$

Tableau 11 : description du modèle dynamique des équipements de communication de type accès maître-esclave

Libellé	Description
	- soit f le flot qui circule entre s_j et t_k alors $D\left(\langle r_i, s_j, t_k, e \rangle \right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ et $D\left(\langle r_i, s_j, t_k, l \rangle \right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ - soit f le flot qui véhicule m_k vers t_j alors $D\left(\langle r_i, t_j, m_k \rangle \right) = (T_d(f) / d_e(r_i)) + d_{mini}(r_i)$ - soit f le flot technique entre un traitement de scrutation et un traitement de réponse, alors $D\left(\langle r_i, t_j, s_k, non \rangle \right) = d_{mini}(r_i)$ et $D\left(\langle r_i, t_j, m_k, non \rangle \right) = d_{mini}(r_i)$ et $D\left(\langle r_i, t_j, s_k, acq \rangle \right) = d_{mini}(r_i)$

1.4.2.3 Les modèles génériques de comportement des traitements techniques

Après avoir décrit le comportement dynamique des éléments du modèle d'implantation, puis celui d'un ensemble représentatif des constituants du modèle organique, il nous reste à présenter le comportement dynamique des *éléments techniques* du modèle d'affectation. La figure 45 présente les entrées-sorties de chaque module.

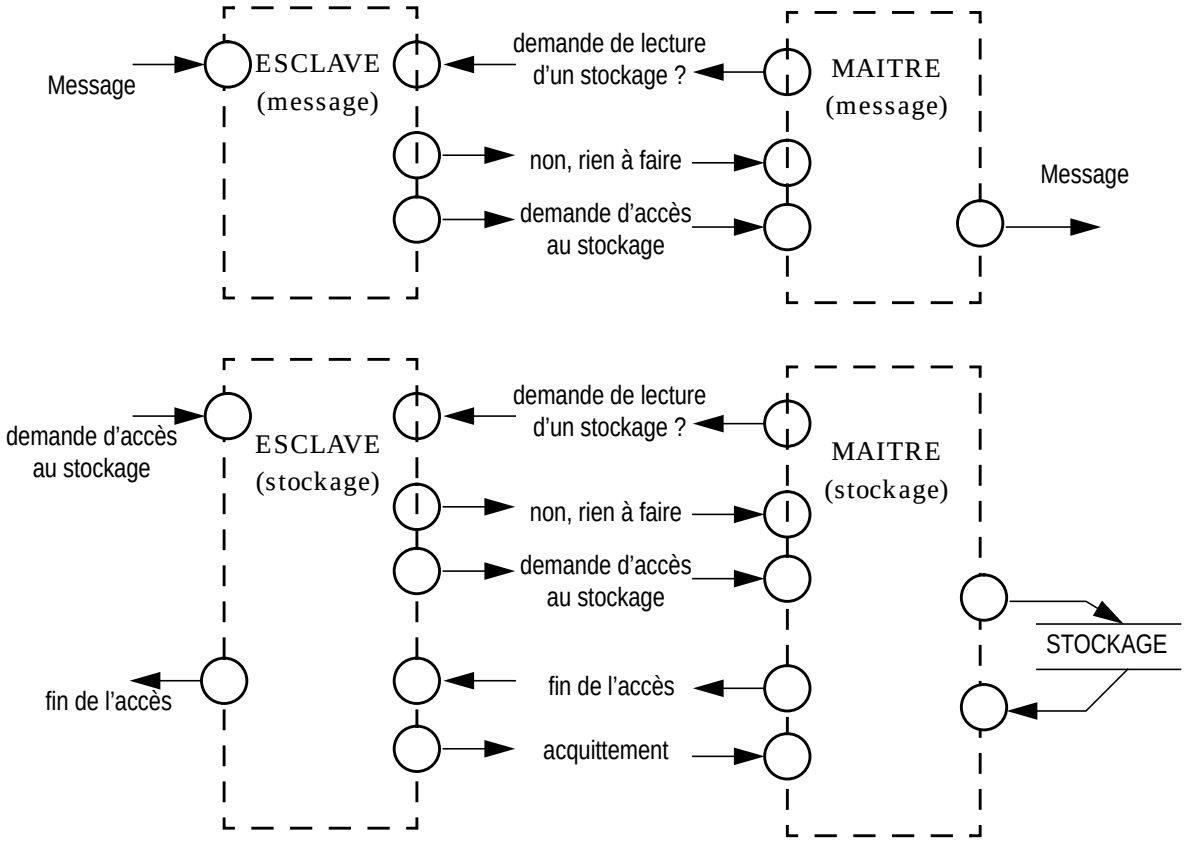


Figure 55 : les entrées-sorties des modèles dynamiques associés aux traitements techniques de scrutation et de réponse

1.4.2.3.1 Les traitements techniques maître-esclave pour transmettre un message

Le modèle dynamique des traitements techniques maître-esclave qui transmettent un message de l'esclave vers le maître est organisé autour d'une boucle de scrutation (P10, T10, P11, P3, T2, P6, P12 et T11). A chaque période de scrutation, le maître demande à l'esclave s'il a un message à transmettre (couleur $\langle t_i, m_j, q \rangle$), et l'esclave répond non (couleur $\langle t_i, m_j, non \rangle$), s'il n'a pas de message à transmettre. Dans le cas contraire, l'esclave transmet le message (P2, T3, P5, P13, T12, P14).

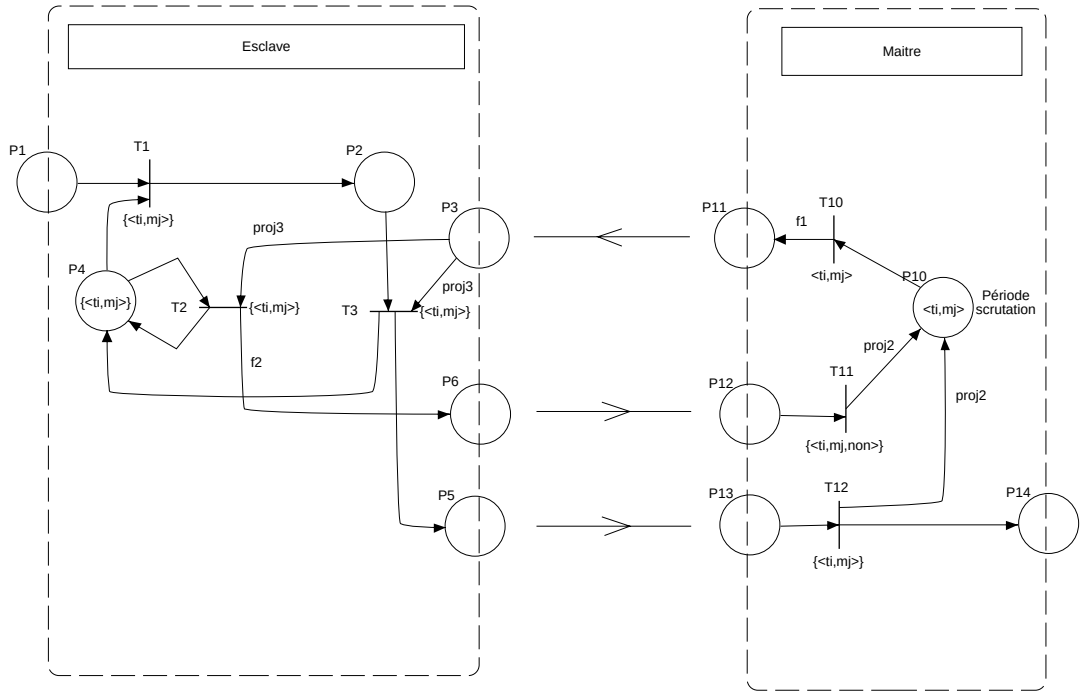


Figure 56 : modèle dynamique des traitements techniques maître-esclave pour la transmission des messages

Tableau 12 : description du modèle dynamique des traitements techniques maître-esclave pour la transmission des messages

Libellé	Description
M4, M10	soit m_j le message transmis par le traitement de scrutation et le traitement de réponse t_i, alors le marquage est : $\sum_{t_i \in T_{scm}} \langle t_i, m_j \rangle$
f1	$f_1(\langle t_i, m_j \rangle) = \langle t_i, m_j, q \rangle$
f2	$f_1(\langle t_i, m_j \rangle) = \langle t_i, m_j, non \rangle$

1.4.2.3.2 Les traitements techniques maître-esclave pour accéder à un stockage

Le modèle dynamique des traitements techniques maître-esclave dont l'esclave doit lire ou écrire une donnée dans un stockage est organisé autour d'une boucle de scrutation (P10, T10, P11, P3, T2, P6, P12 et T11). A chaque période de scrutation le maître demande à l'esclave s'il a une donnée à écrire (couleur $\langle t_i, s_j, e, q \rangle$), ou une donnée à lire (couleur $\langle t_i, s_j, l, q \rangle$), et l'esclave répond non (couleur $\langle t_i, s_j, non \rangle$), s'il n'a pas de requête à faire auprès du maître. Dans le cas contraire l'esclave transmet sa requête (P2, T3, P5, P13, T12, P14), reçoit la reponse à sa requête (P15, T13, P16, P7, T4 et P8) puis émet un acquitement au maître (couleur $\langle t_i, s_j, acq \rangle$, et places P9 et P17).)

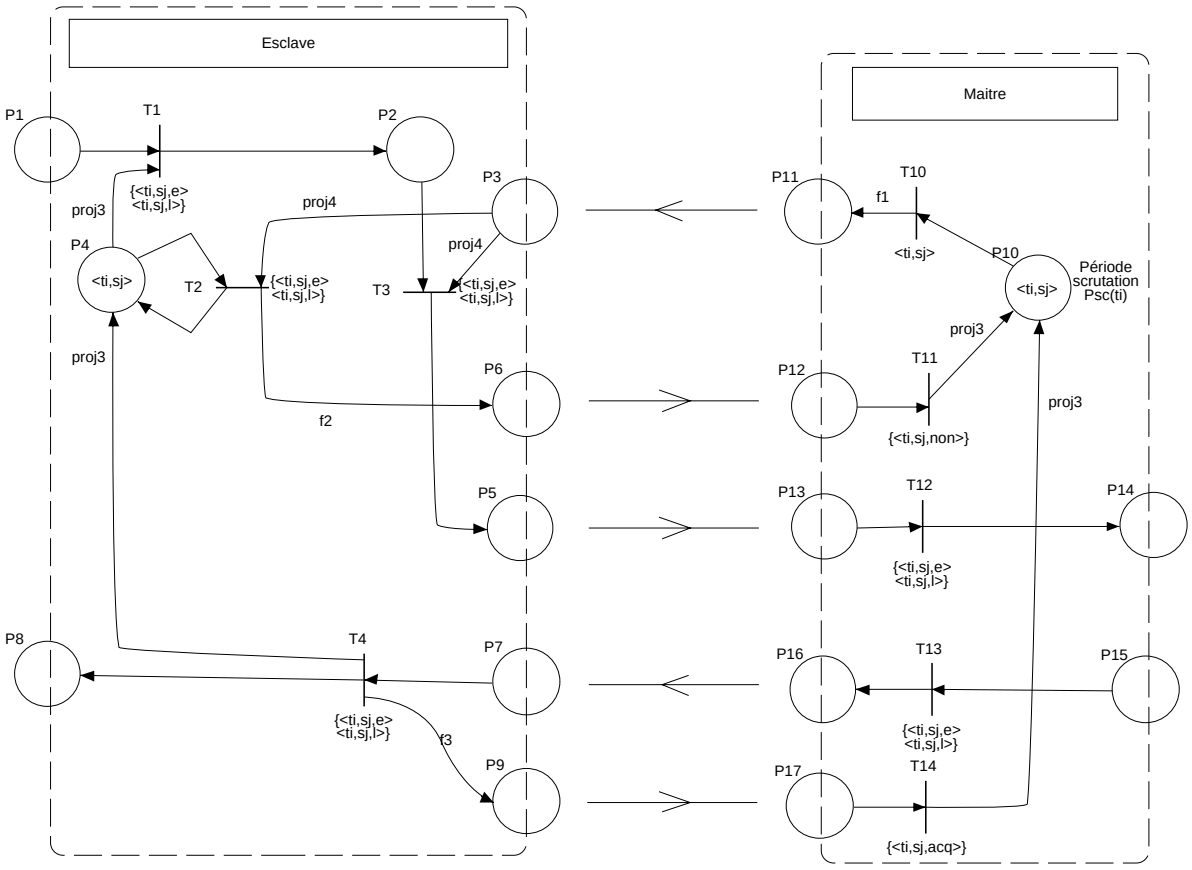


Figure 57 : modèle dynamique des traitements techniques maître-esclave pour la transmission des demandes d'accès aux stockages

Tableau 13 : description du modèle dynamique des traitements techniques maître-esclave pour la transmission des demandes d'accès aux stockages

Libellé	Description
M4, M10	soit s_j le stockage auquel on accède par le traitement de scrutation et le traitement de réponse t_i, alors le marquage est : $\sum_{t_i \in (T_{sce} \cup T_{scl})} \langle t_i, s_j \rangle$
f1	$\forall t_i \in T_{sce}$, $f_1(\langle t_i, s_j \rangle) = \langle t_i, s_j, e, \rangle$ et $\forall t_i \in T_{scl}$, $f_1(\langle t_i, s_j \rangle) = \langle t_i, s_j, l, \rangle$
f2	$f_2(\langle t_i, s_j, e \rangle) = \langle t_i, s_j, non \rangle$, et $f_2(\langle t_i, s_j, l \rangle) = \langle t_i, s_j, non \rangle$
f3	$f_2(\langle t_i, s_j, e \rangle) = \langle t_i, s_j, acq \rangle$, et $f_2(\langle t_i, s_j, l \rangle) = \langle t_i, s_j, acq \rangle$

1.4.2.4 Constitution d'un modèle évaluable à partir des modèles génériques de comportement

Nous venons de construire des modèles génériques de chaque élément de l'architecture, aussi bien des *éléments fonctionnels* du modèle d'implantation, que des *constituants matériels* du modèle organique ou que des *traitements techniques* du modèle d'affectation. Le modèle d'une architecture particulière est obtenu en assurant le «routage» des jetons d'un générique à l'autre. La figure 58 présente les arcs et les transitions support du routage, il s'agit là encore d'un modèle générique, c'est dans la description des fonctions associées aux arcs qu'est particularisé le modèle. Si dans le modèle d'affectation de l'architecture le traitement procédural t et affecté au calculateur p, alors la transition T1 peut être franchie relativement à la couleur $\langle p, t \rangle$, et la fonction f donne $f(\langle p, t \rangle) = \langle t \rangle$. Lorsque le traitement t demande une ressource CPU, il dépose une marque $\langle t \rangle$ dans la place «demande de ressource CPU» du modèle des traitements procéduraux (fig. 46). La transition T1 de la figure 58 assure l'affectation du traitement au calculateur t en déposant une marque $\langle p, t \rangle$ dans la place «demande de ressource CPU» du modèle des calculateurs (fig. 51)

De la même façon, les demandes de lecture d'un stockage sont «routées» dans le modèle des stockages ou dans le modèle des traitements techniques si le stockage n'est pas directement accessible depuis le traitement demandeur.

Une fois les modèles génériques de comportement des éléments de l'architecture établis, et cela une fois pour toute dans un domaine d'application donné, la construction du modèle évaluable d'une architecture est complètement systématique, donc automatisable. Il donne une nouvelle vision du modèle d'affectation qui, cette fois ci, est évaluable par des techniques de simulation. Pour vérifier l'adéquation entre sa solution et le document d'expression des besoins, l'architecte est fréquemment amené à évaluer des temps de réponse entre une sollicitation et ses effets. Evaluer ces temps de réponse, c'est réaliser un scénario de sollicitations, simuler le fonctionnement de l'architecture sous ces sollicitations, et confronter les résultats de la simulation avec ceux attendus dans le guide d'expression des besoins. Pour ce faire, il convient d'utiliser des outils de simulation de réseaux de Petri temporisés [64] [65] [66] [67].

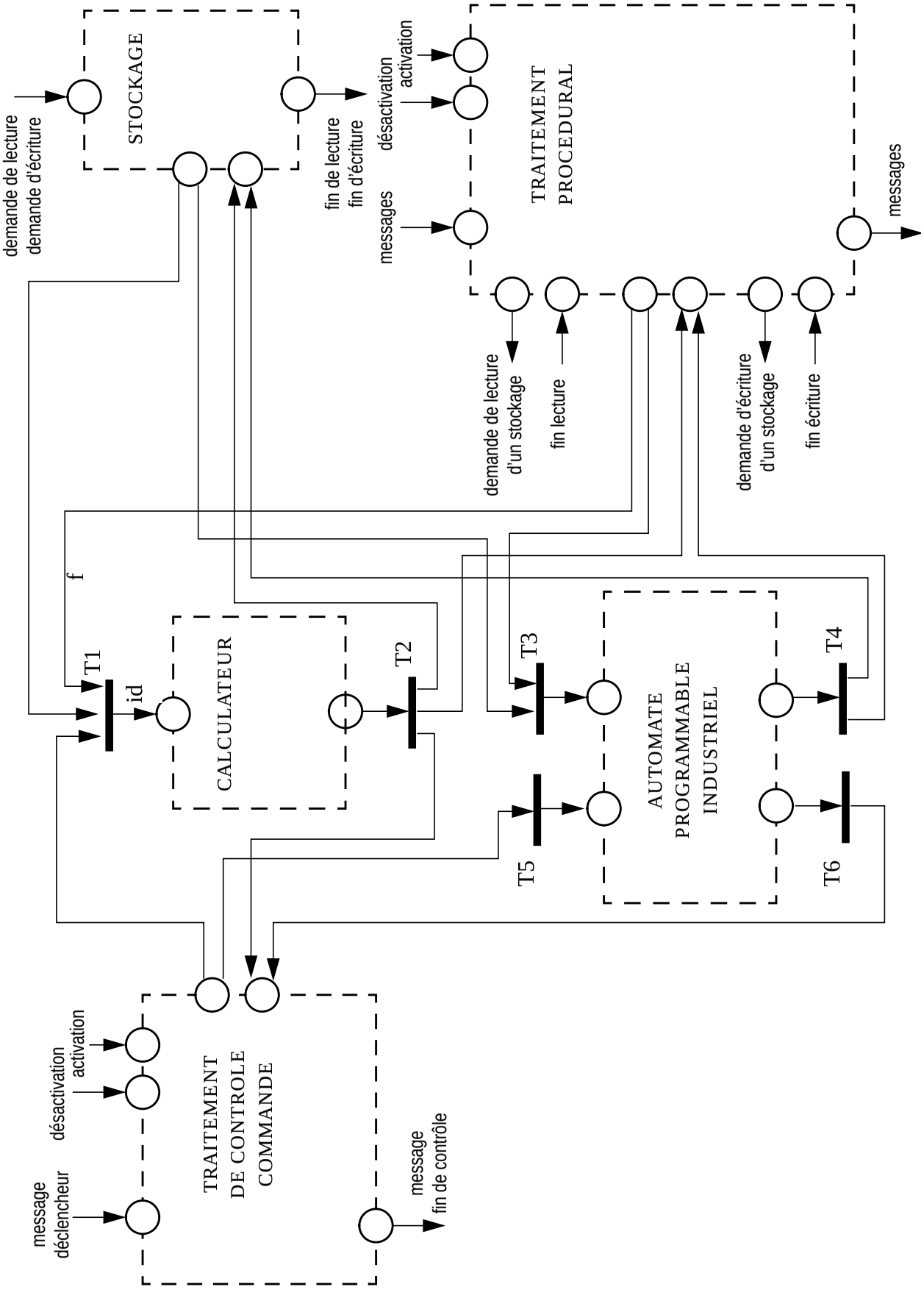


Figure 58 : routage des demandes de ressource CPU

La construction systématique d'un modèle de comportement de l'architecture montre une fois de plus l'intérêt de la formalisation du cœur de l'architecture en phase d'évaluation.

1.4.3 Le modèle d'architecture matérielle

Comme pour le modèle organique, l'absence de formalisme établi, permettant de représenter la structure matérielle d'une architecture de conduite, nous a conduit à adopter une représentation tentant une synthèse des usages. Le formalisme du modèle d'architecture matérielle est calqué sur celui du modèle organique. La seule différence se trouve au niveau de la sémantique attachée aux éléments graphiques du modèle. Là où le modèle organique interprète un élément graphique comme une classe d'équipements, le modèle d'architecture matérielle l'interprètera comme une instance de classe d'équipements, c'est-à-dire comme un équipement X de chez le fournisseur Y.

Pour faire les choix technologiques, qui permettent de transformer un type d'équipement en un équipement particulier, l'architecte a à sa disposition les indicateurs mis en place dans le modèle d'évaluation. Chaque indicateur permet d'effectuer un choix sur une classe d'équipements, on a ainsi par exemple :

- l'indicateur de charge totale d'un calculateur de la formule (10) qui permet de déterminer la puissance minimum du CPU du calculateur à choisir,
- l'indicateur de la formule (13) donnant le temps de cycle d'un automate programmable industriel en fonction de la puissance de traitement de ce dernier, qui fournit à l'architecte une caractéristique essentielles de l'API à choisir,
- l'indicateur de charge d'un équipement de communication à accès aléatoire, tel qu'il est défini dans la formule (14), qui donne une valeur du débit d'informations que devra faire circuler le réseau à choisir,
- l'indicateur de charge d'un équipement de communication à accès maître-esclave, tel qu'il est défini dans la formule (18), qui donne également une valeur du débit d'informations que devra faire circuler le réseau à choisir.

L'architecte peut ainsi appliquer ses propres facteurs correcteurs pour choisir les éléments de l'architecture matérielle. Car si nos indicateurs tiennent compte des traite-

ments techniques et des flots techniques dus à la répartition dans l'architecture de conduite, ils ne peuvent pas tenir compte des traitements et flots techniques dus aux choix technologiques des équipements matériels proprement dits. A titre d'exemple, lorsque l'on choisit pour équipement de communication à accès aléatoire un réseau local ethernet, une correction doit être effectuée entre le débit 10 Mégabits par seconde, caractéristique intrinsèque du réseau, et le débit maxi réel pour l'utilisateur. Cette différence entre le débit maximum théorique et le débit maxi *réel* est induit par l'organisation interne du réseau ethernet. Nombre d'utilisateurs appliquent un facteur de 2 pour passer de l'un à l'autre, il s'agit la d'une «recette» basée sur l'expérience.

C'est lors de l'élaboration du modèle d'architecture matérielle que l'architecte utilise son expérience relative aux technologies. Grâce à notre démarche et notre formalisation de l'architecture de conduite, il n'y a pas eu confusion entre les concepts génériques propres aux calculateur ou aux réseaux, et les caractéristiques technologiques de chaque constructeur.

1.4.4 Les spécifications de chaque constituant

Lors de la construction du modèle d'implantation nous avons montré comment la notion de portabilité de nos travaux dans un projet global d'ingénierie, avait été mise en place grâce à l'expression de la sémantique liant les différents modèles mis en jeu.

Nous nous proposons d'en faire de même pour l'intégration des résultats de la conception d'architecture dans la conception ultérieure de la partie logicielle de chaque constituant matériel de l'architecture.

La figure 59 illustre le procédé de formalisation de la méthode de construction des

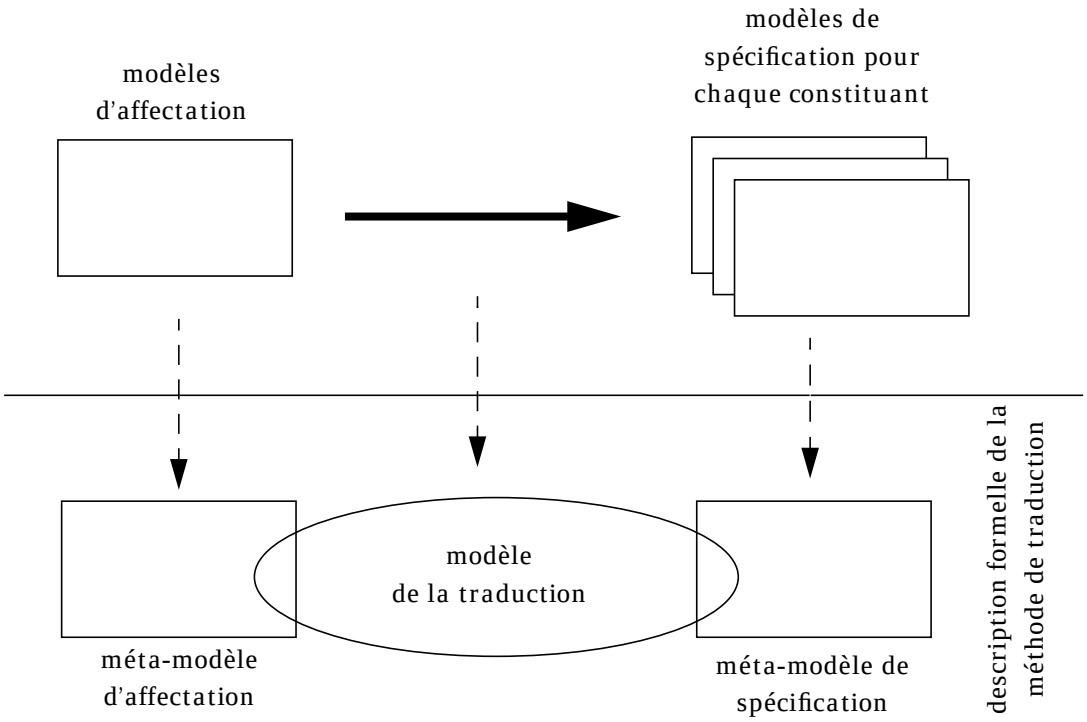


Figure 59 : principe de la formalisation de la construction des spécifications pour chaque constituant de l'architecture

modèles de spécification de chaque constituant de l'architecture. Pour illustrer notre propos nous avons choisi, comme pour la construction du modèle d'implantation, la méthode SART de Hatley et Pirbhai pour constituer les spécifications fonctionnelles de chaque constituant.

Le métamodèle partiel du modèle d'affectation est présenté figure 60, quant à celui

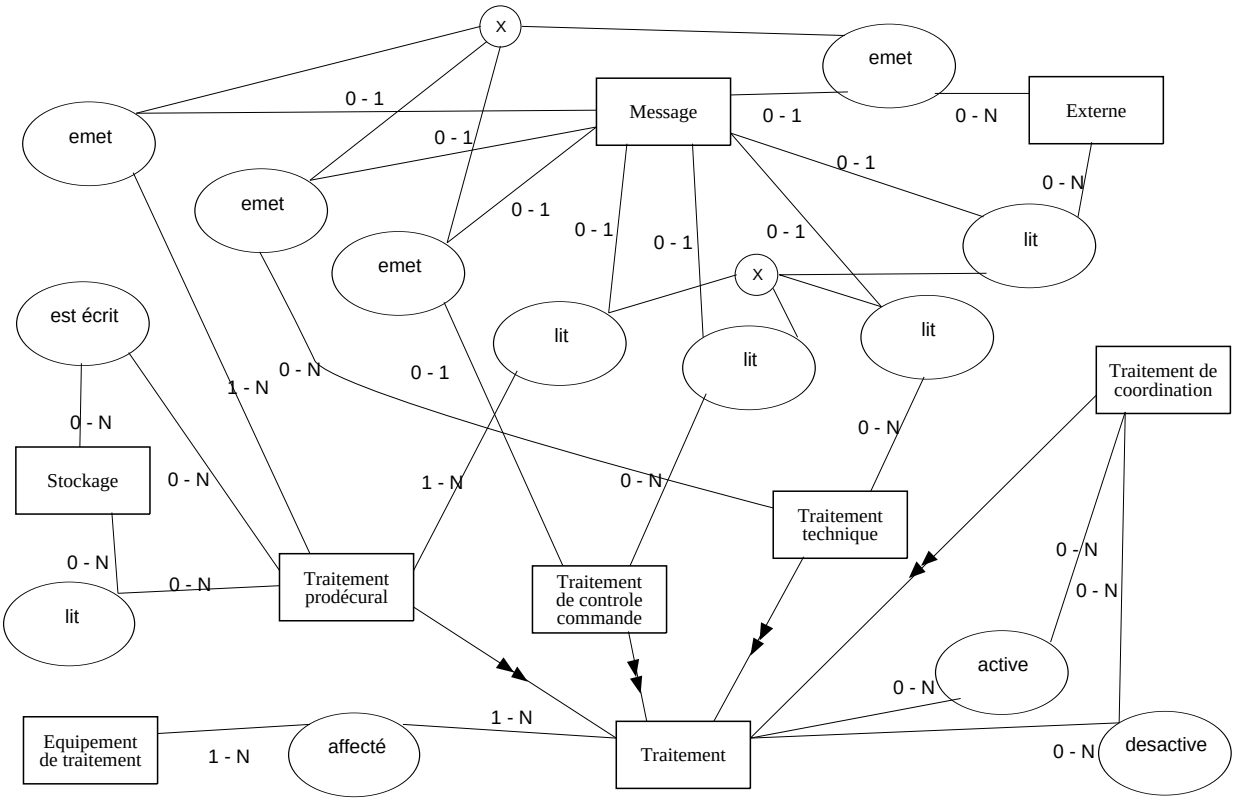


Figure 60 : métamodèle partiel du modèle d'affectation

de SART, on le retrouvera figure 35 dans la division 1.3.1 «Le modèle d'implantation». Le modèle de la traduction d'un modèle d'affectation en spécification par constituant est donnée figure 61.

La génération des modèles de spécification de chaque constituant conclut la phase de conception de l'architecture de conduite, menée par l'architecte. Son travail se termine par la fourniture d'une solution d'architecture qu'il a, après évaluation, jugée la meilleure et la constitution de dossiers de spécification pour la conception du sous-ensemble de la conduite du système de production affecté à chaque constituant matériel de l'architecture.

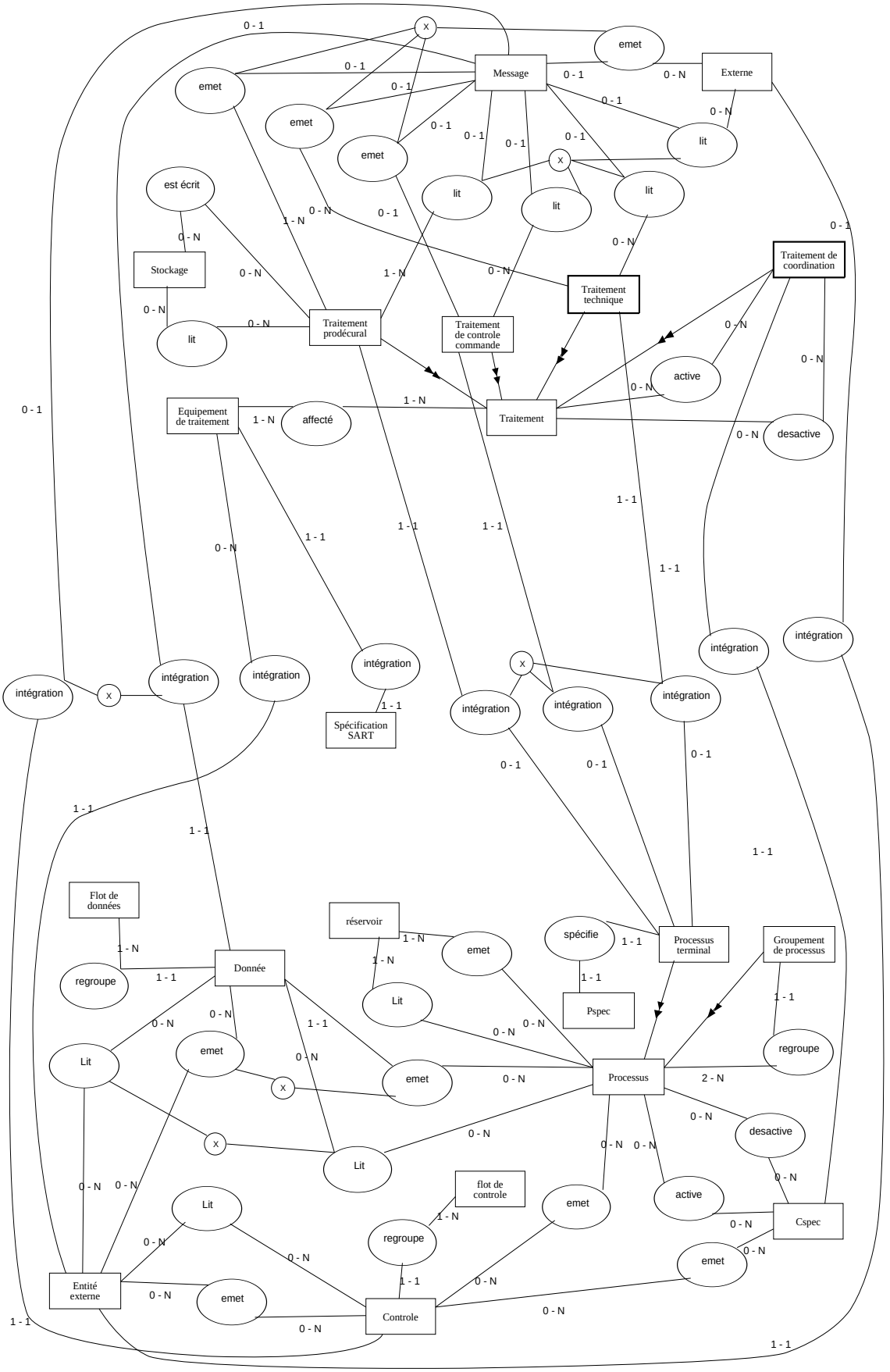


Figure 61 : métamodèle de l'intégration entre le modèle d'affectation et le modèle de spécification de chaque constituant

1.5 Conclusions sur la première partie

Dans cette première partie, nous nous sommes attachés à la formalisation de l'ensemble de la démarche de conception de l'architecture de conduite. Après avoir situé le problème et présenté les principaux travaux évoquant la conception d'architecture, nous avons dégagé les activités de l'architecte, en proposant une méthode de conception d'architecture.

Notre méthode, d'abord présentée sous son aspect fonctionnel, propre à répondre à la question «que faut-il faire pour concevoir une architecture de conduite ?», a ensuite été détaillée par ses modèles pour répondre à la question «comment s'y prend t'on pour construire une architecture ?». Nous avons terminé cette partie en montrant l'intérêt de la formalisation de l'architecture que nous avons proposée lors des phases d'évaluation en répondant à la question «comment évaluer la performance d'une architecture avec les critères propres à chaque architecte ?».

La deuxième partie de ce mémoire va maintenant être consacrée à l'exposé du déroulement de notre démarche en nous appuyant sur un exemple. Nous allons de plus préciser la part de travail fourni par l'architecte et la part d'assistance informatique que nous lui fournissons.

Partie 2

Exemple d'application

2.1 Introduction

Dans la première partie de ce mémoire, nous nous sommes attachés à présenter notre méthode d'assistance à la conception des architectures de conduite au travers de deux points de vue complémentaires. Dans un premier temps, nous avons examiné les activités qui la composent pour avoir un éclairage fonctionnel. Puis dans un second temps, nous avons détaillé les techniques d'élaboration des différents modèles utilisés au sein de la méthode, ainsi que leurs évaluations.

L'objectif de cette deuxième partie est de parcourir la méthode proposée sur un exemple. Cela donnera un point de vue «conduite de projet» au lecteur, tout en illustrant nos propos de la première partie. Nous nous attacherons en particulier à bien mettre en évidence la frontière entre le travail que doit faire l'architecte, et l'assistance procurée par notre méthode.

Pour attendre cet objectif le chapitre qui suit va être présenté sous la forme d'un scénario de construction d'architecture dans lequel trois *acteurs* évolueront :

- l'*ingénieriste*, qui prépare le travail de la conception d'architecture et qui en récupère le fruit,
- l'*architecte* dont la charge est de concevoir la «meilleure» architecture possible,
- les *outils d'assistance* tels qu'ils ont été décrits dans la première partie.

L'exemple support de cette deuxième partie, est un poste de lancement sur un îlot de production organisé autour d'un transfert libre qui distribue les produits sur les postes au fur et à mesure des besoins. L'exemple a été choisi d'une dimension suffisamment réduite pour pouvoir être détaillé dans le corps de ce mémoire, mais néanmoins représentative des problèmes couramment rencontrés lors de la conception d'architecture. Nous utiliserons par la suite le vocable *POSTEL* (POSTE de Lancement) pour désigner le système dont on se propose de concevoir l'architecture de conduite.

Pour assurer le traitement de cet exemple, nous nous sommes munis d'une maquette informatique support des travaux présentés dans la première partie. Elle a permis d'une part de valider les concepts d'intégration entre modèles constamment utilisés dans nos travaux, et en particulier de supporter le cœur formel de la méthode.

D'autre part, elle a permis d'automatiser une partie des tâches systématiques mais laborieuses de notre démarche. La maquette a été élaborée dans l'environnement de conception d'atelier de génie logiciel Graphtalk développé et diffusé par la société Parallax.

2.2 Cahier des charges et spécifications fonctionnelles

2.2.1 Cahier des charges

Description succincte

Le projet a pour but l'automatisation progressive d'un îlot de fabrication existant.

Objectifs généraux

Automatiser la circulation des produits dans un lot de production pour prendre en compte une plus grande diversité des produits.
Automatiser des postes de production pour augmenter la productivité de l'îlot.

Contexte

L'atelier de production est organisé en îlots autonomes (fig. 62), desservis en matières premières par des carristes. Chaque lot reçoit un plan de production à la journée, lui indiquant les lots de pièces à produire.

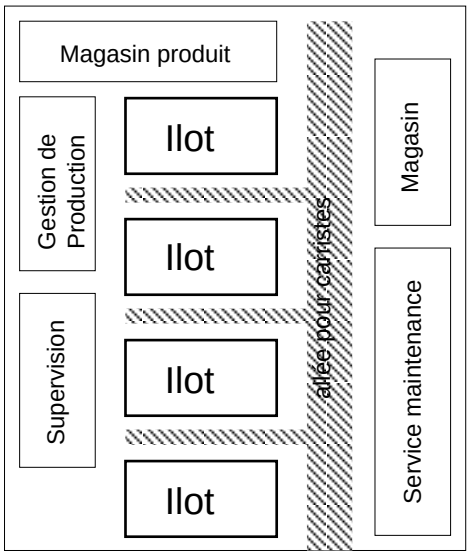


Figure 62 : implantation de l'atelier de production

Dans l'état actuel, l'îlot concerné par le projet est exploité de façon rigide. Cette rigidité doit être interprétée dans le sens où toutes les variantes des produits transformés par l'îlot sont traitées séquentiellement sur tous les postes et toujours dans le même ordre.

Contraintes

Le choix du système, devant assurer la circulation des pièces à l'intérieur de l'îlot, est imposé. La partie opérative devra être réalisée avec le système FlexLink de SKF pour rester homogène avec les îlots d'assemblage déjà réalisés dans l'usine. Un système d'identification Inductel de Télémécanique permettra sur chaque poste de production la reconnaissance des produits.

Description de l'îlot

L'îlot de production concerné par notre l'étude a pour mission l'usinage et le contrôle de petites pièces métalliques de révolution. Pour y parvenir il est actuellement formé de deux postes de production et d'un poste de lancement de production (fig. 63). Chaque poste est apte à réaliser un certain nombre d'opérations sur les produits. Les produits circulent sur des supports de produit appelés "palettes" qui sont équipées du système Inductel d'identification des produits.

Selon le plan de lancement, l'opérateur du poste de lancement prépare une palette en y indiquant toutes les opérations que devra subir successivement le produit (gamme) pour arriver dans l'état fini. Lorsque la palette est prête, il l'envoie sur la ligne principale de transfert (grande boucle) où elle circulera en permanence. Quand un poste est libre, et qu'il identifie le passage d'un produit dont l'opération à effectuer lui est possible, il fait alors dévier la palette sur sa ligne de déviation (petite boucle) afin de la stocker momentanément avant d'effectuer la dite opération sur le produit. L'opération terminée, le poste inscrit dans le système d'identification du produit que l'opération a été effectuée, et redérive la palette sur la ligne principale de transfert.

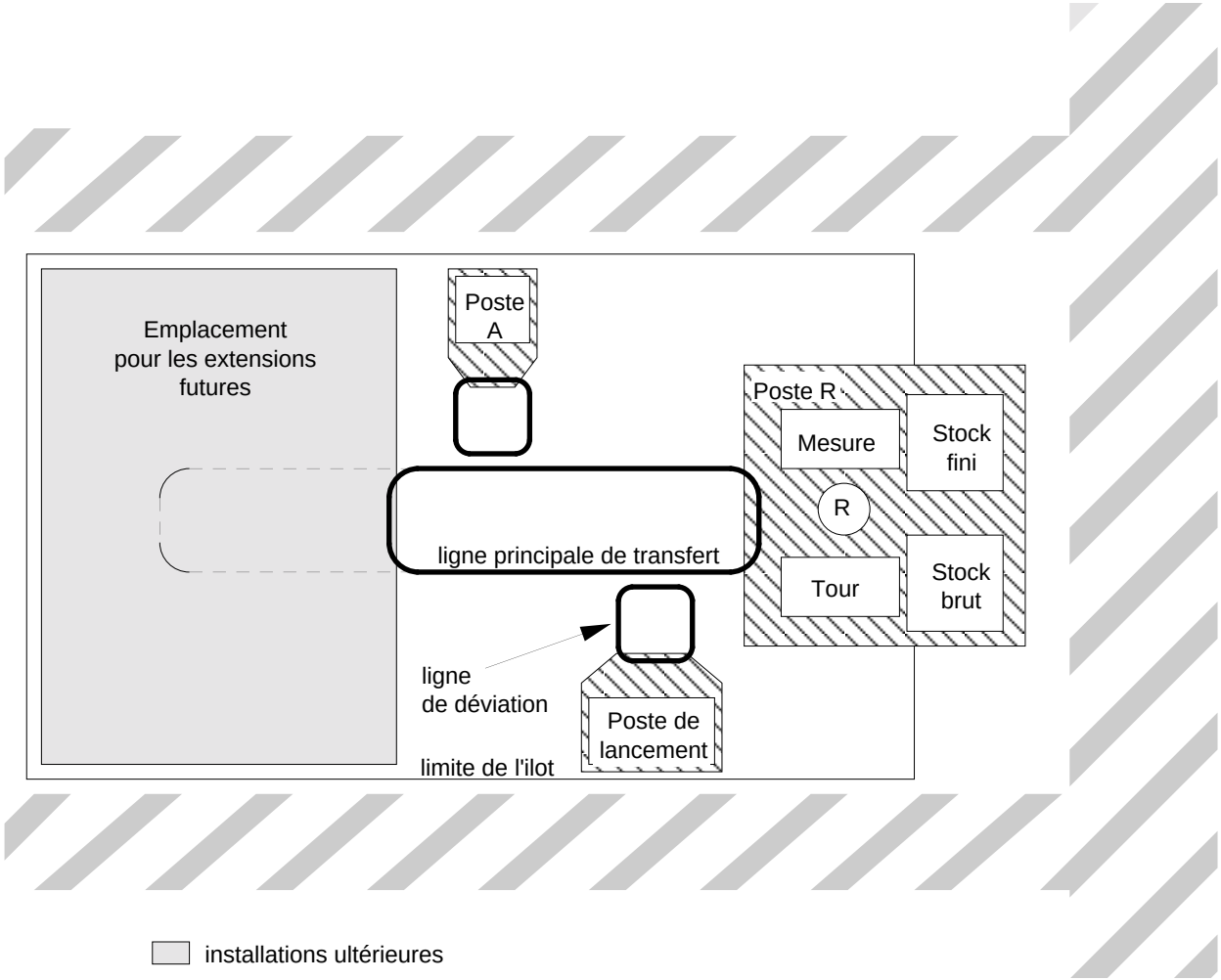


Figure 63 : implantation de l'îlot

Poste de lancement

L'opérateur du poste de lancement reçoit tous les jours un plan de lancement sous la forme d'un listing venant de la gestion de production. Ce listing contient le lancement des produits à effectuer dans la journée.

Exemple de plan de lancement :

heure de lancement	quantité	Référence produit
8h15	5	GVDG 1235 4564
	2	UDFS 4586 1568
	8	KCSH 1684 3655
8h45	12	GVDG 1235 4564
9h15	3	UDFS 4586 1568
	2	KCSH 1684 3655
	5	GVDG 1235 4564

Le rôle de l'opérateur de conduite est double :

à chaque heure de lancement il doit préparer chaque palette correspondant aux produits à réaliser, et les "injecter" sur la boucle principale de transfert,

il réceptionne les palettes vides, extrait le compte rendu de production pour :

- noter les produits réalisés dans le cas où la production s'est bien passée,
- relancer la fabrication d'un produit dont la production n'a pu être menée à terme.

En fin de journée l'opérateur de lancement remet à la gestion de production le bilan de production de la journée contenant :

- la liste des produits effectivement réalisés,
- la liste des produits n'ayant pas été réalisés,
- la liste des rebuts de production.

Un terminal alpha numérique permet un dialogue avec le système d'identification des produits .

Poste de production : poste A

Le poste A est apte à réaliser un certain nombre d'opérations sur les produits. Ces opérations sont identifiées avec une référence unique pour l'ensemble des postes de l'îlot de production.

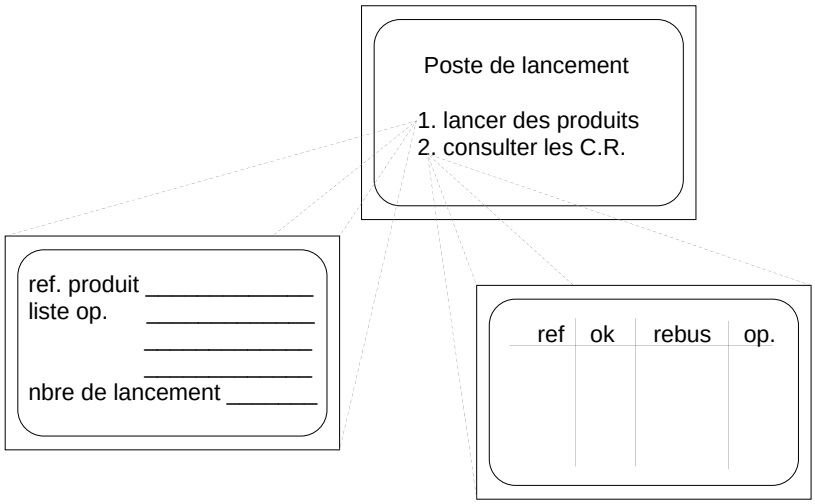


Figure 64 : vue des écrans de dialogue avec les écrans de lancement

Sur un produit une seule opération est effectuée à la fois. Il n'est pas non plus prévu d'enchaîner deux opérations sur un produit, sans que ce dernier ait subi une nouvelle opération sur un autre poste.

En fonctionnement normal, le poste A détourne sur sa ligne de dérivation tous les produits dont il est apte à réaliser l'opération nécessaire, et cela tant que son stockage amont n'est pas plein (la limite du stock amont est fixée à 3 mais peut être modifié après un retour d'expérience de l'exploitation). Lorsque le poste A est libre et qu'une palette est présente dans le stock amont, alors le poste met en place la palette dans sa zone de travail, et effectue l'opération nécessaire. Dès que l'opération est terminée, le poste met à jour le système d'identification des produits et renvoie la palette sur la ligne principale de transfert.

Poste de production : poste R

Le poste R est apte à réaliser un certain nombre d'opérations sur les produits. Ces opérations, qui sont identifiées avec une référence unique pour l'ensemble des postes de l'îlot de production, sont les suivantes :

- charger sur une palette vide (mais préparée) une pièce centrée et dressée,
- mesurer la pièce qui se trouve sur la palette,
- décharger la palette de sa pièce, mesurer cette dernière et la stocker dans le bac des produits finis ou des rebuts selon le résultat de la mesure.

Sur un produit une seule opération est effectuée à la fois. Il n'est pas non plus prévu d'enchaîner deux opérations sur un produit (par exemple un déchargement et un chargement), sans que ce dernier ait subit une nouvelle opération sur un autre poste.

Le poste R n'étant pas sur une ligne de dérivation, mais sur la ligne principale de transfert, il ne devra pas utiliser de stockage amont de palettes pour s'assurer un bon taux de production. Le poste R doit éviter tout arrêt inutile du flot des palettes sur la ligne principale, afin de ne pas "couper" l'approvisionnement en matière d'œuvre des autres postes de production. Lorsqu'une palette se présentera dans sa zone de travail, il la conservera uniquement s'il est en mesure de débuter immédiatement l'opération nécessaire. Sinon il libère la palette, rétablissant ainsi le flots des produits.

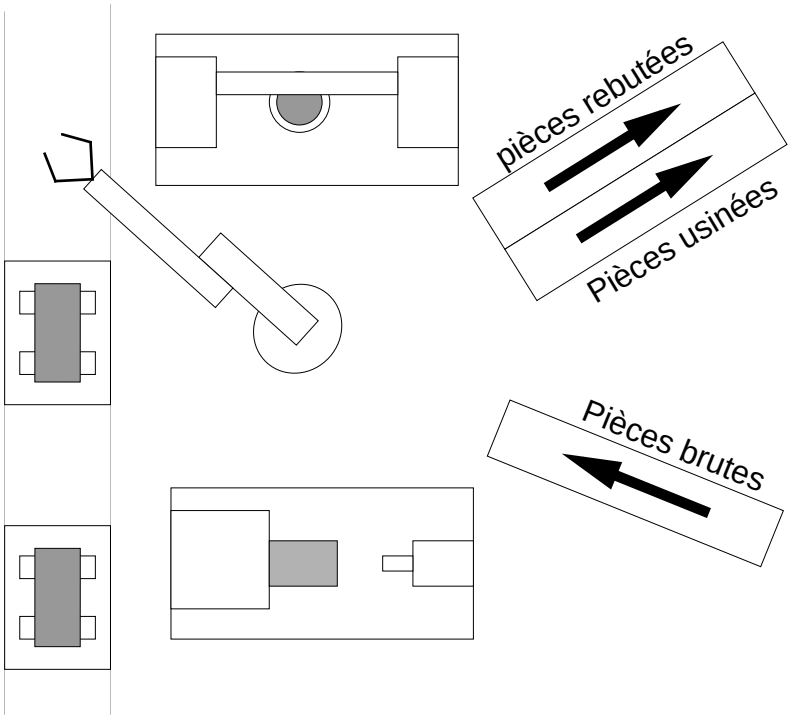


Figure 65 : implantation du poste R

Implantation du poste de lancement

La figure 66 présente les différentes zones du processus situées autour de la boucle de lancement. On y retrouve :

- le poste de lancement avec son système d'identification dynamique,
- la zone de stock amont au poste de lancement,
- l'aiguillage assurant l'entrée et la sortie des palettes, avec son système d'identification dynamique.

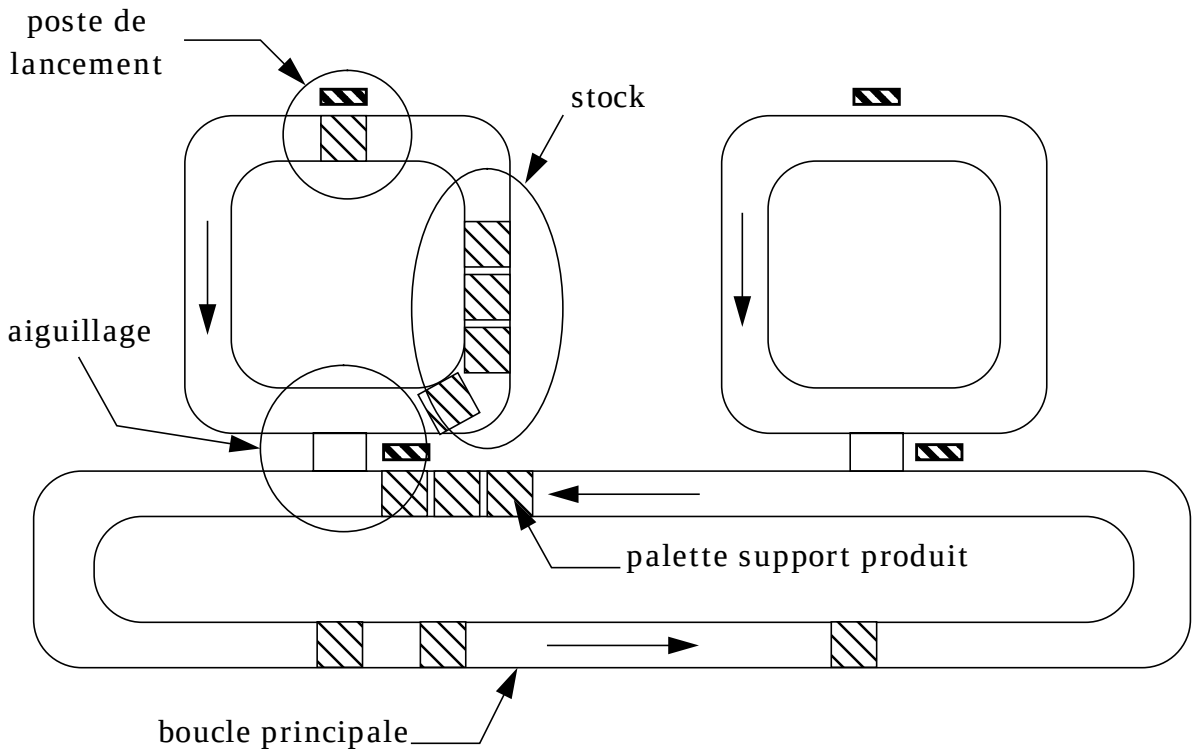


Figure 66 : implantation du poste de lancement

2.2.2 Spécifications fonctionnelles

Les spécifications de POSTEL sont établies à l'aide de la méthode SA-RT, et selon le formalisme de Hatley et Pirbhai. Les modèles produits sont (dans l'ordre d'apparition) :

- le modèle des besoins, constitué, pour chaque niveau, d'un DFD (Diagramme des flots de données), d'un DFC (Diagramme des flots de contrôles), d'une CSPEC (spécification des contrôles) et éventuellement de PSPECs (spécification des processus terminaux),
- les structures de données qui décrivent les flots composés,
- le dictionnaire des besoins.

Modèle des besoins SART

Contexte

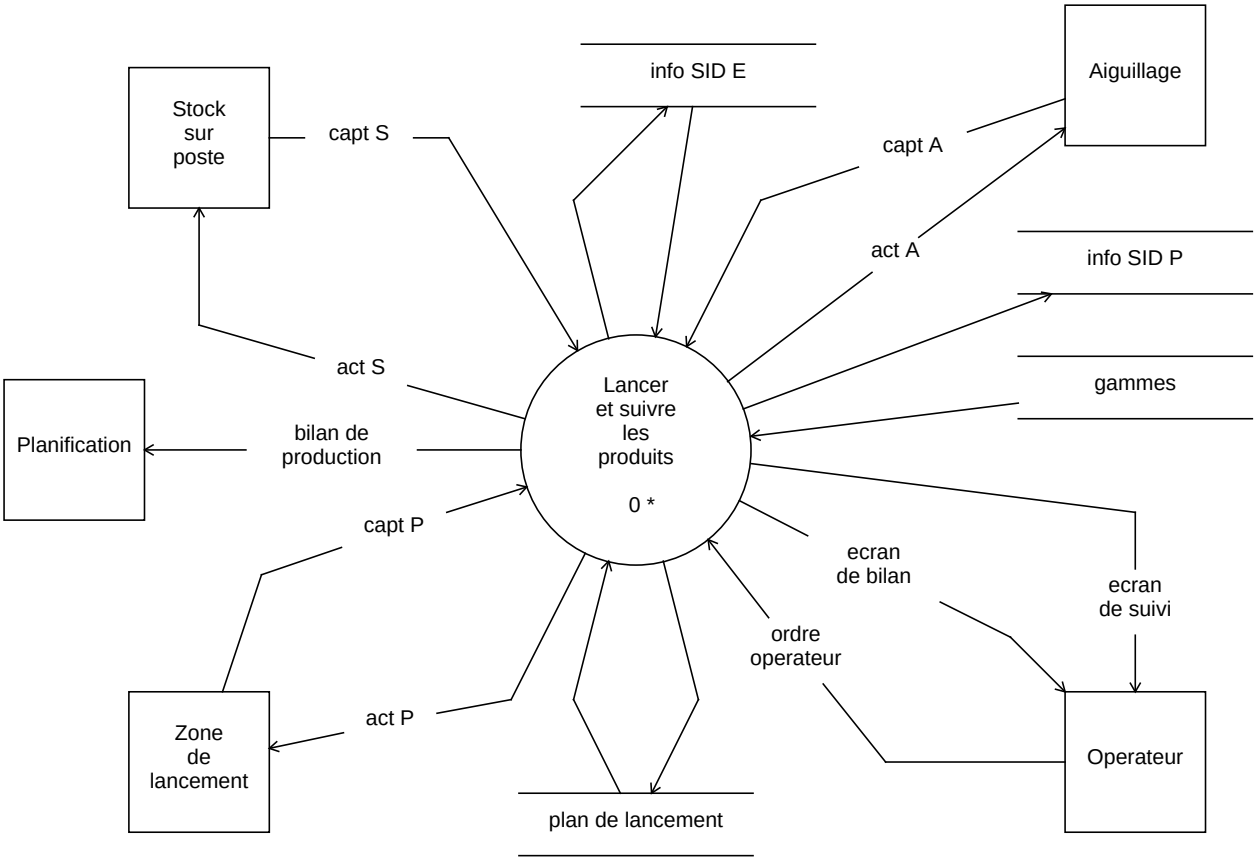


Diagramme de contexte de données

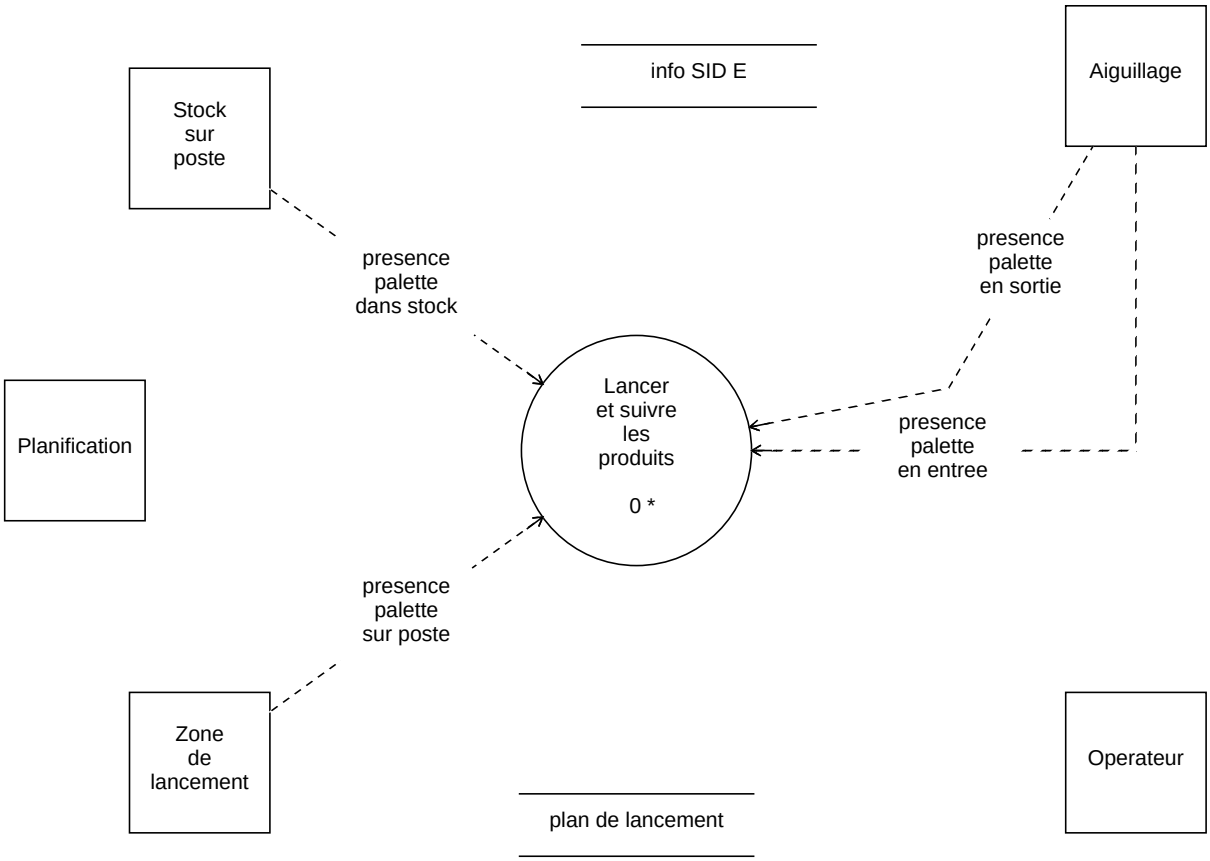
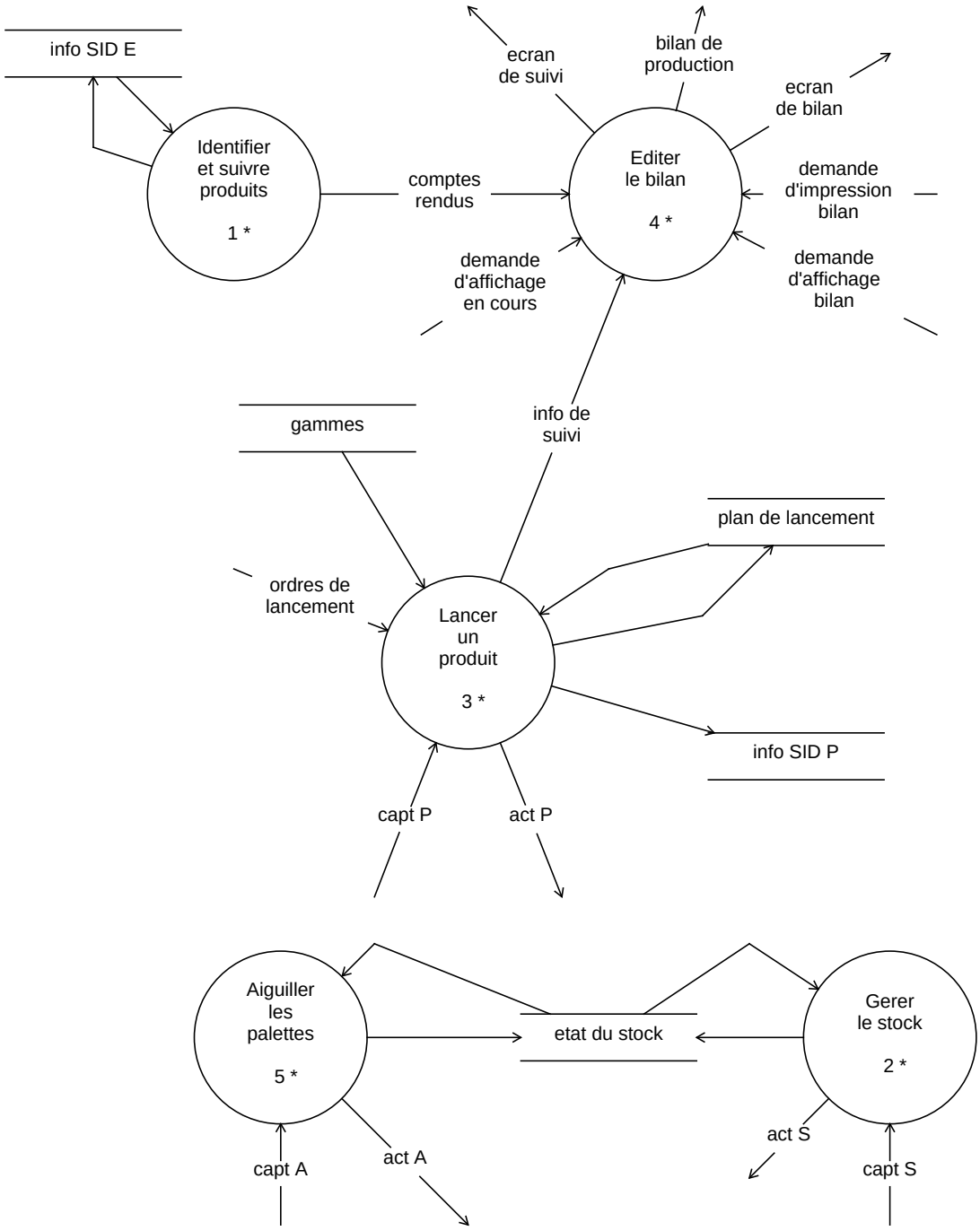
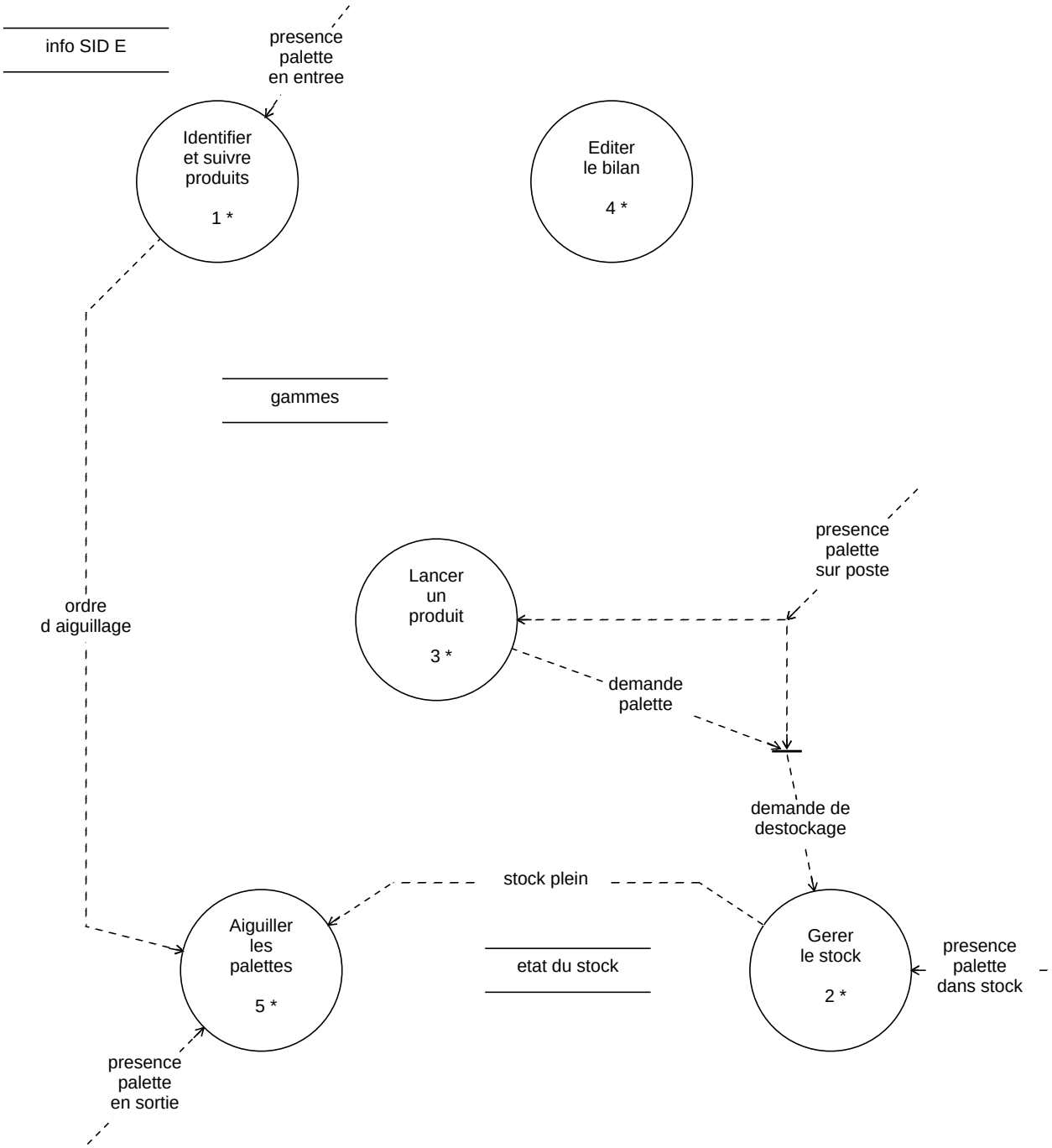


Diagramme de contexte de contrôle

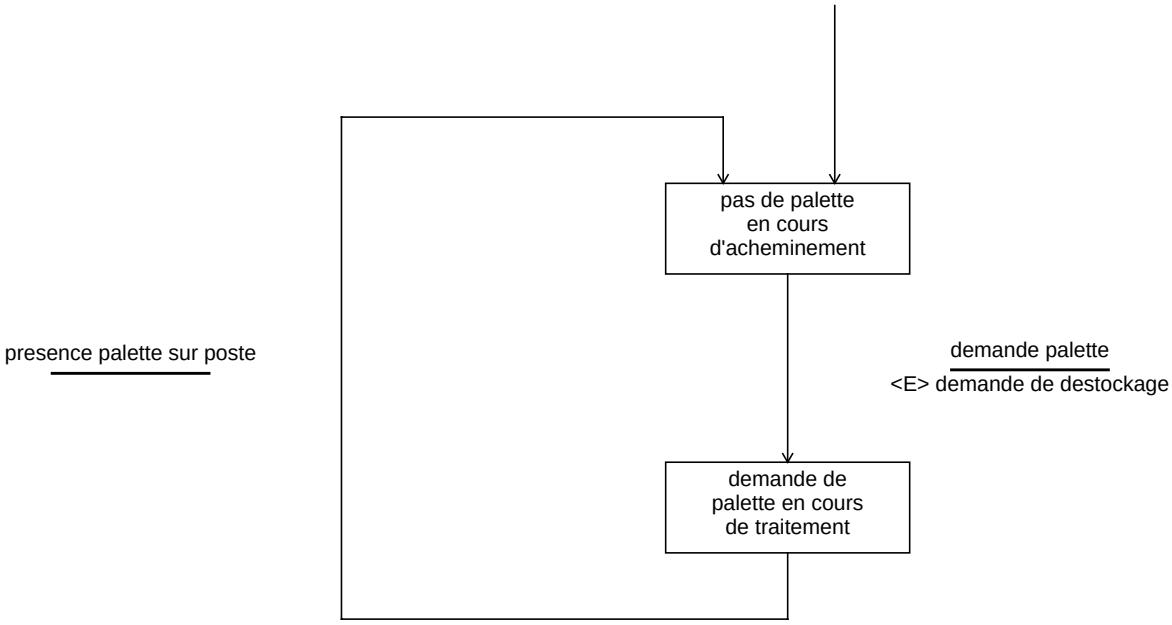
Niveau 0



DFD0 : Lancer_et_suivre_les_produits

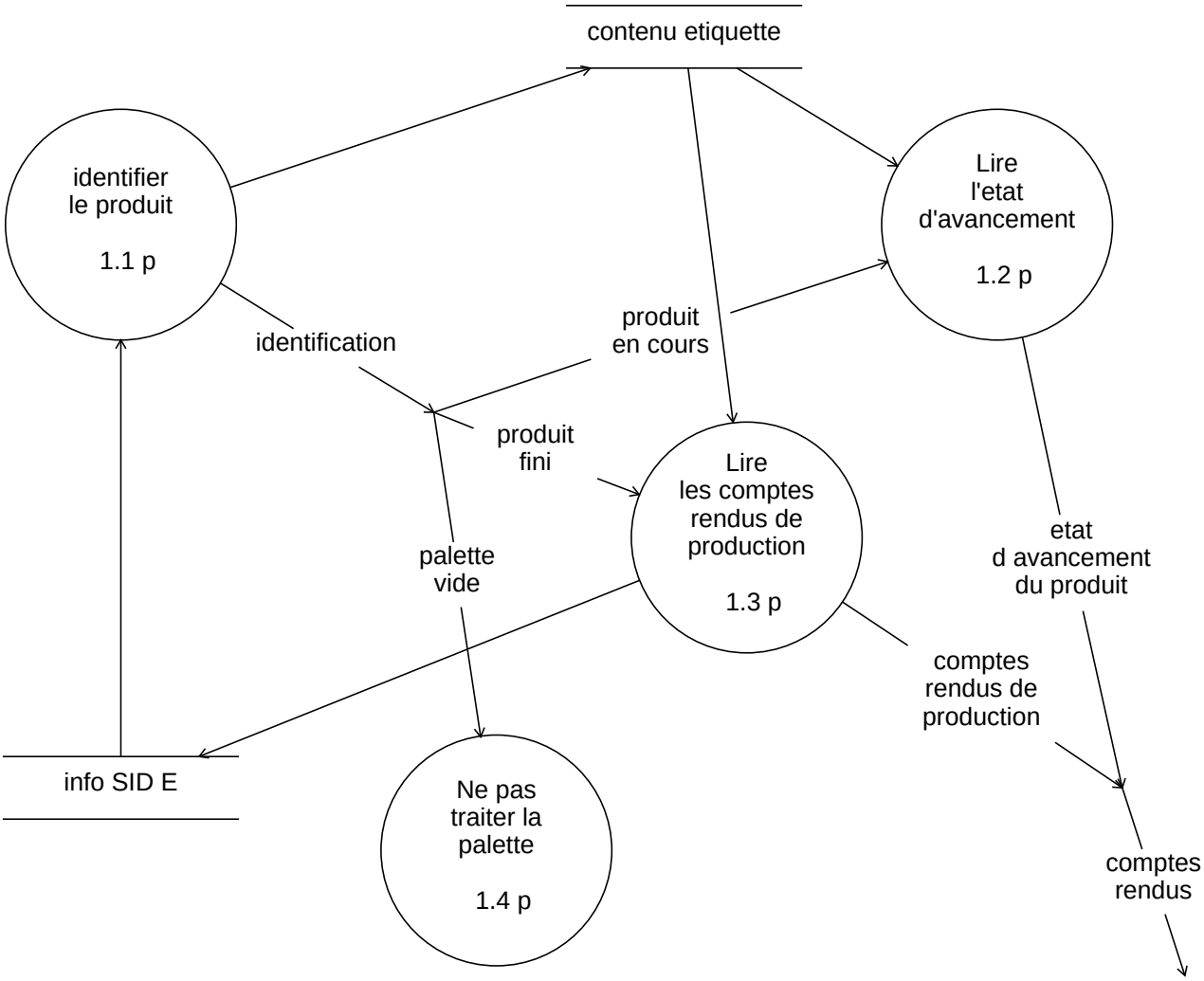


DFC0 : Lancer_et_suivre_les_produits

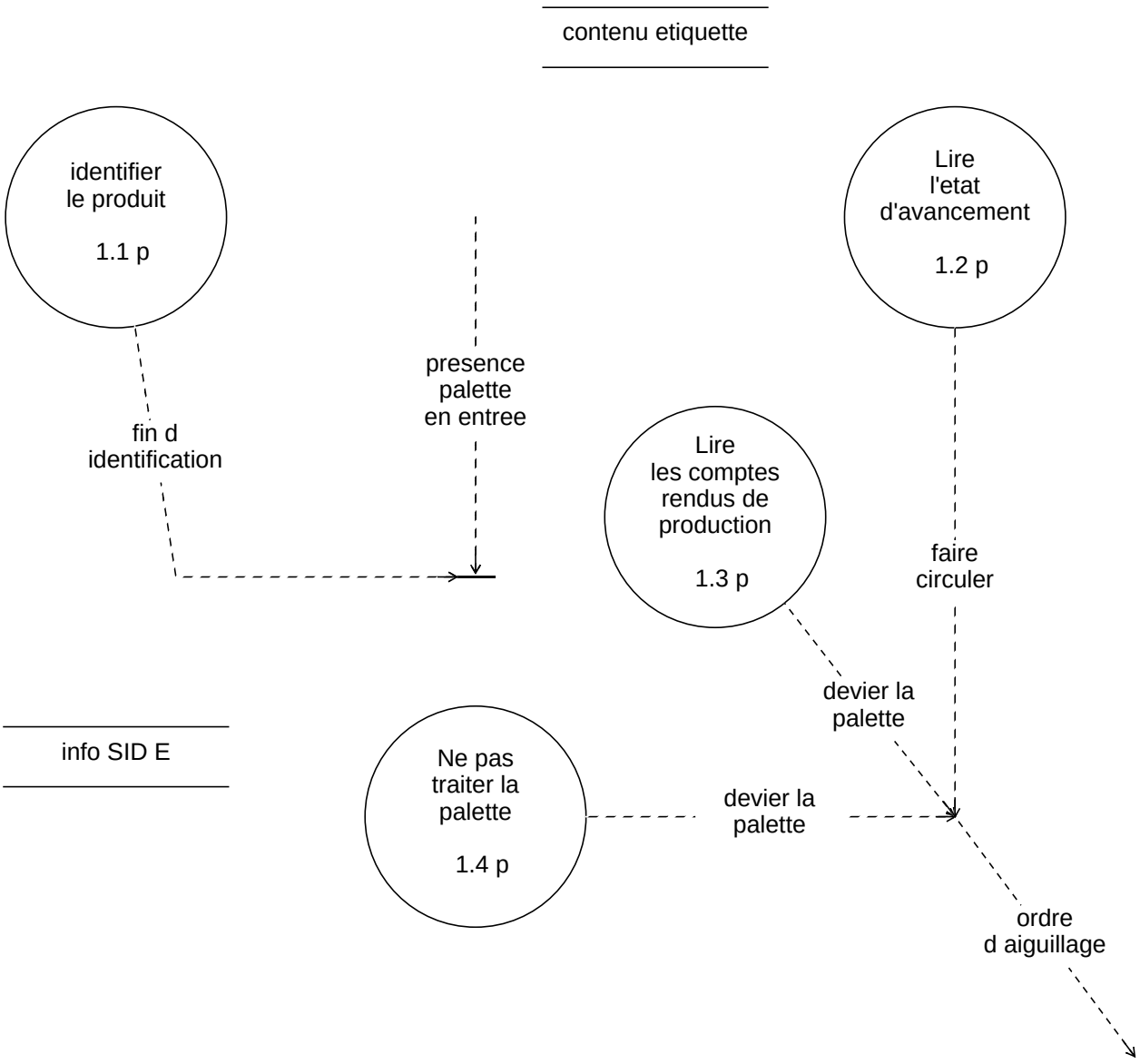


CSpec0 : Lancer_et_suivre_les_produits

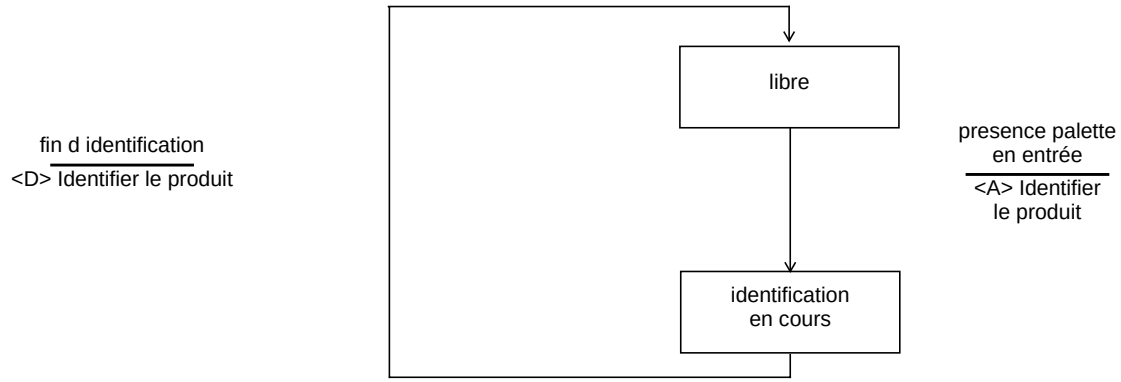
Niveau 1



DFD1 : Identifier_et_suivre_produits



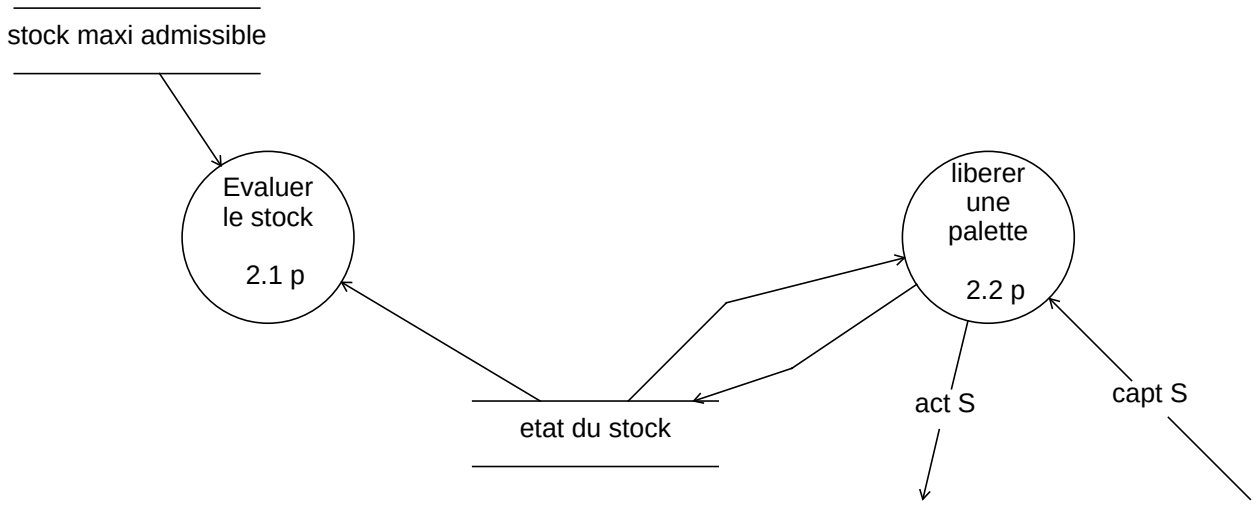
DFC1 : Identifier_et_suivre_produits



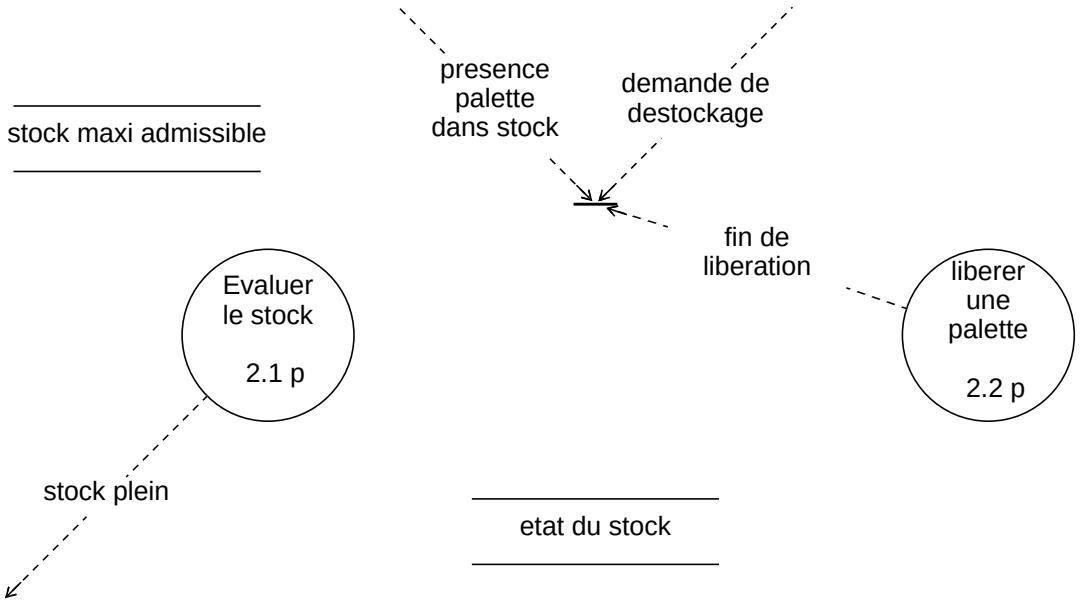
CSpec1 : Identifier_et_suivre_produits

```
 Process 1.1: identifier_le_produit

  description
    Lire les "info SID E"
    Identifier la nature de la production
    selon les cas
      pas de production associee a la palette
        Emettre "palette vide"
      production en cours
        Emettre "produit en cours"
      production terminee
        Emettre "produit fini"
    fin selon
    Emettre les informations relatives au lancement et au suivi vers "contenu
etiquette"
  end pspec


Process 1.2: Lire_l'etat_d'avancement
   description
    lorsqu'un "produit en cours" se presente
      lire le "contenu etiquette"
     Extraire les informations relatives aux phases (effectuees, non effectuer)
      Emettre l'"etat du produit en cours"
  end pspec

Process 1.3: Lire_les_comptes_rendus_de_production
 description
    lorsqu'un "produit fini" se presente
      Lire de "contenu etiquette"
      Extraire les comptes rendus relatifs aux phases de production effectuees
      Emettre l'ordre de "devier la palette"
  end pspec


Process 1.4: Ne_pas_traiter_la_palette

  description
    Si la donnee "palette vide" est recu
      Emettre "devier la palette"
    fin si
  end pspec
```

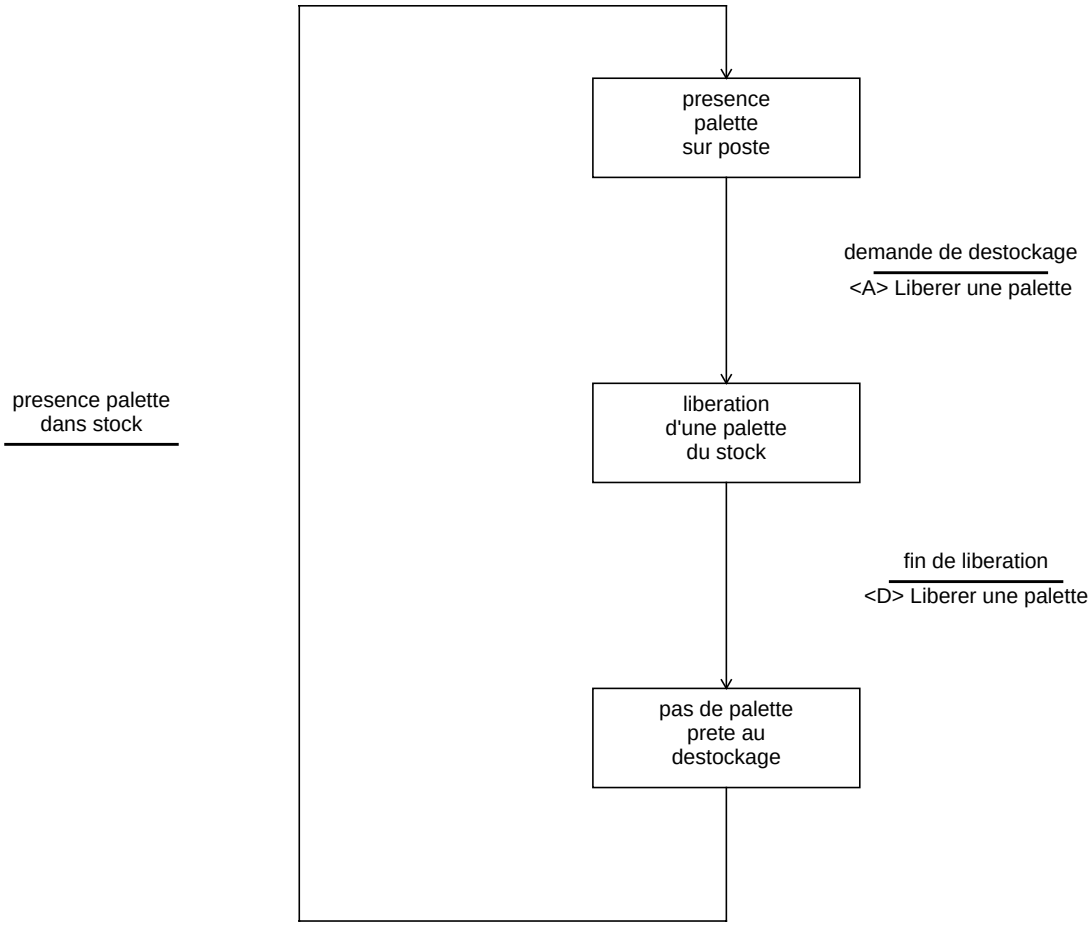
Niveau 2



DFD2 : Gerer_le_stock



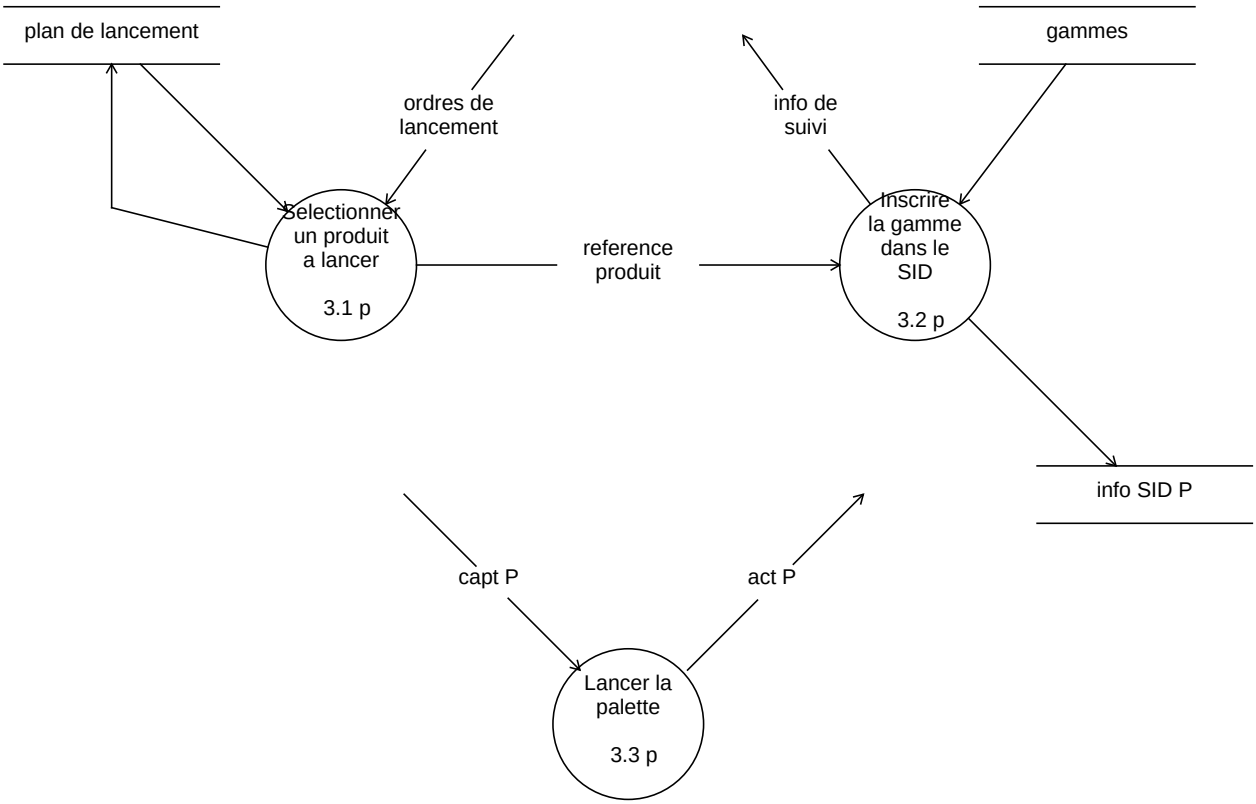
DFC2 : Gerer_le_stock



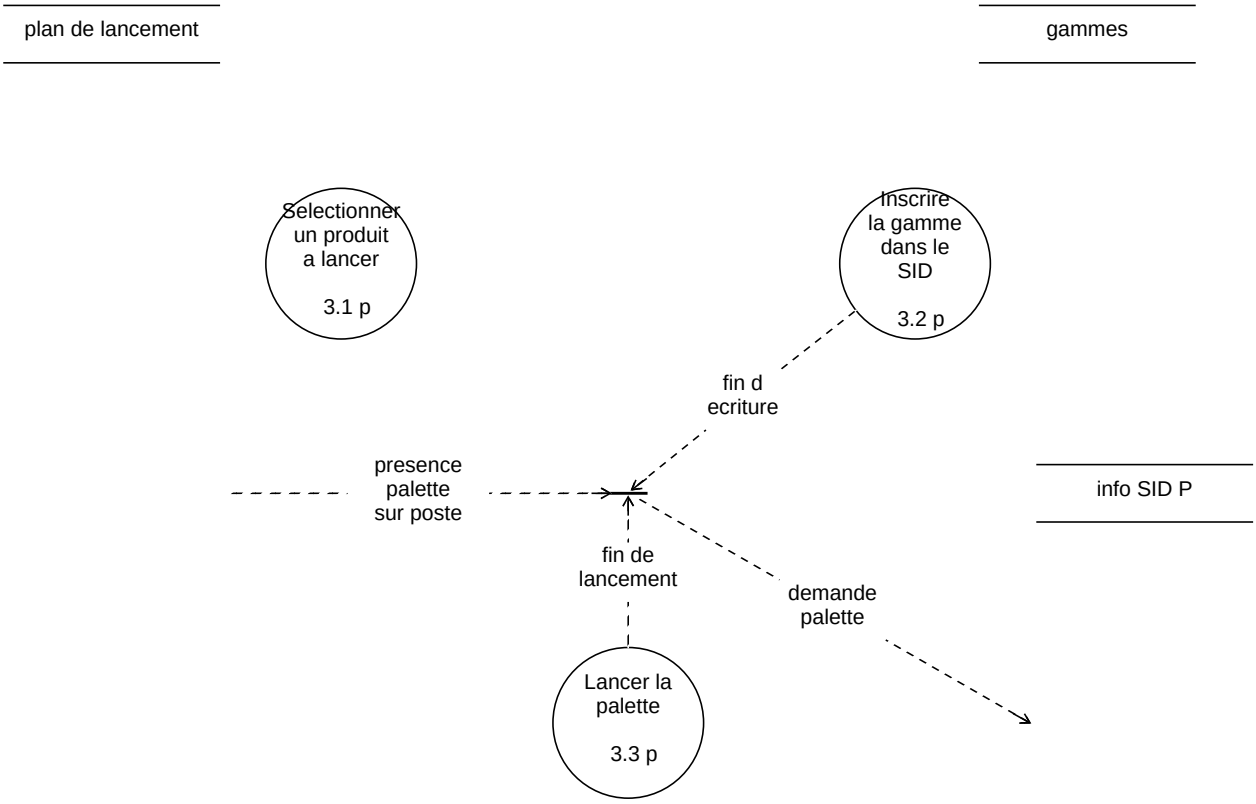
CSpec2 : Gerer_le_stock

```
Process 2.1: Evaluer_le_stock
 description
    des que "etat du stock" est superieur ou egal a "stock maxi admissible"
      emettre "stock plein"
  end pspec


Process 2.2: liberer_une_palette
 description
    controler les actionneurs "act S" au vu des capteurs "capt S" pour liberer
une palette du stock
    emettre "fin de liberation"
  end pspec
```

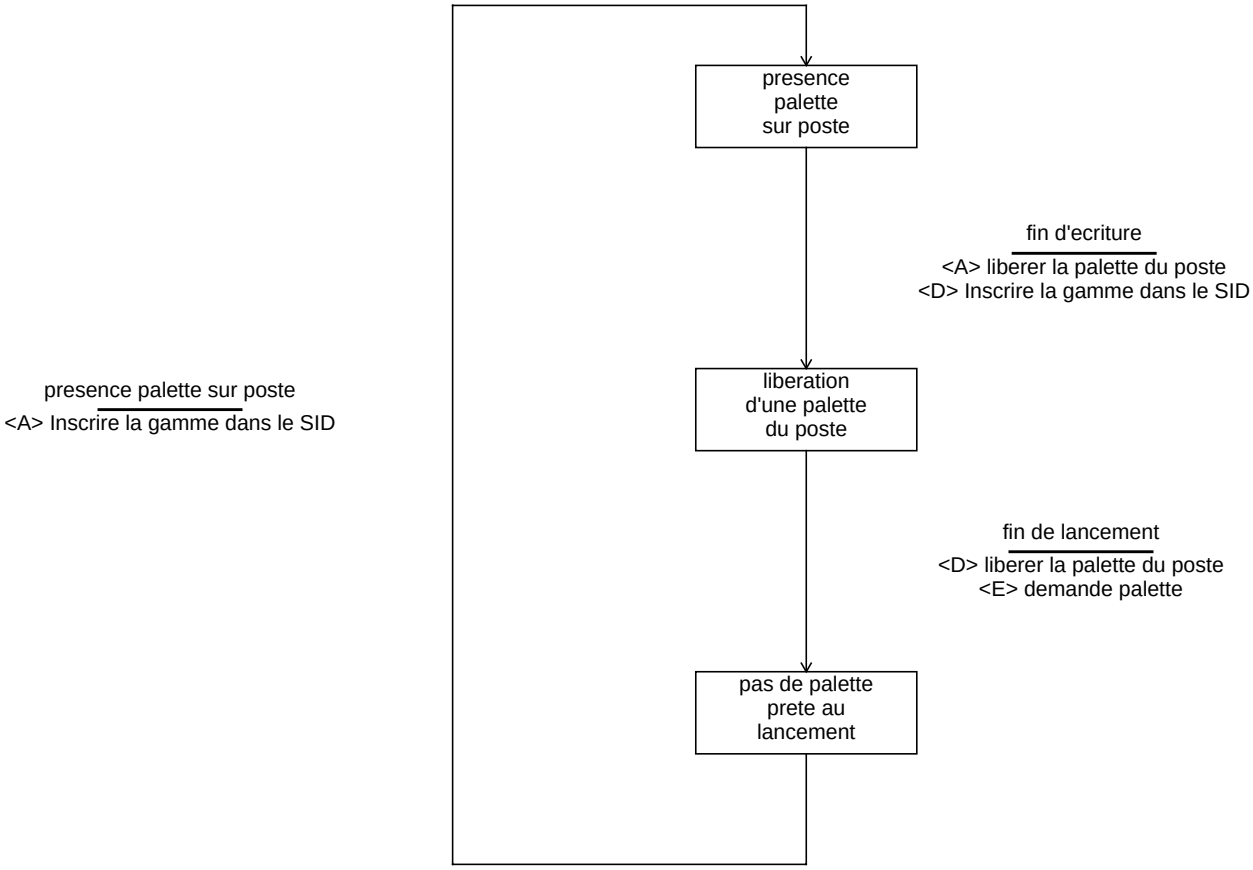
Niveau 3



DFD3 : Lancer_un_produit



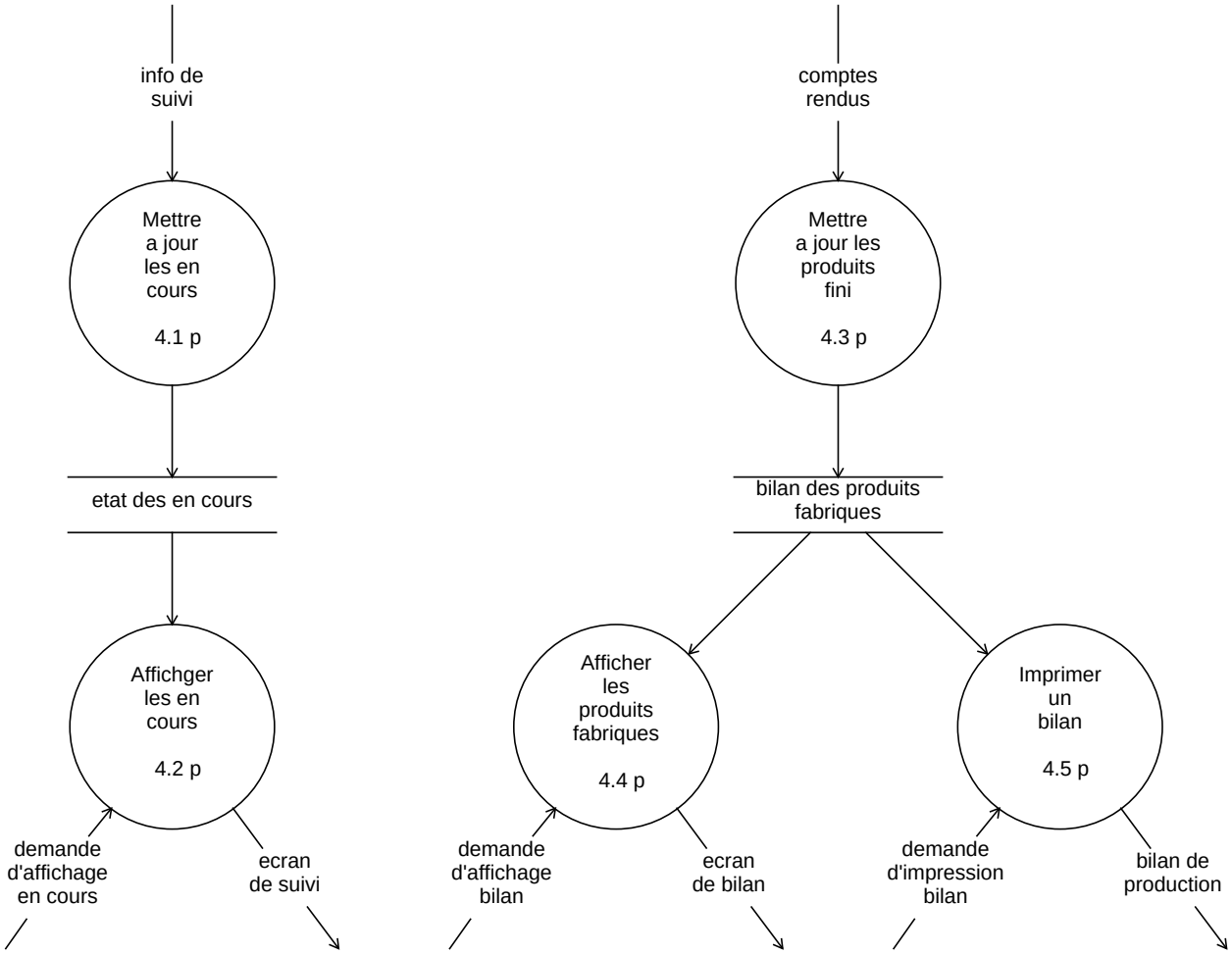
DFC3 : Lancer_un_produit



CSpec3 : Lancer_un_produit

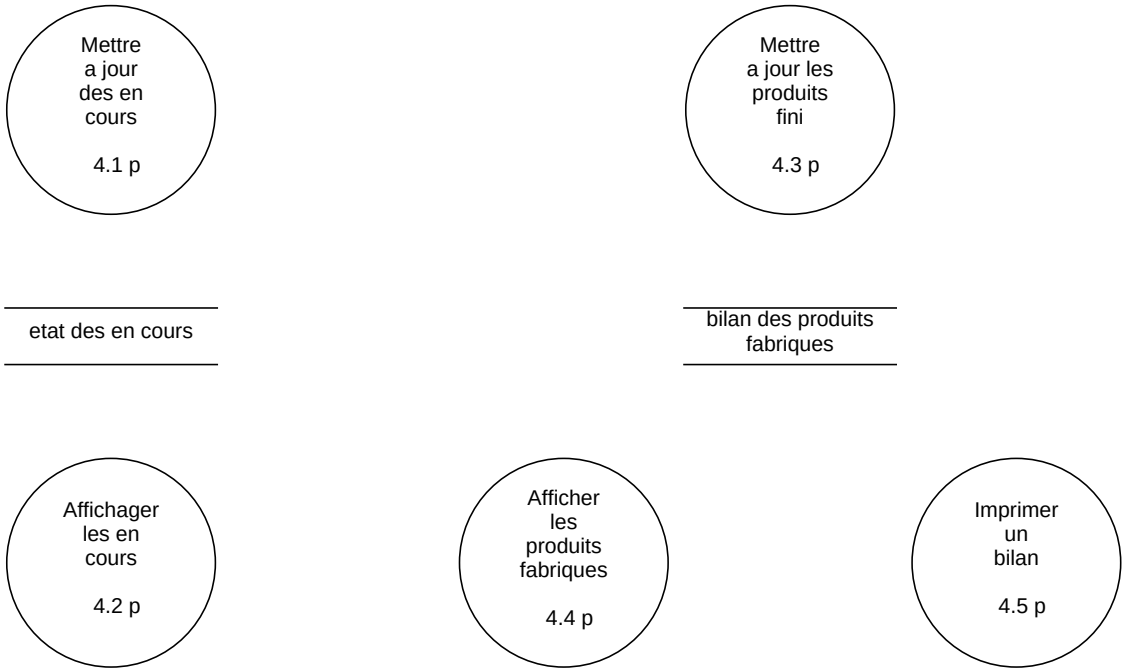
```
Process 3.1: Selectionner_un_produit_a_lancer
 description
    des que "ordres de lancement"
      selectionner dans "plan de lancement" un nouveau produit
      emettre la "reference produit" du produit selectionne
      mettre a jour dans le "plan de lancement" les produits deja lances
  end pspec

Process 3.2: Inscrire_la_gamme_dans_le_SID

  description
    A partir de la "reference produit"
      Rechercher la gamme correspondante
      Ecrire dans le systeme d'identification dynamique ("info SID P"), les
informations necessaires de la gamme
      Emettre "fin d ecriture"
  end pspec

Process 3.3: Lancer_la_palette

  description
    controler les actionneur "act P" au vu des capteurs "capt S" pour liberer
la palette du poste
    emettre "fin de lancement"
  end pspec
```

Niveau 4



DFD4 : Editer_le_bilan

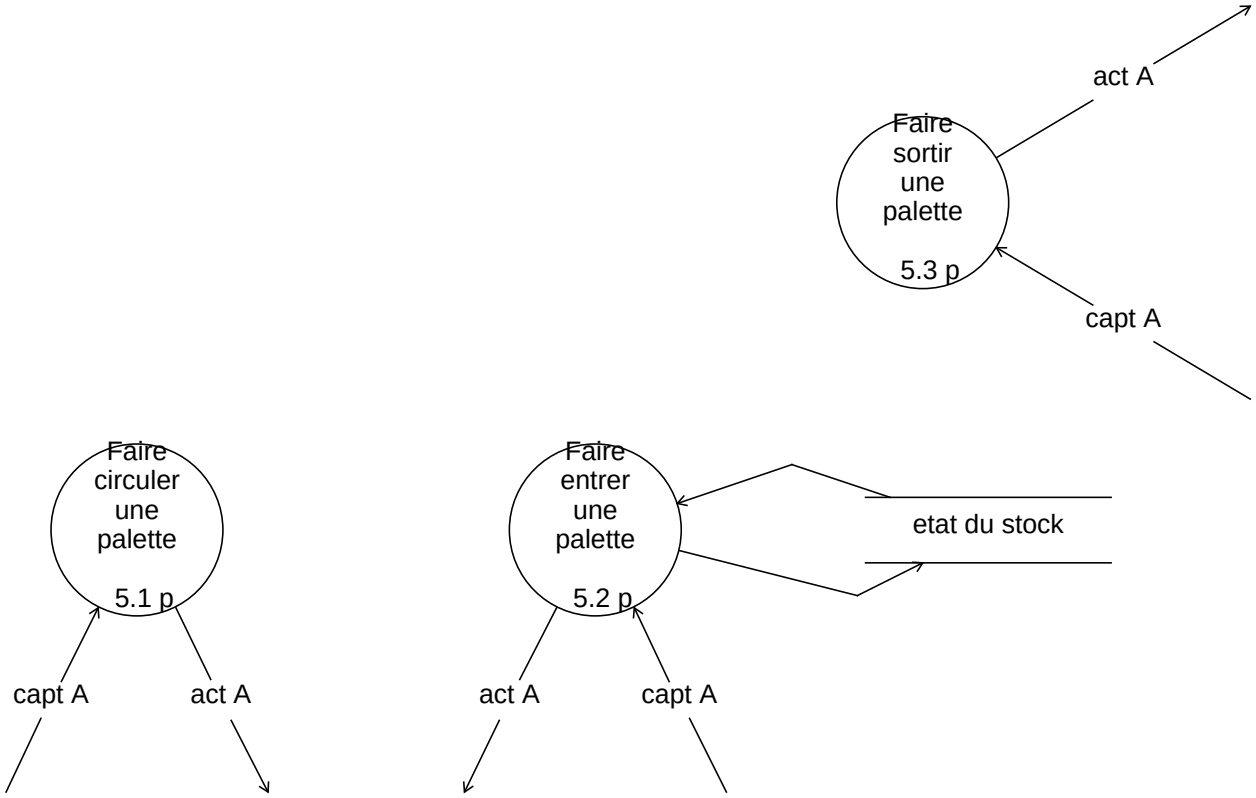
```
Process 4.1: Mettre_a_jour_des_en_cours
 description
    A chaque reception d'un "info de suivi"
       Mettre a jour l'"etat des en cours"
  end pspec

Process 4.2: Affichager_les_en_cours
 description
    lors d'une "demande d affichage en cours"
      Mettre en forme les informations sur l'"etat en cours"
      Emettre "ecran de suivi"
  end pspec
```



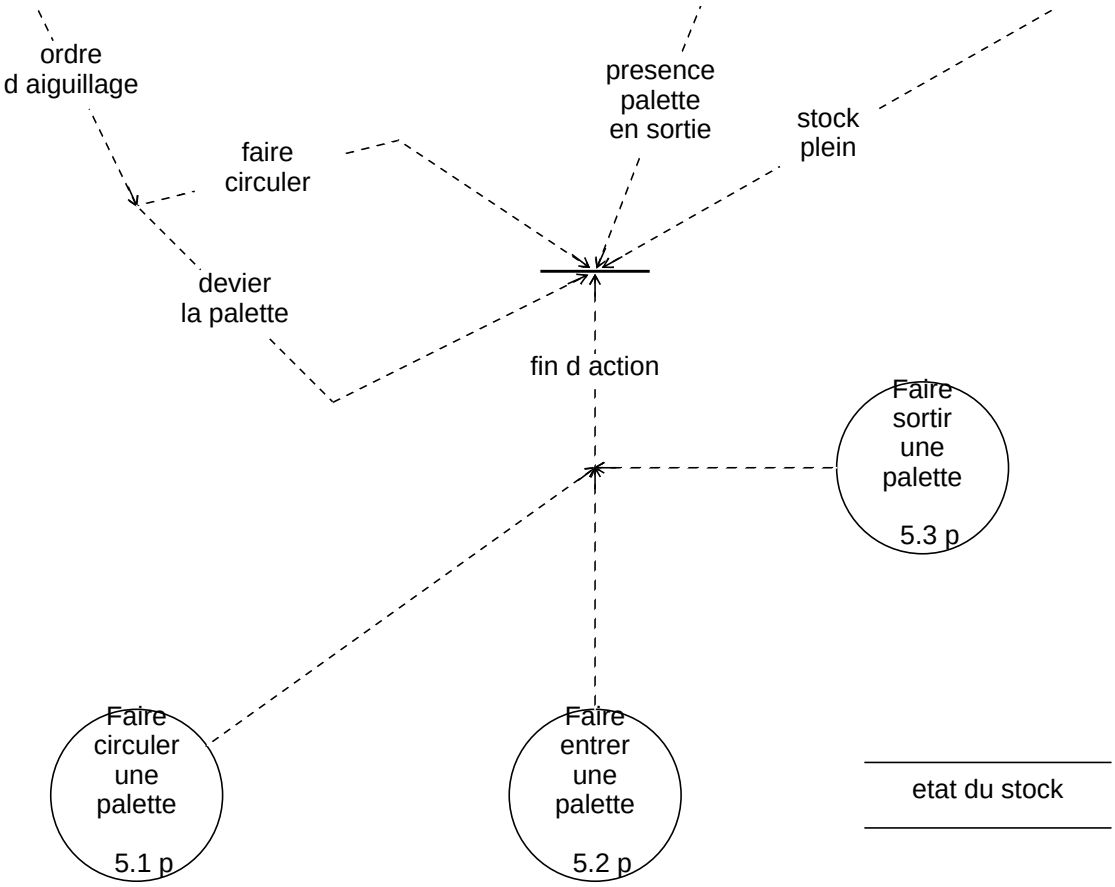
DFC4 : Editer_le_bilan

```
Process 4.4: Afficher_les_produits_fabriques
 description
    lors d'une "demande d affichage bilan"
      Mettre en forme les informations du "bilan des produits fabriques"
      Emettre "ecran de bilan"
  end pspec

Process 4.5: Imprimer_un_bilan
  description
    lors d'une "demande d impression bilan"
      Mettre en forme  les informations du "bilan des produits fabriques"
      Emettre "bilan de production"
  end pspec

Process 4.3: Mettre_a_jour_les_produits_fini

  description
    A chaque reception de "comptes rendus"
      Mettre a jour le "bilan des produits fabriques"
  end pspec
```

Niveau 5

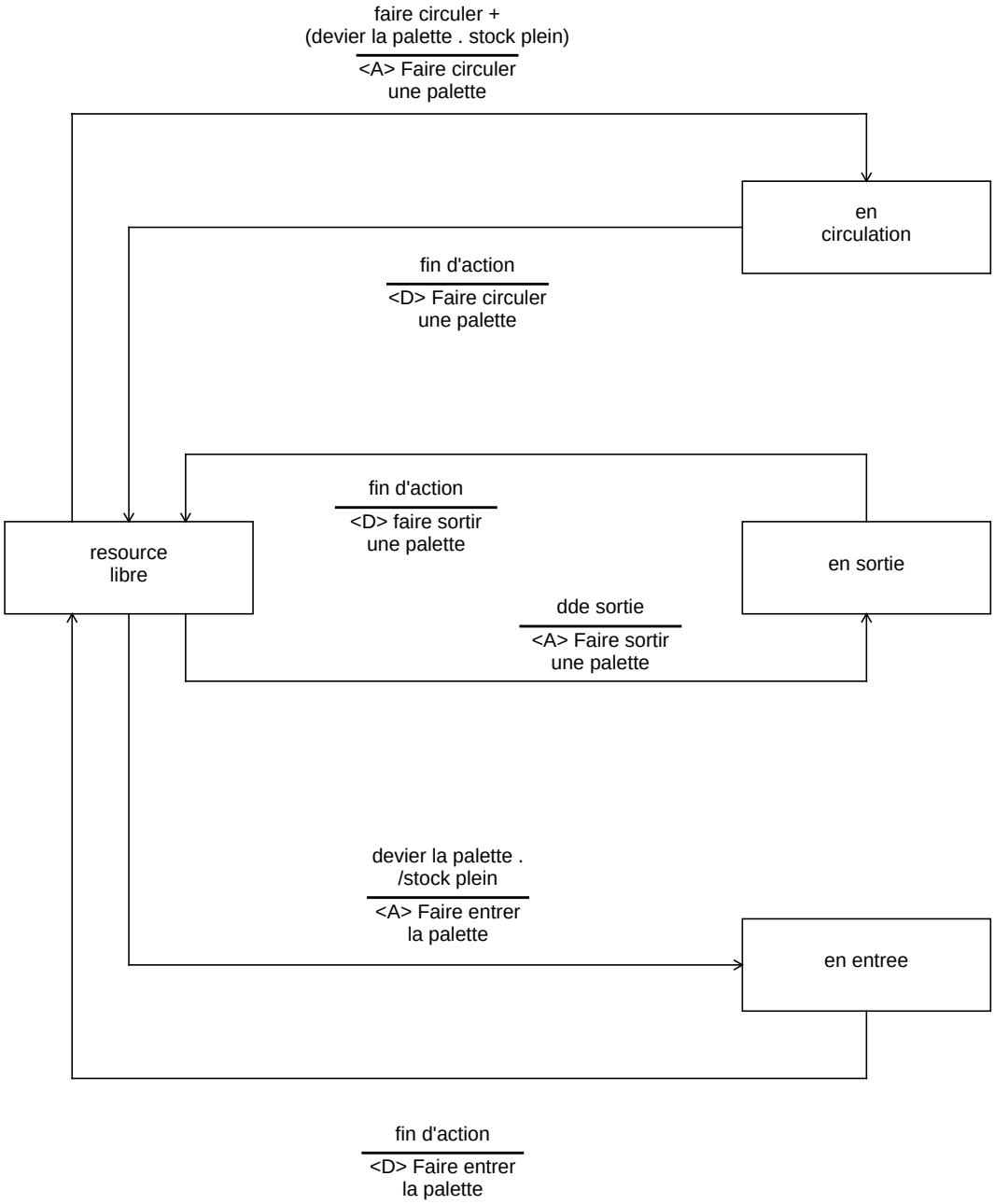


DFD5 : Aiguiller_les_palettes



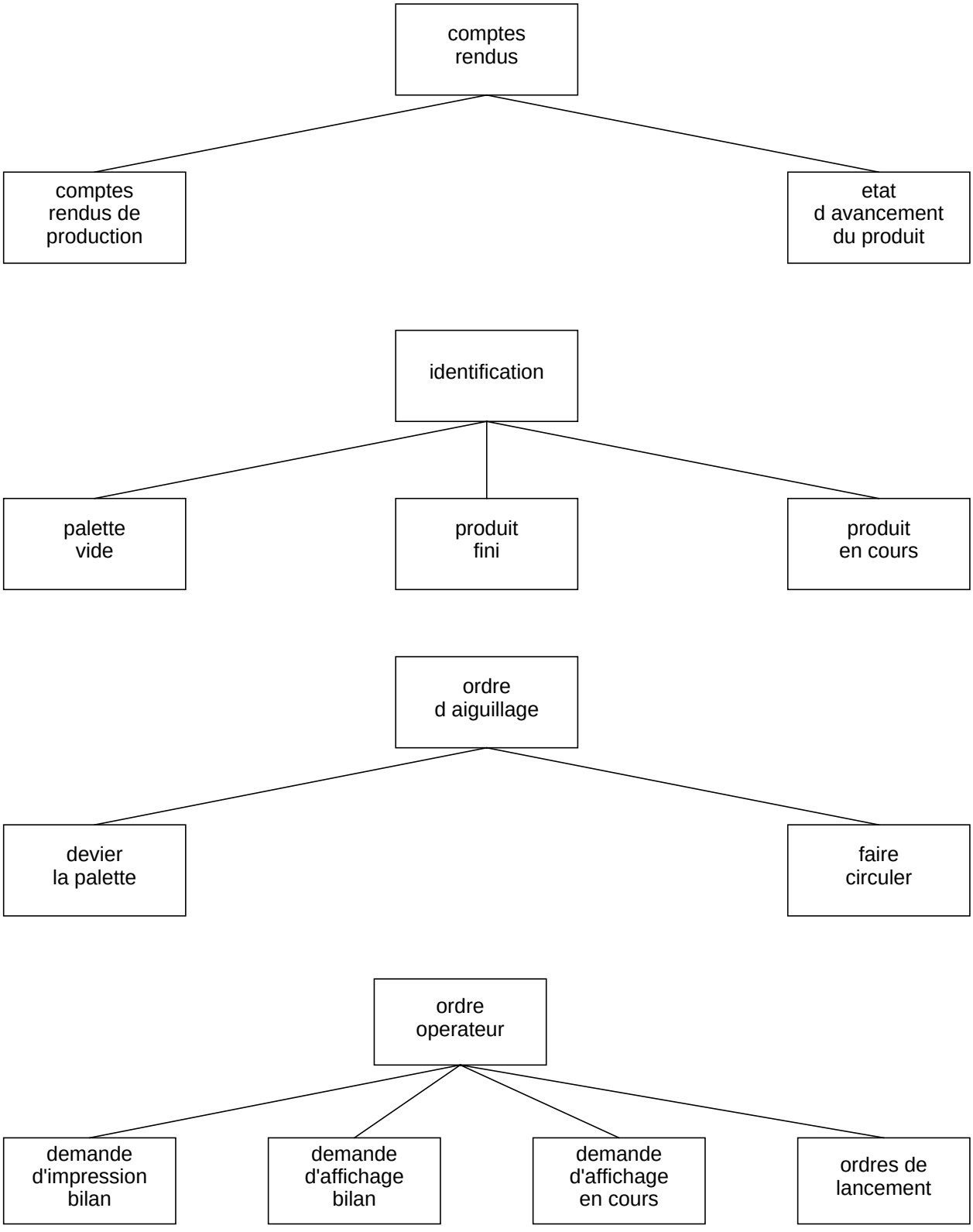
DFC5 : Aiguiller_les_palettes

```
Process 5.1: Faire_circuler_une_palette
 description
    Controler les actionneurs "act A" au vu des capteurs "capt P" pout faire
circuler une palette
    Emettre "fin d action"
  end pspec

Process 5.2: Faire_entrer_une_palette
 description
    Controler les actionneur "act A" au vu des capteurs "capt A" pour faire
entrer une palette
    Emettre "fin d action"
  end pspec

Process 5.3: Faire_sortir_une_palette
 description
    Controler les actionneurs "act A" au vu des capteurs "capt A" pour faire
sortir une palette
    Emettre "fin d action"
  end pspec
```



CSpec5 : Aiguiller_les_palettes

Structure des données



Dictionnaire des besoins

act_A	D	/Ordres a l'attention des actionneurs de l'aiguillage/
act_P	D	/Ordres a l'attention des actionneurs du poste de lancement/
act_S	D	/Ordres a l'attention des actionneurs de la zone de stockage/
bilan_de_production	D	/informations regroupant l'ensemble des comptes rendus sur les differentes phases d'avancement d'un produit/
bilan_des_produits_fabriques	D	/bilan et compte-rendu relatifs aux produits deja fabriques/
capt_A	D	/Informations issues des capteurs de l'aiguillage/
capt_P	D	/Informations issues des capteurs du poste de lancement/
capt_S	D	/Informations issues des capteurs de la zone de stockage/
comptes_rendus	D	Element du flot de donnee : comptes_rendus
comptes_rendus_de_production	D	/comptes rendus sur la production d'un produit maintenant termine/
contenu_etiquette	D	/informations du systeme d'identification dynamique qui vont etre taitees pour le bilan et le suivi/
demande_d'affichage_bilan	D	/demande d'affichage a l'ecran du bilan de production/
demande_d'affichage_en_cours	D	/demande d'affichage a l'ecran des encours de production, c'est-a-dire des informations de suivi de production/
demande_d'impression_bilan	D	/demande d'impression du bilan de production/
demande_de_destockage	C	/demande de liberation d'une palette du stock/
demande_palette	C	/information indiquant que le poste est pres a recevoir une palette en provenance du stock/
devier_la_palette	C	/transferer la palette de la boucle principale vers la boucle de lancement/
ecran_de_bilan	D	/information relatives au bilan de la production/
ecran_de_suivi	D	/informations relatives a l'etat d'avancement des produits qui ont deja ete lances et dont la production n'est pas terminee/
etat_d_avancement_du_produit	D	/etat d'avancement d'un produit en cours de production/
etat_des_en_cours	D	/informations relatives aux en-cours de production (produits en-cours, etat d'avancement,...)/
etat_du_stock	D	/nombre de palette en stock sur la boucle de lancement/
faire_circuler	C	/ne pas transferer la palette sur la boucle de lancement, c'est-a-dire la laisser traverser l'aiguillage sur la boucle principale/
fin_d_action	C	/Une des missions de l'aiguillage (faire circuler, faire entrer ou faire sortir) vient de se terminer/

fin_d_ecriture	C	/fin d'ecriture des informations relatives a la gamme de fabrication du produit dans le systeme d'identification dynamique du poste de lancement/
fin_d_identification	C	/fin d'identification de la palette en amont de l'aiguillage sur la boucle principale/
fin_de_lancement	C	/Une palette vient d'etre lancee/
fin_de_liberation	C	/une palette vient d'etre liberee du stock de la boucle de lancement/
gammes	D	/liste des phases de production a effectuer pour chaque type de produit/
identification	D	Element du flot de donnee : identification
info_SID_E	D	/informations contenues dans le systeme d'identification dynamique de la palette situee a l'entree de la boucle de lancement c'est-a-dire sur la boucle principale/
info_SID_P	D	/information situee dans le systeme d'identification dynamique de la palette situee au niveau du poste de lancement/
info_de_suivi	D	/informations relatives au produit qui vient d'etre lance et necessaire au suivi de son avancement/
ordre_d_aiguillage	C	Element du flot de controle : ordre_d_aiguillage
ordre_operateur	D	Element du flot de donnee : ordre_operateur
ordres_de_lancement	D	/ordres de lancer un produit en production/
palette_vide	D	/pas de production en cours sur la palette identifiee/
plan_de_lancement	D	/description de l'ordre de lancement des produits et informations necessaires pour chaque lancement/
presence_palette_dans_stock	C	/une palette prete a etre destockee est disponible dans le stock de palettes dans la boucle de lancement/
presence_palette_en_entree	C	/une palette se trouve sur la boucle principale, juste en amont de la boucle de lancement, et prete a etre traitee au niveau de l'aiguillage/ Type : non specifie
presence_palette_en_sortie	C	/une palette est situee sur la boucle de lancement, juste en amont de l'aiguillage, prete a etre transferee sur la boucle principale/
presence_palette_sur_poste	C	/une palette est situee sur la boucle de lancement, au niveau du poste de lancement/
produit_en_cours	D	/il y a une production en cours associee a la palette/
produit_fini	D	/la production associee a la palette est arrivee a son terme/
reference_produit	D	/identifiant du type de produit a lancer/
stock_maxi_admissible	D	/nombre maximun de palette que le stock de la boucle de lancement peut accepter/
stock_plein	C	/le stock de palette sur la boucle de lancement a atteind sa capacite maxi/

2.3 Déroulement de la démarche

2.3.1 Script du scénario de conception

A partir du déroulement de la démarche présentée dans la division 1.2.5, et des résultats du chapitre 1.3 «Construction d'une solution d'architecture» et du chapitre 1.4 «Evaluation et choix d'architecture», nous allons donner un vue plus détaillée de notre démarche de conception, en précisant l'organisation des tâches autour des différents intervenants. Nous avons choisi pour cela le formalisme de représentation du *modèle organisationnel des traitements* (MOT) de la méthode *Merise* qui convient parfaitement à nos attentes. Le tableau 14 présente le MOT relatif à notre démarche de conception d'architecture. On y retrouve les trois acteurs : l'architecte, les outils d'assistance que nous lui proposons, et l'ingénieriste. La cinématique de la démarche y est clairement définie, ainsi que la réponse à la question «qui fait quoi ?».

Dans la suite de ce chapitre, nous allons examiner chaque procédure fonctionnelle du MOT en précisant ce que doit faire l'architecte dans les procédures qui sont de son ressort, et les résultats qui lui sont proposés à la fin des procédures fonctionnelles assurées par l'assistante à la conception d'architecture.

.

Tableau 14 : scénario de la conception d'architecture selon notre méthode

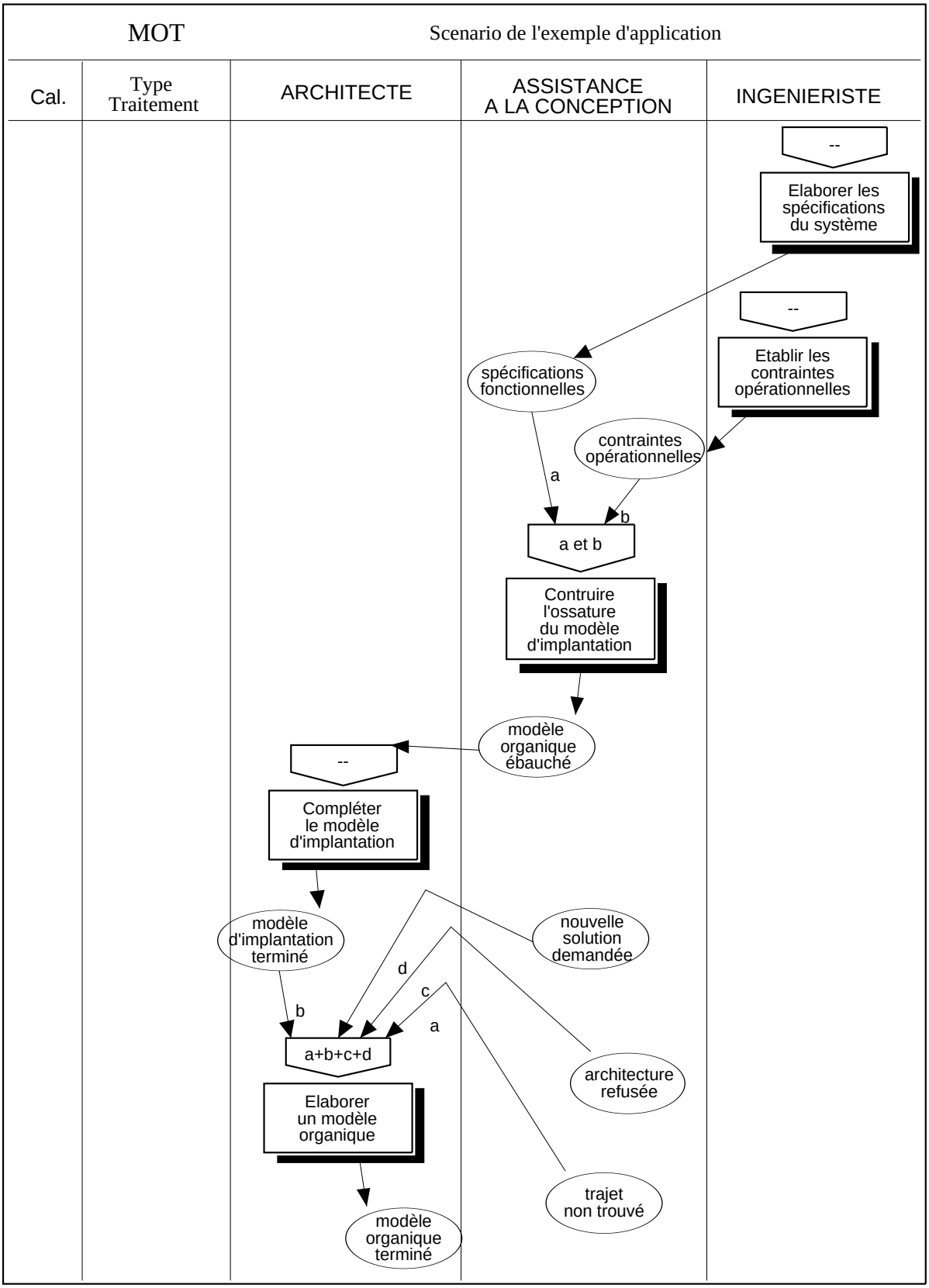


Tableau 14 : scénario de la conception d'architecture selon notre méthode

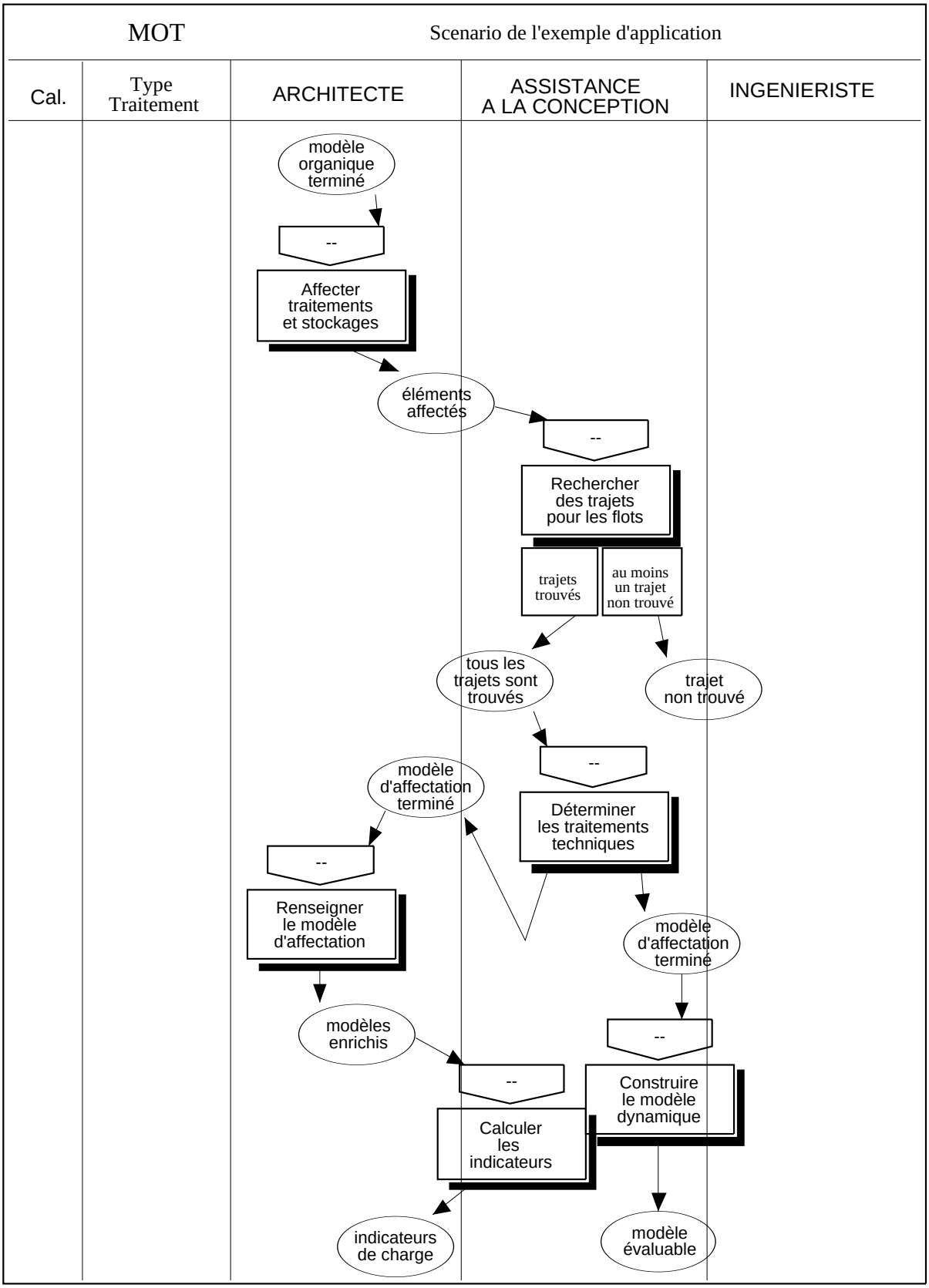


Tableau 14 : scénario de la conception d'architecture selon notre méthode

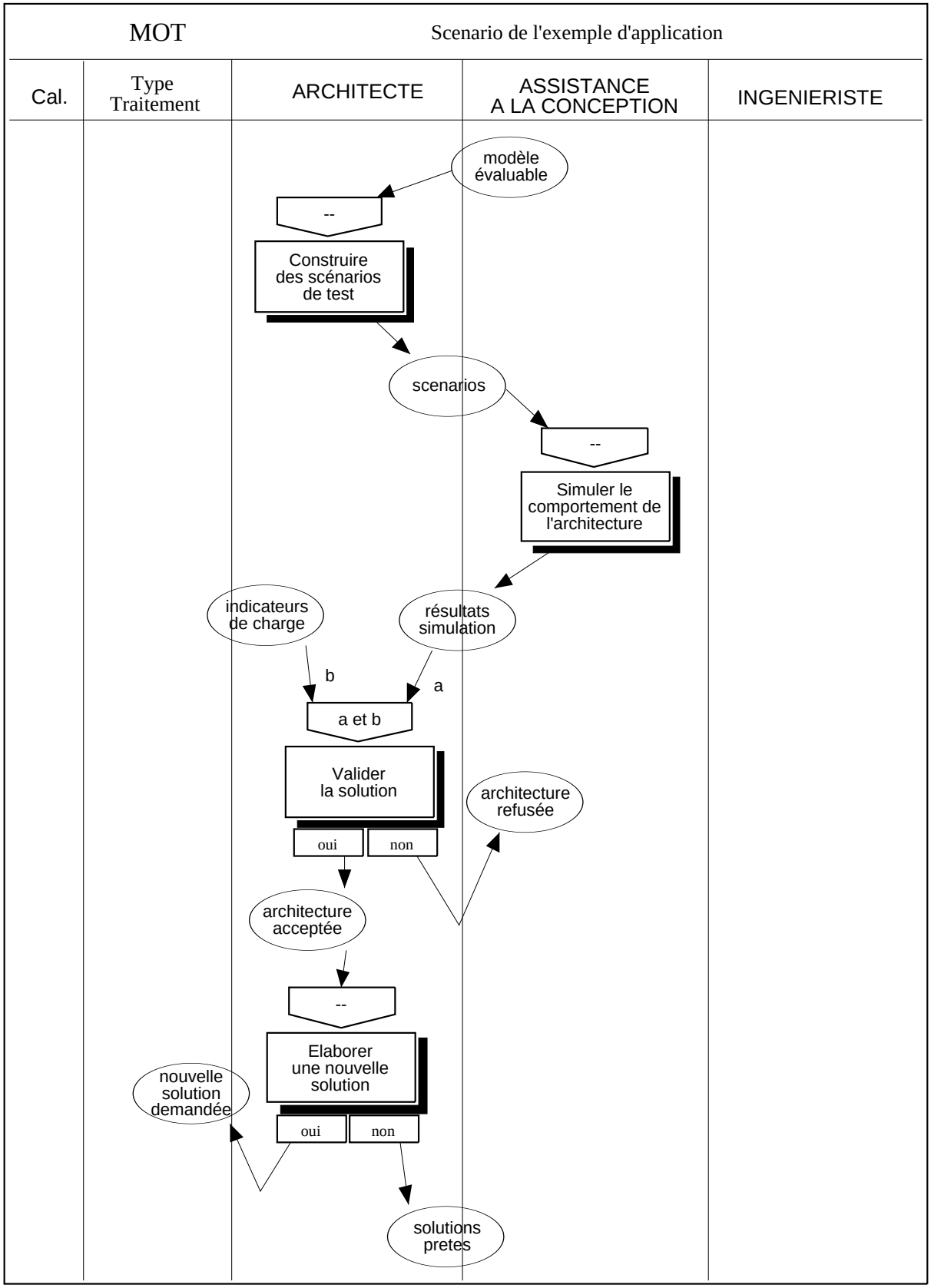
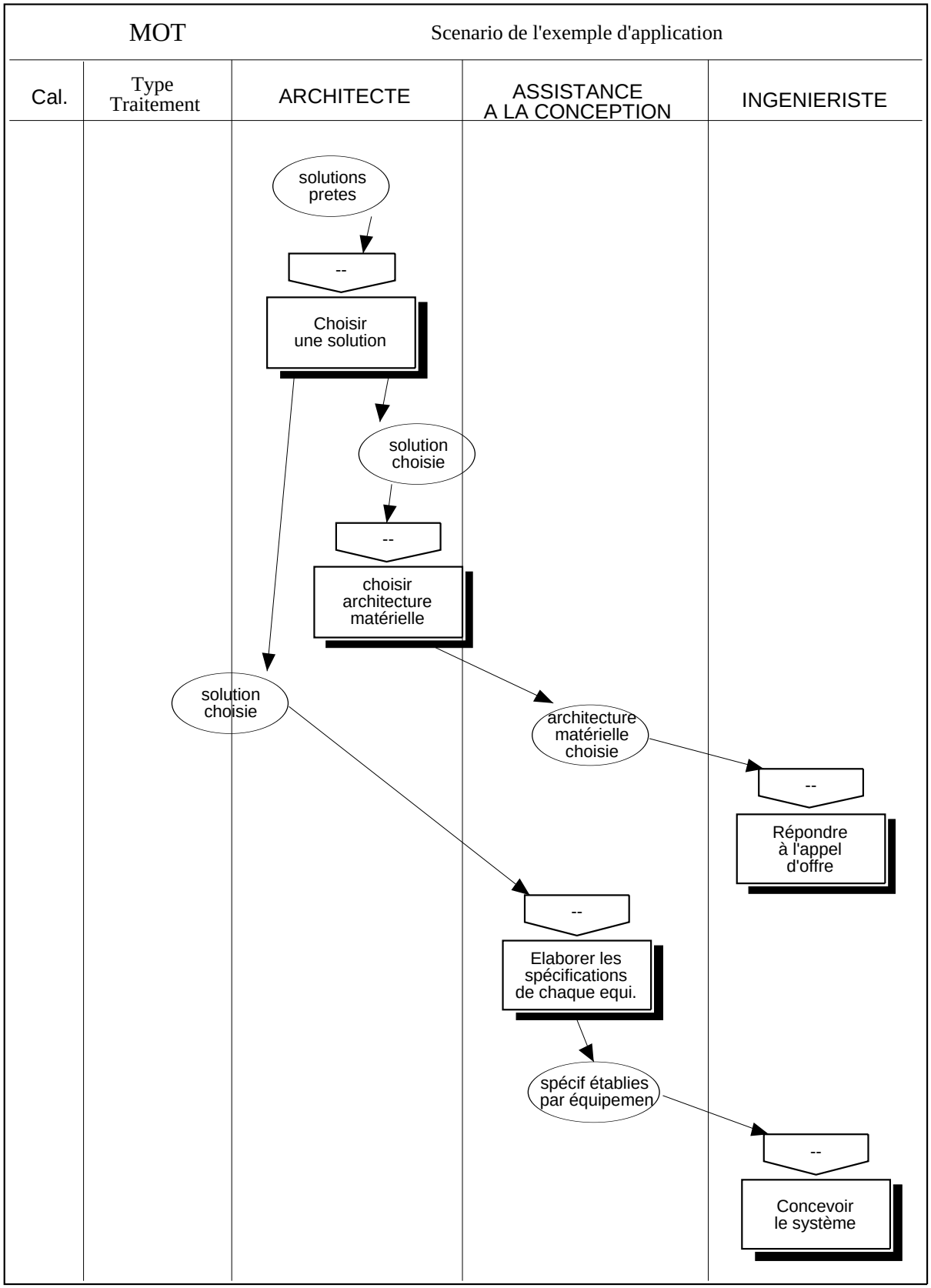


Tableau 14 : scénario de la conception d'architecture selon notre méthode



2.3.2 Construire l'ossature du modèle d'implantation (assistance)

Une ossature du modèle d'implantation est automatiquement élaborée à partir des spécifications SART (fig. 67). On notera que seuls les traitements terminaux du modèle SART sont présents dans ce modèle d'implantation. Bien que ne faisant pas partie de la syntaxe du modèle d'implantation, des ellipses ont été utilisées pour mettre en évidence les regroupements originels des traitements dans le modèle SART. On notera également que les flots du modèle SART comme «identification», qui regroupaient plusieurs flots élémentaires, ont disparu. Seuls les flots élémentaires sont présents sur le modèle d'implantation. Dans cette première phase d'élaboration du modèle d'implantation, l'assistance est donc totale.

2.3.3 Compléter le modèle d'implantation (architecte)

L'architecte doit compléter le modèle que lui fournit l'assistance en indiquant la nature des traitements (procéduraux, ou de contrôle-commande) et en renseignant les triplets $(C(t), D(t), S_c(t))$ associés aux traitements procéduraux, et les couples (m_d, m_f) associés aux traitements de contrôle-commande.

Exemple :
Le traitement 13 «Lire les comptes-rendus de production» est un traitement procédural qui doit être renseigné par l'architecte pour compléter sa description. En effet ce traitement est déclenché lorsque le message «produit fini» apparaît, alors il doit lire le stockage «contenu étiquette», en fin de traitement il écrira des données dans «info SID E» puis il émettra les message «dévier la palette» et «comptes-rendus de production».

Le tableau 15 est le résultat de la saisie, par l'architecte, des informations relatives à la cinématique du modèle d'implantation (elles apparaîssent à l'intérieur des cellules encadrées par un double trait). Dans la colonne «nature des traitements», P désigne un traitement procédural et CC un traitement de contrôle commande (les traitement de coordination n'ont pas de description cinématique).

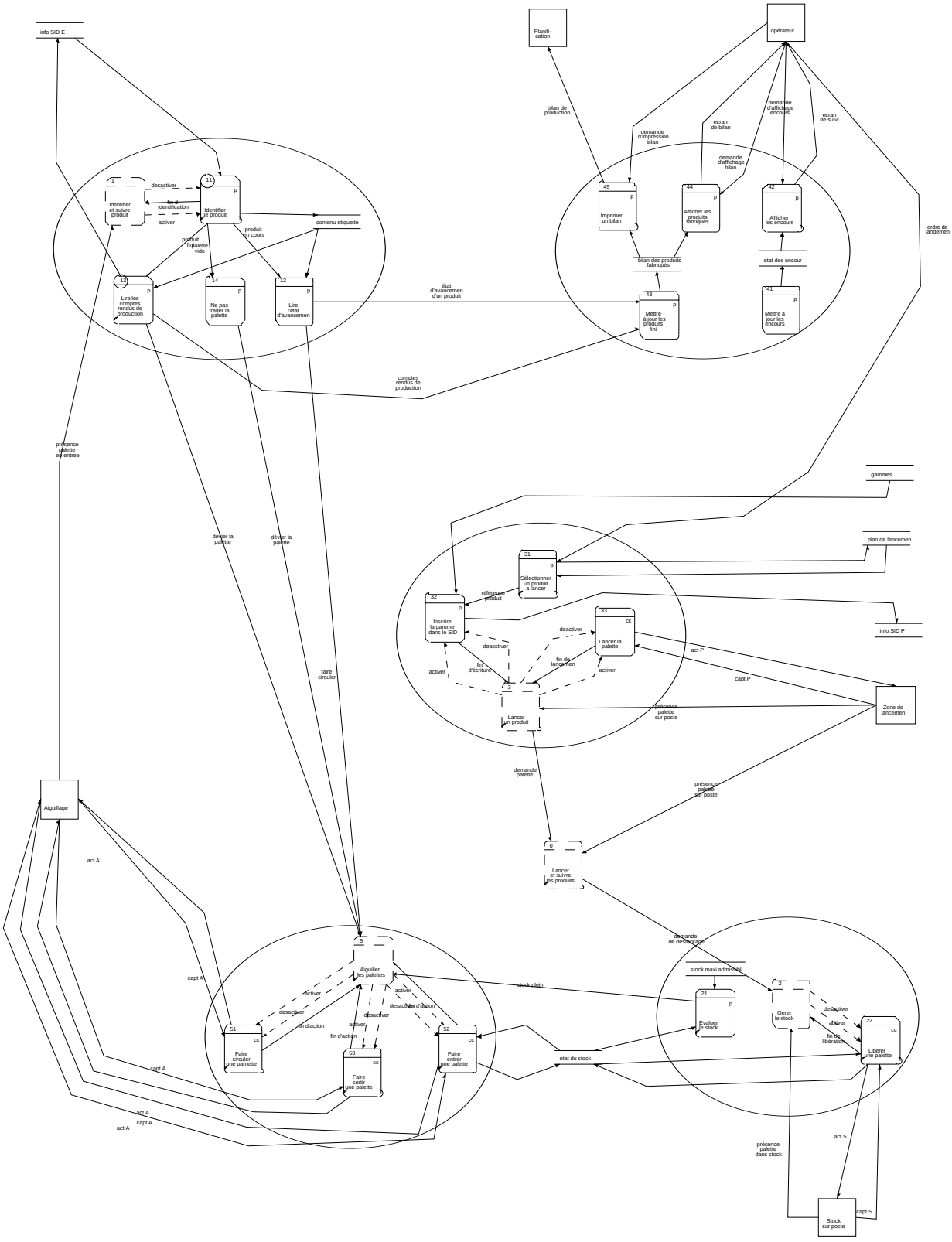


Figure 67 : modèle d'implantation de POSTEL

Tableau 15 : description de la cinématique du modèle d'implantation de POSTEL

Numéro	Traitement	Nature	Description
fourni par l'assistance		A la charge de l'architecte	
11	Identifier le produit	P	({}, {d1, d2, d3}, {s1}) d1 = ({contenu étiquette}, {produit en cours}) d2 = ({}, {palette vide}) d3 = ({contenu étiquette}, {produit fini}) s1 = info SID E
12	Lire l'état d'avancement	P	({c1}, {d1}, {s1}) c1 = {produit en cours} d1 = ({}, {comptes-rendus de production, dévier la palette}) s1 = contenu étiquette
13	Lire les comptes-rendus de production	P	({c1}, {d1}, {s1,s2}) c1 = {produit fini} d1 = ({info SID E}, {état d'avancement d'un produit, faire circuler}) s1 = contenu étiquette
13	Ne pas traiter la palette	P	({c1}, {d1}, {}) c1 = {palette vide} ; d1 = ({}, {dévier la palette})
21	Evaluer le stock	P	({}, {d1, d2}, {s1, s2}) c1 = {produit en cours} d1 = ({}, {stock plein}) ; d2 = ({}, {stock plein}) s1 = stock maxi admissible ; s2 = état du stock
22	Libérer une palette	CC	({}, {fin de libération})
31	Sélectionner un produit à lancer	P	({c1}, {d1}, {s1}) c1 = {ordre de lancement} d1 = ({plan de lancement}, {référence produit}) s1 = plan de lancement
32	Inscrire la gamme dans le SID	P	({c1}, {d1}, {s1}) c1 = {référence produit} d1 = ({info SID P}, {fin d'écriture}) s1 = gammes
33	Lancer la palette	CC	({}, {fin de lancement})
41	Mettre à jour les en-cours	P	({c1}, {d1}, {}) c1 = {état d'avancement d'un produit} d1 = ({état des en-cours}, {})

Tableau 15 : description de la cinématique du modèle d'implantation de POSTEL

Numéro	Traitement	Nature	Description
42	Afficher les en-cours	P	({c1}, {d1}, {s1}) c1 = {demande d'affichage en-cours} d1 = ({}, {écran de suivi}) s1 = état des en-cours
43	Mettre à jour les produits finis	P	({c1}, {d1}, {}) c1 = {comptes rendus de production} d1 = ({bilan des produits fabriqués}, {})
44	Afficher les produits fabriqués	P	({c1}, {d1}, {s1}) c1 = {demande d'affichage bilan} d1 = ({}, {écran de bilan}) s1 = bilan des produits fabriqués
45	Imprimer le bilan	P	({c1}, {d1}, {s1}) c1 = {demande d'impression bilan} d1 = ({}, {écran de bilan}) s1 = bilan des produits fabriqués
51	Faire circuler une palette	CC	({}, {fin d'action})
52	Faire entrer une palette	CC	({}, {fin d'action})
53	Faire sortir une palette	CC	({}, {fin d'action})

2.3.4 Elaborer un modèle organique (architecte)

Pour la conception du modèle organique, l'assistance ne propose naturellement rien, hormis un éditeur graphique proposant les différentes icônes des constituants de commande.

L'architecte doit poser une architecture a priori. Seule son expérience peut lui fournir les premières indications pour évaluer la pertinence de ces choix. La figure 68 présente une architecture posée par l'architecte. Les règles que s'est imposé l'architecte pour obtenir cette architecture sont les suivantes :

- utiliser en priorité un automate programmable industriel pour gérer la commande du processus,

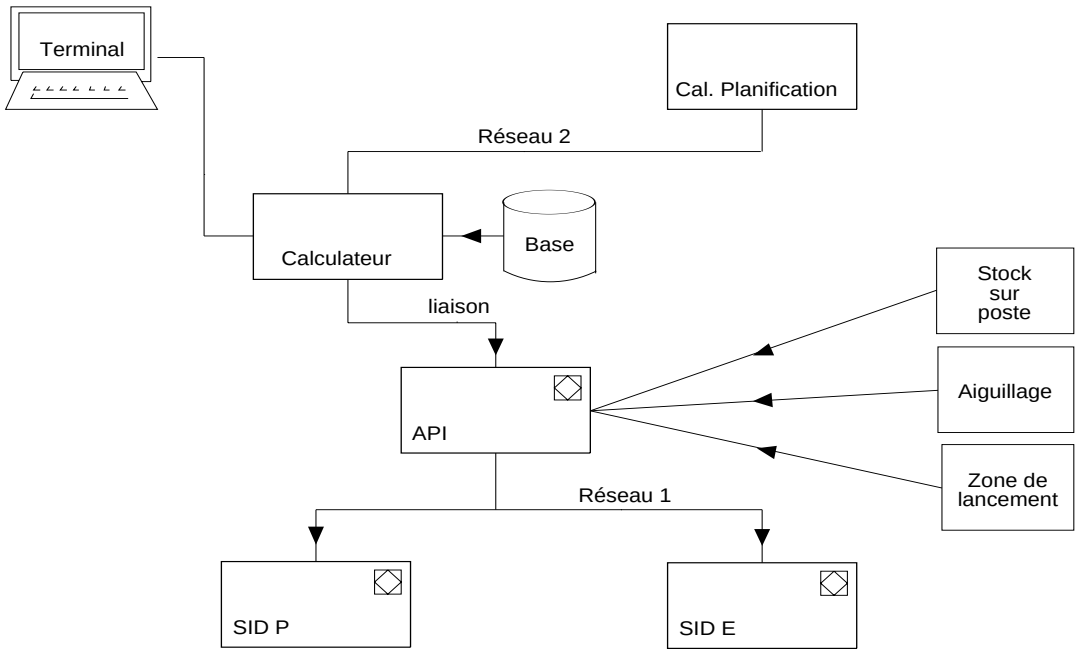


Figure 68 : modèle organique posé par l'architecte

- utiliser de préférence des media de communication maître-esclave avec les automates programmables industriels,
- employer un calculateur pour les fonctions de stockage et de dialogue de haut niveau avec un opérateur.

2.3.5 Affecter traitements et stockages (architecte)

Les outils d'assistance proposent la liste des traitements et des stockages à affecter, et la liste des équipements du modèle organique pouvant les accueillir.

L'architecte doit affecter chaque élément du modèle d'implantation dans le modèle organique, ce qui se traduit par le renseignement de la colonne «Equipement d'architecture» du tableau 16.

Tableau 16 : affectation des traitements et des stockages de POSTEL

Traitements et stockages		Equipement d'architecture
fourni par l'assistance		A la charge de l'architecte
0	Lancer et suivre les produit	API
1	Identifier et suivre produit	API
11	Identifier le produit	API

Tableau 16 : affectation des traitements et des stockages de POSTEL

Traitements et stockages		Equipement d'architecture
12	Lire l'état d'avancement	API
13	Lire les comptes-rendus de production	Calculateur
14	Ne pas traiter la palette	API
2	Gérer le stock	API
21	Evaluer le stock	API
22	Libérer une palette	API
3	Lancer un produit	API
31	Sélectionner un produit à lancer	Calculateur
32	Inscrire la gamme dans le SID	API
33	Lancer la palette	API
41	Mettre à jour les en-cours	Calculateur
42	Afficher les en-cours	Calculateur
43	Mettre à jour les produits finis	Calculateur
44	Afficher les produits fabriqués	Calculateur
45	Imprimer le bilan	Calculateur
5	Aiguiller les palettes	API
51	Faire circuler une palette	API
52	Faire entrer une palette	API
53	Faire sortir une palette	API
S	info SID E	SID E
S	info SID P	SID P
S	gammes	Base
S	plan de lancement	Base
S	contenu étiquette	API
S	bilan des produits fabriqués	Base
S	état des encours	Base
S	état du stock	API
S	stock maxi admissible	API
Ext	Planification	Cal Planification
Ext	Opérateur	Terminal

Tableau 16 : affectation des traitements et des stockages de POSTEL

Traitements et stockages		Equipement d'architecture
Ext	Zone de lancement	Zone de lancement
Ext	Stock sur poste	Stock sur poste
Ext	Aiguillage	Aiguillage

2.3.6 Rechercher des trajets pour les flots (assistance)

L'assistance propose le graphe associé au modèle organique (fig. 69), et recherche s'il existe un trajet possible dans le modèle organique pour chaque flot du modèle d'implantation. Pour notre exemple POSTEL, l'assistance trouve un trajet pour chaque flot, ce qui signifie que l'architecture n'a pas à être remise en cause à ce niveau de sa conception. L'assistance peut aborder directement la phase suivante.

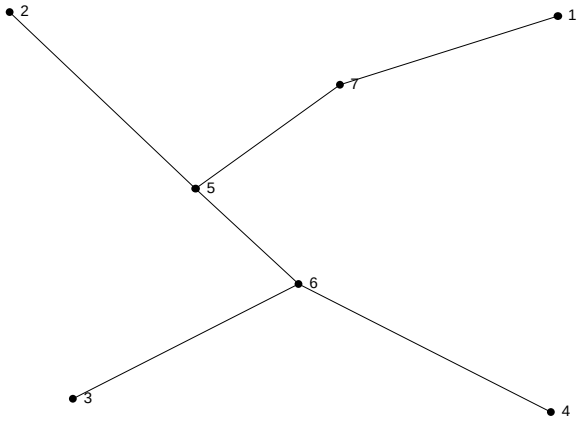


Figure 69 : graphe associé au modèle organique de POSTEL issu de la figure 68

2.3.7 Déterminer les traitements techniques (assistance)

L'assistance propose la détermination systématique des traitements techniques induits par l'affectation des éléments du modèle d'implantation dans le modèle organique. Examinons par exemple le cas du message «état d'avancement d'un produit» émis par le traitement 12 «Lire l'état d'avancement» à destination du traitement 43 «Mettre à jour le produit fini». Après avoir déterminé que la transmission du message se fait depuis un équipement esclave vers un équipement maître, l'assistance ajoute automatiquement deux traitements techniques et les flots nécessaires à la

transmission. La figure 70 présente le résultat pour ce flot, conformément au tableau 3 du chapitre 1.3 «Construction d'une solution d'architecture».

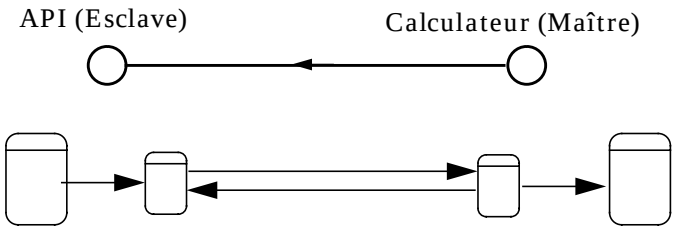


Figure 70 : exemple de traitement technique dû à un media maître-esclave

Dans l'exemple de POSTEL, un trajet unique pour chaque flot est identifié. La construction du modèle d'affectation se fait donc sans aucune intervention de l'architecte. Lorsque l'assistance a terminé son travail, le modèle d'affectation proposé est celui reproduit figure 71.

2.3.8 Renseigner le modèle d'affectation (architecte)

L'architecte doit renseigner le modèle d'affectation en estimant la taille des traitements et des flots. A cette fin, l'architecte doit utiliser les données qu'il possède, ou qu'il est à même d'estimer, relativement aux cadences de production du système dont il conçoit l'architecture de conduite. Pour POSTEL nous estimerons à 2 le nombre de palettes traitées par minute dans la zone d'aiguillage, et à 0,5 palette par minute la fréquence de lancement.

Concernant les traitements procéduraux, l'architecte doit renseigner la taille n_i, et la fréquence d'exécution F_T (voir tableau 17).

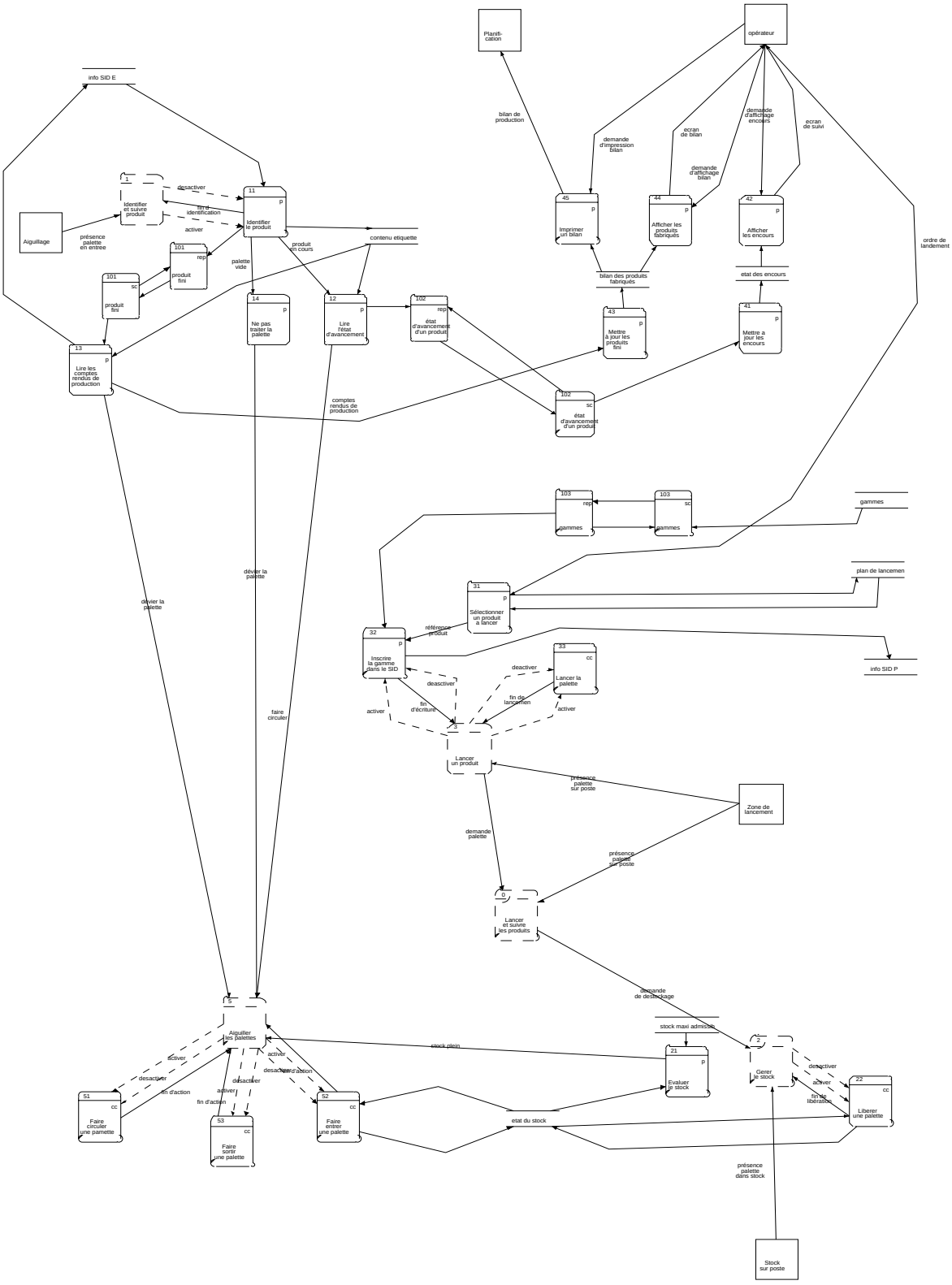


Figure 71 : modèle d'affectation de POSTEL

Tableau 17 : informations statistiques relatives aux traitements procéduraux de POSTEL

Traitements procéduraux		n_i kilo instructions	F_T min^{-1}
fourni par l'assistance		A la charge de l'architecte	
11	Identifier le produit	5	2
12	Lire l'état d'avancement	1	2
13	Lire les comptes rendus de production	1	0,5
14	Ne pas traiter la palette	0,1	0,5
21	Evaluer le stock	0,1	1
31	Sélectionner un produit à lancer	5	0,5
32	Inscrire la gamme dans le SID	2	0,5
41	Mettre à jour les en-cours	5	0,5
42	Afficher les en-cours	5	0,1
43	Mettre à jour les produits finis	5	0,5
44	Afficher les produits fabriqués	5	0,1
45	Imprimer le bilan	5	0,01

Concernant les traitements de contrôle-commande, l'architecte doit renseigner la taille n_{mot} (voir tableau 17).

Tableau 18 : informations statistiques relatives aux traitements de contrôle-commande de POSTEL

Traitements de contrôle-commande		n_{mot} kilo mots
fourni par l'assistance		A la charge de l'architecte
22	Libérer une palette	2
33	Lancer la palette	2
51	Faire circuler une palette	2
52	Faire entrer une palette	5
53	Faire sortir une palette	5

Concernant les flots qui circulent sur un media, l'architecte doit renseigner la taille n_i, et la fréquence d'exécution F_T (voir tableau 17).

Tableau 19 : informations statistiques relatives aux flots qui circulent sur des équipements de communication de POSTEL

Flots		T_d octet	F_d min^{-1}
fourni par l'assistance		A la charge de l'architecte	
sur «liaison»	dévier la palette	10	0,5
	faire circuler	10	2
	état d'avancement d'un produit	1.000	2
	Produit fini	1.000	0,5
sur «réseau»	Lire info SID E	1.000	2
	Ecrire info SID E	1.000	0,5
	Ecrire info SID P	1.000	0,5
entre «calculateur» et «base»	Lire gamme	1.000	0,5
	Lire plan de lancement	10.000	0,5
	Ecrire plan de lancement	1.000	0,5
sur «réseau 2»	bilan de production	10.000	0,005

Concernant les traitements techniques de scrutation, l'architecte doit renseigner la période de scrutation P_{sc} (voir tableau 17), avec une taille de message transmis par scrutation de $T_{sc} = 16 octets$.

Tableau 20 : informations statistiques relatives aux traitements techniques de POSTEL

Traitements de contrôle-commande		P_{sc} seconde
fourni par l'assistance		A la charge de l'architecte
101	produit fini	0,5
102	état d'avancement d'un produit	0,5
103	gammes	1

Concernant le calculateur, l'architecte doit fixer le nombre d'instructions exécutées lors d'un accès à un équipement de stockage. Pour POSTEL on fixe : $n(base) = 10000$.

2.3.9 Calculer les indicateurs (assistance)

L'assistance propose le calcul des indicateurs présentés dans la division 1.4.1 «Evaluation statistique». Les résultats fournis sont les suivants.

- $C(calculateur) = 0,024 \times 10^{-3} Mips$[1] avec, pour le «calculateur», 62% du temps CPU utilisé pour des traitements techniques (accès à l'équipement de stockage «base»),
- $T_{cycle}(API) = \frac{16}{V(API) - 0,22}$ avec T_{cycle} en seconde et V en Kmot/s,
- $C(reseau) = 400 bit/s$ avec 0% de charge technique,
- $C(liaison) = 1600 bit/s$ avec, pour l'équipement de communication «liaison», 79% de flots techniques (scrutation du maître).

2.3.10 Construire le modèle dynamique (assistance)

L'assistance propose le modèle du comportement dynamique de l'architecture de conduite. Ce modèle est obtenu par assemblage des modules génériques de comportement dynamique des éléments du modèle d'implantation, des éléments du modèle organique, et des éléments du modèle d'affectation. La figure 72 présente une vue graphique du modèle de comportement dynamique de l'architecture, bien qu'il n'ait pas pour finalité d'être lu, mais plutôt d'être évalué par simulation.

2.3.11 Construire des scénarios de test d'architecture (architecte)

L'architecte doit, en fonction de ses critères d'évaluation de l'architecture et des contraintes opérationnelles à vérifier, élaborer un échéancier d'événements décrivant un

1. Mips : Million d'instructions par seconde

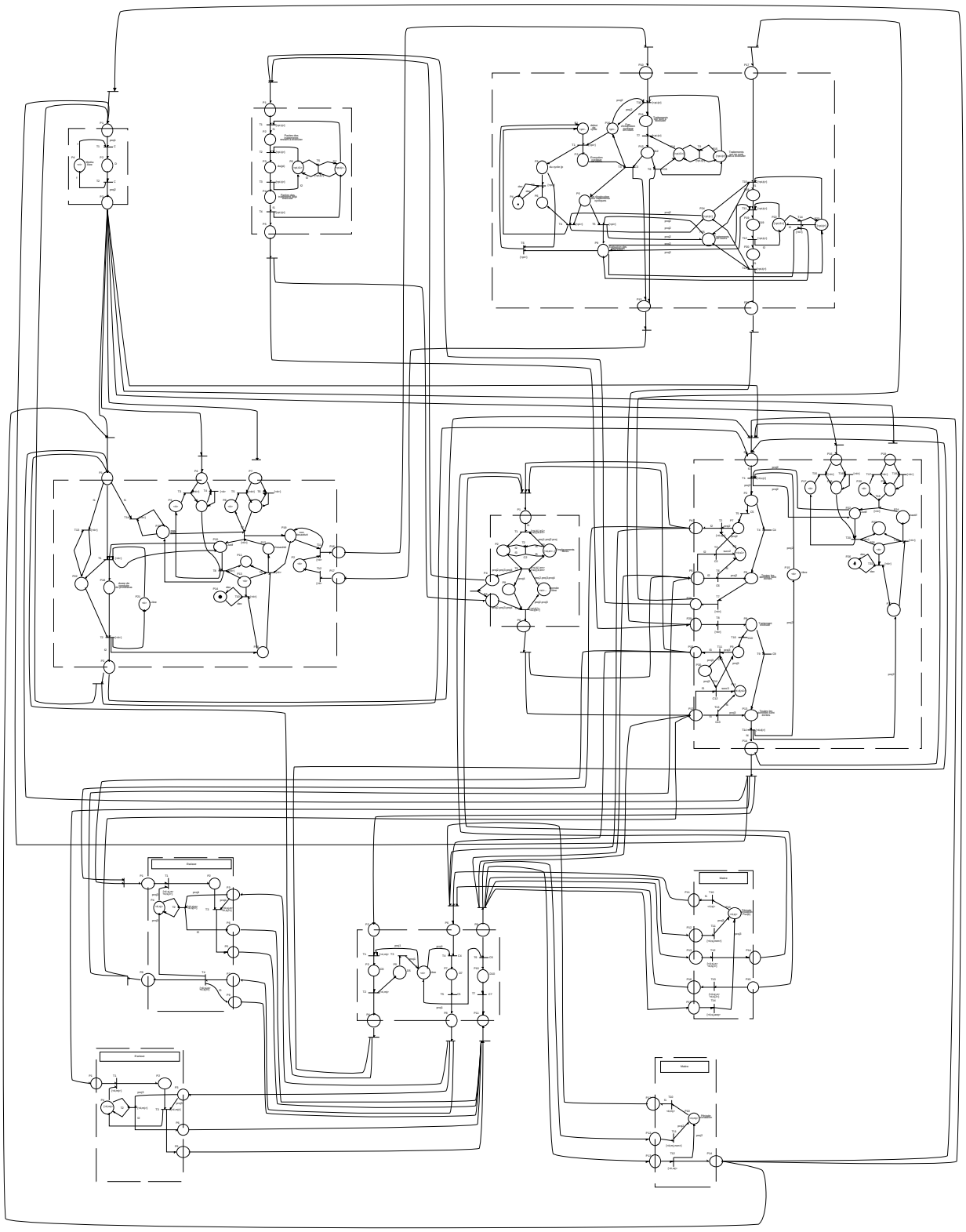


Figure 72 : vue du modèle de comportement dynamique de POSTEL

comportement de l'environnement du système de conduite, sous lequel on attend un certain comportement de l'architecture de conduite elle-même.

2.3.12 Simuler le comportement de l'architecture (assistance)

L'assistance assure l'évolution dans le temps du modèle de comportement dynamique de l'architecture de conduite à partir des sollicitations d'un scénario ayant la forme d'un échéancier. L'architecte obtient en retour une trace de l'évolution des états en fonction du temps, sur laquelle il vérifie les contraintes telles que les temps de réponse ou les taux de charge instantanés des constituants d'architecture.

2.3.13 Valider et choisir une solution (architecte)

L'architecte doit enfin valider la solution d'architecture qu'il vient d'élaborer. Il le fait à partir des indicateurs de charge des équipements de l'architecture organique, où il vérifie que les équipements type ne nécessitent pas des caractéristiques supérieures à ce que propose l'offre du marché des constituants. Il le fait également à partir des résultats de simulation, où la mise en évidence d'un temps de réponse à une sollicitation qui serait supérieur à ce que les contraintes opérationnelles imposent, conduirait l'architecte à remettre en cause sa solution.

L'architecte doit décider s'il souhaite élaborer une nouvelle solution d'architecture de conduite, soit parce que sa première solution n'a pas été validée, soit parce qu'il désire construire une nouvelle solution pour la confronter à la précédente et se donner plus de chance de trouver la «meilleure» solution.

L'architecte doit choisir une solution qu'il juge la meilleure parmi les solutions validées qu'il a élaborées. Il reste maître de ses critères de choix mais il s'appuie sur les résultats de la simulation du comportement dynamique de l'architecture de conduite, et sur les indicateurs statistiques de charge élaborés précédemment.

L'architecte doit alors choisir une architecture matérielle, c'est-à-dire choisir des équipements fournis par des constructeurs. Il doit être vigilant sur les débits des équipements de communication et les puissances CPU des équipements de traitement. La figure 73 représente une architecture matérielle adéquate pour POSTEL.

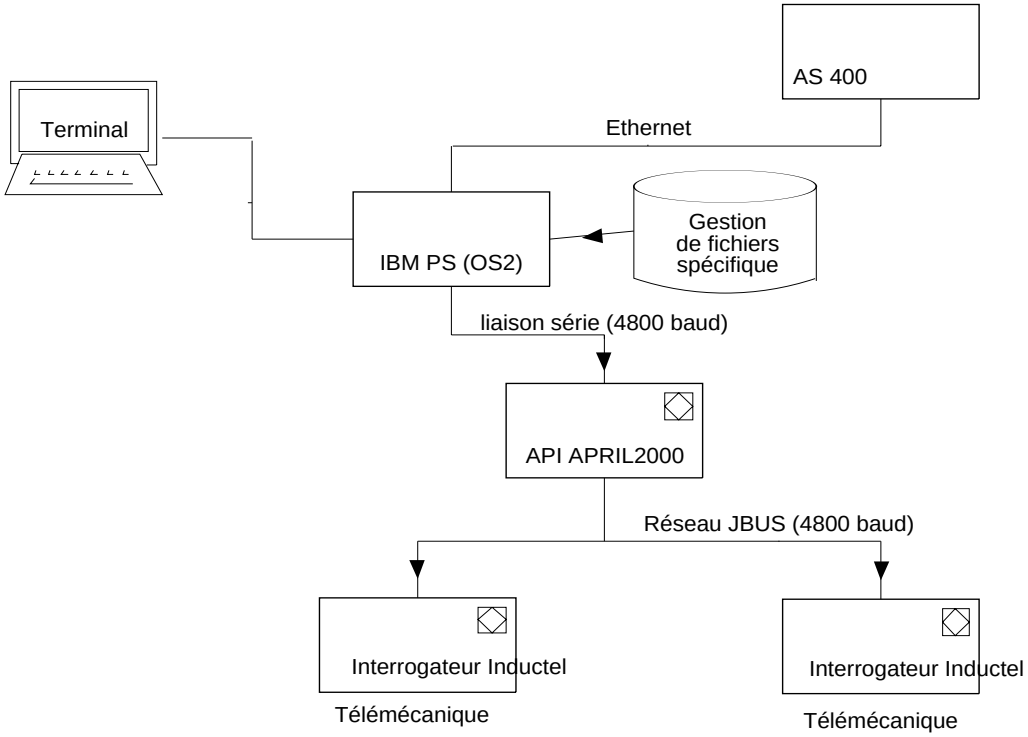


Figure 73 : architecture matérielle de POSTEL

2.3.14 Elaborer les spécifications de chaque sous-système (assistance)

L'assistance propose la constitution des spécifications de chaque constituant du modèle d'architecture matérielle. Dans un premier temps sous la forme d'une projection du modèle d'affectation au travers de chaque constituant matériel. La figure 74 présente le résultat de la projection du modèle d'affectation sur le calculateur du modèle organique de POSTEL. Il devient alors possible de construire le modèle SART correspondant, grace à la formalisation de l'intégration décrite dans la division 1.4.4.

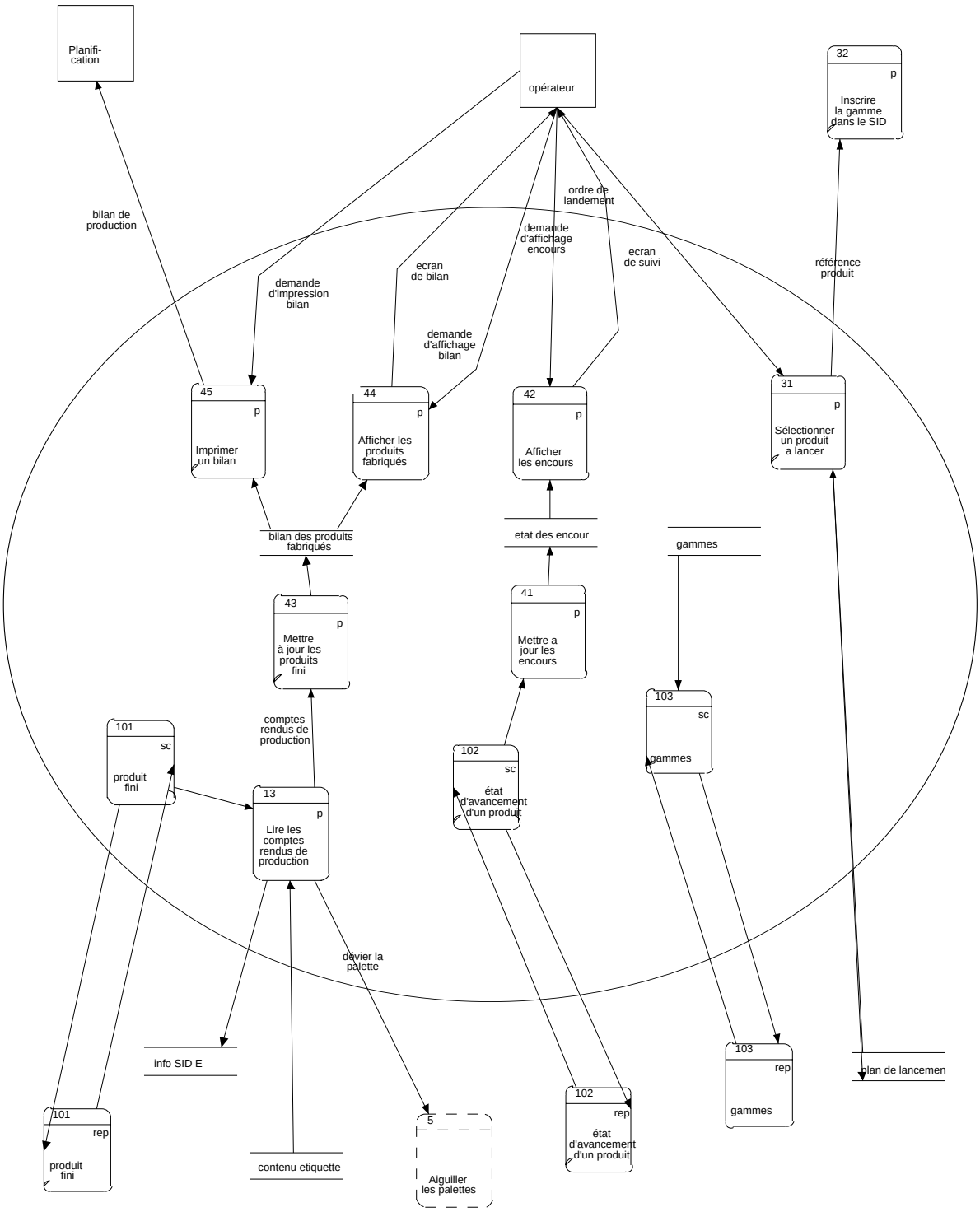


Figure 74 : sous-ensemble du modèle d'affectation de POSTEL relatif au calculateur

2.4 Conclusion sur la deuxième partie

Dans cette deuxième partie, nous avons traité un exemple de conception et d'évaluation d'une architecture de conduite. La méthode de conception que nous proposons a ainsi pu être illustrée avec le point de vue «conduite de projet». La chronologie des activités de la conception d'architecture a donc été mise en évidence. Nous avons ainsi pu faire la part du travail de conception dévolu à l'architecte dans laquelle des choix doivent être faits, et la part du travail de construction de modèles que nous avons pu systématiser grâce à la formalisation de notre approche. L'ampleur des modèles construits dans cet exemple montre à quel point l'automatisation de cette assistance à l'architecte est indispensable. En effet, si l'élaboration d'un modèle organique et l'affectation des traitements dans des constituants d'architecture reste du ressort de l'architecte, l'élaboration du modèle d'affectation avec la détermination des traitements techniques et la construction du modèle évaluable (dynamique du comportement de l'architecture) sont trop laborieuses pour être faites manuellement. Cette deuxième partie permet donc de conclure que la résolution, ou du moins la réduction, du paradoxe de l'architecte passe par l'automatisation de l'assistance que nous avons proposée à l'architecte.

Conclusions

La conception de l'architecture de conduite des systèmes de production, mettant en jeu à la fois une partie logicielle et une partie matérielle, était un terrain quasiment inexploré sur le plan scientifique. Nos travaux ne pouvant se focaliser sur un de ses aspects précis et scientifiquement identifiés, ils ont au contraire abordé *l'ensemble du processus de conception et d'évaluation d'architectures*. Nous avons pour cela développé une méthode de conception, basée sur la manipulation de modèles formels et indépendante des méthodes de conception des systèmes dans lesquelles elle peut être intégrée.

Notre contribution à la conception d'architecture a donc atteint les objectifs énoncés initialement en rendant plus méthodique, plus rigoureuse, et plus formelle la conception d'architecture. Il n'en reste pas moins qu'il s'agit là de travaux initiaux pour lesquels de nombreuses perspectives s'ouvrent à nous.

La première va dans le sens de l'amélioration de la praticabilité de la méthode en l'appliquant à des cas industriels de grande ampleur. Pour ce faire, et compte tenu de la complexité des modèles élaborés, l'assistance que nous avons développée devra être encore plus automatisée afin que l'ergonomie fournie à l'architecte soit plus importante. Seule une amélioration de la *praticabilité* de la méthode peut être garante de sa pénétration dans le monde industriel.

Une deuxième perspective peut être dégagée au niveau de l'évaluation de la sûreté de fonctionnement des architectures de conduite. A partir de notre formalisation de l'architecture, une étude de la robustesse aux défaillances et de la disponibilité peut en effet être développée, en prenant en compte à la fois les aspects matériels et logiciels d'une architecture. Cela permettrait de répondre aux besoins des ingénieristes, souvent exprimés en terme d'évaluation de la sûreté de fonctionnement et de la gestion des modes dégradés des installations industrielles.

Références bibliographiques

Partie 1

Chapitre 1.1 Le problème et son positionnement

[1] VERLINDE (Christian). — Contribution à l'étude des architectures de systèmes automatisés. — Thèse de doctorat de l'Institut National Polytechnique de Lorraine, 1989

[2] PEYRUCAT (Jean-François). — Automatisation ; si on pouvait gagner à tous les coûts… *Mesures*. — n°658, octobre 1993, pp. 30-34

[3] ROSNAY (Joël de). — Le macroscope, Vers une vision globale. — Éditions du Seuil, 1975. — 18 cm, 351 p. (Points)

[4] DURAND (Daniel). — La systémique, 5ème éd. — Presses universitaires de France, 1992 [1ère éd. 1979]. — 17 cm, 126 p. (Que sais-je ?)

[5] LE MOIGNE (Jean-Louis). — La modélisation des systèmes complexes. — Paris, Dunod, 1990. — 22 cm, 330 p. (Afcet-système)

[6] MÉLÈSE (Jacques). — L'analyse modulaire des systèmes de gestion, une méthode efficace pour appliquer la théorie des systèmes au management. — Puteaux, Edition Hommes et techniques, 1972. — 235 p.

[7] RICHARD (Jacques), BAJIC (Eddy) et VÉRON (Michel). — Système d'information d'une cellule d'usinage auto-contrôlée. *Ingénierie des systèmes d'information*. — Vol 1, n°3, 1993, pp. 373-392

[8] JAULENT (Patrick). — Génie logiciel les méthodes, SADT, SA, E-A, SA-RT, SREM, SYS-P-O, OOD, HOOD... — Armand Colin, 1990. — 23 cm, 288 p.

[9] Centre Coopératif de Génie Automatique. — Fascicule «Carte des activités de développement d'un système automatisé de production». — CCGA, Saint-Ouen

[10] PANETTO (Hervé). — Une contribution au Génie Automatique : le Prototypage des Machines et des Systèmes Automatisés de Production. — Thèse de doctorat de l'Université de Nancy I, 1991

[11] LHOSTE (P.), PANETTO (H.), MOREL (G.) et ROESCH (M.). — Une Méthode de Spécification et de Conception de la Partie commande des Machines et Systèmes Automatisés de Production Manufacturière. *Actes de la conférence GAMI Génie Automatique et Production Industrielle*. — Saint-Ouen, 12-13 Mars 1991

[12] ZAYTOON (J.), NIEL (E.), A. MILLE (A.) et JUTARD (A.). — A temporal SADT for automated manufacturing systems. *Proceeding of International conference on industrial engineering and production management*. — Mons, 2-4 June 1993, pp. 154-164

[13] ZAYTOON (Janan). — Extension de l'analyse fonctionnelle à l'étude de la sécurité opérationnelle des systèmes automatisés de production. — Thèse de doctorat de l'Institut National des Sciences Appliquées de Lyon, 1993

[14] LONCHAMP (Jacques), GODART (Claude) et DERNIAME (Jean-Claude). — Les environnements intégrés de production logiciel. *Technique et science informatiques*. — Vol. 11, n°1, 1992, pp. 31-95

[15] MINOT (R.) et SOUFFLET (D.). — PCTE, une plateforme d'intégration normalisée, Génie Logiciel & Système Expert. — n°22, mars 1991, pp. 10-14

[16] Association Française de Normalisation. — PR Z 68-901, Génie Automatique, Représentation des systèmes de contrôle et de commande des systèmes automatisés, Contribution à l'intégration des outils XAO du Génie Automatique : modèle conceptuel Base-PTA. — Paris la défense, 1992

[17] PANETTO (Hervé). — Une contribution au Génie Automatique : le Prototypage des Machines et Systèmes Automatisés de Production. — Thèse de doctorat de l'Université de Nancy I, 1991

[18] TIXADOR (J.M.). — Une contribution au génie automatique : la SPécification EXécutable des Machines et Systèmes Automatisés de Production. — Thèse de doctorat de l'Université de Nancy I, 1989

[19] BARAKAT (O.). — Contribution à la modélisation et à la simulation orientée objet des systèmes flexibles de production — Thèse de doctorat de l'Université de Franche-Comté, 1991

[20] BOURRIÈRES (J.P.). — Contribution à la modélisation intégrée des systèmes robotisés d'assemblage.— Thèse d'état, Besançon, 1990

[21] LHOTE (François). — Approche dis-continue du génie automatique dans le domaine des systèmes de production d'objets composés. *Actes de ADPM'92*. — Paris, 29-30 Janvier 1992, pp. 121-129

[22] DESCOTES-GENON (B.) et LADET (P.). — Modélisation, simulation et commande de systèmes de production à l'aide des réseaux de Petri colorés, application à un atelier d'assemblage. *Revue d'automatique et de productique appliquées*. — Vol. 3, n°1, 1990, pp. 97-113

[23] SAYAT (Belkacem) et LADET (Pierre). — Spécification de la commande coordonnée d'un système de production. Utilisation conjointe du grafcet et des réseaux de Petri. *Actes de GRAFCET'92*. — Paris, 25-26 Mars 1992, pp. 169-77

[24] BOUREY (J.P.) et GENTINA (J.C.). — Conception structurée et modulaire d'architectures de contrôle réparti en production flexible manufacturière. *APII*. — Vol. 25, n°4, 1991, pp. 349-376

[25] CRUETTE (D.). — Méthodologie de conception des systèmes complexes à événements discrets : application à la conception et la validation hiérarchisée de la commande de cellules flexibles de production dans l'industrie manufacturière. Thèse de doctorat de l'Université des Sciences et Technologies de Lille, 1991

[26] HEBRARD (A.), BOUREY (J.P.) et GENTINA (J.C.). — Concept d'interface intelligent et structuration de la commande de systèmes de production flexibles de type manufacturier. *Proceedings of the first European Control Conference*. — Grenoble, 2-5 Juillet 1991, pp. 2123-2127

[27] AMAR (S.), CRAYE (E.) et GENTINA (J.C.). — Une méthode hiérarchique de spécification et de prototypage des systèmes de production flexible. *APII*. — Vol. 26, n°5-6, 1992, pp. 483-514

[28] DEBOUCHÉ (Jean-Luc). — Contribution à la conception des systèmes de contrôle-commande. — Thèse de doctorat de l'Université des Sciences et Technologies de Lille, 1993.

[29] PRUNET (F.), STURLESE (J.L.), CASALOT (C.), GINESTE (E.), PANAGET (D.), DECHENAUX (G.) et LLORCA (P.). — Méthodologie et implantation automatique de commande d'automatisme à l'aide de la chaîne PIASTRE. *APII*. — Vol. 21, n°4, 1987, pp. 299-321

[30] GIACCONE (Thierry). — Etude, simulation fonctionnelle et implantation d'un logiciel de commande d'un processus en productique à l'aide d'un outil de CAO : PIASTRE. — Rapport de DEA, USTL, Montpellier, 1989

[31] GIACCONE (Thierry). — Modèle structuré de spécification, de conception et de mise au point de systèmes à événements discrets. — Thèse de doctorat de l'Université de Montpellier II, 1991

[32] CHAPURLAT (Vincent), GIACCONE (Thierry), MONNERET (Gérald), PEREYROL (Frédéric), PRUNET (François), et SIMOTTEL (Dan). — A structured model for specification of discrete events systems: the ACSY model. *APII*. — Vol. 27, n°1, 1993, pp. 65-80

[33] LESAGE (Jean-Jacques) et KIEFER (François). — A methodological integration between functional and data analysis. *Actes du 23ème CIRP séminaire international sur les systèmes de production*. — Nancy, 6-7 Juin 1991

[34] LESAGE (Jean-Jacques) et ROUSSEL (Jean-Marc). — Hierarchical approach to grafcet using forcing order. *APII*. — Vol. 27, n°1, 1993, pp. 115-122

[35] DENIS (Bruno), LESAGE (Jean Jacques) et TIMON (Guy). — Toward a theory of integrated modelling. *Proceedings of IFAC workshop on CIM in Process and manufacturing industries*. — Espoo (Finland), 23-25 Novembre 1992

[36] WARD (Paul T.) et MELLOR (Stephen J.). — Structured development for real-time systems (3 Vol.)— Englewood Cliffs, Prentice-Hall, 1986. — 25 cm, 503 p. (Yourdon press computing series)

[37] HATLEY (Derek J.) et PIRBHAI (Imtiaz A.). — Stratégies de spécification des systèmes temps réel (SA-RT), trad. de l'américain [Strategies for real-time system specification]. — Paris, Masson, 1991. — 24 cm, 345 p. (Méthodes Informatiques et Pratique des Systèmes)

[38] CALVEZ (Jean-Paul). — Spécification et conception des systèmes, une méthodologie. — Paris, Masson, 1990. — 24 cm, 614 p. (Manuels informatiques Masson)

[39] CALVEZ (Jean-Paul). — Spécification et conception des systèmes, études de cas. — Paris, Masson, 1990. — 24 cm, 276 p. (Manuels informatiques Masson)

[40] DELCUVELLERIE (Jean-Luc). — Connaissances en conception des architectures de systèmes informatisés d'automatisation et outil d'analyse de la robustesse aux défaillances d'architectures réparties. — Thèse de doctorat de l'Institut National Polytechnique de Lorraine, 1989

[41] DE MARCO (T.). — Structured analysis and system specification. — Englewood Cliffs, Prentice Hall, 1978 (Yourdon press computing series)

[42] Association Française de Normalisation. — X 50-150, Analyse de la valeur, Analyse fonctionnelle, Vocabulaire. — Paris la défense, 1990

Chapitre 1.2 Présentation générale de la méthode

[43] HIDDE (Axel R.) et ZÖLLNER (Bernd). — Computer-aided manual workstation as an extension of the product-independent flexible factory automation concept. *Computer in industry*. — Vol. 16, n°3, 1991, pp. 225-237

[44] HIDDE (Axel R.). — Management of information in a system of heterogeneous distributed data bases using the example of a PCB assemblage. *International journal of computer integrated manufacturing*. — Vol. 4, n°6, 1991, pp. 323-330

Chapitre 1.3 Construction d'une solution d'architecture

[45] TARDIEU (Hubert), ROCHFELD (Arnorld) et COLLETI (René). — La méthode merise, Tome 1, Principes et outils. — Paris, Les éditions d'organisation, 1986. — 328 p.

[46] BRUYN (William), JENSEN (Randall), KESKAR (Dinesh) et WARD (Paul). — ESML: an extended systems modeling language based on the data flow diagram. *ACM software engineering notes*. — Vol. 13, n°1, 1988, pp. 1-19

[47] FRANCE (Robert B.). — Semantically extended data flow diagrams: a formal specification tool. *IEEE transaction on software engineering*. — Vol. 18, n°4, 1992, pp. 329-346

[48] WARD (Paul T.). — The transformation schema: an extension of the data flow diagram to represent control and timing. *IEEE transaction on software engineering*. — Vol. 12, n°2, 1986, pp. 198-210

[49] BRODE (John). — Rules for transforming IDEF diagrams to coloured Petri nets. — Cambridge (Massachusetts), Meta Software Corporation, 1988. — 3 p.

[50] DENIS (Bruno), LESAGE (Jean Jacques) et TIMON (Guy). — Toward a theory of integrated modelling. *Revue Sciences et techniques de la conception*. — Vol. 2, n°1, 1993, pp. 87-96

[51] LESAGE (Jean-Jacques), DENIS (Bruno) et TIMON (Guy). — Une approche formelle de la modélisation. *Actes du symposium international «La conception en 2000 et au-delà, outils et technologie»*. — Strasbourg, 24-27 Novembre 1992

Chapitre 1.4 Evaluation et choix d'architecture

[52] Association des Exploitants d'Équipements de Mesure de Régulation et d'Automatisme. — Système automatisés de production, Guide d'analyse des besoins (aide à la conception d'architecture). — Verneuil en Halatte, EXERA, 1990. — 30 cm

[53] BOUREY (J.P.) et GENTINA (J.C.). — Structuration de la partie procédurale du système de commande de cellules flexibles de production. *Actes du Congrès Automatique 1988*. — Grenoble, 10-12 Octobre 1988, pp. 233-242

[54] MOITESSIER (F.), AUBRY (J.F.), DERNIAME (J.C.) et ZANNE (C.). — Une méthode de conception des systèmes de commande pour des processus hybrides rapides. *Actes de ADPM'92*. — Paris, 29-30 Janvier 1992, pp. 3-9

[55] ZAYTOON (Janan). — Extension de l'analyse fonctionnelle à l'étude de la sécurité opérationnelle des systèmes automatisés de production. — Thèse de doctorat de l'Institut National des Sciences Appliquées de Lyon, 1993

[56] AJMONE (M.) et CHIOLA (G.). — Construction of generalized stochastic Petri net models of bus oriented multiprocessor systems by stepwise refinements: a case study. *Proceedings of the International conference modelling techniques and tools for performance analysis.* — Sophia Antipolis, 5-7 Juin 1985, pp. 265-278

[57] DI STEPHANO (Antonella), MIRABELLA (Orazio) et ZAPPALÀ (Carlo). — Featuring FDDI in a process control environment. *Computer in industry.* — Vol. 21, n°1, 1993, pp. 35-49

[58] NATARAJAN (K.S.). — Performance analysis of prefetch strategies for database access. *Proceedings of the international conference modelling techniques and tools for performance analysis.* — Sophia Antipolis, 5-7 Juin 1985, pp. 213-229

[59] RIAT (Jean-Christophe). — Validation par simulation d'architecture de commande d'atelier dans l'industrie de production manufacturière. Application aux systèmes automatisés de production d'un grand constructeur automobile. — Thèse de doctorat de l'Université des Sciences et Technologies de Lille, 1992.

[60] GRESSIER (M.E.). — La modélisation au moyen des réseaux de Petri Stochastiques : application aux systèmes informatiques fortement synchrones. — Thèse de doctorat du Conservatoire National des Arts et Métiers, 1987

[61] LEPOLD (Roland). — Performability evaluation of degradable computer systems based on stochastic Petri nets. — Thèse de l'institut de recherche polytechnique de l'Université de Haute Alsace, 1992

[62] MERCIER DES ROCHETTES (R.), DESCOTES-GENON (B.) et LADET (P.). — De la spécification à la commande d'un atelier d'assemblage : utilisation des réseaux de Petri colorés. *Actes du Congrès Automatique 1988.* — Grenoble, 10-12 Octobre 1988, pp. 243-252

[63] DAVID (René) et ALLA (Hassane). — Du Grafcet aux réseaux de Petri. — Paris, Hermès, 1989. — 23 cm, 424 p. (Traité des nouvelles technologies, série automatique).

[64] BESOMBES (Béatrix). — Un système d'aide à la conduite d'atelier flexible basé réseaux de Petri colorés. — Thèse de doctorat de l'Université Lyon I, 1990

[65] FITRZYK (Dominique). — Atelier logiciel pour automaticien : contribution à la réalisation d'un outil d'analyse et de simulation de réseaux de Petri. — Mémoire d'ingénieur CNAM, 1993

[66] TANKOANO (J.), BOUDEBOUS (D.) et DERNIAME (J.C.). — PETRI-S : un simulateur de systèmes de production automatisés décrits à l'aide de réseaux de Petri interprétés colorés. *APII*. — Vol. 25, n° 1, 1991, pp. 1-30

[67] VALETTE (R.), THOMAS (V.) et BACHMANN (S.). — SEDRIC : un simulateur à événements discrets basé sur les réseaux de Petri. *APII*. — Vol. 19, 1985, pp. 423-436

Annexe A

Les activités de la conception de commande des systèmes de production

AUTEUR : B. Denis

DATE : 26 Novembre 1993

LECTEURS : J.J. Lesage

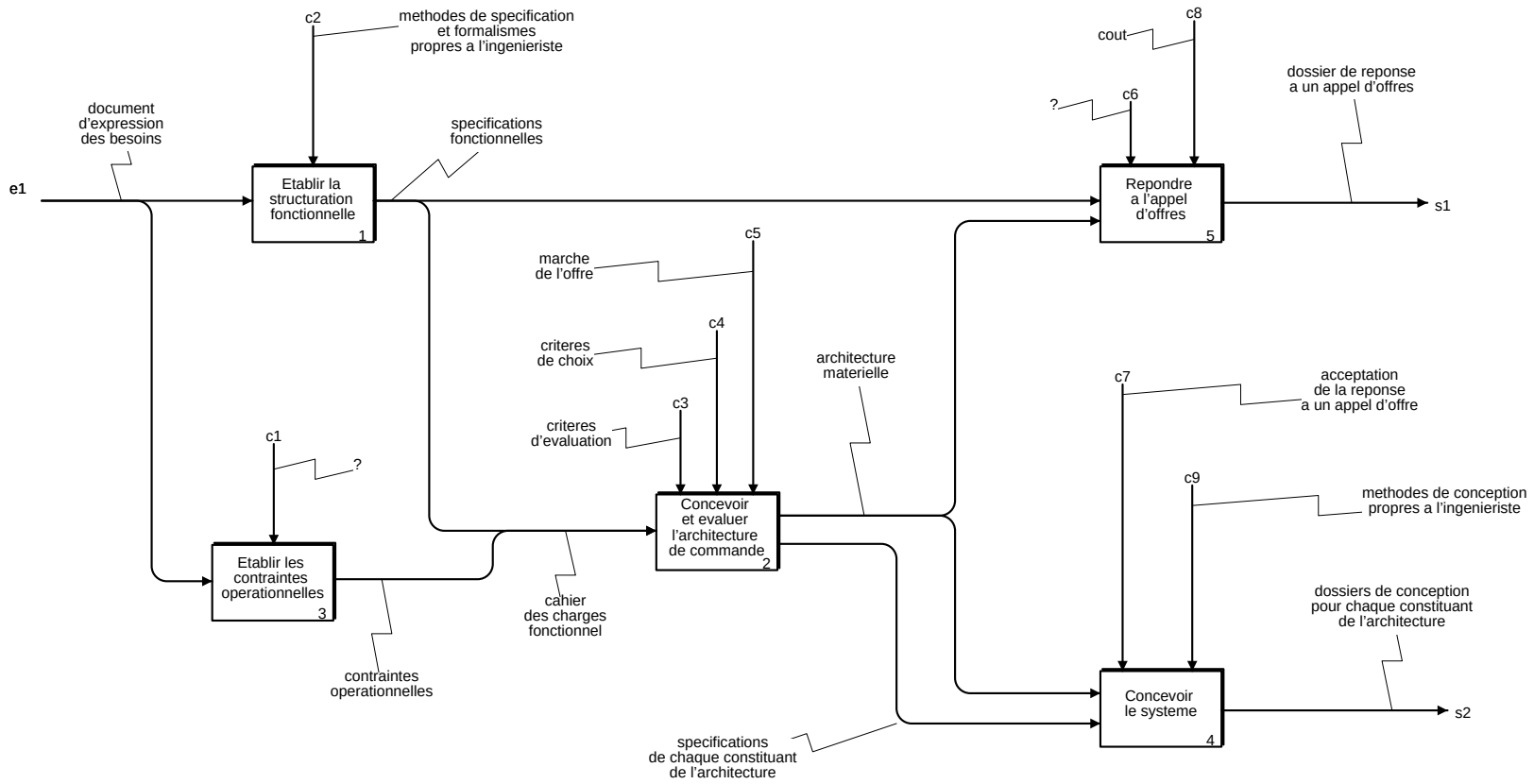
MODELE : Conception et evaluation d architecture

Point de vue : concepteur d'une demarche
de conception et d'evaluation d'architecture
de conduite des systemes complexes.

Objectif : mettre en evidence la place de la conception
et de l'evaluation d'architecture dans le travail
de l'ingenieriste, ainsi que les differentes activites
qui y prennent place.

VERSION V1.3

AUTEUR : B. Denis MODELE : Conception et evaluation d architecture DATE : 26 Novembre 1993	LECTEURS : J.J. Lesage	CONTEXTE : NEANT



A-1	concevoir une architecture de conduite dans un environnement d'appel d'offres	VERSION V1.3

AUTEUR : B. Denis MODELE : Conception et evaluation d architecture DATE : 26 Novembre 1993	LECTEURS : J.J. Lesage	CONTEXTE : NEANT

GLOSSAIRE (A-1)

ACTIVITE DU DIAGRAMME (As0) : Etablir la structuration fonctionnelle

document d'expression des besoins
Document elabore en phase d'analyse des besoins et a partir du cahier des charges de consultation fourni lors d'un appel d'offre

methodes de specification et formalismes propres a l'ingenieriste

specifications fonctionnelles
Description des fonctions attendues du systeme de conduite du systeme de production, independamment des moyens utilises pour les assurer

ACTIVITE DU DIAGRAMME (A-0) : Concevoir et evaluer l'architecture de commande

cahier des charges fonctionnel
Document qui regroupe les specifications fonctionnelles (resultat de l'analyse fonctionnelle) et les contraintes operationnelles du systeme a concevoir.

criteres d'evaluation

criteres de choix
Criteres de selection des architectures de conduite, avec lesquels l'architecte decide quelle est la "bonne" architecture de conduite

marche de l'offre
Liste des constituants d'architecture disponibles sur le marche, ainsi que leurs principales caracteristiques

architecture materielle
Description organique des constituants d'architecture et de leur interconnexion. Dans l'architecture materielle, les constituants d'architecture sont decrits en terme de refe reference a un materiel fourni par les constructeurs du marche de l'offre

specifications de chaque constituant de l'architecture
Description des fonctions attendues de chaque constituant d'architecture, independamment des moyens utilises dans le constituant pour les realiser.

AUTEUR : B. Denis MODELE : Conception et evaluation d architecture DATE : 26 Novembre 1993	LECTEURS : J.J. Lesage	CONTEXTE : NEANT

ACTIVITE DU DIAGRAMME (As0) : Etablir les contraintes operationnelles

?

contraintes operationnelles
Ensemble de contraintes ne relevant pas des besoins fonctionnels mais qui sont fixees des le debut du projet, soit pour s'adapter a un environnement existant soit pour im imposer une partie de la solution

ACTIVITE DU DIAGRAMME (As0) : Concevoir le systeme

acceptation de la reponse a un appel d'offre

methodes de conception propres a l'ingenieriste

dossiers de conception pour chaque constituant de l'architecture

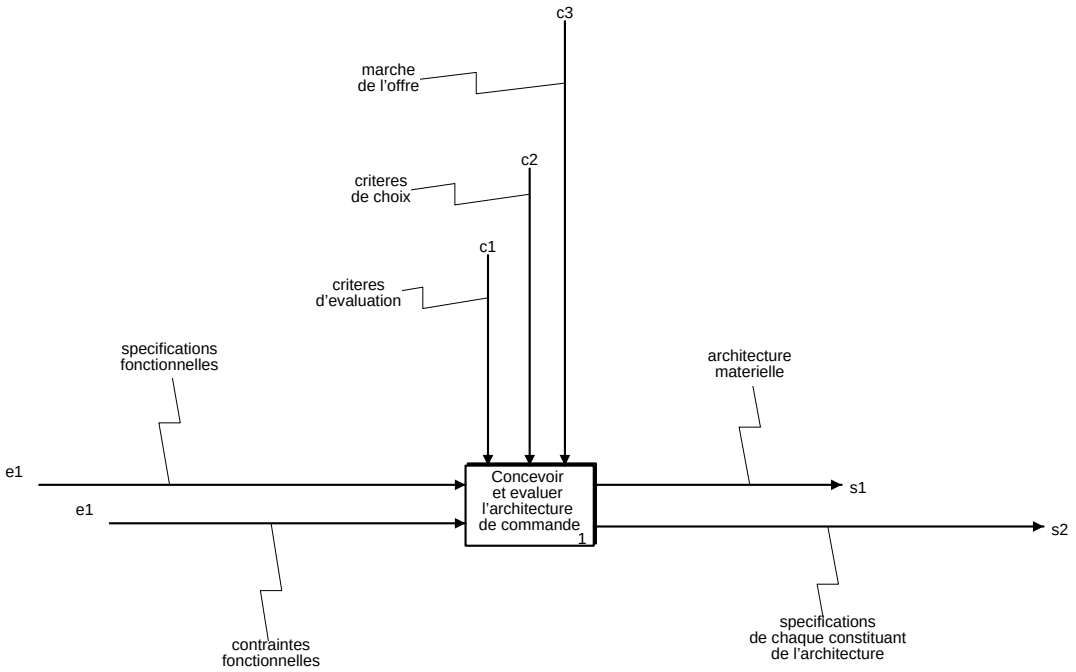
ACTIVITE DU DIAGRAMME (As0) : Repondre a l'appel d'offres

?

cout

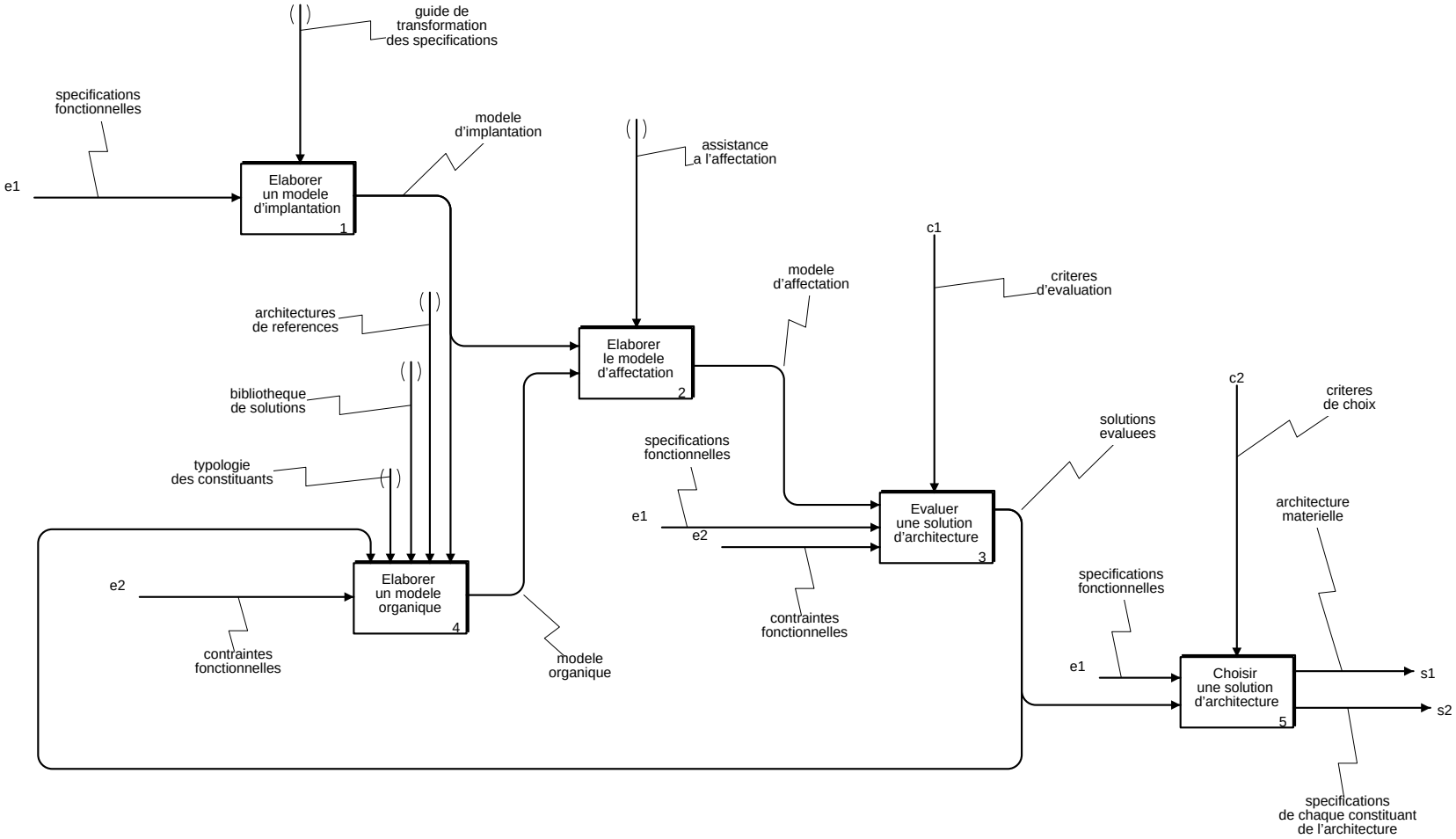
dossier de reponse a un appel d'offres

AUTEUR : B. Denis MODELE : Conception et evaluation d architecture DATE : 26 Novembre 1993	LECTEURS : J.J. Lesage	CONTEXTE : concevoir une architecture de conduite dans un environnement d'appel d'offres



A-0	Concevoir et evaluer l'architecture de commande	VERSION V1.3

AUTEUR : B. Denis MODELE : Conception et evaluation d architecture DATE : 26 Novembre 1993	LECTEURS : J.J. Lesage	CONTEXTE : Concevoir et evaluer l'architecture de commande



A0	Concevoir et evaluer l'architecture de commande	VERSION V1.3

Résumé :

Les travaux exposés dans ce mémoire portent sur la *conception de l'architecture de conduite des systèmes automatisés de production*. Ils ont été développés au sein du Laboratoire Universitaire de Recherche en Production Automatisée (LURPA - EA 1385). Il s'agit de travaux *prospectifs* dont l'objectif est de construire un *cadre méthodologique formel* de conception et d'évaluation d'architectures.

La première partie est consacrée à l'exposé du cadre méthodologique retenu et comporte quatre chapitres dans lesquels sont successivement abordés :

- la problématique générale de la conception d'architectures de conduite, ainsi qu'une revue des travaux scientifiques dans le domaine,
- la présentation générale de la méthode proposée avec le point de vue des activités de conception, d'évaluation et de choix d'architecture,
- la présentation détaillée de la construction d'architecture avec cette fois le point de vue des différents modèles produits,
- la présentation détaillée d'évaluation et de choix de l'architecture retenue avec encore une fois le point de vue des modèles d'évaluation.

La seconde partie du mémoire est destinée à préciser et à valider par l'exemple la méthode proposée. Pour ce faire, nous l'avons appliquée à la conception de l'architecture de conduite d'un poste de lancement d'un transfert libre. Le plan de cette étude suit de manière ordonnée la démarche que nous avons présenté dans la première partie.

Mots clés :
architecture de conduite, métamodélisation, conception intégrée, système de production, évaluation de performance, réseaux de Petri colorés et temporisés

Abstract :

This work concerns the control architecture design of production systems. It has been conducted at the University Laboratory for Automated Production Research (LURPA - EA 1385). The realization of a formal modelling framework, which provides a way to design architecture and a way to analyse performances, is the subject of this thesis.

The first section gives an explanation of modelling framework which has been developed. Its has been divided in four chapters dealing with:

- the problem of control architecture design, and a review of the previous works done in this research field,
- the general presentation of proposed method with a functional point of view, and proposed tools which ensures performance evaluation,
- the detailed presentation of control architecture building with a produced model point of view,
- the detailed presentation of performance evaluation of built architecture.

The second section provides an example of control architecture design to validate the proposed framework. This case study has been treated in order to show steps of the proposed method.

Key words :
control architecture, metamodelling, integrated design, production system, performance evaluation, coloured timed Petri nets