7e Conférence Internationale de MOdélisation et SIMulation - MOSIM'08 - du 31 mars au 2 avril 2008 – Paris- France
« Modélisation, Optimisation et Simulation des Systèmes : Communication, Coopération et Coordination.».

PILOTAGE ASSISTE PAR SIMULATION DISTRIBUEE DANS L'INDUSTRIE DU SEMI CONDUCTEUR

P. PUJO[*], G. ZACHAREWICZ[°], C. FRYDMAN[*], N. GIAMBIASI[*]

[*]LSIS
Av. Esc. Normandie Niemen
13397 MARSEILLE cedex 20
Patrick.Pujo@lsis.org

[°]IMS-LAPS / GRAI
Université Bordeaux - CNRS
351, cours de la Libération, 33405 TALENCE cedex
Gregory.Zacharewicz@laps.ims-bordeaux.fr

RESUME : *Après avoir exposé toute la complexité des processus de production dans l'industrie du semi conducteur, les auteurs montrent comment la simulation distribuée peut apporter une aide au pilotage des installations. L'approche proposée tend à s'appuyer sur les capacités de différents cadres de modélisation formelle pour décrire les processus opérationnels et les processus décisionnels, à l'aide d'éditeurs graphiques adéquats. Ensuite, ces différents modèles de processus sont transformés en modèles G-DEVS (Generalized Discrete EVent Specification) interconnectés. Enfin, tous ces modèles sont exécutés simultanément par simulation dans un environnement HLA (High Level Architecture). Il devient alors possible d'observer le comportement de tout ou partie du système de production, et d'en tirer les informations propres à aider à la prise de décision lors du pilotage des installations. Les différentes briques de cette approche seront illustrées via l'exemple simple d'une production dans une usine de 'Front End'.*

MOTS-CLES : *G-DEVS, HLA, Simulation distribuée, WorkFlow, JIS Z 8206, micro électronique*

1. INTRODUCTION

En suivant depuis plus de 30 ans la loi de Moore (tous les 18 mois, la densité surfacique de transistors dans les puces électroniques double…), la fabrication en micro électronique se doit d'être de plus en plus performante. Outre la sophistication des équipements, les usines de fabrication sont dotées d'outils de gestion et de systèmes d'information très élaborés. Toutefois, ceci ne suffit pas pour appréhender toute la complexité des différents types de processus (opérationnels et décisionnels) inhérent à ce type de production. Aussi, il est nécessaire d'avoir des outils pour évaluer très rapidement l'incidence de telle ou telle prise de décision. Pour cela, le développement d'un système de simulation a été entrepris : il s'agit de créer un système de pilotage assisté via la simulation, avec l'équivalent d'une usine virtuelle, alimentée en données par l'usine réelle, sur laquelle peuvent être testées en permanence les décisions à prendre, afin d'en analyser l'impact avant la mise en œuvre. Un tel système de simulation, qui doit s'appuyer sur des modèles d'origines multiples, ne peut s'envisager qu'au travers de la simulation distribuée. Cet article présente les premiers résultats obtenus lors de cette réalisation. Après une caractérisation des processus dans l'industrie micro électronique, nous présentons ici les formalismes de modélisation de processus que nous avons retenus pour couvrir l'ensemble de ces processus. Puis, nous exposons les mécanismes de transformation de ces modèles vers un formalisme unique G-DEVS. Ensuite, nous montrons comment ces modèles G-DEVS peuvent être mis en œuvre dans l'environnement de simulation distribuée HLA. Un exemple de simulation distribuée, relatif au cheminement d'un produit, est ensuite développé.

2. CONTEXTE DE L'ETUDE

Les systèmes de production manufacturière dans l'industrie du semi conducteur sont actuellement caractérisés par une complexité inégalée. Indépendamment des objectifs de productivité et de rationalité communs à toutes les entreprises, la micro électronique fournit des produits de plus en plus sophistiqués, résultant de l'association de millions de transistors dont la surface élémentaire est de l'ordre du micromètre carré, et fiables à la fois. Par ailleurs, les clients finaux exigent des taux de défectueux de l'ordre du millionième, ce qui nécessite le déploiement de managements Six Sigma (Pinaton et Campion, 2004). Parallèlement, la haute technicité des produits génère un renouvellement rapide de l'offre, du fait de l'évolution des technologies et de la demande du marché : la durée de vie d'un produit est de plus en plus courte, et il arrive fréquemment qu'elle soit de l'ordre de ses durées cumulées de mise au point, d'industrialisation et de production, ce qui ne laisse que peu de latitudes de réaction si des problèmes apparaissent lors de la fabrication. De tout ceci émerge la nécessité d'une plus grande dynamique dans le fonctionnement des équipements dans les unités de production 'Front End' (de la plaquette de silicium vierge au wafer comportant plusieurs puces gravées et testées). Ces équipements sont toutefois contraints par les aspects technologiques et physiques des parties de processus qu'ils assument. Il en résulte qu'une large part de la performance de l'activité de fabrication repose sur les actions de pilotage, dont un des objectifs est de prévenir les dysfonctionnements.
La véritable difficulté rencontrée ici provient de la grande complexité, quantitative et qualitative d'une part, et aux niveaux opérationnels et décisionnels d'autre part.

2.1. Processus opérationnels et décisionnels

L'unité de fabrication 'Front End' de Rousset, qui emploie environ 3000 personnes (dont 1000 ingénieurs) travaille 24 heures sur 24 et 7 jours sur 7. Elle supporte la fabrication d'une quinzaine de technologies différentes, correspondant à environ 350 produits génériques actifs (c'est-à-dire pouvant être différentiés par un codage physique de programmes ou de données). Le temps de cycle moyen est d'environ 60 jours, il correspond en moyenne à 360 opérations réalisées sur 250 équipements de production. Ceci sous entend que les gammes sont bouclées : chaque produit passe plusieurs fois par certains équipements, c'est-à-dire que certaines parties du processus sont parcourues plusieurs fois, pour des dépôts-gravures successifs. Parallèlement aux activités de production, une grande partie des activités correspondent à des activités de contrôle en vue de la surveillance du processus et du maintien de la qualité des produits. 20000 contrôles sur équipement et 60000 contrôles sur produit sont ainsi réalisés hebdomadairement ! Le système de pilotage doit donc intégrer cette dimension : analyser tous ces résultats, détecter ceux qui sont représentatifs ou potentiellement porteurs d'un futur dysfonctionnement, émettre un diagnostic et concevoir une stratégie corrective, puis enfin déployer les actions précédemment décidées.
La gestion de production conventionnelle (planification – ordonnancement – MES) s'avère dans ces conditions insatisfaisante : il y a trop de perturbations internes dans le processus, et ces perturbations sont difficiles à gérer. En effet, certaines prises de décision de pilotage en production doivent être prises relativement à des avis d'experts relevant des bureaux d'étude, selon des protocoles bien définis appelés ECN (Engineering Change Notice).
L'obtention de l'efficience du pilotage se heurte donc en micro électronique à une complexité relevant de 3 aspects quantitatif, structurel et qualitatif, que nous allons détailler.

2.2. Analyse de la complexité des processus

2.2.1 Complexité quantitative

Les gammes de fabrication des circuits, appelées ***Route***, décrivent l'itinéraire que suit le produit de la première opération à la dernière, nécessaire à l'obtention d'un produit particulier. Les routes diffèrent selon leur nature, le nombre et la durée des opérations. Une ***Opération*** est la brique élémentaire d'une route reliée à un travail (i.e. diffusion, photolithographie, gravure, implantation, etc.). Chaque opération est composée par une séquence ordonnée d'***Event*** nommée ***Script***. Un script donne la définition claire et détaillée des tâches : c'est l'équivalent du contrat de phase. Les Events (évènements) appelés Steps sont toutes les actions réalisées qui sont décrites via le CAM system (Computer Aided Manufacturing). Ce dernier gère l'état des lots en cours de fabrication et envoie à chaque équipement les ***recipes*** (recette qui correspond à une étape de traitement, dans un équipement donné : il s'agit des paramètres nécessaires à la réalisation de l'opération). Tous les éléments principaux de processus de fabrication sont pilotés, coordonnés et maîtrisés par WorkStream, un outil de pilotage de type Manufacturing Execution System (MES).
Cette description très hiérarchisée et détaillée des tâches à accomplir doit être ramenée à la variété de produits évoquée au paragraphe précédent, ainsi qu'au nombre d'opérations. Une tentative de regroupement des routes par familles ne diminue quasiment pas cette complexité. Nous pouvons rajouter à cela que la production s'effectue par lot contenant plusieurs wafers, mais que sur un wafer peuvent être gravé plusieurs types de circuits. Par ailleurs, la localisation d'un circuit sur son wafer peut influencer la qualité du résultat final, certains procédés de diffusion pouvant générer des défauts physiques d'homogénéité dans la fabrication.
Pour répondre à ce défi relevant de la complexité quantitative, les unités de fabrication sont dotées de systèmes d'information performants, capables d'emmagasiner toute la dynamique du système. Ainsi, toutes les évolutions sont sauvegardées, mais également tous les contrôles effectués, au moins le temps de fabrication du lot en question (60 jours en moyenne). Il faut impérativement, lors d'un éventuel incident, être capable de parcourir à nouveau l'intégralité de la route suivie par le produit, et d'en analyser tous les paramètres. Ceci aboutit à des moyens de sauvegarde considérables de l'ordre de plusieurs centaines de tera octets.
Il va de soi que de tels systèmes d'information sont aussi complexes et sensibles que les systèmes opérationnels de production qu'ils soutiennent.

2.2.2 Complexité structurelle

Le traitement des ECN s'apparente à une action de pilotage du pilotage. En effet, il s'agit de modifier les conditions dans lesquelles s'effectue le pilotage opérationnel des équipements (figure 1).

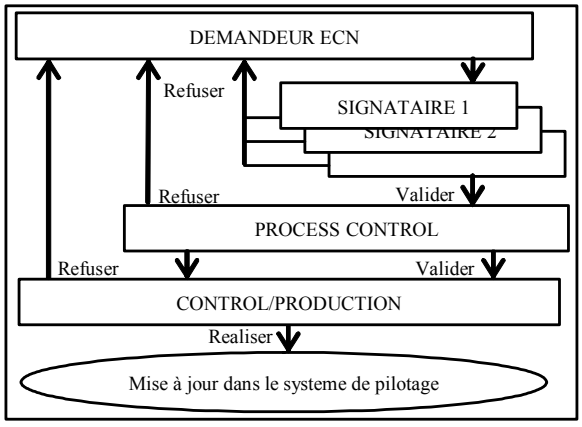

Figure 1. Circuit de décision de pilotage du pilotage

3 types d'acteurs peuvent demander la mise en œuvre d'un ECN : des ingénieurs Device, Process ou Process Control. Les ingénieurs Device font souvent des modifications sur une route ou un produit, à des fins d'amélioration technique. Les ingénieurs Process de-

mandent plutôt des modifications sur des Events, par exemple sur une recette en vue d'une amélioration de process. Les ingénieurs Process Control font des demandes de qualification d'event, de changements de limite de spécification ou contrôle (SPC) afin d'améliorer le rendement (accroitre les limites augmente le nombre de contrôles corrects, donc de produits acceptés). A la fin de sa demande, le demandeur doit définir la liste des approbateurs qui doivent valider sa demande. Ces signataires sont des ingénieurs de Device, Process, Manufacturing, Quality. Ils ne peuvent pas modifier une demande, mais seulement ajouter un commentaire et valider / refuser.
Si tous les approbateurs liés à une demande l'ont validée, elle est transférée dans la liste d'attente du Process/Control pour la validation finale.
Il existe une douzaine de types d'ECN, selon lequel le circuit de validation diffère. Pour améliorer la fiabilité et la dynamique du traitement des ECN, les procédures s'appuient sur des WorkFlows, ce qui permet de ramener le délai de 15 jours à quelques jours par rapport à un traitement manuel.

2.2.3 Complexité qualitative

Paradoxalement, cette amélioration importante dans la réactivité se traduit par une plus grande difficulté pour le pilotage, qui voit son horizon de réaction raccourcir.
Tout incident en production signalé par un contrôle, par exemple un paramètre critique sur une étape de process, pour lequel est observé une sortie des limites de la carte de contrôle correspondante, se traduit par un arrêt de la production du lot en question et/ou par la suspension de l'équipement. L'opérateur suit alors une ***Ocap*** (Out Control Action Plan) qui lui indique quelles sont les actions à suivre pour interrompre la production de ce lot.
Ceci effectué, la suite des opérations nécessite l'avis de différents spécialistes. En fonction de la criticité de l'incident, soit l'arrêt est définitif, c'est-à-dire que le défaut est irréversible et le lot doit être détruit (***Scrapé***), soit il est mis en attente (***Holdé***) pour une reprise ultérieure, sous forme de retouche ou de dérogation.
Dans tous les cas, une ECN est nécessaire pour la suite des opérations : soit il s'agit de mettre en œuvre des moyens pour que le ***Scrap*** ne se reproduise plus, soit il s'agit d'éviter à l'avenir un ***Hold*** inutile, soit il s'agit d'une fausse alerte sans conséquence (une transaction ***Release*** permet de débloquer le lot), soit il s'agit, lorsque le process s'y prête, de refaire un Step défectueux (une route de ***Rework*** permet d'annuler et de refaire certaines opérations qui n'auraient pas été réalisées dans des conditions adéquates (exemple : le dépôt d'une résine)).
Dans la majeure partie de ces cas, c'est l'analyse qualitative, l'expérience et/ou l'expertise qui guide les choix de validation ou non des approbateurs de l'ECN.
Dans certains cas, cela peut aboutir à des opérations de contrôle technique et de maintenance de l'équipement.
Dans tous les cas, le fait d'accélérer les procédures ECN oblige le pilotage opérationnel à réintégrer rapidement le lot dans le planning de production. Là où auparavant, un incident se traduisait par une libération de charge sur le planning, charge qui était reportée sur un planning hebdomadaire ultérieur, il faut maintenant considérer l'incident comme n'ayant que des conséquences temporaires, donc lié à une reprise de production rapide, sans pour autant savoir exactement ce qu'il y aura à faire.

2.2.4 Incertitude de la prise de décision en pilotage

Nous venons de constater que donner simultanément de la robustesse au fonctionnement du process, par la consultation d'experts, et de la réactivité, en organisant rapidement ces consultations, génère pour le pilotage une grande difficulté. En effet, même si certaines réactions peuvent être facilement anticipées, en prenant comme hypothèse la simple validation de la demande contenue dans l'ECN, il demeure une incertitude sur cette validation et surtout sur la date de retour de l'information. C'est là qu'intervient l'utilité de la simulation pour le pilotage opérationnel.

2.3. Aide au pilotage via simulation

Par simulation, il est en effet possible d'appréhender cette date, qui correspond à la date de reprise au plus tôt du lot : il ne reste alors qu'à l'associer aux tâches de reprise à insérer dans l'ordonnancement courant et à lancer un ré-ordonnancement. Ces tâches sont alors réalisées par le MES.
Pour réaliser une telle simulation, il faut bien entendu avoir une vue globale de tous les processus en interactions, autant au niveau opérationnel qu'au niveau décisionnel. C'est ce que nous allons développer ici.
Toutefois, l'usage de la simulation peut être étendu pour l'aide à la compréhension de la complexité des systèmes de production dans l'industrie du semi conducteur. En effet, la simulation peut s'avérer très utile pour la validation de décision de pilotage et pour la mise au point des règles correspondantes.

3. PILOTAGE ASSISTE PAR SIMULATION

Nous venons de voir qu'il faut assister le pilotage par une fonction simulation. En fait, cette fonction vient en complément des 3 fonctions de base que sont la décision, le lancement et le suivi (figure 2). La simulation n'a pas de rôle 'actif' dans la mesure où elle ne prend pas de décision, mais permet au pilotage de les prendre à partir d'informations calculées à partir des informations réelles prélevées sur les équipements et traduisant l'état du système opérationnel, via la fonction suivi du MES.

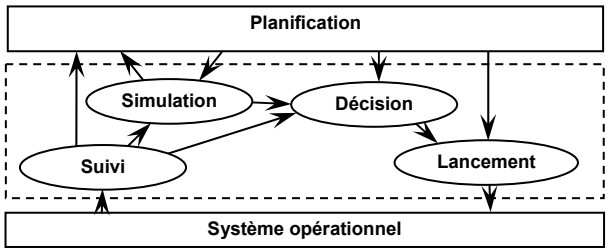

Figure 2. Place du pilotage opérationnel proactif et de ses principales fonctions (Pujo *et al.*, 2004).

3.1. Pilotage assisté par simulation distribuée

Nous avons vu qu'il existe différents types de processus à simuler simultanément. Une des difficultés réside dans le fait que ces processus ne relèvent pas du même formalisme, car ils ne relèvent pas des mêmes métiers. Des spécialistes les décrivant ne seraient pas capables de se mettre d'accord sur le choix d'un formalisme unique de description. Nous avons choisi donc de laisser les spécialistes s'exprimer dans le formalisme de leur spécialité. Puis, pour obtenir un fondement plus sûr, des propriétés de vérification et faciliter la mise en œuvre de la simulation, nous avons préféré traduire tous les modèles en un formalisme unique, plus précis et plus rigoureux, à partir duquel pourront être effectuées les simulations souhaitées.
Par ailleurs, ces processus sont le plus souvent exprimables par morceaux, le comportement de tel ou tel sous ensemble ne pouvant s'exprimer de manière intelligible que sur une partie réduite du fonctionnement de l'ensemble :
- soit il s'agit du comportement du système réduit à un seul produit,
- soit il s'agit du comportement du système réduit au fonctionnement nominal,
- soit il s'agit au contraire du traitement des cas exceptionnels, quand se produit un dysfonctionnement,
- soit il s'agit d'une vue purement décisionnelle ou purement opérationnelle…

Le système global sera donc décrit par juxtaposition et interaction de nombreux modèles, pour lequel nous devons envisager des utilisations contextuelles : les liens (couplages) entre ces modèles se feront selon les cas de manière implicite ou explicite. Pour cela, la simulation distribuée est la meilleure solution envisageable.

3.2. Différents types de processus à formaliser

Pour essayer de réduire la complexité de l'environnement de simulation distribuée à mettre en place tout en validant complètement l'approche proposée, nous avons réduit le nombre de formalismes de description des processus à deux : l'un pour les processus décisionnels, l'autre pour les processus opérationnels.

3.2.1 Processus décisionnel en WorkFlow

Les processus décisionnels utilisés dans l'entreprise, plus particulièrement au niveau des ECN évoqués au paragraphe 2.2.2 font déjà l'objet d'une modélisation sous la forme de workflows. Nous avons vérifié que tous les processus décisionnels pouvaient être facilement décrits sous forme de workflows. Un éditeur graphique, LSIS-WME (pour Workflow Model Editor), permet de représenter par exemple un flux logique de production (ou gamme, ou route) en respectant des règles grammaticales basées sur des prémisses issus de la WfMC afin d'en assurer la représentation explicite (WfMC, 1999), (Zacharewicz *et al*. 2006a). La figure 3 montre le processus décisionnel qui permet de suivre le parcours d'un produit (ou d'un lot) d'un même produit tout au long de sa route. Un parcours informationnel est assuré simultanément et parallèlement à la transformation du produit physique au niveau du processus opérationnel.

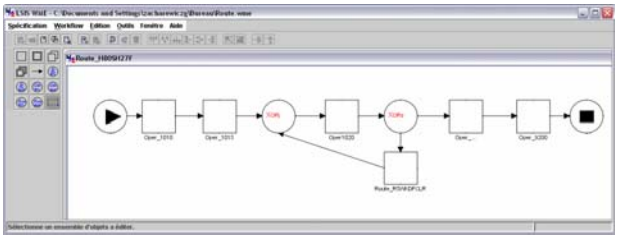

Figure 3. Exemple de processus décisionnel : extrait d'une route

La figure 4 donne un aperçu d'un ECN : la procédure de demande de modification des limites de contrôle sur un équipement de production et sa structure logique.

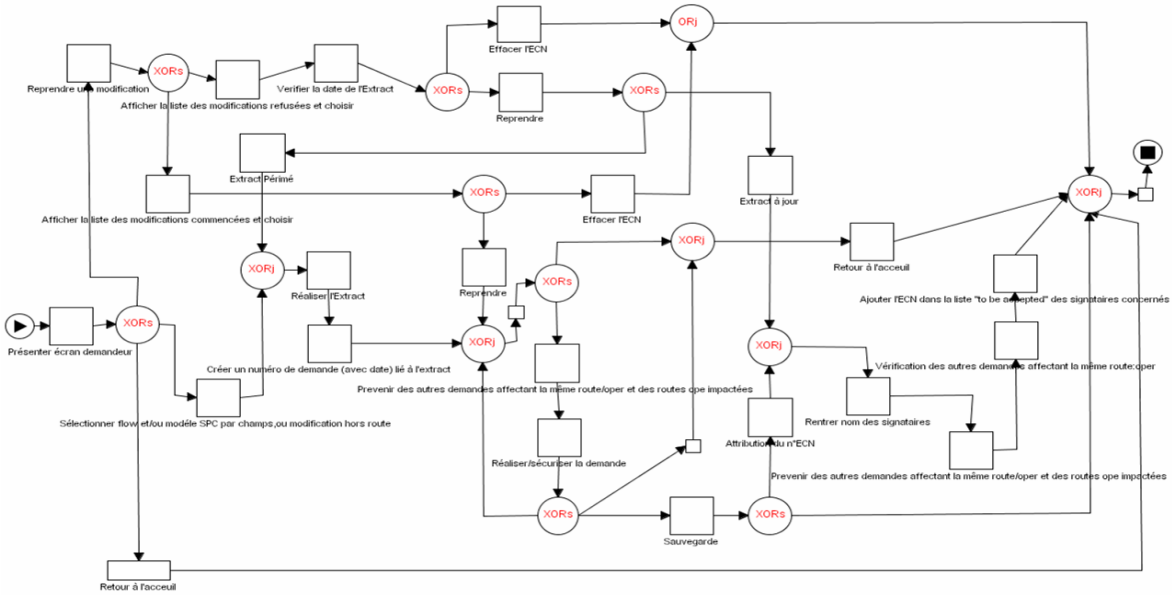

Figure 4. Vue générale d'un processus décisionnel : procédure d'un ECN en Statistical Process Control

Ces processus workflows sont sauvegardés en XML. Plus que le détail du processus ici présenté, il faut retenir que chaque modèle est paramétrable au niveau modèle métier et au niveau modèle DEVS. Nous pouvons en particulier spécifier des fonctions de transitions complexes sur une base de fonctions mathématiques ad hoc.

3.2.2 Processus opérationnels en JIS Z 8206

Les processus manufacturiers font l'objet de modélisations depuis les débuts de l'ère industrielle. Les premières représentations intéressantes datent des années vingt, avec les travaux des époux Gilbreth, qui proposaient un ensemble de 26 symboles graphiques correspondant aux différents équipements types de l'époque. Cette première représentation a fait l'objet en 1947 d'une standardisation, avec un vocabulaire graphique de 5 symboles, correspondant aux 5 grandes classes d'activités rencontrées dans un système manufacturier : l'opération (transformation, assemblage…), le transport, le stockage, l'attente (l'encours ou WIP) et les activités de contrôle et d'inspection.

Cette représentation, reprise par la norme japonaise JIS Z 8206 (JIS, 1982), nous est apparu comme le meilleur standard pour représenter les processus opérationnels. Un éditeur graphique développé au LSIS : LSIS_PME (Production Model Editor) (figure 5) permet de décrire des processus opérationnels, sauvegardés en XML.

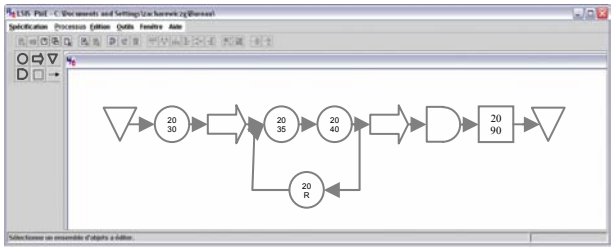

Figure 5. Exemple de processus opérationnel correspondant à la route figure 3

3.2.3 Synchronisations entre opérationnel et décisionnel

Ces différents modèles ont bien entendu des interactions. Nous pouvons identifier 2 types d'interactions : celles qui existent obligatoirement, donc de façon structurelle et automatique, et celles qui sont nécessitées par le fonctionnement, et qui sont définies au cas par cas.

Les premières, que nous qualifierons d'implicites, permettent de relier par exemple les processus décisionnels et opérationnels qui suivent les produits. Ainsi, les processus opérationnels n'intègrent aucune règle de pilotage, mais chacune des 5 classes d'activité génère en entrée un ensemble de vérifications validant la faisabilité de l'activité :

- est-ce que la ressource est bien compatible avec la tâche demandée sur le produit ? (une erreur de configuration de la route est toujours possible),
- est-ce que la tâche en question est autorisée ? (cette question est très fréquemment pertinente, concernant notamment les problèmes de *Hold* et de *Scrap* de la production).
- est-ce que les bons paramètres de configuration de l'équipement sont téléchargés ? (en particulier, lors d'une ECN visant une quelconque modification dans l'utilisation d'une machine, il faut éviter de produire avec des paramètres identifiés comme potentiellement mauvais. Il faut donc attendre la fin du traitement de l'ECN pour savoir quelles seront les conditions de production, et si la *recipe* a été changée).
- …

La réponse à ces pré-conditions est obtenue via un dialogue automatique et systématique de synchronisation et de validation entre l'activité en question dans le processus opérationnel et l'activité correspondante dans le processus décisionnel décrivant la route à suivre.

Passée cette étape, que nous pouvons qualifier de pré synchronisation (figure 6), le processus décisionnel entre en état d'attente de fin de tâches, tandis que le processus opérationnel simule la tâche sur l'équipement. Pour l'instant, nous faisons l'hypothèse simplificatrice que tout se passe bien. Nous pourrons par la suite intégrer des comportements stochastiques de pannes ou de dérive de l'équipement. A la fin de la tâche, le processus opérationnel valide le passage à la prochaine étape de traitement et rend compte de l'accomplissement de la tâche au processus décisionnel, au travers d'un état comportant plusieurs attributs. Il s'agit alors d'actions de post synchronisation. Le processus décisionnel peut alors valider le passage à l'opération suivante, mais il peut également déclencher d'autres processus, comme des routes de *rework* ou des ECN.

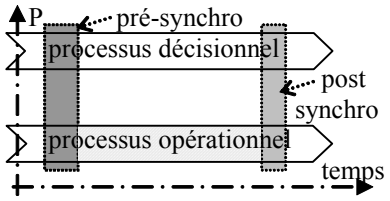

Figure 6. Chronogramme de synchronisation entre processus décisionnel et opérationnel.

D'autres types de synchronisation inter processus sont également possibles. En particulier, la modélisation via workflows permet de considérer une approche modulaire et une composition hiérarchique, concrètement une opération élémentaire d'un workflow peut être également un workflow pris dans une bibliothèque de processus prédéfinis.

De plus, des opérateurs logiques permettent de créer des processus divergents, des processus convergents et des alternatives paramétrables par le contenu du flux…

3.3. Principe d'interopérabilité des modèles

Les descriptions orientées 'processus' ne sont pas réputées pour être fondées sur des concepts formels forts (Van Der Aalst, 2002), ce qui limite la vérification des propriétés sémantiques et la validation des modèles ainsi décrits. Pour faciliter l'utilisation conjointe de tous ces modèles grâce à une vérification systématique, nous

avons préféré simplifier l'exécution de la simulation grâce à une transformation préalable de tous les modèles dans un formalisme unique commun à tous les processus, G-DEVS.

3.3.1 Choix d'un formalisme unificateur G-DEVS

Il existe divers formalismes pour aborder la modélisation des systèmes complexes à évènements discrets : citons les statecharts, les réseaux de Petri ou le formalisme DEVS. DEVS, défini par (Zeigler, 1973) comme une spécification formelle de modèles à événements discrets (DEVS : Discrete EVent system Specification), est un formalisme abstrait universel indépendant de l'implémentation. Il permet de modéliser facilement des systèmes complexes. Enfin, il intègre intrinsèquement le temps, permet de manipuler conjointement les deux concepts d'état et d'évènement. Il introduit notamment la possibilité d'évolution autonome du modèle grâce à la durée de vie des états, associée à la fonction de transition interne. Le concept de modèles basiques-couplés, introduit ultérieurement (Zeigler, 1984), fournit un moyen de construire des modèles composés et hiérarchiques, en réutilisant des descriptions stockées en librairie. Enfin, DEVS propose une spécification formelle de la sémantique opératoire des mécaniques de simulation clairement séparée des modèles (Zeigler *et al*, 2000) : modularité, hiérarchisation des modèles, réutilisation (définition de bibliothèques), utilisation explicite du temps, indépendance lors de l'implémentation d'un langage particulier.. Nous avons plus particulièrement choisi d'utiliser G-DEVS, qui est bien adapté aux processus multi-variables. G-DEVS (Generalized Discrete EVent Specification) (Giambiasi *et al,* 1995 ; Giambiasi *et al,* 2000) est une extension du formalisme DEVS qui permet d'intégrer en lieu et place d'entrées/sorties constantes par morceau des entrées/sorties polynomiales par morceau. Les valeurs des entrées/sorties sont toujours discrètes, mais les trajectoires d'entrées/sorties sont définies polynomiales par morceaux, chaque polynôme d'ordre N n'est décrit que par ses N coefficients et le passage d'un morceau à un autre demeure un évènement discret. G-DEVS est dans notre cas particulièrement adapté, car nous pouvons attacher à chaque évènement un vecteur de valeurs qui permet de répondre au caractère complexe de la description des produits en cours de production.

3.3.2 Mécanisme de transformation en G-DEVS

Le mécanisme de transformation pour passer des descriptions orientées 'processus' (workflow et JIS Z 8206) en modèles G-DEVS (figure 7) est fondé sur l'utilisation d'un langage formel de description d'un processus, compatible avec une représentation XML. Nous avons créé ce langage en référence à la représentation XML des workflows proposées par la WfMC. Grâce à cela, la validation syntaxique du modèle est obtenue en référence à un DTD (Document Type Definition) de workflow élaboré par la WfMC (WfMC, 2005). Nous avons extrait les concepts essentiels à la définition d'un DTD pour les processus opérationnels depuis le document de la WfMC. Un processus P est alors défini par une structure algébrique telle que :
P = (Nom, RESS, A, Ct, ARC, IT, ST), où :

- Nom, le nom du processus,
- RESS, l'ensemble des ressources du processus,
- A, l'ensemble des tâches du processus,
- Ct, l'ensemble des contrôleurs du processus,
- ARC, l'ensemble des arcs du processus,
- IT, l'ensemble des items du processus,
- ST, l'ensemble des stocks du processus.

Ces processus sont pris en compte après transformation par LSIS_DME (DEVS Model Editor). Une bibliothèque de modèles G-DEVS 'génériques' contient les différents éléments d'un processus.

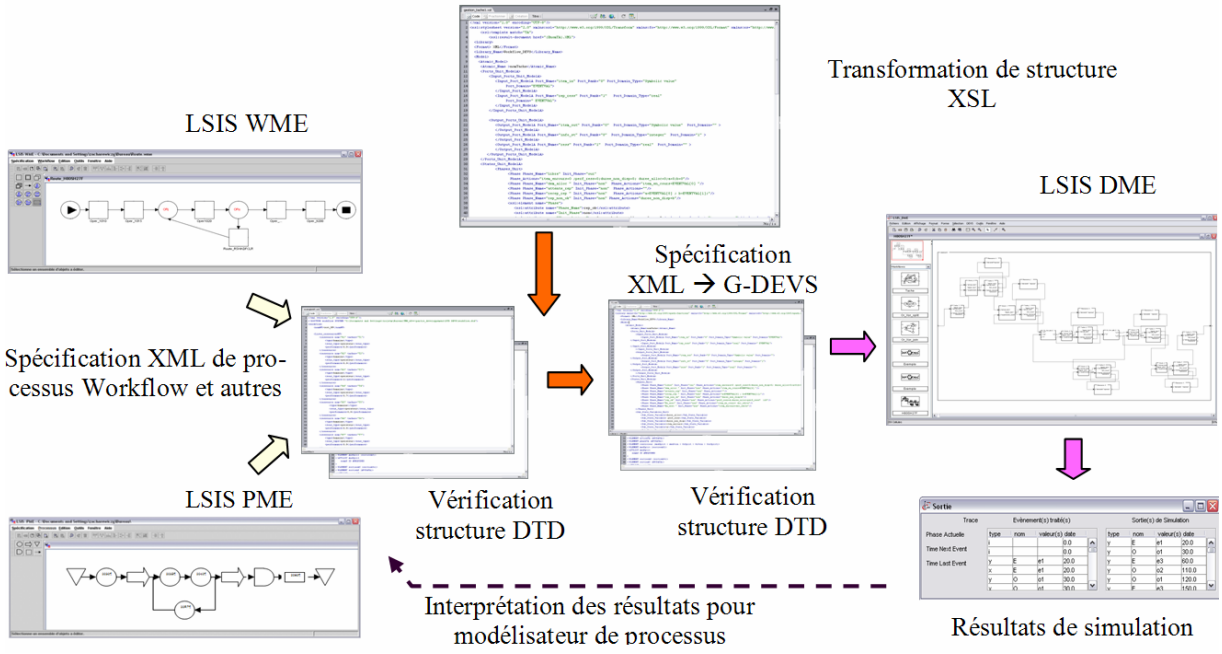

Figure 7. Mécanisme de transformation de formalismes de spécialistes en formalisme G-DEVS

Chacun de ces modèles est défini sous la forme d'un modèle couplé. L'algorithme de transformation utilisé a été proposé dans (Zacharewicz, 2006).

3.4. Mise en œuvre de la simulation distribuée via HLA

3.4.1 HLA (High Level Architecture)

HLA est une spécification d'architecture logicielle créée pour le DoD américain, via son département 'modélisation & simulation' pour pouvoir exécuter une simulation globale composée de simulations distribuées et hétérogènes. Dans HLA, chaque simulation participante, appelée fédéré, dialogue avec les autres fédérés dans une fédération HLA. HLA est entièrement normalisé. Par exemple, la norme IEEE (M&S) P1516.2-2000 décrit comment s'effectuent les communications entre modèles et la mise en application via un «coordinateur central» : le RTI (Run Time Infrastructure) (figure 8). Les fédérés interagissent en utilisant des services proposés par le RTI. Notamment, ils peuvent 'publier' pour informer de leur intention de fournir des informations et 'souscrire' pour en récupérer d'autres provenant d'autres fédérés. Ces informations ont des structures orientées objet. Deux types d'objets sont échangés dans HLA, les classes d'objet, persistant pendant la durée de la simulation, et les classes d'interaction, qui ne subsistent qu'au cours de transmissions entre fédérés... Enfin, pour respecter les relations de causalité temporelles lors de la simulation, HLA offre divers mécanismes de gestion et d'avancement du temps, avec les algorithmes de synchronisation de simulation distribuée (optimistes et pessimistes) (Fujimoto 1998) (Zacharewicz *et al*. 2006b).

3.4.2 G-DEVS / HLA

L'éditeur graphique LSIS_DME (pour DEVS Model Editor) (Hamri et al, 2007) a été étendu pour pouvoir créer des modèles G-DEVS HLA conformes (Zacharewicz et al, 2006), afin de permettre une simulation en mode distribué. La structure d'un modèle G-DEVS peut être maintenant décomposée en différents modèles qui seront implémentés dans des fédérés (figure 8) pour constituer une fédération HLA (c'est-à-dire un modèle couplé G-DEVS distribué).

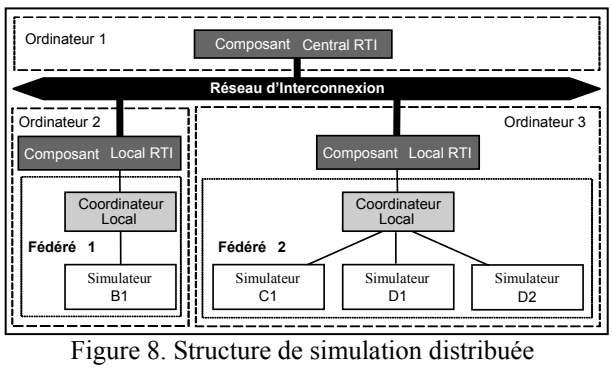

Figure 8. Structure de simulation distribuée G-DEVS/HLA

4. REALISATION ET EXEMPLE

L'illustration faite ici montre la faisabilité des concepts développés au §3 pour résoudre les problèmes soulevés au §2. Cette réalisation reste à industrialiser pour pouvoir être déployée sur l'ensemble de l'unité de production.
Cela dit, les 3 grandes briques fonctionnelles que sont l'édition des processus, la transformation en G-DEVS et la simulation distribuée via HLA sont opérationnelles. Nous revenons ici sur le premier et le troisième point.

4.1. Edition graphique de processus

Tous les processus peuvent être décrits graphiquement via un éditeur implémenté en Java et donc indépendant des systèmes d'exploitation. L'ensemble des 3 éditeurs LSIS-WME (workflow), LSIS-PME (JIS Z 8206) et LSIS-DME (DEVS, G-DEVS…) constitue une suite homogène et cohérente, en termes de graphisme, de fonctionnement et de format de sauvegarde XML. Nous ne présenterons ici que le premier de ces éditeurs.

4.1.1 LSIS-WME

LSIS_WME implémente les fonctionnalités suivantes :
• Création d'une spécification Workflow hiérarchisée graphique.
• Gestion des différents contrôleurs de flux : And Join, And Split, Or Join, Or Split, Xor Join et Xor Split
• Edition de tous les paramètres d'une tâche : Nom, Type, Priorité, Description, Action, Ressources
• Vérification syntaxique en ligne du Workflow : Vérification des connexions interdites, et de la structure explicite du Workflow.
D'autres fonctionnalités sont en cours de développement : Gestion avancée des ressources, Création et gestion de triggers, Gestion des multi-instances et de la hiérarchie du flux, Editions avancées des transitions du flux, Exports vers différents formats.
Une copie d'écran a été donnée à la figure 3. Ce workflow est un processus décisionnel qui permet de suivre la route logique du produit et de s'assurer ainsi de son avancement.

4.1.2 Traduction d'un workflow en G-DEVS

Pour continuer cet exemple, nous avons à la figure 9, visualisé dans LSIS-DME, le même processus que celui de la figure 3, traduit en modèles G-DEVS couplé.
En pratique, chaque opération du workflow correspond à un template générique de modèle G-DEVS couplé qui est implémenté lors de la traduction avec les informations contenues dans le modèle du workflow.
Chaque élément de ces modèles couplés est lui-même composé de modèles G-DEVS atomiques.
Ceci est illustré dans les figures suivantes : la première opération du workflow 'oper_1013' est composée de 3 modèles G-DEVS couplés : 'stock 0', 'ressource 0' et 'tache 0' (figure 10), 'tache 0' est décrit sous forme de G-DEVS atomique en figure 11.

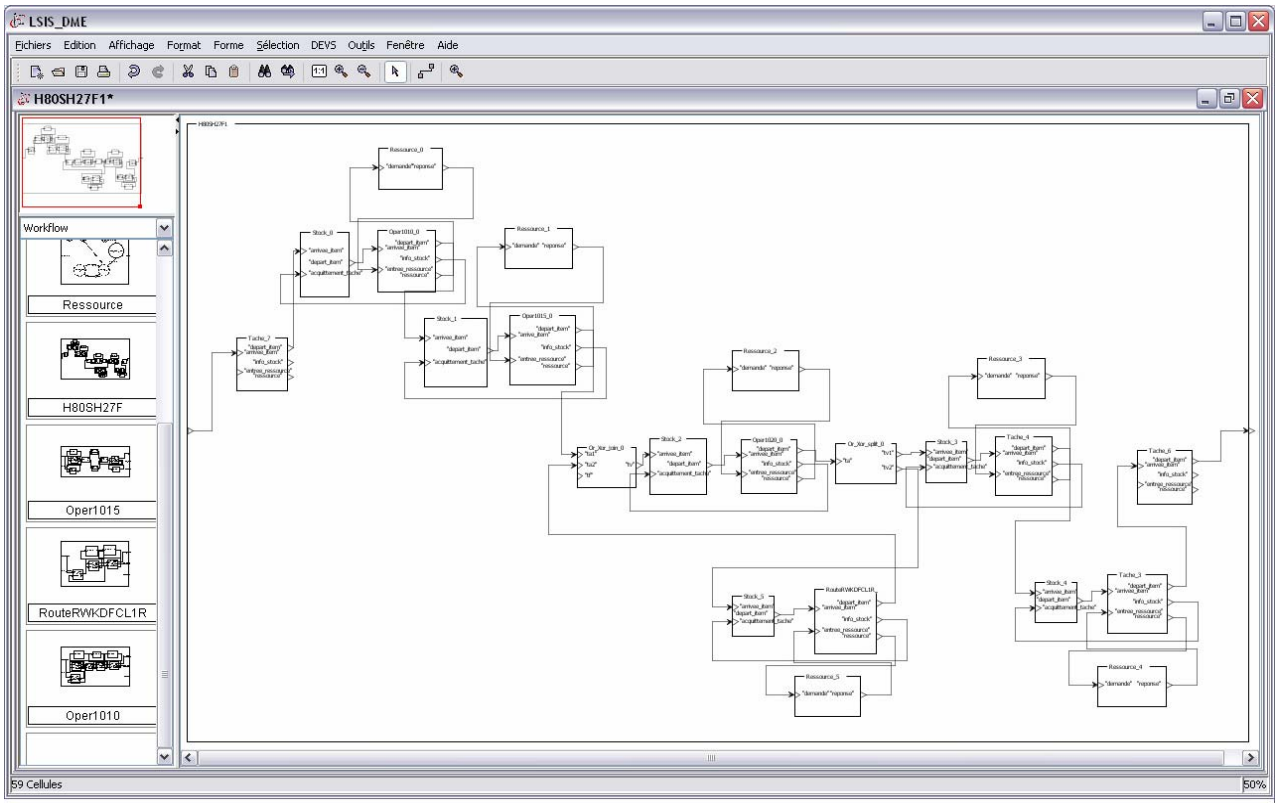

Figure 9. Editeur Graphique LSIS-DME : modèle d'une route au niveau logique, en G-DEVS

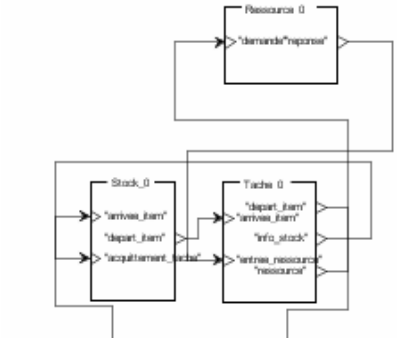

Figure 10. Traduction de l'opération Oper_1013 en modèles G-DEVS couplés

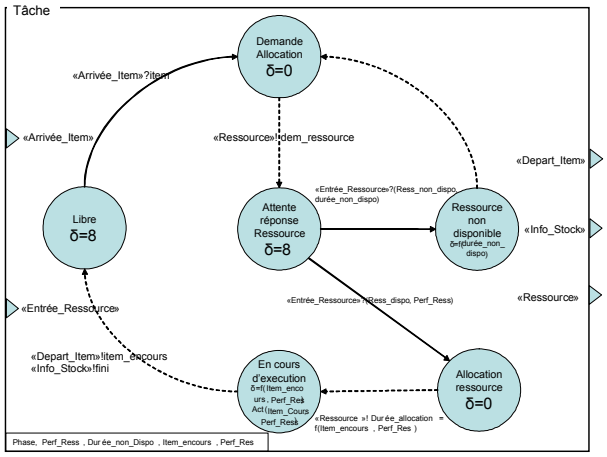

Figure 11. Décomposition du modèle G-DEVS couplé 'tache 0' en G-DEVS atomique

4.2. Mise en œuvre de la simulation via HLA

La fédération HLA regroupe ainsi un ensemble de simulations (appelées fédérés) de modèles G-DEVS s'échangeant des informations. Pour effectuer une simulation, une fédération requiert un fichier qui décrit notamment les objets et interactions partagées (FOM File). Il spécifie les données partagées et leurs modes d'échange entre fédérés (figure 12). La mise en oeuvre de HLA implémente un composant central (Central RTI Component) en charge de l'échange de données synchronisé entre les fédérés. Enfin, chaque fédéré implémente une classe héritant de méthodes pour invoquer les services HLA en communiquant avec un Composant Local du RTI (LRC) qui assure l'interopérabilité (interface Java, C++…) avec le CRC.

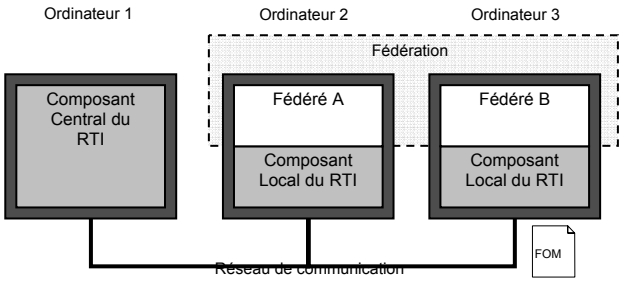

Figure 12. Vue logique des composants de RTI -HLA

L'ensemble de fédérés correspondant à notre exemple (figures 3, 5, 9-11) est montré en figure 13.

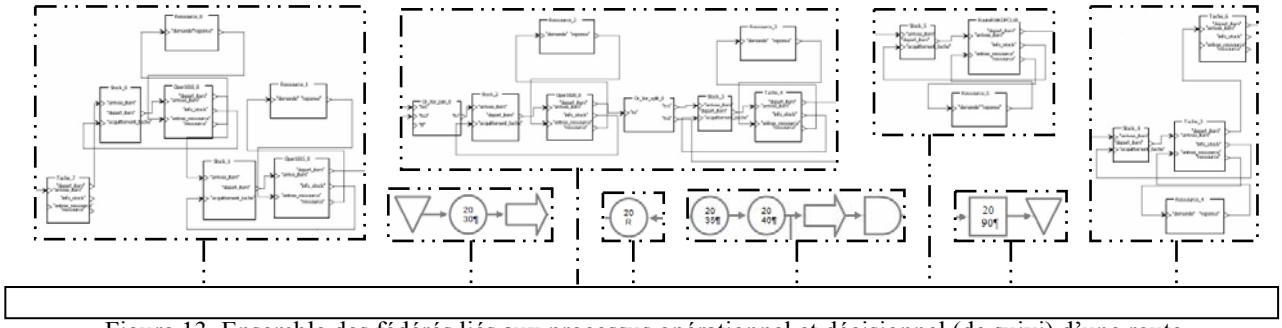

Figure 13. Ensemble des fédérés liés aux processus opérationnel et décisionnel (de suivi) d'une route.

Lors de la phase d'implémentation des concepts HLA dans chaque fédéré, le programme du fédéré reçoit en héritage la classe *FederateAmbassador*. Cet ajout permettra au fédéré d'utiliser les services du RTI, d'implémenter les services de callback, et de pouvoir notamment faire appel au service de demande d'autorisation de traitement du prochain événement. Cet appel *NextMessageRequest*() auprès du RTI permet d'obtenir l'autorisation de traiter le prochain événement local planifié dans le cas d'une simulation dirigée par les événements. Le RTI détermine en fonction du LBTS (Lower Bound on Time Stamp ; HLA version 1.3) et des messages à destination de ce fédéré, s'il peut lui accorder le droit de traiter son prochain événement. S'il autorise le fédéré à traiter le message demandé, il envoie le service de rappel *TimeAdvanceGrant()†*, mais si le fédéré possède des messages en attente de réception avant la date de son prochain événement local, le RTI les lui délivre auparavant avec les services de call back *ReceiveInteraction()†* ou *ReflectAttributesValues()†*.

Lorsque le fédéré reçoit une autorisation ou des messages du RTI, il les fait parvenir à ses successeurs. En retour, il reçoit la nouvelle date de ses modèles successeurs et éventuellement un message de sortie qu'il envoie au RTI avec le service *sendInteraction()*. Si les attributs de l'objet partagé « coordinateur local » ont changé, il en informe la fédération avec le service *sendAttributesUpdates()*.

HLA utilise la notion de Lookahead. Le Lookahead L est la durée au delà de sa date actuelle pendant laquelle un fédéré atteste qu'il n'émettra pas de message. Les fédérés peuvent modifier dynamiquement la valeur de leur Lookahead (en cours d'exécution). Dans le cas d'une augmentation par un fédéré de son Lookahead, le RTI prend en compte directement cette modification. Dans le cas d'une diminution le RTI prend en compte la demande mais elle ne sera effective qu'après que le fédère ait avancé son temps de l'ancienne valeur de son Lookahead, ceci pour respecter la causalité.

Le calcul du Lookahead de chaque fédéré est donc un élément fondamental dans l'exécution d'une simulation distribuée. Dans ce cas, il correspond au délai minimum d'un modèle pour émettre un événement de sortie à partir d'un événement reçu. Ce 'retard' correspond pour les modèles G-DEVS à la prochaine exécution de la fonction de sortie λ(s), donc associée à la plus courte durée de vie des états D(s) qui peut s'écouler à partir de l'état courant. Toutes ces notions fondamentales en simulation distribuée sont largement développées dans (Zacharewicz, 2006).

5. CONCLUSION

Nous avons présenté un environnement de simulation distribuée pour systèmes de production intégrant des processus décisionnels de pilotage et des processus opérationnels de transformation, stockage et inspection. A partir d'une spécification des processus opérationnels selon la norme JIS Z 8206 et des processus de pilotage sous forme de workflows standards, nous générons des modèles distribués G-DEVS. Après avoir montré comment s'effectue la transformation d'un processus en modèles G-DEVS, nous avons expliqué comment est mise en œuvre une simulation globale de l'ensemble de ces modèles via une architecture HLA, qui permet l'interconnexion entre les composants des modèles. Nous avons illustré l'utilisation de cet environnement avec un exemple pris dans le domaine de la micro électronique.

Ces travaux valident la faisabilité de cette approche de simulation distribuée via HLA à l'étude des synchronisations entre lignes de production et mécanismes de pilotage à plusieurs niveaux. L'usage de la simulation pour l'aide à la compréhension du comportement complexe des systèmes de production dans l'industrie micro électronique s'avère ainsi très prometteuse, en s'appuyant ici sur un contexte scientifique formel (G-DEVS), et non sur des applications industrielles au comportement mal maîtrisé.

La spécification de simulation distribuée HLA nous apparaît comme particulièrement adaptée pour la mise en œuvre de la structure de simulation distribuée présentée précédemment. En effet, cette norme permet la réutilisation et l'interopérabilité de composants de simulation sans la nécessité de les recoder. De plus, l'implémentation de la spécification d'interface (RTI) propose des services (tels que la prise en considération du Lookahead) qui peuvent considérablement accélérer l'exécution de la simulation distribuée, tout en assurant le respect des relations de causalité existantes ainsi que la coordination des messages entre les modèles.

Ceci est fondamental si nous envisageons une application de l'approche à la totalité de l'unité de production, constituant ainsi une véritable usine virtuelle travaillant en parallèle de l'usine réelle et servant à tester par anticipation les choix de pilotage. En effet, ceci nous

amène un problème quantitatif lié au nombre très élevé de processus à prendre en considération, et il nous faudra alors s'appuyer sur des mécanismes fiables, robustes et automatiques pour décrire les problèmes de synchronisation inter-processus.
Enfin, HLA permet d'envisager la mise en œuvre de couplages simulation-réalité. En effet, les approches distribuées telles que HLA permettent d'assurer l'interoperabilité et l'inter fonctionnement de logiciels hétérogènes et l'interaction partie simulée partie logiciels temps réel. L'objectif n'est plus alors la performance mais plutôt la communication temps réel entre les différentes entités, toujours dans le respect de la causalité.

Les travaux présentés ici sont financés dans le cadre d'un projet impliquant le Laboratoire des Sciences de l'Information et des Systèmes, le Conseil Général des Bouches du Rhône et ST Microelectronics – Zone industrielle de Rousset 13106 Rousset cedex, France, que nous remercions pour leur soutien

REFERENCES

Defense Modeling and Simulation Office (DMSO). U.S. Dept of Defense. 1998. *High Level Architecture Rules* Version 1.3. IEEE P1516.2, Standard M&S.

Fujimoto R. M., 1998. Time Management in the High Level Architecture. *Transaction of the SCS Simulation*, **71**, (6 December), 388-400.

Fowler J. W. and O. Rose, 2004. Grand Challenges in Modeling and Simulation of Complex Manufacturing Systems. *Simulation*, **80**, (9), pp469-476.

Giambiasi, N., Frydman, C. and Escudé, B. 1995. Hierarchical/Multi-View Modelling and Simulation. *Proceedings of the 7th European Simulation Symposium* Erlangen Nuremberg - Germany.

Giambiasi N., B. Escude and S. Ghosh, 2000. G-DEVS A Generalized Discrete Event Specification for Accurate Modeling of Dynamic Systems. *Transactions of the SCS International*, **17**, (3), 120-134.

Hamri M. E.-A. et G. Zacharewicz, 2007. LSIS-DME: An Environment for Modeling and Simulation of DEVS Specifications. *Proceedings of AIS-CMS International Modeling and Simulation Multiconference*, Buenos Aires - Argentina, February 8-10, pp. 55-60. ISBN 978-2-9520712-6-0

Institute of Electrical and Electronic Engineers. 2001. *High Level Architecture (HLA) - Federate Interface Specification* IEEE Standard for Modeling and Simulation (M&S). std 1516.2-2000, March.

JIS Z 8206: 1982. Graphical symbols for process chart, *Japanese Standards Association*

Pinaton J., Ph. Campion, 2004, De l'efficience de la production par une approche 'Qualité Totale' de type PDCA chez STMicroelectonics, *GDR MACS, Journées de synthèse STP,* Aix-en-Provence, octobre.

Pujo P., M. Pedetti and F. Ounnar, 2004. Pilotage proactif des lignes de production kanban par modélisation DEVS et simulation temps réel. *Proceedings of MOSIM'04*, *5° Conférence Francophone de MOdélisation et SIMulation*, Nantes, France, 1-3 septembre.

Ramamurthi V., M. E. Kuhl, and K. D. Hirschman, 2005. Analysis of production control methods for semiconductor research and development fabs using simulation. *Proceedings of the 37° Winter Simulation Conference,* Orlando, California, pp2177-2185, ISBN:0-7803-9519-0

Riccardi V., P. Pujo and C. Frydman, 2004. DEVS Modelling For The Proactive Control By Simulation Of Kanban Production Lines. *Proceedings of the International Workshop on Modeling & Applied Simulation* (*MAS 2003*), Italy, 2-4 octobre 2003.

Rose O., 2000. General simulation applications in semiconductor manufacturing: why do simple wafer fab models fail in certain scenarios? *Proceedings of the 32° Winter Simulation Conference,*Orlando, Florida, pp1481-1490, ISBN:0-7803-6582-8

Song H.S. and T.G. Kim, 1994. The DEVS framework for discrete event systems control. *Proceedings of the 5th Conference on AI, Simulation and Planning in High Autonomous Systems*, Gainesville, USA, pp228-234.

Van der Aalst W., *Workflow management: models, methods, and systems*, MIT Press, Cambridge, MA, 2002

Workflow Management Coalition, 1999. *Terminology & Glossary*. WfMC-TC-1011, 3.0, Feb.

Workflow Management Coalition, 2005. *Workflow Process Definition Interface — XML Process Definition Language (XPDL)*. WFMC-TC-1025, Oct.

Zacharewicz G., M. E.-A. Hamri, W. Trojet, C. Frydman and N. Giambiasi, 2006. G-DEVS / HLA Environment for Distributed Simulations of Workflows. *Proceedings of the International Conference on Modeling and Simulation - Methodology, Tools, Software Applications (M&S-MTSA'06*), SCS, pp. 206 - 211, Calgary, Alberta, Canada, July 31 - August 02

Zacharewicz G., N. Giambiasi and C. Frydman, 2006. Lookahead Computation in G-DEVS/HLA Environment. *Simulation News Europe Journal (SNE)* special issue 1 "Parallel and Distributed Simulation Methods and Environments", **16**, (2), 15-24, ISSN 0929-2268

Zacharewicz G., 2006. *Un environnement G-DEVS/HLA : application à la modélisation et simulation distribuée de WorkFlow*, Thèse de Doctorat en Sciences, Université Paul Cézanne, Marseille, 30 novembre.

Zacharewicz G. and M. E.-A. Hamri, 2007. Flattening G-DEVS / HLA structure for Distributed Simulation of Workflows. *Proceedings of AIS-CMS International Modeling and Simulation Multiconference*, pp. 11-16, Buenos Aires - Argentina, February 8-10, ISBN 978-2-9520712-6-0

Zeigler B.P., 1973. *Theory of Modelling and Simulation*. Ed. John Wiley, New York.

Zeigler B.P., 1984. *Multifacetted Modelling and Discrete Event Simulation*. Academic Press, London.

Zeigler B.P., H. Praehofer and T. G. Kim, 2000. *Theory of Modeling and Simulation.* 2nd Edition, Academic Press, New York, US