Conception formelle de documents hypermedias portables

Roberto Willrich

Année 1996

Thèse

Présentée au Laboratoire d'Analyse et
D'Architecture des Systèmes du C.N.R.S
en vue de l'obtention du titre de DOCTEUR
DE L'UNIVERSITE PAUL SABATIER DE
TOULOUSE
SPECIALITE: Informatique Industrielle

par

Roberto WILLRICH

CONCEPTION FORMELLE DE DOCUMENTS HYPERMEDIAS PORTABLES

Soutenance le 30 Septembre 1996 devant le jury :

MM.	Claude BETOURNE	- *Examinateur*
	Richard CASTANET	- *Rapporteur*
	Jean-Pierre COURTIAT	- *Examinateur*
	Michel DIAZ	- *Directeur de Thèse*
	Hervé GUYENNET	- *Examinateur*
	Eric HORLAIT	- *Rapporteur*
	Guy JUANOLE	- *Président*
	Alain LEGER	- *Examinateur*
	Jean François SCHMIDT	- *Examinateur*

Rapport LAAS n° 96376
Thèse préparée au Laboratoire d'Analyse et d'Architecture
des Systèmes du CNRS

7, avenue du Colonel Roche
31077 TOULOUSE Cedex

Avant Propos

Les travaux présentés dans ce mémoire ont été effectués au Laboratoire d'Analyse et d'Architecture des Systèmes du Centre National de la Recherche Scientifique, dirigé par Monsieur le Professeur A. COSTES que je remercie pour son accueil.

J'exprime ma profonde gratitude à M. DIAZ, Directeur de Recherche au CNRS et responsable du groupe Outils et Logiciels pour la Communication, qui m'a accueilli dans ce groupe et a accepté la lourde charge de diriger mes rechercher. Je le remercie pour les conseils, l'enthousiasme et la confiance qu'il m'a renouvelée à maintes occasions.

J'exprime mes plus vifs remerciements à:

Monsieur C. BETOURNE, Professeur à l'Université Paul Sabatier,
Monsieur R. CASTANET, Professeur à l'Université de Bordeaux I,
Monsieur J.P. COURTIAT, Chargé de Recherche au CNRS,
Monsieur M. DIAZ, Directeur de Recherche au CNRS,
Monsieur H. GUYENNET, Professeur de l'Université de Franche-Comté,
Monsieur E. HORLAIT, Professeur de l'Université Paris VI
Monsieur G. JUANOLE, Professeur à l'Université Paul Sabatier,
Monsieur A. LEGER, Ingénieur du CCETT
Monsieur J.F. SCHMIDT, Ingénieur de l'Airbus Training,

pour l'honneur qu'il me font en participant au Jury et particulièrement à Messieurs R. CASTANET et E. HORLAIT pour avoir accepté la charge de rapporteur, et également à Monsieur G. JUANOLE pour me faire l'honneur de présider ce Jury. Je remercie aussi Monsieur P. RULLIER d'Airbus Training pour avoir analysé notre travail.

J'adresse un remerciement tout particulier à Messieurs P. SENAC, P. DE SAQUI-SANNES et à Madame V. BAUDIN THOMAS, pour leur collaboration aux travaux effectués dans le cadre de cette thèse et pour leur amitié.

Mes remerciements s'étendent à l'ensemble de mes collègues du groupe OLC et de l'ENSICA, qui m'ont assuré de leur soutien amical et de leur bonne humeur.

Toute ma reconnaissance à l'ensemble du personnel technique/administratif du LAAS pour leur soutien dans la réalisation de mon travail de thèse.

Finalement, je tiens à remercier le CNPq pour le support financier qui a rendu possible mon séjour en France.

Table des Matières

1. Introduction 1

2. Modélisation de documents hypermédias 5

2.1. Hypermédia 6
2.1.1. Documents hypertextes 6
2.1.2. Documents multimédias 7
2.1.3. Documents hypermédias 7
2.2. Echange des documents hypermédias 8
2.3. Les exigences d'un modèle hypermédia 9
2.3.1. Structure du contenu 10
2.3.2. Structure conceptuelle 11
2.3.2.1. Les composants et les groupes de composants 11
2.3.2.2. Les chemins de parcours et les structures des informations 12
2.3.2.3. Composition temporelle du document 13
2.3.3. Structure de présentation 15
2.3.3.1. Spécification des caractéristiques de présentation des composants 16
2.3.3.2. Composition spatiale du document 16
2.3.4. Interactions 16
2.3.5. Modélisation de la structure conceptuelle 17
2.3.5.1. Lignes temporelles 17
2.3.5.2. Composition via des points de référence 18
2.3.5.3. Composition hiérarchique 18
2.3.5.4. Réseaux de Petri 19
2.3.6. Présentation des documents hypermédias 19
2.4. Modèles multimédias et hypermédias 20
2.4.1. Modèle de référence d'hypertexte Dexter 21
2.4.2. Amsterdam Hypermedia Model (AHM) 22
2.4.3. La norme HyTime 24
2.4.4. Modèle de composition hiérarchique de [Hudson, 93] 25
2.4.5. Modèles basés sur l'approche de composition via des points de références 27
2.4.5.1. L'approche orientée objet de [Steinmetz, 90] 27
2.4.5.2. Le système MODE 28
2.4.5.3. Le système Firefly 29
2.4.6. Systèmes basés sur les organigrammes 31
2.4.6.1. IconAuthor 32
2.4.6.2. Eyes M/M 33
2.4.7. Modèles basés sur les réseaux de Petri 33
2.4.7.1. OCPN 33
2.4.7.2. OCPN avec caractérisation des présentations 35
2.4.7.3. A-OCPN 36
2.4.7.4. XOCPN 36
2.4.7.5. RTSM 38
2.4.7.6. Trellis 39

2.4.7.7. MORENA 40
2.4.7.8. MHPN 41
2.4.7.9. RdPFT 42
2.4.7.10. RdPHFT 44
2.5. Conclusion 46

3. La norme MHEG 53

3.1. Les classes MHEG, rt-objets et canaux 54
3.1.1. Classe ACTION 55
3.1.2. Classe LINK 57
3.1.3. Classe SCRIPT 59
3.1.4. Classe CONTENT 59
3.1.5. Classe MULTIPLEXED CONTENT 59
3.1.6. Classe COMPOSITE 60
3.1.7. Classe CONTAINER 61
3.1.8. Classe DESCRIPTOR 61
3.2. La machine MHEG 61
3.3. La synchronisation dans MHEG 62
3.3.1. Pré-requis de synchronisation identifiés par MHEG 63
3.3.2. Niveaux de synchronisation dans MHEG 63
3.4. Exemple de représentation MHEG 63
3.5. Conclusion 65

4. Le modèle RdPHFT Interprété (RdPHFT-I) 67

4.1. Introduction au modèle RdPHFT-I 68
4.1.1. Structure du contenu 68
4.1.2. Structure de présentation 70
4.1.2.1. Spécification des caractéristiques de présentation 71
4.1.2.2. Spécification des canaux 72
4.1.3. Structure conceptuelle 73
4.1.3.1. Interprétation du modèle RdPHFT 73
4.1.3.2. Structuration du niveau de Synchronisation Logique 74
4.3. Définition formelle du modèle RdPHFT-I 75
4.4. Les apports du modèle RdPHFT-I 78
4.4.1. L'édition de la durée de présentation 78
4.4.2. Attente de synchronisation 78
4.4.3. Définition des présentations alternatives 79
4.4.4. Détection des conflits d'utilisation des ressources partagées 79
4.4.5. Description du document 79
4.5. Limitations du modèle RdPHFT(-I) 80
4.5.1. Limitation à une composition de présentations 80
4.5.2. Relations conditionnelles 80
4.5.3. Granularité de synchronisation 81
4.5.4. Représentation des actions dynamiques 82
4.5.5. Méthodes d'interaction du modèle RdPHFT-I 82

4.5.6. Les liens contexte 83
4.5.7. Les synchronisations continues et intra-média 83
4.6. Exemple de spécification RdPHFT-I 84
4.6.1. Spécification de la structure conceptuelle 86
4.6.2. Spécification de la structure du contenu 90
4.6.3. Spécification de la structure de présentation 91
4.7. Conclusion 91

5. Traduction RdPHFT-I en MHEG 93

5.1. MHEG et le modèle RdPHFT 93
5.1.1. Correspondants MHEG des éléments du modèle RdPHFT 94
5.1.1.1. Les places 94
5.1.1.2. Les transitions et les arcs 95
5.1.1.3. Le contrôle de la validité temporelle 96
5.1.2. Représentation des synchronisations MHEG en RdPHFT 96
5.1.2.1. Synchronisation élémentaire 97
5.1.2.2. Synchronisation enchaînée 97
5.1.2.3. Synchronisation cyclique 97
5.1.2.4. Synchronisation conditionnelle 97
5.1.3. Viabilité de la traduction RdPHFT/MHEG 97
5.2. La procédure de traduction RdPHFT-I en MHEG 98
5.2.1. Représentation MHEG de la structure du contenu 99
5.2.2. Représentation MHEG de la structure conceptuelle et de présentation 101
5.2.2.1. Attribut Description 101
5.2.2.2. Attribut Elements 101
5.2.2.3. Attribut Availability-Start-Up 102
5.2.2.4. Attribut Rt-Availability-Start-Up 102
5.2.2.5. Attribut Specific-Behaviour 103
5.2.3. Représentation MHEG de la description d'un document hypermédia 108
5.3. Exemple de traduction RdPHFT en MHEG 108
5.4. Conclusion 113

6. Vers la conception de documents hypermédias : méthodologie et environnement 115

6.1. Vers la conception de documents hypermédias MHEG 115
6.1.1. L'environnement de développement de documents hypermédias MHEG 116
6.1.1.1. Éditeur 117
6.1.1.2. Vérificateur 118
6.1.1.3. Simulateur 118
6.1.1.4. Traducteur 118
6.1.2. Vers la présentation des représentations MHEG générées 120
6.2. Vers l'interprétation Java de spécifications RdPHFT-I 121
6.2.1. Le modèle RdPHFT-I interprétable en Java 122
6.2.2. L'environnement Java de développement de documents hypermédias 122
6.3. Conclusion 124

7. Conclusion 127

8. Bibliographie 131

Annexe A. Les classes du modèle RdPHFT-I 139

A.1. Les Classes Données 139
A.2. Les Classes Canal 141
A.3. Les Classes Présentation 141

Annexe B. L'interpréteur Java RdPHFT-I 145

B.1. Format du fichier d'entrée de l'interpréteur RdPHFT-I 146
B.1.1. Description de la grammaire 146
B.1.2. Format du fichier RdPHFT-I 146
B.1.3. Exemple 147
B.2. L'interpréteur RdPHFT-I 150
B.2.1. Module Ordonnanceur 152
B.2.2. Module PrésentationTexteImage 153
B.2.3. Module PrésentationAudio 154
B.2.4. Module PrésentationBouton 155
B.2.5. Module PrésentationAnimationGraphique 156
B.3. Exemple de structure d'interprétation 157
B.4. Dictionnaire des données 159

Chapitre 1

Introduction

Le développement technologique des systèmes d'information et de communication à haute vitesse a permis l'apparition de nouvelles applications dans le domaine des systèmes répartis. Une des principales tendances dans ce contexte est l'intégration de différents médias de présentation, comme le texte, la voix, les images et les vidéos dans un très vaste domaine d'applications informatiques distribuées (par exemple, téléenseignement, publications électroniques, travaux coopératifs). Ces systèmes sont appelés systèmes multimédias et hypermédias. Dans ce travail, nous considérons qu'un système hypermédia est un système multimédia dans lequel les informations monomédias et multimédias sont accédées et présentées à l'aide des mécanismes de navigation basés sur des liens. Les documents hypermédias définissent les structures des informations hypermédias.

Les informations utilisées, stockées et échangées par des applications multimédias et hypermédias représentent un investissement important. Ainsi, il est vital que ces informations soient utilisables dans un monde en rapide évolution tant au niveau des systèmes que des technologies. Pour cela, il est nécessaire de définir un langage commun pour tous les systèmes de communication multimédias/hypermédias qui apparaîtront dans les années à venir [Colaitis, 92].

Les systèmes hypermédias existants sont malheureusement basés sur des concepts et des modèles différents, et apparaissent à ce titre comme des systèmes fermés. Ainsi les composants hypermédias créés à partir d'une solution ad-hoc dans un système particulier ne peuvent être ni réutilisés ni transférés facilement. Afin de résoudre ce problème, plusieurs modèles et formats d'échange multimédias/hypermédias ont été proposés; ils définissent des concepts syntaxiques et sémantiques permettant de réaliser des systèmes multimédias et hypermédias ouverts. En particulier, le modèle de référence hypertexte Dexter [Halasz, 90] est une tentative de capturer, de façon formelle et informelle, les abstractions importantes rencontrées dans un vaste domaine de systèmes hypermédias existants et futurs.

En parallèle, des normes internationales pour la représentation d'informations multimédias et hypermédias ont été développées : MHEG [ISO 13522], HyTime [Newcomb, 91] et HyperODA [Appelt, 93]. Ces normes internationales sont destinées à permettre le transfert et le traitement de ces types d'informations dans un système ouvert. Par exemple, la future norme internationale ISO/IEC MHEG (*Multimedia Hypermedia Expert Group*) [ISO 13522] définit un codage (une représentation) d'informations multimédias et hypermédias, permettant l'échange de plusieurs types de médias, des structures repré-

sentant la composition temporelle et spatiale de ces médias ainsi que les interactions lors de leurs présentations.

A cause de la complexité structurelle des systèmes hypermédias, leur conception ne peut pas être abordée efficacement et aisément en utilisant directement les normes internationales précédemment citées. Ainsi, un modèle logique de plus haut niveau doit être utilisé pour faciliter la description des documents hypermédias. Le travail développé dans ce mémoire de thèse propose un tel modèle de spécification de documents hypermédias qui permet ensuite la génération automatique de représentations MHEG.

Comme dans le cas général des systèmes et applications répartis, la spécification et la validation sont deux étapes fondamentales du cycle de développement des documents hypermédias. Ainsi, ce développement requiert l'utilisation d'un modèle définissant un formalisme capable de spécifier et de valider le comportement logique (la navigation basée sur les liens) et temporel du document. La spécification de documents hypermédias par l'utilisation de modèles informels ou semi formels peut être source d'ambiguités et ne permet pas l'utilisation d'outils permettant l'analyse des propriétés de la spécification. A l'inverse, les modèles formels permettent la construction d'une spécification précise et non ambiguë. De plus, seule la représentation formelle peut mener à des analyses réellement significatives d'un système, et seules les techniques d'analyse formelle sont adéquates pour vérifier précisément les propriétés importantes.

Il existe plusieurs modèles formels de systèmes hypermédias ([Stotts, 90], [Halasz, 90]) et multimédias ([Little, 90], [Diaz, 93]). Mais il n'y a pas à ce jour de modèle formel permettant l'expression facile et complète, à travers une approche unifiée, des structures des systèmes hypermédias répartis. A partir de cette constatation et prenant en compte les concepts établis par le modèle de référence hypertexte Dexter et le modèle multimédia RdPFT (*Réseaux de Petri à Flux Temporels*), [Diaz, 93], [Sénac 95a] et [Sénac, 96a] proposent un nouveau modèle formel, nommé *Réseaux de Petri Hiérarchisés à Flux Temporels* (RdPHFT). Ce dernier permet une spécification formelle unifiée et précise de la synchronisation logique et temporelle à l'intérieur des systèmes hypermédias distribués. Ce modèle n'a pas pour objectif la description complète de documents hypermédias; par exemple, il ne fournit pas les moyens de spécifier les informations d'accès ou les caractéristiques spatiales et sonores des présentations qui composent un document.

Ce mémoire propose d'abord une version interprétée du modèle RdPHFT, appelée RdPHFT-I, qui étend le modèle RdPHFT afin de spécifier les données et les caractéristiques spatiales et sonores des présentations. Le terme «interprétation» signifie que l'on a étendu la sémantique des places du modèle RdPHFT. Cette sémantique a été définie à partir des concepts proposés par la norme MHEG, cela afin de permettre la représentation de toutes les informations nécessaires à la génération automatique de représentations MHEG.

Au-delà du modèle RdPHFT-I, nous proposons une méthodologie de prototypage de documents hypermédias dans laquelle la spécification et l'analyse des documents sont faites à l'aide du modèle RdPHFT-I et l'implantation est supportée par la production automatique de représentations MHEG. Pour la présentation de ces représentations MHEG, il est nécessaire d'utiliser des systèmes supportant des applications MHEG, à savoir des interpréteurs MHEG (des exemples d'interpréteur MHEG sont présentés dans [Meyer-Boudnik, 95]). Le but de ce travail n'est pas d'obtenir un modèle permettant la spécification de tous les documents hypermédias représentables en MHEG, mais seulement un sous-ensemble structuré et modulaire dont le comportement peut être spécifié au moyen de réseaux RdPHFT.

L'objectif de cette méthodologie est de fournir une base formelle solide pour la spécification et l'analyse de documents hypermédias, et pour la traduction de cette spécification en une représentation MHEG. Un environnement de développement interactif, incluant un éditeur graphique, un analyseur et un traducteur MHEG, est nécessaire à la création de documents hypermédias MHEG. Nous avons développé un tel environnement de développement de représentations MHEG.

Enfin, dans le contexte du développement formel de documents hypermédias, nous proposons une variante de la méthodologie précédente permettant la création de documents hypermédias interprétables par une application Java. Java [Sun, 95] est un langage orienté-objet et multi-tâches permettant la réalisation des applications hypermédias sur Internet. Une application Java peut être transférée via le réseau et interprétée par une machine virtuelle indépendante d'un système d'exploitation. Ainsi le langage Java permet de développer des applications multimédias portables. Nous avons développé une application Java permettant de décrire dans la forme d'un RdPHFT-I un document, puis de l'analyser et de l'interpréter (de le présenter).

Les travaux présentés dans ce mémoire ont été conduits au sein du groupe de recherche Outils et Logiciels pour la Communication (OLC) du Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS (LAAS-CNRS). Les thèmes de recherche traditionnellement abordés par le groupe OLC portent essentiellement sur la conception formelle de protocoles de communication. Les limitations des protocoles actuellement utilisés pour le transport d'informations multimédias ont amené le groupe OLC à conduire une réflexion sur les nouvelles exigences induites en matière protocolaire par les profils applicatifs multimédia et hypermédias. Il est apparu que cette étude devait en tout premier lieu débuter par une définition claire et non ambiguë des caractéristiques fondamentales des applications multimédias et hypermédias. L'expertise et la longue pratique du groupe OLC dans le domaine des techniques de spécification formelles a conduit ces travaux vers l'utilisation des méthodes formelles afin de décrire précisément les problématiques du multimédia et de l'hypermédia. En particulier, ces travaux ont abouti à la définition du modèle RdPHFT [Sénac, 96b]. Bien sûr, le passage de la description formelle des documents hypermédias à leur implémentation est un problème critique devant être résolu afin de garantir la validité et la qualité des applications multimédias et hypermédias. C'est pourquoi dans le cadre de cette thèse, nous sommes consacrés à la définition et à la mise en œuvre d'une approche formelle de conception de documents hypermédias portables basée sur le modèle RdPHFT.

Cette thèse a été réalisée dans le cadre du projet CESAME (Conception Formelle de Systèmes Distribués Coopératifs Haut-débit Multimédia) [Diaz, 94a], financé par France-Télécom et associant le CNET, le CCETT et le CNRS. Le choix de la norme MHEG pour la représentation de documents hypermédias a été pris conformément au souhait émis par France Télécom.

Ce mémoire est organisé de la façon suivante :

Le chapitre 2 est consacré à la problématique de la modélisation de documents hypermédias. En premier lieu, nous présentons les concepts généraux des systèmes hypertextes, multimédias et hypermédias. Ensuite, nous introduisons les principales exigences qui doivent être satisfaites par un modèle hypermédia. Sur cette base, nous présentons et critiquons les principaux modèles multimédias et hypermédias. En particulier, nous avons vérifié que le modèle RdPHFT satisfait presque tous les besoins d'un modèle de composition logique et temporelle de documents hypermédias. Néanmoins, le modèle RdPHFT ne permet pas la description complète de documents hypermédias.

Le chapitre 3 présente les concepts MHEG utilisés pour la définition du modèle RdPHFT-I et la présentation de la procédure de traduction d'une spécification RdPHFT-I en une représentation MHEG.

Le chapitre 4 propose le modèle RdPHFT-I, qui étend le modèle RdPHFT afin de permettre la spécification complète de documents hypermédias. Cette extension a été définie à partir des concepts proposés

par la norme MHEG, cela afin de permettre la génération automatique de représentations ouvertes MHEG à partir du modèle RdPHFT.

Le chapitre 5 introduit une procédure de traduction d'un réseau RdPHFT-I vers une représentation MHEG. La définition de cette procédure permettra la génération automatique d'une représentation finale (un code interprétable par une machine) d'un document hypermédia à partir de sa spécification RdPHFT-I. Cette représentation pourra donc être utilisée dans un système hypermédia ouvert.

Le chapitre 6 présente une méthodologie formelle de prototypage de documents hypermédias dans laquelle la spécification et l'analyse des documents sont faites à l'aide du modèle RdPHFT-I et l'implantation est supportée : soit par la production automatique de représentations MHEG qui peut être échangée et présentée dans un système ouvert; soit par l'interprétation de la spécification RdPHFT-I par une application Java qui peut être transférée sur Internet et présentée sous plusieurs systèmes d'exploitation. Ce chapitre présente aussi deux outils mettant en œuvre la méthodologie de développement de documents hypermédias qui ont été implémentés dans le cadre de ce travail : l'outil RdPHFT-I MHEG permettant la génération de représentations MHEG; et l'outil Java RdPHFT-I permettant l'interprétation de spécifications RdPHFT-I.

Enfin, le chapitre 7 conclut ce travail.

Chapitre 2

Modélisation de documents hypermédias

Les systèmes multimédias et hypermédias répartis sont très complexes, principalement à cause du grand nombre de fonctionnalités qui leur sont associées. Une liste partielle de ces fonctionnalités englobe : la gestion d'objets répartis, l'architecture de documents hypermédias, les formats d'échange d'informations, les langages de script, les formats de données médias, les outils, les services des systèmes d'exploitation, les protocoles et les architectures de communication. Dans [Buford, 94a], l'auteur propose un cadre de travail pour l'organisation et la définition des interrelations entre les différentes fonctionnalités des systèmes multimédias et hypermédias. Ce cadre de travail est composé de quatre modèles :

- **Modèle d'Information Multimédia** : il comprend la modélisation des données pour le stockage, l'accès et le traitement d'informations multimédias. Un modèle multimédia doit permettre la spécification des composants d'un document, des associations et des relations entre ces composants et des interactions avec l'utilisateur.
- **Modèle de Traitement Multimédia Réparti** : il englobe les services du système d'exploitation et les outils de création d'applications multimédias/hypermédias. [Buford, 94a] présente un modèle à 7 couches (application, toolkits, orchestration, services média, services de système et service de matériel) contenant les fonctionnalités nécessaires au traitement d'applications multimédias réparties.
- **Modèle de Conférence Multimédia** : il fournit les abstractions pour la communication multi-destinations, le transfert en temps-réel, le courrier électronique et la téléphonie avec vidéo.
- **Modèle de Multiservices de Communication** : il contient le modèle de communication avec une architecture de réseau, de protocoles de communication et d'interfaces.

Dans ce travail, nous traitons uniquement des modèles d'informations hypermédias, c'est-à-dire que nous étudions la modélisation des composants de documents hypermédias et de leur composition. La section 2.1 introduit les principaux concepts hypermédias. La section 2.2 présente les modes d'échange de documents hypermédias. La section 2.3 introduit les principales exigences qui doivent être satisfaites par un modèle hypermédia. Ensuite, la section 2.4 présente les principaux modèles multimédias et hypermédias. Enfin, la section 2.5 présente les conclusions de ce chapitre.

2.1 Hypermédia

L'hypermédia combine différents types de médias de présentation offerts par le multimédia avec la structure d'information offerte par l'hypertexte [Hardman, 94].

2.1.1 Documents hypertextes

Un **document hypertexte** est une structure d'information organisée de façon non linéaire : les données sont stockées dans un réseau de nœuds connectés par des liens (voir la base de données hypertextes de la figure 2.1) :

- Les **nœuds** contiennent des unités d'information composées de textes et d'informations graphiques. En général, un nœud représente un concept ou une idée, exprimés d'une façon textuelle ou graphique.
- Les **liens** définissent des relations logiques (ou sémantiques) entre les nœuds, c'est-à-dire entre les concepts et les idées. La notion d'**ancre**, introduite par le modèle de référence hypertexte Dexter [Halasz, 90] (voir section 2.4.1), permet la spécification d'une partie de l'information qui sera source ou destination d'un lien. Une ancre hypertexte spécifie une séquence de caractères dans le texte ou le nœud entier. Les ancres source et destination d'un lien peuvent être définies dans différents nœuds ou dans un même nœud.

En général, quand le lecteur d'un document hypertexte clique sur une ancre source, le lien est suivi, causant la présentation de l'ancre destination. Ainsi, le lecteur peut **naviguer** dans le document à travers les ancres et les liens. Parfois, un nœud est aussi défini comme une unité de navigation.

Il y a d'autres mode d'accès aux informations que la navigation par les liens hypertextes. Par exemple, à partir d'un mécanisme de haut niveau, le lecteur peut demander une recherche par mots-clefs, aller directement à un nœud spécifique, ou parcourir la liste de nœuds précédemment visités.

La figure 2.1 présente une correspondance entre les informations et les ancres sur l'écran, et les nœuds, les liens et les ancres dans la base de données hypertexte. Au début, le contenu du nœud *LAAS* a été présenté. Ensuite, le lien ancré sur la phrase «*Comment se rendre au LAAS*» dans le nœud *LAAS* a été suivi, causant la présentation de l'information associée au nœud *Itinéraire*.

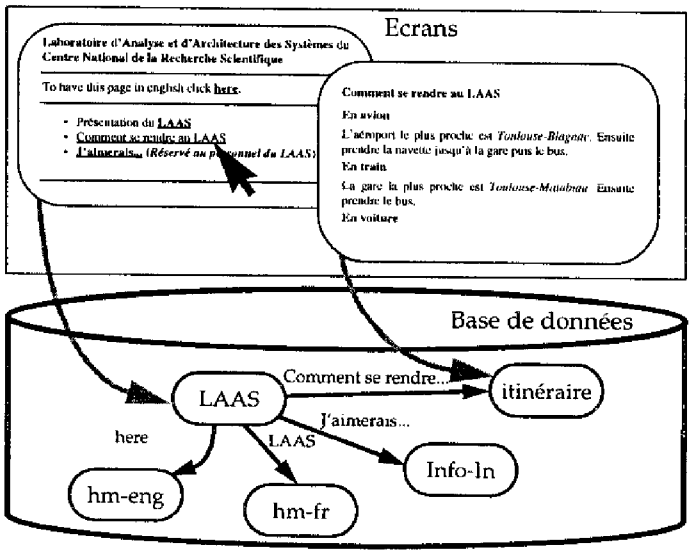


Figure 2.1 Base de données hypertextes

Comme les textes et les images sont des informations statiques (l'information n'évolue pas dans le temps), la notion de temps n'est pas utilisée lors de la spécification des documents hypertextes classiques. Dans l'hypertexte, le temps de présentation des informations est déterminé par le lecteur du document en suivant les liens.

2.1.2 Documents multimédias

Un **document multimédia** peut être vu comme une spécification des activités qui sont définies pour coordonner la présentation d'une collection de composants (figure 2.2). Ces composants peuvent être constitués de différents types de médias avec des caractéristiques temporelles différentes [Steinmetz, 90] : les **médias dépendants du temps** ou **dynamiques** (ou encore continus) qui évoluent dans le temps, comme les séquences vidéo et audio, ou les animations graphiques; et les **médias indépendants du temps** ou **statiques** (ou encore discrets), comme les textes, les graphismes et les images[1]. De plus, ces composants peuvent être stockés dans différents sites d'un système réparti.

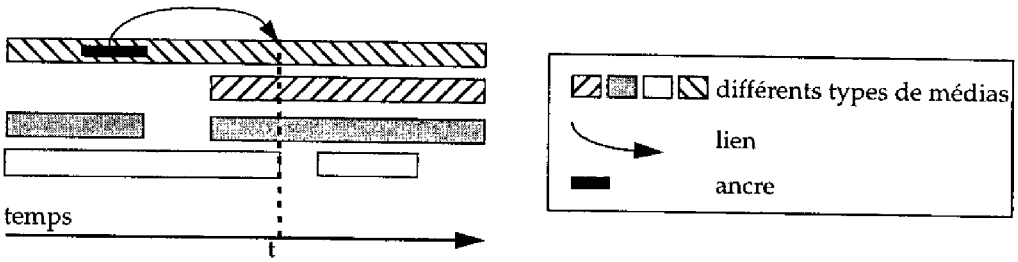


Figure 2.2 Document multimédia [Hardman, 94]

Pour l'orchestration de la présentation des composants d'un document multimédia, l'auteur peut définir la position de départ et de fin des présentations, et/ou les relations conditionnelles et temporelles entre et à l'intérieur des présentations (voir section 2.3.3). Cette description de l'ordre temporel relatif de présentation des composants du document est appelée **scénario multimédia**.

Les documents multimédias peuvent être interactifs. Typiquement, il y a deux méthodes d'interaction à partir desquelles le lecteur peut contrôler la présentation [Hardman, 94] :

- Une méthode de contrôle de la présentation, qui permet d'ajuster la référence de temps utilisée pour la présentation. Avec ce mécanisme, le lecteur peut arrêter, répartir, faire l'avance ou le retour rapide (et quelques fois la recherche) de la présentation du document, cela à partir d'une interface similaire au contrôle d'un magnétoscope.
- Une méthode d'interaction similaire à un lien hypertexte rudimentaire défini par une ancre et un lien. En suivant ce lien, la présentation «saute» vers une autre partie du document. Cette méthode d'interaction est illustrée dans la figure 2.2 : quand l'utilisateur clique sur l'ancre, la présentation saute vers le point t.

2.1.3 Documents hypermédias

Un **document hypermédia** est une combinaison de documents hypertextes et de documents multimédias (illustré par la figure 2.3). Il représente une évolution naturelle de l'hypertexte, dans laquelle les notions ou les concepts des nœuds hypertextes peuvent maintenant être exprimés par différents types de médias de présentation. L'inclusion des données multimédias augmente le pouvoir d'expression de l'information contenue dans le document et rend la présentation de celui-ci plus attractive et réaliste.

1. Lors de la composition du document, un certain caractère temporel peut être associé aux médias statiques. Par exemple, l'auteur peut associer une durée de présentation à une information textuelle.

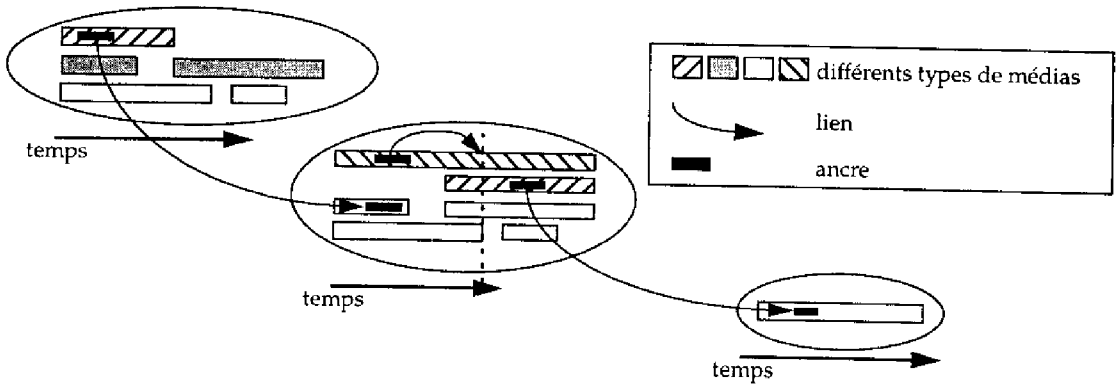


Figure 2.3 Document hypermédia

Comme dans les documents hypertextes, les ancres permettent la spécification d'une partie de la présentation d'un média qui sera source ou destination d'un lien hypermédia. Par exemple, une ancre peut être une séquence de trames vidéo ou une région d'une image.

Les liens hypermédias peuvent être temporisés. Ce type de lien est activé pendant un intervalle temporel et il peut être suivi automatiquement en fonction des contraintes temporelles. Ce type de lien introduit la notion de systèmes hypermédias actifs [Stotts, 90].

L'inclusion des données multimédias dans les structures hypertextes introduit un aspect temporel pour la spécification de documents hypermédias. Ainsi, il est nécessaire qu'un modèle hypermédia fournisse des mécanismes permettant l'intégration temporelle d'une variété de médias statiques ou dynamiques. Enfin, un modèle hypermédia doit fournir les mécanismes de description des interactions possibles avec le lecteur du document, c'est-à-dire qu'il doit permettre la définition des liens hypermédias, extension des liens hypertextes (avec un caractère temporel), et des autres méthodes d'interaction.

2.2 Echange des documents hypermédias

Les documents hypermédias sont situés dans des serveurs multimédias, dans lesquels des fragments d'information stockés par différents serveurs multimédias (n'importe où dans le monde mais aussi dans le disque local) peuvent être accédés par les systèmes clients (figure 2.4). Les serveurs multimédias en général fournissent la capacité de stockage et de capture des informations multimédias, et peuvent être dédiés à un type de média (les serveurs d'images, les serveurs de vidéos, etc.).

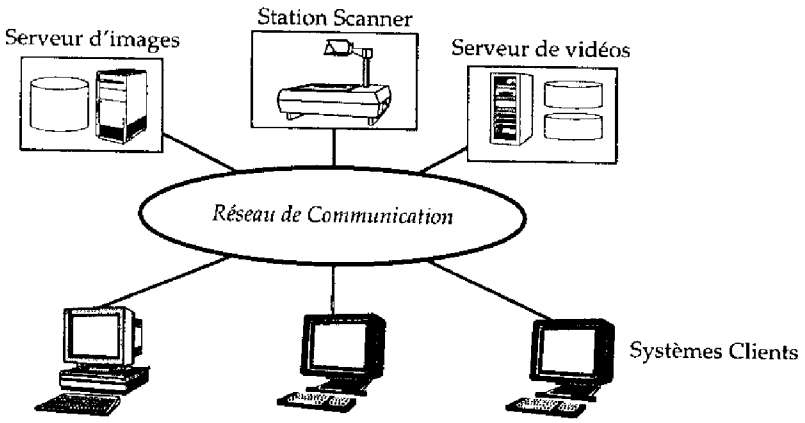


Figure 2.4 Clients et serveurs multimédias

Techniquement, le transfert des informations multimédias et hypermédias peut être réalisé par deux modes de transmission différents [Fluckiger, 95] :

- **Transmission asynchrone ou télé-chargement** : le document (ou une partie) est transféré sur le réseau de communication comme un fichier complet ou un ensemble de fichiers, et délivré au système client où le document sera présenté. Ce processus est asynchrone, la présentation étant effectuée à l'instant choisi par le système client ou par l'utilisateur.
- **Transmission en temps-réel et présentation synchronisée** : le document, ou une parties, est transféré d'un serveur et présenté simultanément, après un délai de transmission et de traitement, sur le système client. Ainsi, les textes, les images, ou les séquences sonores sont injectés dans le réseau sous la forme de séquences binaires, resynchronisés et présentés dans le système récepteur.

Lors de la présentation d'un document hypermédia, ces deux types de transmission peuvent être utilisés pour l'accès et la présentation des informations. Par exemple, un document peut être composé d'un ensemble de textes, d'images et de séquences audio et vidéo. Si le système client n'a pas la capacité de stockage permettant le télé-chargement complet du document, ce dernier doit être divisé en parties. Ces parties doivent être transférées indépendamment, quoi qu'elles puissent être liées par des relations logiques et temporelles. Ainsi, une partie initiale peut être télé-chargée dans le système client, et quand le lecteur sélectionne la présentation d'une autre partie qui n'est pas disponible localement, le système client doit demander la transmission asynchrone ou en temps-réel de cette partie.

2.3 Les exigences d'un modèle hypermédia

Un modèle hypermédia induit une technique de description de documents hypermédias. Basées sur la norme ISO ODA[2] (*Office Document Architecture*) [ISO 8613], plusieurs approches ([Meghini, 91], [Hoepner, 92], [Karmouch, 93], [Kerhervé, 93]) divisent la description des documents multimédias et hypermédias en trois parties : la **structure logique**, qui décrit les différentes parties logiques du document, ou composants, et leurs relations logiques; la **structure de présentation**, qui décrit comment, où et quand les différents composants seront présentés; et la **structure du contenu**, qui décrit les informations qui constituent les composants.

[Fluckiger, 95] affirme que la partition de la description des documents hypermédias dans ces trois structures offre plusieurs avantages. La séparation entre la structure logique et de présentation, par exemple, permet la réutilisation de la structure logique lors de la présentation du document sur des dispositifs différents, où seule une simple modification ou adaptation de la structure de présentation doit être réalisée; et la séparation entre la description de la présentation et du contenu à présenter est nécessaire parce que les documents hypermédias peuvent utiliser les mêmes données dans différents contextes [Schloss 94], de plus ces données peuvent être utilisées par différents documents.

La séparation entre la sémantique pure (la structure logique) et la définition des instants d'apparition des composants (le quand, défini par la structure de présentation) est contestée en partie par l'auteur de ce mémoire. Cela parce qu'un document hypermédia n'est pas seulement ce qu'il contient, mais aussi quand ces contenus sont présentés. Ainsi, en nous basant sur l'architecture proposée par [Klas, 90], nous considérons qu'un modèle hypermédia idéal devrait permettre une description multi-niveaux incluant les structures suivantes : la **structure conceptuelle**, qui décrit les différentes parties logiques du document, ou composants, leurs relations logiques et à quels instants ces composants sont

2. Les types de documents qui peuvent être représentés par le modèle d'architecture de document ODA sont ceux fréquemment trouvés dans les travaux bureautiques tels que les rapports, les lettres, etc (uniquement des informations statiques). La norme ODA définit une architecture de document avec deux structures : la structure logique (les objets de base et composite) et de présentation (p.ex. pages, blocs).

présentés; la **structure de présentation**, qui décrit comment et où les différents composants seront présentés; et la **structure du contenu**, qui décrit les informations qui constituent les composants.

L'unique différence entre cette classification et la première est que «le quand» de la présentation des composants est transféré de la structure de présentation vers la structure logique. Ainsi, contrairement à l'affirmation de [Fluckiger, 95], lors de la présentation du document en différents systèmes clients, la structure logique et le comportement temporel (la structure conceptuelle) peuvent être réutilisables. Par contre, les présentations (la structure de présentation et du contenu) non supportées par le système client doivent être adaptées, alors que le «quand» de la présentation des composants doit être respecté. Par exemple, si un composant est constitué d'une présentation audio et si le système client ne permet pas la présentation de ce type de média, le contenu de ce composant peut être remplacé par une information textuelle. Mais les comportements temporels de ces présentations doivent être équivalents.

En dehors de ces trois structures, un modèle hypermédia doit permettre aussi la spécification des méthodes d'interaction possibles. Ainsi, il doit permettre la définition des ancres, des liens hypermédias et des autres méthodes d'interaction des documents multimédias.

A partir de la description complète du document, un système hypermédia peut utiliser deux approches différentes pour présenter le document : via une interprétation directe de la description logique (comme éditée par l'auteur) du document; ou via une traduction de cette description logique vers une **représentation physique** qui sera ensuite stockée, transférée et présentée.

Cette section présente les principales exigences qui doivent être satisfaites par un modèle hypermédia. Cette étude est divisée en six parties : la modélisation des structures du contenu (section 2.3.1), conceptuelle (section 2.3.2), et de présentation (section 2.3.3) d'un document hypermédia; la spécification des mécanismes d'interactions (section 2.3.4); les approches de modélisation de la structure conceptuelle des documents hypermédias (section 2.3.5); et enfin les approches de présentation du document (section 2.3.6).

2.3.1 Structure du contenu

Une des étapes nécessaires à la réalisation d'un document hypermédia est la génération ou capture des différentes informations qui vont être utilisées pour la composition du document. Par exemple, l'auteur peut utiliser une caméra pour enregistrer une séquence vidéo ou bien il peut créer une information graphique à partir d'un éditeur spécialisé. Dans ce travail, nous appelons ces informations des **données primitives**.

Les données primitives peuvent être constituées de deux types d'information : des **données médias**, qui peuvent être vues par le lecteur du document (audio, vidéo, texte, etc.); et des **données non-médias** qui ne peuvent pas être présentées immédiatement (document HyperODA et HyTime, etc.). Toute ces données peuvent être stockées dans une base de données locale ou bien être réparties sur différents serveurs multimédias.

La structure du contenu est responsable de la description des données primitives. Par exemple, elle peut être constituée d'un ensemble de données primitives et de leurs descripteurs. La paire données primitives et description de ces données est appelée par la suite **objet multimédia** (quoiqu'il puisse spécifier des données non-médias ou encore des données monomédias). Le terme «objet» est utilisé ici dans une acception large si bien que la représentation de cette information ne suit pas nécessairement une approche orientée-objet.

Au niveau de la structure du contenu, un modèle hypermédia doit permettre, entre autre, la spécification des informations d'accès et de manipulation des données primitives, et les valeurs originelles des

caractéristiques spatiales, sonores et temporelles de présentation (par exemple la vitesse et la taille originelles de présentation d'une séquence vidéo).

2.3.2 Structure conceptuelle

La structuration conceptuelle spécifie les composants et les groupes de composants d'un document hypermédia, et la composition logique et temporelle de ces composants.

2.3.2.1 Les composants et les groupes de composants

La structure conceptuelle contient la définition des composants sémantiques du document, permettant la partition du discours du document, par exemple, en chapitres et paragraphes, ou une série de séquences vidéos et de titres. Un exemple de cette structuration est présenté dans la figure 2.5.

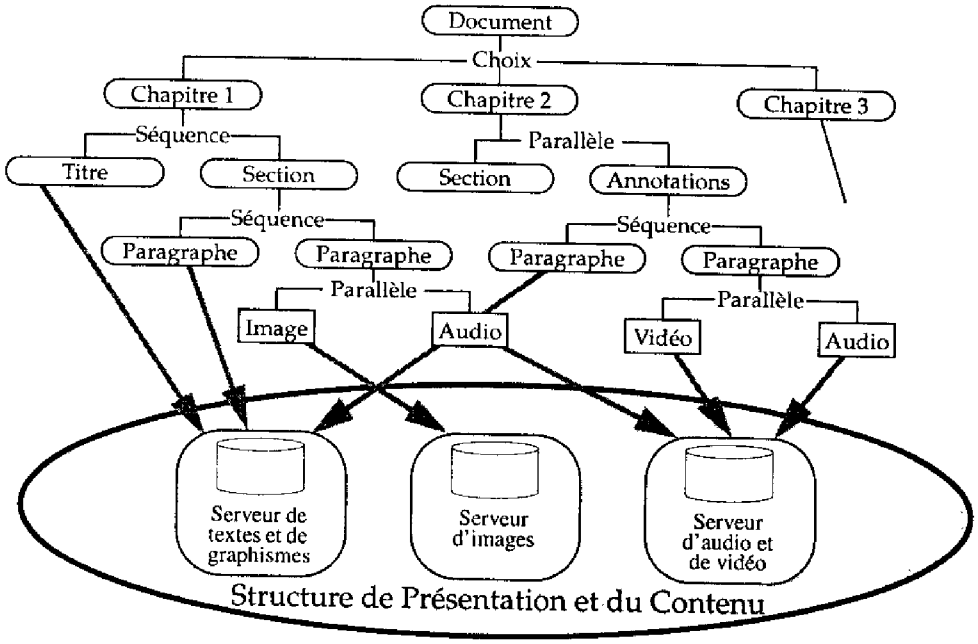


Figure 2.5 Exemple de structure conceptuelle d'un document hypermédia

Comme pour la programmation de systèmes de grande taille, la spécification de documents hypermédias devient délicate et très complexe lorsque la taille augmente. La structuration conceptuelle est aussi utile pour la construction de présentations complexes à partir de petits groupes d'information. Ainsi, la description du document pourrait être divisée en sections, chaque section pouvant être créée de façon indépendante des autres sections [Hardman, 95] : les présentations qui expriment ensemble une idée ou un concept peuvent être groupées et ce groupement peut être réutilisé n'importe où dans le document. De plus, ces mécanismes de structuration permettent la composition du document à partir de techniques *top-down* et/ou *botton-up*. Ils permettent aussi la définition de relations logiques et temporelles entre ces groupes et la définition de contraintes temporelles associées à un groupe de présentations. En conclusion, la structure conceptuelle introduit le bénéfice de la modularité, de l'encapsulation et de mécanismes d'abstraction.

La définition d'un composant du document doit être réalisée indépendamment des informations concernant la représentation du contenu du composant lui-même [Klas, 90], qui sont décrites par la structure de présentation et du contenu (comme illustré par la figure 2.5). Ainsi, les composants de la structure conceptuelle doivent faire uniquement mention des informations que ces composants représentent. Afin de simplifier la tâche de spécification d'un document hypermédia, cette mention doit être

indépendante du type d'information (données médias ou non-médias) ou de sa localisation géographique (locale ou répartie).

Lors du stockage et de la transmission d'un document, certaines parties de la structure conceptuelle et/ou les contenus des composants peuvent être incluses explicitement ou implicitement dans la structure du document [Fluckiger, 95] :

- **Inclusion explicite** : la structure du document hypermédia contient les données primitives elles-mêmes. Ainsi, la structure du document et les contenus inclus explicitement sont stockés dans un même fichier et transférés conjointement.
- **Inclusion implicite** : la structure du document hypermédia fait référence aux données primitives ou autres parties de la structure conceptuelle. Ainsi, dans les modes de transmission en temps-réel et de télé-chargement (section 2.2), seule une partie du document doit être transférée quand l'inclusion implicite est utilisée. Lors de la présentation du document, le système client doit accéder de façon distante aux parties incluses implicitement dans le document.

2.3.2.2 Les chemins de parcours et les structures des informations

La structure conceptuelle définit aussi les chemins de parcours du document hypermédia. Fondamentalement, deux types de liens peuvent être définis [Shackelford, 93] :

- **Lien structurel** : ce type de lien fournit la structure de base d'un document hypermédia. Quand le lecteur suit ces liens, le discours (ou la structure) de base est préservé comme il a été créé par l'auteur du document. Dans la figure 2.6, les traits qui lient les composants (représentés par les carrés) sont des liens structurels.
- **Lien hyper-structurel** : ce type de lien permet la définition des relations qui traversent la structure de base d'un document. Les liens hyper-structurels peuvent être divisés en liens **associatifs** et **référenciels** [Ginige, 95]. Les liens associatifs permettent la connexion de concepts associés et les liens référenciels fournissent des informations additionnelles à un concept. Dans la figure 2.6, les flèches grises représentent les liens hyper-structurels du document.

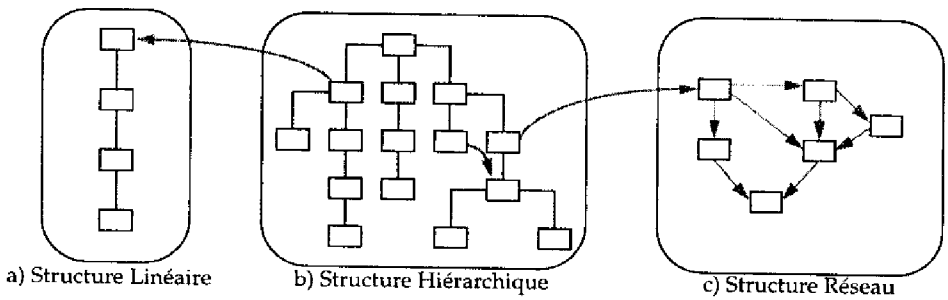


Figure 2.6 Types de liens et de structures

Nous pouvons distinguer trois façons de structurer l'information [Ginige, 95] :

- **Structure linéaire** (figure 2.6a) : elle est utile dans les documents dits «tours guidés», où le document fournit une procédure d'aide avec une liste d'actions discrètes qui doivent être exécutées dans un ordre particulier. Par exemple, les structures linéaires sont présentes dans les matériels d'entraînement, parce qu'un instructeur doit enseigner les modules d'une façon séquentielle. L'auteur peut inclure des questions en plusieurs points et si l'utilisateur donne de mauvaises réponses, des informations additionnelles peuvent être affichées via des liens référenciels.
- **Structure hiérarchique** (figure 2.6b) : elle est utilisée pour les documents comparables aux livres (organisés en chapitres et paragraphes), et l'auteur peut utiliser, par exemple, des liens référenciels

vers un glossaire ou une section de référence.

- **Structure réseau** ou graphe (figure 2.6c) : elle contient des liens associatifs. Ces liens sont de natures sémantiques et pragmatiques, et sont véritablement non-séquentiels. La structure réseau est plus adaptée pour l'organisation d'informations de type encyclopédie.

Un document hypermédia peut contenir à la fois ces trois types de structures; ainsi, un modèle hypermédia idéal devrait supporter tous ces types de structures.

2.3.2.3 Composition temporelle du document

La structure conceptuelle définit aussi la composition temporelle d'un document hypermédia, qui consiste en la description des instants de départ et d'arrêt des présentations des composants et des relations temporelles et conditionnelles entre ces présentations. Ces relations sont établies entre des événements définis à l'intérieur des différentes présentations. Il y a deux types d'événements qui peuvent arriver pendant la durée d'une présentation :

- Les **événements synchrones** (prévisibles) : ce sont les événements dont les positions dans le temps sont déterminées à l'avance. Un exemple d'événement synchrone est l'instant auquel une certaine trame vidéo ou un échantillon sonore sera présenté. La position de ces événements peut être déterminée seulement dans les conditions idéales (sans prendre en considérations les délais imprévisibles, comme une charge de réseau excessive).
- Les **événements asynchrones** (imprévisibles) : ce sont les événements dont les positions dans le temps ne peuvent pas être déterminées en avance, comme l'instant où un logiciel arrive dans un état particulier ou bien les interactions avec le lecteur du document.

Les relations temporelles

Les relations temporelles définissent les positions temporelles relatives entre et à l'intérieur des composants d'un document hypermédia. La description de ces relations requiert un **modèle temporel**. Par rapport à son unité élémentaire, deux classes de modèles temporels peuvent être identifiées [Wahl, 94] :

- **Les modèles basés sur les points** : ce sont des modèles dont les unités élémentaires sont les événements. Il y a trois relations déterministes qui peuvent être définies entre deux événements : un événement doit arriver avant, simultané avec, ou après un autre événement. Etant donné qu'une relation temporelle non-déterministe résulte d'une disjonction des relations temporelles déterministes, 2^3 relations non-déterministes peuvent être représentées par cette approche (par exemple, un événement doit arriver avant ou simultanement avec un autre événement).
- **Les modèles basés sur les intervalles** : ce sont des modèles dont les unités élémentaires sont les intervalles. [Hamblin, 72] et [Allen, 83] définissent qu'il y a 13 relations temporelles possibles entre deux intervalles (figure 2.7) : *avant, précède, chevauche, pendant, commence, finit* et *égale*, auxquelles s'ajoutent les relations inverses (exceptée la relation *égale*). 2^{13} relations non-déterministes peuvent être exprimées par cette approche.

[Wahl, 94] montre que les modèles temporels basés sur les intervalles sont plus appropriés aux systèmes multimédias :

- Les approches basées sur les intervalles couvrent un nombre plus grand de relations que les approches basées sur les points.
- Les approches basées sur les intervalles offrent une abstraction de plus haut niveau pour la description des relations temporelles multimédias, car les durées de présentation des composants apparaissent comme des intervalles temporels (ces présentations ont des durées différentes de zéro et infini).

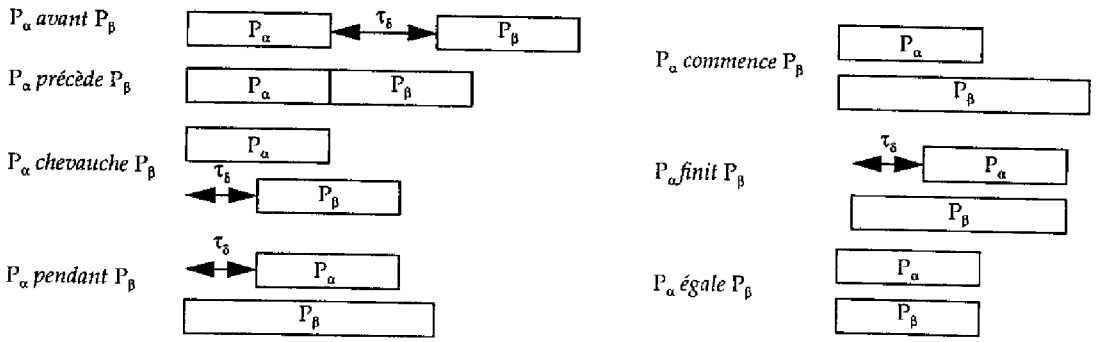


Figure 2.7 Les relations fondamentales entre deux intervalles [Allen, 83]

Les relations temporelles peuvent définir deux types de synchronisation :

- **Synchronisation inter-média** : ce sont des relations temporelles entre événements ou intervalles définis dans différentes présentations. Par exemple, «le début de la présentation B doit arriver 5 s après la fin de la présentation A». La synchronisation fine et suivie entre différents médias, par exemple la *lip-synchronization* (la synchronisation entre le mouvement des lèvres et la voix) est appelée **synchronisation continue.**
- **Synchronisation intra-média** : ce sont des relations temporelles entre événements ou intervalles définis à l'intérieur d'un objet multimédia dépendant du temps, par exemple, définies entre les trames d'une séquence vidéo ou entre les échantillons sonores d'une séquence audio.

En général, les synchronisations intra-médias sont des dépendances temporelles naturelles qui sont définies implicitement lors de la production des données primitives (dans la structure du contenu). Le schéma de synchronisation intra-média peut être changé par l'auteur du document lors de la définition de la structure de présentation. Par exemple, la synchronisation entre les trames d'une séquence vidéo peut être définie par la vitesse de présentation de cette séquence. Les synchronisations inter-média sont généralement des dépendances temporelles artificielles spécifiées explicitement par l'auteur de la structure conceptuelle d'un document. Ainsi, en général les synchronisations inter-médias sont décrites par la structure conceptuelle d'un document, et les synchronisation intra-média font partie de la structure de présentation.

Les relations temporelles peuvent être définies de façon statique. Ce processus qui consiste à organiser explicitement les relations temporelles entre les composantes est nommé **orchestration**. En contraste, les relations temporelles dites **relations en-direct** (ou vivantes) sont des relations qui sont définies de façon dynamique [Fluckiger, 95]. Des exemples des relations en direct sont : la synchronisation entre une vidéo et une audio enregistrées en direct, ou les synchronisations dans un travail coopératif. Dans ce mémoire, nous traitons uniquement des relations temporelles orchestrées, qui sont les relations les plus couramment rencontrées dans les documents hypermédias.

Pour l'orchestration d'un document hypermédia, il est nécessaire d'utiliser un modèle de spécification des relations logiques, conditionnelles et temporelles (ces modèles seront présentés en détail dans la section 2.3.5). Ces modèles peuvent décrire de façon **procédurale** ou **déclarative** la composition du document. La description procédurale décrit complètement la structure de contrôle de présentation du document à partir de la description ordonnée des actions. Une description déclarative de l'orchestration décrit les contraintes temporelles entre les présentations sans prendre en considération l'ordre chronologique des actions (un événement dans le futur peut influencer une action dans le passé). Ce dernier requiert le calcul de l'ordonnancement des présentations qui composent un document hypermédia. Ce calcul peut être réalisé en temps de compilation (*compile-time computation*) et/ou en temps de présentation (*run-time computation*) [Blakowski, 96].

A cause du non-déterminisme de la durée de traitement des informations dans les systèmes hypermédias répartis (causée par le réseau de communication, et les délais d'accès aux bases de données, etc.), les relations temporelles envisagées ne peuvent pas toujours être garanties; c'est pourquoi un modèle hypermédia doit permettre à l'auteur du document de spécifier des **méthodes de tolérance** de synchronisation [Wynblatt, 95]. Ainsi, l'auteur peut spécifier quels compromis de synchronisation sont acceptables et les moyens de spécifier des traitements d'exception lors de la violation de ces contraintes.

Les relations conditionnelles

Une relation conditionnelle est définie par une condition, associée aux états d'un ensemble de composants (par exemple, un état de présentation, de préparation ou d'exécution), et des actions qui seront appliquées à un ensemble de composants quand la condition est satisfaite. Par exemple, «si A est en cours de présentation et si le lien L est activé, alors présenter B» est une relation conditionnelle.

La Synchronisation dans un système Réparti

Les besoins de spécification des relations temporelles d'un document multimédia ou hypermédia sont dépendants du type de transfert du document (voir section 2.2) :

- Pour le télé-chargement de documents hypermédia, les informations de synchronisation sont délivrées (complètement ou en parties) avant le départ de la présentation du document (ou d'une partie du document).
- Pour la présentation synchronisée de scénarios multimédias, il y a trois approches fondamentales pour la livraison des informations de synchronisation pour les systèmes récepteurs [Blakowski, 96] :
  - Délivrance des informations de synchronisation avant le départ de la présentation : l'implémentation de cette approche est simple et elle permet une facile manipulation dans le cas des plusieurs sources d'objets multimédia. Par exemple, [Chassot, 95] utilise ce type d'approche. Le désavantage est le délai causé pour le transport de la spécification de la synchronisation avant la présentation;
  - Utilisation d'un canal de synchronisation additionnel : aucun délai n'est produit par cette approche, mais le canal de communication additionnel est nécessaire et il peut être source d'erreurs dues au délai ou à la perte des unités de synchronisation.
  - Multiplexage des flux de données et des informations de synchronisation : dans ce cas les informations de synchronisation et les objets multimédias sont délivrés conjointement. Par exemple, [Courtiat, 94b] [Courtiat, 96a] utilise ce type d'approche. Le problème est la difficulté de définir une connexion de transport fournissant les qualités de service requises par tous les médias véhiculés [Chassot, 96].

2.3.3 Structure de présentation

La modélisation de la structure de présentation d'un document hypermédia englobe la description des caractéristiques spatiales, sonores et temporelles de chaque présentation du document[3]. L'auteur doit spécifier les caractéristiques de présentation de chaque composant et la composition spatiale de ces présentations à un instant donné. La spécification des caractéristiques de présentation englobe la description de la façon dont le composant sera vu (description des caractéristiques spatiales) et/ou entendu (description des caractéristiques sonores) par le lecteur du document. La composition spatiale

3. Il y a d'autres types de traitement des composants que leurs présentations, comme la préparation des données et l'exécution d'un exécutable. Quelques fois, l'auteur peut personnaliser le traitement de l'information; par exemple, un script peut définir des paramètres d'entrée dont les valeurs peuvent être définies par l'auteur. Dans ce travail, nous ne prendrons pas en considération ce type de caractérisation d'une présentation multimédia.

décrit les relations spatiales entre les présentations.

2.3.3.1 Spécification des caractéristiques de présentation des composants

A partir de la structure conceptuelle du document, l'auteur doit définir le contenu des composants (c.-à-d. associer un objet multimédia au composant) et particulariser ses caractéristiques de présentation. Par exemple, l'auteur peut changer les caractéristiques originelles de présentation des objets multimédias (comme le volume ou la vitesse de présentation), ou il peut encore définir de nouvelles caractéristiques (comme la position spatiale de présentation d'une image).

Afin de modéliser une présentation, un modèle hypermédia doit permettre la spécification, entre autre, des informations suivantes :

- Les caractéristiques temporelles de la présentation des informations dynamiques, comme la vitesse de présentation, la position de début et de fin de présentation et le nombre de répétitions.
- Les caractéristiques spatiales des présentations visuelles, comme la taille, la position et le style de présentation.
- Les caractéristiques des présentations sonores, comme le volume de présentation.
- Les dispositifs de sortie, appelé canaux, sur lesquels les informations seront présentées et vues par le lecteur du document (p.ex., une fenêtre, un canal audio).
- Des présentations alternatives peuvent être définies afin de remplacer la présentation principale si celle-ci ne peut pas être présentée dans un certain système (présentation hypermédia adaptable au ressources disponibles), s'il y des problèmes d'accès à l'information principale, ou si les contraintes temporelles ne sont pas satisfaites.

2.3.3.2 Composition spatiale du document

La méthode la plus courante de positionnement de la présentation des composants à l'écran est la définition de la position spatiale absolue (figure 2.8a) ou relative (figure 2.8b) de chaque présentation dans un système de coordonnées virtuelles.

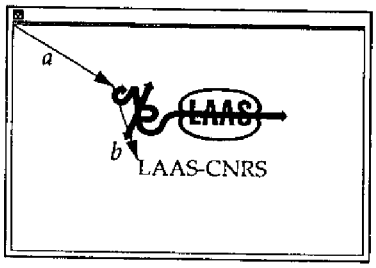


Figure 2.8 Composition spatiale

2.3.4 Interactions

Un des principaux aspects qui caractérisent un document hypermédia est son caractère interactif : le lecteur du document dispose d'un ensemble de possibilités permettant le contrôle de la présentation. Quatre méthodes d'interactions peuvent être identifiées [Hardman, 95] :

- **Navigation** : cette méthode permet la spécification d'un ensemble de choix donnés au lecteur du document afin qu'il puisse sélectionner la présentation d'un concept parmi plusieurs. Elle est généralement réalisée par la création d'un lien ou script qui attache des ancres origines à des ancres destinations.
- **Contrôle de la présentation** : c'est la méthode d'interaction fréquemment trouvée dans les docu-

ments multimédias (section 2.1.2), où le lecteur peut arrêter, répartir, faire l'avance ou le retour rapide de la présentation du document.

- **Contrôle de l'environnement** : cette méthode permet de particulariser l'environnement de présentation du document. Par exemple, le lecteur peut désactiver le canal audio afin d'inhiber les présentations audios ou il peut encore changer la taille d'une fenêtre.
- **Interactions d'application** : dans les méthodes précédentes, l'auteur crée le document et le lecteur peut seulement interagir avec. Il y a des applications qui requièrent des mécanismes spécifiques. Par exemple, dans les applications de télé-enseignement, un modèle devrait permettre la spécification de la notion de suivi des élèves (par exemple, un historique des activités de l'élève) et d'examen (par exemple, à partir d'un champ de saisie de caractères). Une autre méthode d'interaction est la recherche par mots-clefs. Ces types de mécanismes d'interactions sont supportés par des outils spécialisés.

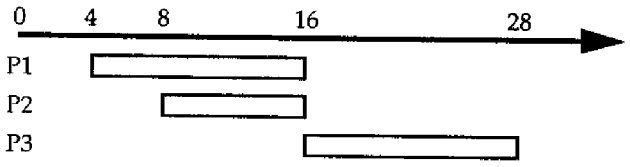
Un modèle hypermédia idéal devrait permettre la spécification de tous ces types d'interaction. De plus, afin de simplifier la tâche de structuration conceptuelle d'un document hypermédia, un modèle devrait fournir une approche uniforme de représentation des composants, des relations logiques et temporelles, et des mécanismes d'interaction.

2.3.5 Modélisation de la structure conceptuelle

La description de la structure conceptuelle, c'est-à-dire la définition des composants et de la composition logique et temporelle de la présentation de ces composants, est une étape critique du développement des documents hypermédias. Les principales approches de base de composition de documents multimédias et hypermédias sont : ligne temporelle, composition hiérarchique, composition via des points de références et les réseaux de Petri.

2.3.5.1 Lignes temporelles

La ligne temporelle permet l'alignement des présentations sur un axe temporel. La figure 2.9 présente un exemple de ligne temporelle. Les systèmes MAEstro [Drapeau, 91], QuickTime [Apple, 91], et la norme HyTime [Newcomb, 1991] (section 2.4.3) utilisent des modèles de composition temporelle basés sur cette approche.



1. P1 est présenté de 4 jusqu'à 16
2. P2 est présenté de 8 jusqu'à 16
3. P3 est présenté de 16 jusqu'à 28

Figure 2.9 Exemple de ligne temporelle

Cette technique offre pour principal avantage une grande simplicité d'expression des schèmes de synchronisation. De plus, l'auteur a une vision très nette des informations qui seront présentées et à quel moment elles le seront. Malheureusement, cette méthode présente plusieurs limitations :

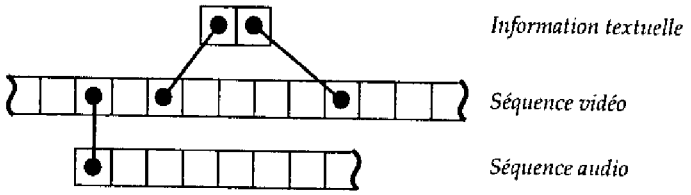
- Elle permet seulement la définition de l'alignement temporel idéal de présentations, dans la mesure où elle définit les points de départ et de fin idéaux des présentations des composants du document.
- Elle requiert la connaissance exacte de la durée de présentation d'un objet multimédia. L'auteur doit manipuler les segments d'information de façon manuelle, afin d'obtenir les comportements temporels désirés (par l'édition des données ou par le changement de la vitesse de présentation).

- Comme cette approche ne fournit pas de mécanismes de structuration, ni la représentation de relations logiques, elle ne permet pas la définition de la structure conceptuelle complète des documents hypermédias.

Quelques modèles multimédias et hypermédias utilisent des versions étendues de l'approche ligne temporelle, permettant la définition de relations logiques et temporelles entre présentations (par exemple, l'approche proposée par [Hirzalla, 95]). Dans tous les cas, les modèles résultants de ces extensions deviennent très complexes et perdent l'avantage principal du modèle ligne temporelle de base qui est la simplicité de compréhension du comportement temporel du document.

2.3.5.2 Composition via des points de référence

Dans l'approche de composition via des points de référence, les présentations sont vues comme des séquences de sous-unités discrètes [Blakowski 92]. La position d'une sous-unité (p.e., une trame d'une séquence vidéo) dans un objet est appelée un **point de référence**. Les informations discrètes (p.e., texte) ont seulement deux points de référence : le début et la fin de présentation. La synchronisation entre présentations est définie par des connexions entre les points de référence. La figure 2.10 illustre la composition via des points de référence.



Les carrés représentent les points de références : des trames pour l'information vidéo et des échantillons sonores pour l'information audio.

Figure 2.10 Synchronisation via des points de référence

Un avantage du concept de point de référence est la facilité de changer la vitesse de présentation [Haindl, 96], parce qu'il n'y a pas une référence explicite au temps.

Les limitations de l'approche de composition via des points de référence sont les suivantes :

- Les synchronisations sont définies par un ensemble de connexions entre points de référence. Lors de la présentation du document, les irrégularités de présentations sont ignorées. Il n'y a pas de possibilité de tolérance à ce niveau.
- L'utilisation unique d'une approche de composition via des points de références ne permet pas la spécification de délais, ni la définition de synchronisations conditionnelles autres que les connexions entre les points de référence.
- Cette approche n'est pas appropriée pour la description de la structure conceptuelle des documents hypermédias. Car elle ne propose aucun mécanisme de structuration et qu'il n'y a pas de distinction entre le composant et son contenu.

2.3.5.3 Composition hiérarchique

Dans la composition hiérarchique de documents hypermédias, les présentations d'un document sont vues comme un arbre (figure 2.11). Dans cet arbre, les nœuds internes dénotent des relations temporelles entre les sous-arbres sortants. Ces relations temporelles sont définies par des **opérateurs de synchronisation**. Les principaux opérateurs de synchronisation sont les opérateurs série et parallèle. Ces opérateurs définissent qu'un ensemble d'actions seront exécutées de façon sérielle ou parallèle. Ces actions peuvent être atomiques ou composées [Blakowski, 92] : une action atomique manipule la pré-

sentation d'une information, l'interaction avec l'utilisateur ou un délai; les actions composées sont une combinaison d'opérateurs de synchronisation et d'actions atomiques.

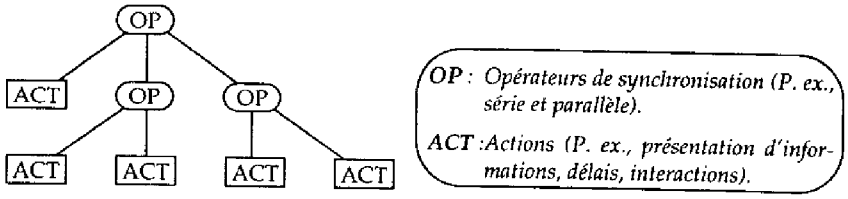


Figure 2.11 Composition hiérarchique

L'approche de composition hiérarchique présente quelques limitations :

- Cette approche ne permet pas la représentation de relations temporelles au delà des frontières hiérarchiques. Ainsi, les liens hyper-structurels (section 2.3.1) ne peuvent pas être spécifiés par cette approche.
- Les relations logiques et temporelles entre présentations ne sont pas représentées d'une façon naturelle. Au contraire de l'approche ligne temporelle, l'auteur n'a pas une vision d'ensemble des informations présentées et de leur date de présentation.
- Un ensemble d'actions peut être synchronisé seulement par rapport au début ou à la fin d'un ensemble d'actions. Cela signifie, par exemple, que la présentation des sous-titres dans certaines parties d'une séquence vidéo requiert le découpage de la séquence vidéo en plusieurs composantes consécutives.

2.3.5.4 Réseaux de Petri

Un réseau de Petri est un bon candidat pour la modélisation de documents hypermédias, à cause principalement de sa représentation graphique, de la facilité de modélisation de schémas de synchronisation, et de la possibilité d'analyser d'importantes propriétés [Murata, 89] qu'il offre. Mais, les approches de composition basées sur le modèle de réseau de Petri ont quelques limitations :

- De manière similaire à la composition hiérarchique, dans la plupart des modèles basés sur les réseaux de Petri, un ensemble d'actions peut être synchronisé seulement par rapport au début ou à la fin d'un ensemble d'actions.
- Cette approche ne contient pas les concepts appropriés pour représenter des synchronisations conditionnelles plus complexes ni pour les méthodes d'interaction d'application (section 2.3.4).
- Dans [Hudson, 93], les auteurs affirment qu'une autre limitation de cette approche est sa complexité : dans un certain sens, ce formalisme tend à être trop puissant et, en général, de trop bas niveau pour l'utilisation directe par un auteur d'un document.

2.3.6 Présentation des documents hypermédias

Un modèle hypermédia peut être vu comme un modèle logique de haut niveau qui facilite la tâche de création des documents hypermédias. Un système hypermédia peut utiliser deux approches différentes pour présenter le document : via une interprétation (présentation) directe de la description logique du document; ou via une traduction de la description logique vers une **représentation physique** qui sera ensuite stockée, transférée et interprétée (ou exécutée). Un **modèle physique** (ou modèle de stockage) définit la représentation physique d'un document hypermédia qui peut être traitée par un système hypermédia.

Les formats d'échange multimédia et hypermédia sont applicables à plusieurs contextes [Buford, 94a] :

- comme un modèle de stockage pendant la création et l'édition de documents multimédias (et hypermédias), pour de nouvelles compositions et pendant le stockage de matériaux qui peuvent être utilisés dans le processus d'édition;
- comme un format pour l'émission de médias dans une forme finale;
- comme un format d'émission en temps-réel d'un serveur vers les clients connectés via des réseaux pour le téléenseignement, l'information-sur-demande, etc;
- pour l'échange d'informations multimédias entre applications.

L'adoption de normes internationales de représentation d'informations hypermédias et multimédias (par exemple, MHEG [ISO 13522], HyTime [Newcomb, 94] et HyperODA [Appelt, 93]) comme modèle physique des documents hypermédias est grandement recommandée. Les documents hypermédias ainsi stockés pourraient être transférés dans un système ouvert et présentés via des interpréteurs de la représentation normalisée ou bien encore traduits vers une représentation spécifique à un système hypermédia.

L'interprétation (présentation) directe de la description logique du document requiert l'implémentation d'un outil d'interprétation de la description logique (une solution ad-hoc). Dans ce cas, afin de permettre la présentation d'un document hypermédia dans un système ouvert, ce système doit supporter la traduction de la description logique du document vers un format normalisé de représentation des informations hypermédias et multimédias, ou bien, l'outil d'interprétation doit être portable.

2.4 Modèles multimédias et hypermédias

Plusieurs modèles multimédias et hypermédias ont été proposés. Selon le type d'interface avec l'auteur du document, nous pouvons classer ces modèles en **orientés-langage** et **graphiques**. D'autres classifications des paradigmes de création de documents hypermédias peuvent être trouvées en [FAQ, 96] et [Ginige, 95].

Les approches de plus bas niveau sont les modèles orientés-langage (utilisées, par exemple, par [Gibbs, 91], [Herman, 94], [Klas, 90], [Vazirgiannis, 93], MHEG [ISO 13522]). Ces modèles ont un pouvoir d'expression très grand, mais la spécification de la composition d'un document hypermédia dans une forme textuelle est difficile à produire et à modifier ([Ackermann, 94], [Hudson, 93]). De plus, les compositions temporelles entre présentations sont difficiles à identifier.

Les modèles graphiques, par exemple basés sur les approches lignes temporelles et réseaux de Petri, ont l'avantage d'illustrer de façon graphique la sémantique de synchronisation temporelle. Ces types de modèles simplifient la spécification de contraintes temporelles et réduisent le temps de création de documents hypermédias.

Les modèles multimédias et hypermédias peuvent être aussi classés en modèles **formels** et **non formels**. La spécification de documents hypermédias par l'utilisation de modèles informels ou semi-formels peut être source d'ambiguïtés et ne permet pas l'utilisation d'outils d'analyse. Par contre, les modèles formels permettent la construction d'une sémantique précise et non ambiguë d'un document hypermédia et ils permettent aussi l'application de méthodes d'analyse.

La section précédente a identifié les principales exigences qui doivent être satisfaites par un modèle hypermédia. Basée sur cette étude, cette section présente et critique les principaux modèles multimédias et hypermédias.

2.4.1 Modèle de référence d'hypertexte Dexter

Le modèle de référence d'hypertexte[4] Dexter est une tentative de capturer, d'une façon formelle ou informelle, les abstractions importantes trouvées dans la plupart des systèmes hypertextes [Halasz, 90]. Dexter est un modèle à trois couches (figure 2.12) :

- La couche *Within-component* décrit la structure interne des données primitives. La définition de cette couche ne fait pas partie du modèle Dexter, mais de modèles de référence conçus spécifiquement pour modéliser la structure d'applications particulières, de documents, ou de types de données.
- La couche *Storage* modélise le réseau hypertexte par la définition des composants du document et de ses liens. Le mécanisme d'ancrage (présenté par la suite) définit la séparation entre cette couche et la couche *Within-component*.
- La couche *Run-time* est responsable de la manipulation du réseau hypertexte au moment de la présentation du document. L'interface entre cette couche et la couche *Storage* est réalisée par la spécification des caractéristiques de présentation des composants.

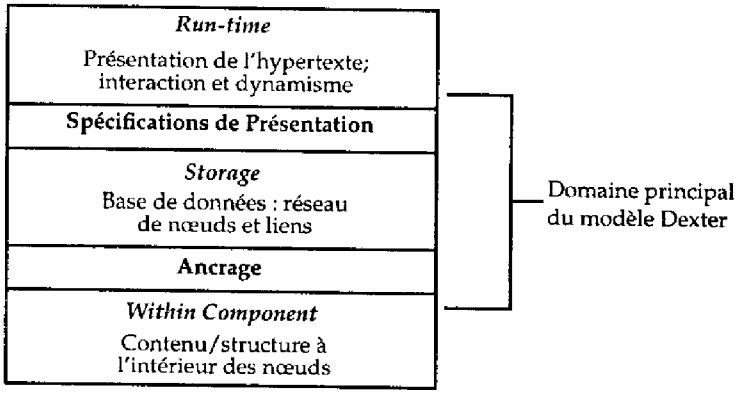


Figure 2.12 Les couches du modèle Dexter

Le domaine principal du modèle de référence Dexter est la couche *Storage*. Dans cette couche, l'entité fondamentale et l'unité de base d'adressage est le **composant**. Il y a trois types de composants :

- Un **composant atomique** représente des données primitives. Chaque composant atomique contient un identificateur unique, le contenu lui-même (par inclusion explicite) et les informations sur le composant. Ces informations décrivent les propriétés du composant :
  - Une liste d'ancres. Une ancre est une région à l'intérieur des données primitives qui peut être utilisée comme source ou destination d'un lien.
  - La spécification de la présentation, qui contient les informations originelles de présentation du composant.
  - Un ensemble d'attributs/valeurs arbitraires. Ces attributs sont utilisés, par exemple, par la définition de mots-clés d'un composant.
- Les **composants composites** fournissent les moyens de structuration d'un document hypertexte. Un composite est une simple collection de composants atomiques, composites et liens.
- Les **liens** sont des entités qui représentent des relations entre composants. Un lien est défini par une liste de spécifieurs. Chaque spécifieur définit :
  - l'identificateur du composant et de l'ancre qui est un point final d'un lien;
  - la direction spécifie si le point défini par ce spécifieur est un point d'origine du lien, de destination ou d'origine et de destination;

4. Dans le modèle Dexter, les documents hypertextes englobent les documents hypermédias.

- la spécification de la présentation.

La description multi-couche du modèle Dexter ne présente pas les mêmes niveaux d'abstraction que l'architecture de description de documents hypermédias présentée dans la section 2.3 :

- Les composants décrivent et contiennent les données primitives, il n'y a pas de distinction entre la structure conceptuelle et le contenu. Ainsi, les objets multimédias ne peuvent pas être réutilisés.

  De plus, la notion de composant composite ne fournit pas un mécanisme d'abstraction adéquat pour la représentation de la structure du document, où un ensemble de composants devrait être traités comme un objet unique. Cela n'est pas les cas du modèle Dexter qui ne permet pas la définition d'une ancre attachée à un composite (un ensemble de composants) [Grønbaek, 94].
- La couche *Storage* du modèle Dexter ne fournit pas les moyens de définir des relations temporelles et conditionnelles entre les composants du document.

2.4.2 *Amsterdam Hypermedia Model* (AHM)

Le AHM [Hardman, 93] est un modèle hypermédia obtenu à partir d'une extension du modèle de référence Dexter et du modèle multimédia CMIF (*CWI*[5] *Multimedia Interchange Format*) [Bulterman, 91]. Le modèle AHM résout quelques problèmes du modèle Dexter, qui sont : l'absence de la notion de temps dans la couche *Storage*; l'absence de moyen de spécifier l'information de présentation de haut niveau; et l'absence de la notion de contexte pour les ancres.

Contrairement au modèle de référence Dexter, le modèle AHM permet la représentation des relations temporelles entre l'ensemble des présentations qui composent un document hypermédia. Un exemple de composition temporelle d'un document hypermédia est présenté dans la figure 2.13. Ci-dessous, nous présentons les éléments utilisés pour la représentation de cette composition :

- Les composants atomiques du modèle Dexter représentent les données primitives et un ensemble d'informations sur ces données. Le modèle AHM spécifie de plus : la durée de présentation et le canal[6] où ces données seront présentées. Ainsi, dans le modèle AHM un composant atomique représente la paire données primitives/présentation.

  Au contraire du modèle Dexter, un composant atomique peut contenir les données encodées (inclusion explicite) ou leur faire référence (inclusion implicite). La référence est utilisée dans le cas où il y a plusieurs présentations de la même information (pour la réutilisation des données primitives).

5. CWI : Centrum voor Wiskunde en Informatica.

6. Dans le AHM, un canal est un dispositif de sortie abstrait utilisé pour la présentation de composants. Un canal spécifie les caractéristiques de présentation par défaut des données présentées (par exemple, format de caractère ou volume).

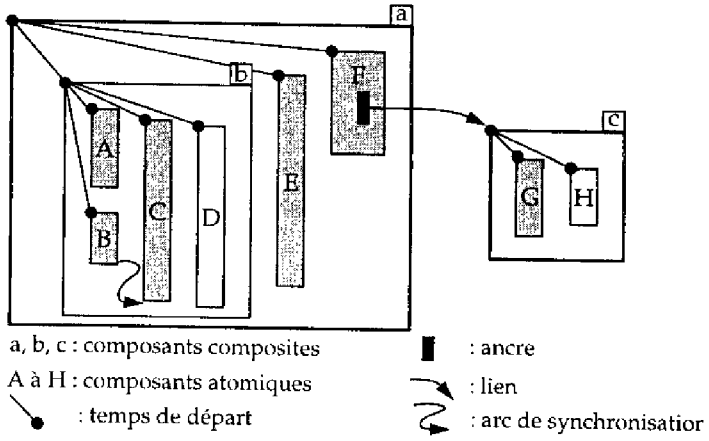


Figure 2.13 Les relations temporelles dans le modèle AHM [Hardman, 94]

- Les composants composites du modèle Dexter représentent une simple collection de composants atomiques, composites et liens, c'est-à-dire une composition logique des données primitives. Dans le modèle AHM, ils représentent une composition logique et temporelle des présentations. Les attributs ajoutés à un composant composite sont :
  - le type de composition, qui peut être *parallèle* ou *choix*. Dans une composition parallèle, tous les composants sont présentés. Dans un composite choix, un seul composant sera présenté (le mécanisme de sélection est réalisé par le support de *run-time*).
  - le temps de départ de présentation d'un composant. Ce temps est relatif au départ de présentation du composite père.
  - les arcs de synchronisation, qui permettent la représentation des relations temporelles entre deux composants. Cette relation temporelle est définie par un intervalle de synchronisation (contenant un temps idéal et la déviation acceptable), et le type de synchronisation. Ce type indique si cet intervalle est relatif au départ ou à la fin de présentation d'un composant, ou encore s'il s'agit d'un *offset*[7] à partir du début d'une présentation. Une deuxième partie du type indique si la relation est forte ou faible. Dans le cas d'une relation forte, si la relation n'est pas respectée une erreur arrive et la présentation s'arrête. Dans ce cas d'une relation faible, si la relation n'est pas respectée un avertissement arrive et la présentation continue.
- Le modèle AHM a ajouté la notion de contexte aux liens, où [Hardman, 94] : un **contexte source** pour un lien est la partie d'une présentation hypermédia affectée par la poursuite d'un lien, un **contexte destination** est la partie qui est affichée à l'arrivée à la destination du lien. Le modèle AHM définit trois types de lien contexte :
  - *continue*, la présentation source ne s'arrête pas lors de la poursuite du lien;
  - *pause*, la présentation source s'arrête, mais lors d'une reprise de présentation, elle se fera à partir du point d'arrêt;
  - *replace*, la présentation source s'arrête, et lors d'une reprise de présentation, elle se fera à partir du point initial.

Les synchronisations conditionnelles plus complexes (utilisées par exemple, dans la sélection d'un composant composite du type choix) ne sont pas représentables dans la couche *Storage* du modèle AHM. Cela est contrôlé par la couche *Run-time*. Ainsi, il n'y a pas moyen de spécifier ce type de synchronisation.

Le concept d'arc de synchronisation n'est pas suffisamment puissant pour exprimer la *lip-synchronisa-*

7. Cet *offset* ne définit pas une position temporelle, mais un point de référence à l'intérieur de la présentation.

tion [van Rossum, 93]. Cela est possible seulement si l'auteur définit un certain nombre d'arcs de synchronisation entre les trames vidéos et les échantillons sonores.

Le modèle AHM permet la spécification de contraintes temporelles à travers les arcs de synchronisation. Aucun mécanisme de contrôle lors de la violation de ces contraintes n'est fourni. Si une contrainte temporelle n'est pas respectée, une condition d'erreur existe ou bien la contrainte est ignorée. De plus, ce modèle ne propose aucune méthode d'analyse de la composition logique et temporelle du document (cela est hors du domaine du modèle).

Comme le modèle Dexter, AHM ne suit pas le modèle multi-niveau présenté dans la section 2.3. Les composants font uniquement référence aux données primitives. Ainsi, l'AHM ne permet la réutilisation que des données primitives par différents composants atomiques. La description de ces données n'est pas réutilisable.

L'environnement CMIFed

Le CMIFed (*CWI Multimedia Interchange Format Editor*) [Van Rossum 93] est un environnement d'édition et de présentation de documents hypermédias basé sur le modèle AHM. Il y a deux façon de voir un document CMIF pendant sa création : la vision hiérarchique et la vision des canaux (figure 2.14). La vision hiérarchique montre le document comme une collection de composants qui sont présentés en série ou en parallèle. La vision canaux présente les composants associés aux canaux abstraits; elle est présentée sous la forme de lignes temporelles, indiquant les informations de synchronisation.

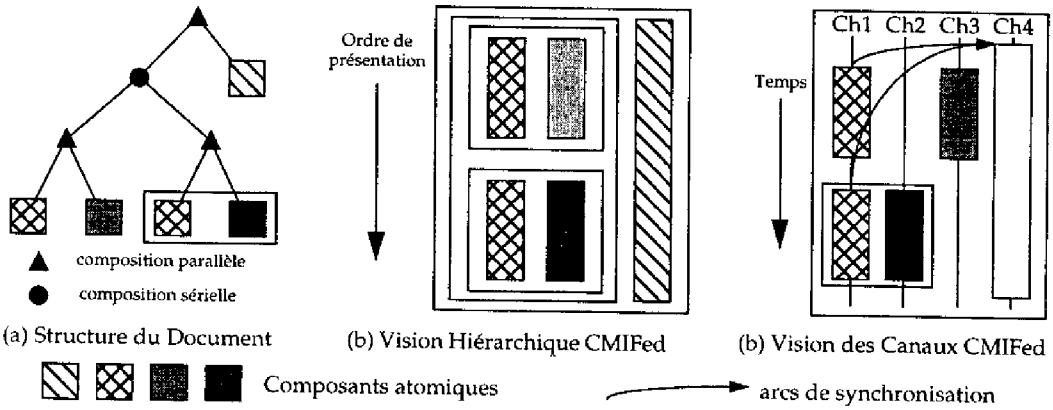


Figure 2.14 Hiérarchie et vision des canaux du CMIFed [Hardman, 94]

Dans CMIFed, la présentation est créée par la définition de la structure hiérarchique du document (figure 2.14b). Les informations temporelles sont déduites à partir de la structure hiérarchique et des durée des composants à l'intérieur de cette structure (une synchronisation plus fine peut être ajoutée dans la vision des canaux).

L'origine d'un lien dans CMIFed est une ancre définie à l'intérieur d'un composant atomique, et la destination peut être une ancre, un composant atomique ou composite.

2.4.3 La norme HyTime

Le langage de structuration de documents hypermédias HyTime [Newcomb, 90] spécifie comment certains concepts universels de tous les documents hypermédias peuvent être représentés en utilisant le langage SGML (*Standard Generalized Markup Language*) [ISO 8879]. Ces concepts comprennent l'association de plusieurs types d'information par l'utilisation d'hyper-liens, le placement et les interrelations de ces informations sur des systèmes de coordonnées virtuelles. Ces systèmes de coordonnées peuvent

représenter l'espace, le temps, ou n'importe quelle dimension quantifiable.

HyTime peut être vu comme un meta-langage pour la définition de types de documents multimédias, dans la mesure où il s'agit d'un langage permettant la définition des concepts à partir desquels l'auteur peut décrire la structure conceptuelle d'un document. Cette définition est réalisée par les *Définitions de Type de Document* (DTD). Par exemple, une DTD peut définir la structuration du document en chapitres, sections et paragraphes. Chaque DTD requiert un logiciel de présentation d'application HyTime approprié [Rutledge, 93].

HyTime utilise un modèle temporel basé sur la ligne temporelle (section 2.3.5.1) pour la composition temporelle de documents hypermédias [Erfle, 94]. Dans cette approche, une présentation (ou un autre traitement de données) est définie comme un événement. Contrairement à l'approche ligne temporelle générale (section 2.3.5.1), HyTime permet aussi l'expression de relations entre événements :

- le début d'un événement peut être synchronisé avec le début d'un autre événement.
- la fin d'un événement peut être synchronisée avec le début d'un autre événement.
- la fin d'un événement peut être synchronisée avec la fin d'un autre événement.
- la durée d'un événement peut être affectée comme égale à la durée d'un autre événement.

Pour exprimer des délais entre événements (p.e., la vidéo commence 10 sec. après l'audio), il est nécessaire d'utiliser des événements artificiels [Erfle, 94].

HyTime permet la description de la structure conceptuelle du document. Mais ce langage ne permet pas la description des caractéristiques de présentation, ni des modèles d'interaction du document. Ces caractéristiques et modèles peuvent être décrits via des constructeurs non HyTime (un code SGML générique), mais ces constructeurs ne sont pas connus par le langage HyTime. Ainsi, un logiciel de présentation d'application HyTime requiert l'inclusion de sémantiques d'interprétation spécifiques de l'application [Buford, 94b]. De plus, HyTime repose sur un modèle orienté-langage et il est donc difficile d'utilisation (un environnement d'édition graphique de documents hypermédias pourrait simplifier l'utilisation de HyTime).

2.4.4 Modèle de composition hiérarchique de [Hudson, 93]

Dans [Hudson, 93], les auteurs proposent une approche hiérarchique de composition de documents multimédias. Les composants du document sont représentés par des objets primitifs ou composites (construits à partir d'objets primitifs et composites). Ces objets sont représentés en quatre parties :

- Le **contenu** : il contient explicitement les données primitives.
- L'ensemble de **ressources nécessaires** : indiquant les matériels et les autres ressources nécessaires à la présentation du contenu.
- Une **machine temporelle** : il s'agit d'un composant actif qui contrôle le temps de présentation des différentes parties des données primitives (le «quand» de la présentation). Par exemple, un type de machine temporelle est l'horloge périodique, utilisée pour le contrôle de présentation d'une séquence vidéo avec une vitesse constante.
- Un **driver de présentation** : il s'agit d'un élément passif responsable de la manipulation et de la présentation du contenu de l'objet basé sur les stimuli temporels de la machine temporelle (le «comment» et le «quoi» de la présentation). Les drivers de présentation sont généralement dépendants du type de média.

Le type d'objet et les quatre parties ci-dessus impliquent certaines propriétés et paramètres que l'objet doit avoir. Ces paramètres et propriétés définissent le comportement et les caractéristiques de l'objet.

Des propriétés communes à tous les objets sont utilisées par les techniques d'analyse et de synthèse. Ces propriétés communes sont :

- **Durée** : elle peut être déterminée si l'auteur définit une valeur, non-déterminée si l'auteur ne définit pas cette valeur, ou indéterminée si la durée est imprévisible.
- **Réductibilité** : elle définit la capacité de changer la durée de présentation et la méthode de modification (par ex., via le changement de la vitesse, via l'arrêt ou la répétition de la présentation, etc.)
- **Capacité de contrôle de flux** : elle indique la capacité d'arrêter, de reprendre, de finir, de changer de direction, et de changer la vitesse d'une présentation.

[Hudson, 93] utilise une approche hiérarchique (introduite dans la section 2.3.5.3) pour la composition temporelle des objets qui composent un objet composite. Ce dernier est structuré comme un arbre avec des nœuds externes occupés par des objets primitifs ou composites et des nœuds internes occupés par des opérateurs de composition. Les opérateurs de composition sont les suivants :

- **Séquentiel** : une superposition entre les présentations peut être définie.
- **Parallèle** : il définit la présentation concurrente de plusieurs présentations. Il y a deux critères de terminaison : *arbitraire* et *sélection*. Ces critères sont accompagnés d'un compteur. Dans le cas d'une terminaison arbitraire, les présentations parallèles finissent quand un nombre spécifié de présentations finissent (sans identification des présentations elles-mêmes). Dans la terminaison *sélection*, la présentation parallèle finit quand la présentation sélectionnée finit.
- **Ligne temporelle** : fréquemment une présentation est utilisée comme une présentation majeure accompagnée d'autres présentations conduites par l'évolution temporelle de la présentation majeure.
- **Escape** : l'opérateur escape est similaire à l'opérateur ligne temporelle. Mais dans ce cas, la fin de la présentation majeure interrompt les autres présentations demandées (les autres filles) et aussi le demandeur (qui a commandé l'exécution de l'escape). Dans l'opérateur ligne temporelle, le demandeur est stoppé jusqu'à la fin des présentations demandées.
- **Synchronisation continue** : cet opérateur permet la définition de la synchronisation de différents médias dans une granularité fine.
- **Conditionnel** : cet opérateur fournit la possibilité de créer des présentations conditionnelles. L'interaction avec l'utilisateur et l'histoire de la présentation (le temps actuel de présentation, l'état de sortie des différentes présentations et les contrôles de flux précédemment appliqués) peuvent être utilisées pour déterminer si une présentation sera effectuée ou non.
- **Sélection** : il permet la représentation d'un choix (faite par un lecteur du document) d'une présentation parmi un nombre de présentations. Le paramètre *time-out* spécifie le temps d'attente maximale pour l'occurrence de l'interaction et le paramètre défaut spécifie quelle action d'interaction doit être simulée si le *time-out* arrive.
- **Répétition** : il permet de représenter des boucles dans la présentation. Un facteur définit le nombre de répétitions.

La figure 2.15 présente un exemple d'utilisation de l'approche de composition proposée par [Hudson, 93]. Dans cette application, appelée mini_musique, d'abord un audio et une image sont présentés en parallèle. Ensuite, l'utilisateur peut choisir entre étudier quelques termes musicaux, écouter des musiques ou demander une aide. Si le lecteur ne fait rien pendant 10 secondes, un audio est présenté.

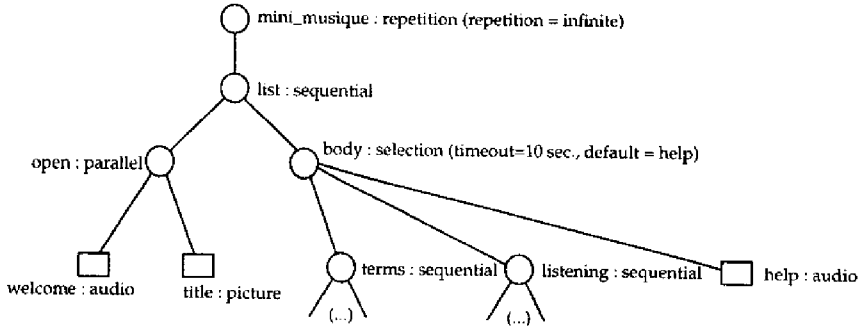


Figure 2.15 Approche de composition [Hudson, 93]

Cette approche permet l'utilisation de techniques d'analyse pour la détection de conflits de ressources partagées, par exemple, la présentation de deux audio dans un canal unique. Lors de la réalisation d'un analyseur, [Hudson, 93] propose des facilités de corrections pour résoudre le problème de conflit d'utilisation des ressources partagées. Par exemple, pour résoudre un conflit entre deux séquences audio, la présentation d'une séquence audio pourrait être plus prioritaire que l'autre audio; on pourrait aussi bien mixer les deux séquences audio dans une seule présentation.

Un autre type d'analyse proposé par [Hudson, 93] est la vérification des contraintes de synchronisation temporelles. Par exemple, un avertissement est donné à l'auteur si deux présentations sont forcées à démarrer et finir dans un même instant mais sont de durées différentes. Lors de la réalisation d'un analyseur, [Hudson, 93] propose aussi des facilités de corrections. Par exemple, l'analyseur peut proposer la modification de l'opérateur de composition pour tronquer la présentation la plus grande, ou attendre la fin des deux présentations. D'autres solutions consistent à réduire ou allonger les présentations (*stretching* ou *shrinking*).

L'approche proposée par [Hudson, 93] permet la description des structures du contenu, conceptuelle, et de présentation. Mais elle n'est pas multi-niveaux. Cette approche ne fait pas de distinction entre les objets multimédias (la structure du contenu) et les composants (la structure conceptuelle), et de ce fait la réutilisation d'objets n'est pas possible.

Cette approche force l'utilisation d'une structure uniquement hiérarchique (et linéaire) du document. Elle ne permet pas la définition de relations logiques et temporelles entre éléments de différents niveaux de la structure. Ainsi, les liens hyper-structurels (section 2.3.2) ne peuvent pas être représentés par cette approche. De plus, elle ne propose pas de méthodes de tolérance de synchronisation.

2.4.5 Modèles basés sur l'approche de composition via des points de références

2.4.5.1 L'approche orientée objet de [Steinmetz, 90]

Dans [Steinmetz, 90], l'auteur propose un modèle multimédia orienté-objet qui utilise une composition via des points de référence (section 2.3.5.2). Dans cette approche, les présentations, les interactions et autres activités d'une application sont modélisées par des objets actifs.

Pendant l'exécution, un objet exécute des opérations de synchronisation, afin d'assurer l'ordre temporel des événements. Un *event stamp* est défini comme un point de synchronisation lors de l'exécution d'une opération de synchronisation. Un *event stamp* peut être vu comme un point de référence de l'objet.

Dans les systèmes multimédias répartis, rien ne peut être dit à propos du délai entre deux *event stamps* qui doivent être synchronisés. [Steinmetz, 90] introduit trois concepts pour la définition de contraintes temporelles sur ce délai : *Timemin*, le délai minimum acceptable entre deux événements de synchroni-

sation; *Timeave*, le délai idéal entre événements de synchronisation; *Timemax* ou *time-out*, le délai maximum acceptable entre événements de synchronisation.

[Steinmetz, 90] introduit aussi le concept de *Restricted Blocking* qui permet la définition du comportement d'objets en attente d'un point de synchronisation. Par exemple, dans le cas d'une présentation vidéo, cette action peut être la présentation de la dernière trame de la séquence vidéo (ou la présentation d'une autre vidéo) jusqu'à la réalisation de la synchronisation.

La figure 2.16 présente un exemple de programmation de synchronisation en utilisant le modèle proposé par [Steinmetz, 90]. Dans cet exemple, deux objets, une séquence vidéo A et un audio B, ont leurs fins de présentation comme point de synchronisation. Le délai de synchronisation entre ces deux objets devrait être entre 1 s de l'objet audio précédant l'objet vidéo et de 2 s de l'objet vidéo précédant l'objet audio. Si l'objet audio atteint le point de synchronisation avant l'objet vidéo, mais pendant l'intervalle admissible, une musique *musique_Bach* est présentée jusqu'à la réalisation de la synchronisation. Dans le cas de l'objet vidéo, la dernière image sera présentée. Une procédure d'exception est exécutée quand un *event stamp* arrive hors de l'intervalle admissible.

Dans ce modèle, au contraire de l'approche via des points de référence, la définition de délais entre présentations est possible grâce aux fonctions *Timemin*, *Timeave* et *Timemax*. De plus, l'auteur peut spécifier les limites de tolérance de synchronisation et il peut aussi spécifier des traitements d'exception lors de la violation de ces contraintes.

```
program of object_A :                    program of object_B :
-- full-motion video                     -- audio
-- from WORM                             -- from CD_ROM
play (experiment_fluid)                  play (experiment_fluid)
...                                      ...
SYNCHRONIZE                              SYNCHRONIZE
    WITH object_B AT end                     WITH object_A AT end
    MODE restricted_blocking                 MODE restricted_blocking
    WHILE_WAITING display_last_image         WHILE_WAITING play (music_Bach)
    TIMEMIN 0                                TIMEMIN 0
    TIMEMAX 1s                               TIMEMAX 2s
    TIMEAVE 0                                TIMEAVE 0
    EXCEPTION display_last_image             EXCEPTION play (music_Bach)
```

Figure 2.16 Exemple de programmation de synchronisation proposé par [Steinmetz, 90]

L'approche proposée par [Steinmetz, 90] présente cependant certaines limitations :

- Cette approche n'offre pas une représentation explicite des interactions et ne dispose pas de mécanismes de structuration.
- L'auteur ne propose pas de méthode d'analyse temporelle.
- Cette approche ne suit pas un modèle multi-niveaux.

2.4.5.2 Le système MODE

Le système MODE (*Multimedia Objects in a Distributed Environment*) [Blakowski, 92] est composé de deux outils orientés-objets qui permettent la spécification graphique et l'exécution de présentations multimédias synchronisées. Dans cette approche, les classes d'objets peuvent avoir un ou plusieurs attributs des classes *Information*, *Présentation* et *Transport*. Un objet de la classe *Information* représente une unité d'information de base. Si les objets *Information* sont présentés, ils génèrent un ou plusieurs objets *Présentation*. Un objet *Présentation* contient toutes les données et les méthodes nécessaires à la présentation de l'information. Si un objet doit être présenté dans un nœud distant, l'objet *Information* ou les objets *Présentation* appropriés sont convertis dans un objet *Transport* et transportés vers le nœud distant. Un objet *Transport* contient toutes les méthodes et les données nécessaires pour transférer, com-

presser et décompresser un objet.

A chaque présentation particulière du document, une ou plusieurs présentations alternatives peuvent être spécifiées. Ces présentations peuvent être choisies si la présentation préférée ne peut pas être utilisée à cause de limitations du système client ou du réseau de communication. Par exemple, si un système client n'est pas capable d'afficher une certaine information sonore, une information textuelle remplace une séquence audio.

Le système MODE utilise une approche de synchronisation via des points de référence étendus afin de permettre la manipulation d'intervalles temporels fixes (de délais), d'objets avec durées imprévisibles, et de conditions qui peuvent être générées par l'environnement hétérogène réparti. Deux objets spéciaux sont proposés :

- L'objet *timer*, un objet dynamique virtuel où les points de référence sont séparés d'un millième de seconde.
- L'objet *interaction*, qui modélise la communication avec l'utilisateur. Ces objets peuvent invoquer des méthodes qui peuvent causer le démarrage, l'arrêt, la continuation et la fin d'une présentation, la simulation de l'avance-rapide ou du retour d'une présentation, et le saut vers un point d'une présentation.

De manière similaire au concept de *Restricted Blocking* de [Steinmetz, 90], le système MODE propose l'utilisation de trois types d'actions pour définir le comportement des présentations qui doivent être synchronisées :

- Action d'attente : une action peut être exécutée lorsqu'une présentation arrive dans le point de synchronisation et attend la réalisation de la synchronisation. Les actions possibles sont : continuer la présentation de la dernière position, suspendre, ou annuler le point de synchronisation;
- Action d'accélération : une action peut être exécutée sur la présentation en retard, par exemple, accélérer la vitesse de présentation ou sauter vers le point de synchronisation;
- Action de saut : quand une présentation n'arrive pas à temps dans le point de synchronisation, il est possible de sauter vers ce point.

Un éditeur graphique a été construit à fin de faciliter la spécification des relations temporelles et spatiales entre les objets. Cet éditeur présente trois visions du document :

- la vision de présentation, à partir de laquelle l'auteur peut exécuter un objet unique;
- la vision temporelle, qui présente tous les objets (représentés par des rectangles) alignés dans un ordre temporel et les points de synchronisation entre les objets (notés par des lignes verticales);
- la vision de composition visuelle (*layout*), à partir de laquelle l'auteur peut éditer les caractéristiques spatiales de présentation dans un point dans le temps.

Les limitations du système MODE sont les suivantes :

- Il n'est pas basé sur une approche formelle et ne contient aucune méthode d'analyse du comportement logique et temporel d'un document.
- Ce système n'est pas appropriée pour la description de la structure conceptuelle des documents hypermédias, parce qu'il ne propose aucun mécanisme de structuration.

2.4.5.3 Le système Firefly

Le Firefly ([Buchanan, 92], [Buchanan, 93]) est un système utilisé pour la construction de documents multimédias et hypermédias. Dans ce système, un document est composé de trois parties[8] : les segments médias (ou les données primitives); la spécification niveau-média qui décrit le comportement temporel individuel de chaque segment média; et une spécification niveau-document qui décrit le

document.

La spécification niveau-média décrit le comportement temporel des segments médias. La description niveau-média d'un segment média est appelé *item média* et englobe :

- Les *événements*, qui représentent des points dans le temps (prévisibles ou imprévisibles) définis pendant la présentation d'une information (par exemple, le départ d'une vidéo, la fin d'un logiciel ou les interactions avec l'utilisateur).
- Les *durées*, qui spécifient la durée entre deux événements consécutifs définis dans une présentation. Une durée est décrite par un intervalle qui définit une durée minimale, optimale et maximale. Si ces valeurs ne sont pas égales, la durée de présentation d'une information est dite modulable. Cette contrainte temporelle n'est pas utilisée au moment de la présentation du document, mais seulement pour déterminer (dans une phase de compilation) la durée idéale de présentation.
- Les *coûts de réduction et d'allongement*, qui sont associés à des durées modulables pour spécifier la pénalité pour sélectionner une durée autre que l'optimale. Des méthodes de réduction et d'allongement sont définies pour chaque présentation modulable (par exemple, une information peut voir sa durée de présentation changée par la modification de la vitesse de présentation).
- Une *référence*, qui pointe vers les segments médias réels décrits par l'*item média*.

La spécification niveau-document décrit le comportement temporel d'un document et est basée sur une approche de composition via des points de référence. Les composants de la spécification niveau-document sont les suivants :

- Des *items médias* présentés dans le document (décrits par le niveau-média).
- Des *contraintes temporelles* qui représentent les relations temporelles entre les événements. Chaque restriction temporelle est définie comme une relation entre un événement source et un événement destination. Il y a deux classes de relations temporelles :
  - les égalités temporelles, pour exprimer que deux événements, A et B, doivent arriver de façon simultanée, ou que A doit précéder B d'un certain temps;
  - les inégalités temporelles, pour exprimer qu'un événement A doit précéder B d'une durée non spécifiée, ou d'une durée maximale, ou d'une durée minimale et maximale. Les contraintes d'inégalités temporelles permettent la définition d'une tolérance lors de la synchronisation.
- Des *opérations* qui peuvent être associées à des événements. Ces opérations permettent de contrôler le comportement des présentations. Ces opérations peuvent changer le temps (comme le changement de la vitesse de présentation), ou ne pas changer le temps (comme l'affectation d'un nouveau volume à une audio ou la réalisation un *zoom* sur une image).
- Des *durées et coûts*, qui permettent la personnalisation des durées et des coûts des segments médias pour un document spécifique. Si l'auteur ne définit pas ces informations, les durées et les coûts définis dans le niveau-média seront adoptés.
- Des *contrôles d'événements imprévisibles*, qui permettent à l'auteur d'activer ou de désactiver les événements imprévisibles du document.

Dans l'approche ligne temporelle, l'auteur d'un document doit éditer les données primitives pour réduire ou allonger la durée d'une présentation. L'approche proposée par [Buchanan, 93] fournit les moyens pour la réalisation d'une réduction/allongement automatique des durées de présentation. Cela permet la synchronisation de points internes multiples, même si les données ne sont pas précisément à la durée désirée. La détermination du temps auquel les événements doivent arriver est basée

8. Les références [Buchanan, 93] et [Buchanan, 92] diffèrent sur la description de ces parties. Nous avons adopté ici celle de [Buchanan, 93].

sur les contraintes de durée de présentation et les contraintes de synchronisation. Cela est résolu par l'utilisation de la programmation linéaire qui minimise le coût total de réduction et d'allongement de la durée des présentations (une erreur arrive si un temps ne peut pas être associé à des événements de façon que les contraintes du documents soient satisfaites). Ainsi, les durées des présentations peuvent être ajustées afin de produire un scénario temporel optimal.

Le système Firefly propose une représentation graphique à partir de laquelle l'auteur peut créer des documents hypermédias. Par exemple, la figure 2.17 décrit un cours d'électricité et de magnétisme. Cette représentation graphique est basée sur les lignes temporelles (section 2.3.5.1). Chaque item média a sa propre ligne temporelle (les items média ayant une durée imprévisible sont représentés par des lignes pointillées). Les nœuds du graphe représentent des événements : les rectangles représentent les événements de départ et d'arrêt (ainsi chaque item est représenté par deux rectangle connectés), et les cercles représentent des événements internes. Les contraintes temporelles entre ces événements sont représentées par des arcs typés qui lient les événements.

Les limitations du système Firefly sont les suivantes :

- Cette approche a été initialement définie pour les documents multimédias et elle ne fournit pas de moyens de représenter le niveau des nœuds hypermédias; cette approche n'est donc pas structurée.
- Les liens hypermédias ne sont pas représentés dans la vision temporelle du document et on n'a pas une indication nette du moment où un lien est actif. Par exemple, l'événement asynchrone *Define_Electrolyte* de la figure 2.17 arrive lors d'une interaction avec l'utilisateur, mais l'instant d'activation (désactivation) de l'ancre associée à cet événement n'est pas représenté.
- Ce système permet la définition de contraintes temporelles entre événements, mais il ne fournit aucun mécanisme de contrôle lors de la violation de ces contraintes.
- Aucune mention n'est faite quant à la description des caractéristiques spatiales et sonores des présentations.

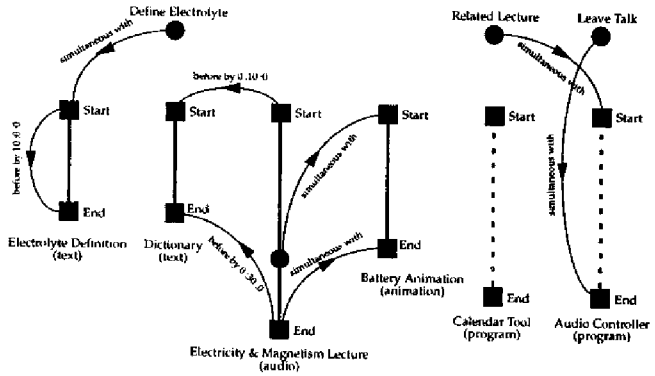


Figure 2.17 Représentation graphique du système Firefly [Buchanan, 93]

2.4.6 Systèmes basés sur les organigrammes

La création d'un document hypermédia par l'utilisation d'une approche organigramme est similaire à sa programmation (p.e., en utilisant le langage C), mais avec l'aide d'une interface graphique. Cette interface fournit des icônes de haut niveau, afin de visualiser et manipuler la structure du document.

Un ensemble d'icônes est arrangé dans un graphe qui spécifie des interactions et des chemins de contrôle de présentation d'un document hypermédia. En général, les fonctionnalités associées à chaque icône peuvent être modifiées en utilisant des menus et des éditeurs associés. La figure 2.18 présente un exemple d'organigramme.

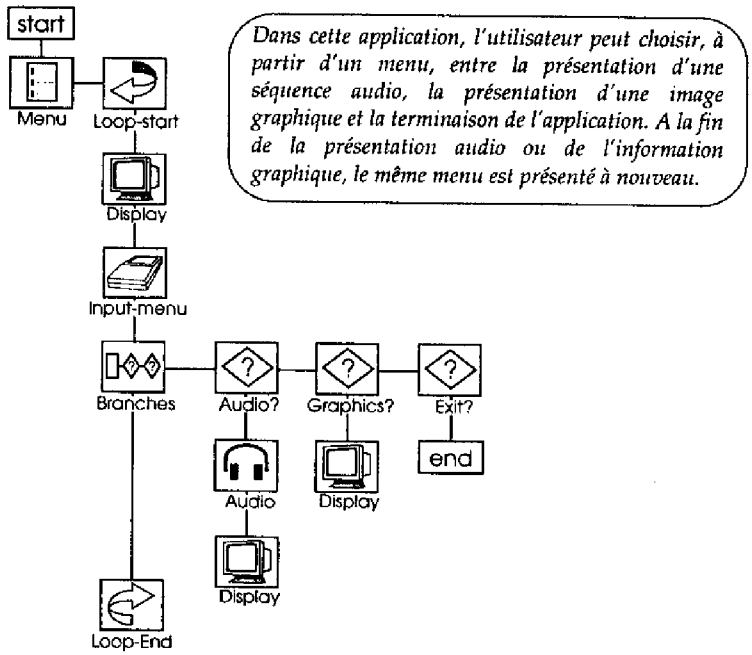


Figure 2.18 Exemple d'organigramme pour la composition de documents hypermédias

L'approche organigramme de création d'applications hypermédias est simple d'utilisation pour des petites applications. Par contre, dans des applications complexes, la compréhension et la manipulation sont compliquées.

L'approche organigramme est adoptée, par exemple, par les logiciels IconAuthor [AimTech, 93] et Eyes M/M [Eyes, 92].

2.4.6.1 IconAuthor

Dans le logiciel IconAuthor [AimTech, 93], les organigrammes sont construits à partir d'une librairie d'icônes. Ces icônes représentent des actions comparables à celles rencontrées dans les langages de programmation conventionnels (boucles, si-alors, procédures). Les icônes représentent aussi des fonctions de présentations et d'interactions.

Dans la librairie d'icônes, les icônes sont groupées dans les familles suivantes :

- **Flux** : elles contrôlent le flux d'exécution d'une application, c'est-à-dire que ces icônes déterminent quelle est la prochaine icône à exécuter (p.e., branches, si-alors, boucles, menus).
- **Entrée** : ces icônes permettent aux utilisateurs d'interagir avec l'application.
- **Sortie** : ces icônes contrôlent la sortie qui est générée par l'application. Les sorties peuvent être n'importe quel type d'information textuelle ou graphique qui est présentée à l'écran.
- **Données** : des icônes qui permettent la manipulation de plusieurs types de données. Elles peuvent stocker, manipuler et accéder des données originaires de plusieurs sources, comme une entrée de l'utilisateur ou une base de données.
- **Multimédia** : ces icônes représentent une variété de médias de présentation.
- **Extensions** : ces icônes permettent à l'application multimédia d'exécuter ou de communiquer avec d'autres applications (p.e., exécuter le *snapshot*[9]).
- **Clientèle** : icônes définies par l'utilisateur.

9. *Snapshot* est un utilitaire disponible dans l'environnement OpenWindows qui permet la capture d'une partie de l'écran.

L'auteur d'une application multimédia peut changer les valeurs des paramètres associés aux icônes via des boîtes de dialogue. Par exemple, à partir d'une boîte de dialogue, l'auteur peut définir la position et la couleur d'une fenêtre (représentée par une icône de sortie fenêtre). Le logiciel IconAuthor contient aussi des éditeurs pour la création de textes, graphiques et animations.

Dans l'approche adoptée par IconAuthor, l'organigramme spécifie le flux de contrôle de présentations monomédias. Avec l'utilisation exclusive des icônes proposées par ce logiciel, l'auteur ne peut représenter ni le parallélisme ni les relations temporelles entre présentations. Pour représenter cette composition temporelle, l'auteur doit créer de nouvelles icônes (via une icône Clientèle).

2.4.6.2 Eyes M/M

Dans l'environnement d'édition d'applications multimédias Eyes M/M [Eyes, 92], les icônes sont combinées dans un graphe qui représente le flux temporel de présentation. Ce graphe est équivalent aux réseaux de Pétri temporisés [Koegel, 92] : les branches représentent des chemins d'exécution parallèle; les nœuds représentent des points de synchronisation de deux ou plusieurs chemins parallèles.

Il y a trois catégories d'icônes :

- les icônes actions représentent les éléments d'une présentation, comme une séquence vidéo ou une image. Ces icônes fournissent deux types de synchronisation : un délai optionnel qui précède l'action et un mode asynchrone, dans lequel l'exécution du graphe n'est pas bloquée pendant le traitement d'une action.
- les icônes entrées représentent des mécanismes d'interaction avec le lecteur du document;
- au contraire du logiciel IconAuthor, le Eyes M/M permet l'expression du parallélisme. Les icônes de contrôle de flux représentent des mécanismes de synchronisation inter-flux. Ce sont les icônes AND et OR. L'icône OR permet l'attente de la fin d'au moins un flux d'entrée pour activer le flux de sortie (les autres flux d'entrée ne sont pas interrompus). L'icône AND permet l'attente de la fin de tous les flux d'entrée pour activer le flux de sortie.

2.4.7 Modèles basés sur les réseaux de Petri

Dans cette section, nous analysons dix modèles pour le multimédia et l'hypermédia basés sur les réseaux de Petri : OCPN [Little, 90] et ses extensions ([Iino, 94], [Prabhakaran, 93], [Qazi, 93]), RdPFT [Diaz, 94b], RTSM [Yang, 96], Trellis temporisé [Stotts, 90], RdPHFT [Sénac, 95a], MORENA [Botafogo, 95], et enfin PHPN [Wang, 95].

2.4.7.1 OCPN

Little et al. [Little, 90] proposent une technique de spécification formelle pour la composition temporelle de documents multimédias. Cette spécification est ensuite traduite en un schéma de base de données afin de permette le stockage et l'accès aux composants d'un document multimédia. [Little,90] propose aussi un algorithme de présentation du document.

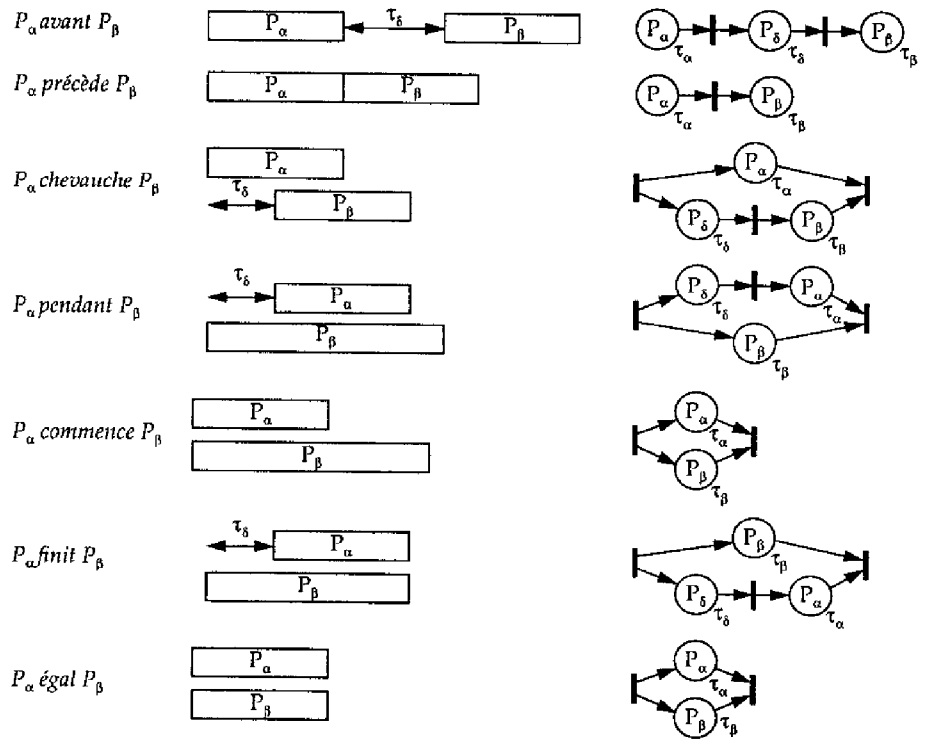


Figure 2.19 Les constructeurs OCPN [Little, 90]

Le modèle de composition temporelle

[Little, 90] propose un modèle de réseau de Petri temporisé (réseau de Petri avec temporisation dans les places), appelé *Object Composition Petri Nets* (OCPN), pour la spécification de scénarios de synchronisation de présentations multimédias statiques. Dans ce modèle, les scénarios de synchronisation sont construits en utilisant une approche de structuration minimaliste, basée sur sept constructeurs fondamentaux de synchronisation. Ces constructeurs correspondent aux sept relations fondamentales entre deux intervalles (section 2.3.2.3). La figure 2.19 présente les correspondants OCPN de ces relations.

Chaque place du réseau OCPN représente le traitement d'un objet multimédia ou bien un délai. Une place est associée aux ressources nécessaires et le temps requis pour le traitement modélisé. Les transitions d'un réseau OCPN représentent les points de synchronisation d'un scénario multimédia.

A cause de l'utilisation exclusive des sept schémas de synchronisation, les réseaux OCPN ont une structure de graphe marqué sans circuit. Ces réseaux sont aussi définis comme saufs, et ainsi, ils offrent de bonnes propriétés pour la vérification des propriétés temporelles [Diaz, 94b].

Le schéma de base de données

Le modèle OCPN est utilisé comme une technique de spécification visuelle de scénarios de présentation multimédia. Afin de permettre le stockage, la communication, et la reproduction d'un document multimédia spécifiée par un réseau OCPN, un schéma de base de données a été défini. Ce schéma est le modèle physique du modèle logique OCPN et préserve la sémantique du modèle OCPN.

Le schéma de base de données de [Little, 90] est basé sur l'approche de composition hiérarchique (section 2.3.5.3). Dans cette approche, l'arbre de composition hiérarchique a trois types de nœuds :

- Les nœuds terminaux, qui représentent des présentations d'un objet multimédia (dans leur plus fine granularité de synchronisation). Cet objet est spécifié par un type de média (texte, image, vidéo, etc.), par un attribut non spécifié et par une référence aux données primitives.

- Les nœuds non-terminaux, qui spécifient des relations temporelles entre deux présentations. Les relations temporelles possibles sont les 7 relations fondamentales entre deux intervalles [Allen, 83] (figure 2.19). Un nœud non-terminal spécifie : un opérateur de composition (*avant, précède, chevauche, pendant, commence, finit* et *égal*); les durées des nœuds fils P_α et P_β (τ_α et τ_β); et une durée additionnelle permettant la définition de délais (τ_δ).
- Les nœuds *meta*, qui spécifient aussi des relations temporelles, mais entre un nombre quelconque de présentations.

La figure 2.20 présente un exemple simplifié de composition temporelle en utilisant l'approche de composition hiérarchique de [Little, 90]. Cette composition spécifie que trois sections de présentation de diapositive, accompagnées d'une information audio, seront présentées en séquence. Et dans chaque section, les présentations de la diapositive et de l'information audio seront traitées de façon parallèle et auront les mêmes durées.

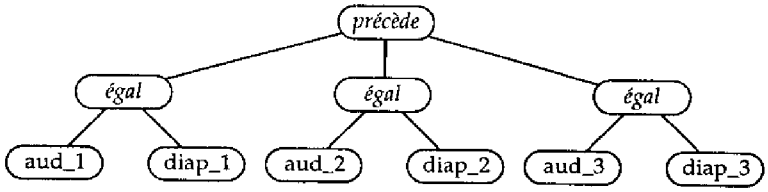


Figure 2.20 Composition hiérarchique de [Little, 90]

La technique de modélisation, de stockage, et de présentation de documents multimédias proposée par [Little, 90] présente quelques limitations :

- Seule l'expression des relations temporelles est possible. Les relations conditionnelles ne sont pas supportées par le modèle OCPN.
- Le modèle OCPN ne permet pas la spécification de tolérances temporelles admissibles pour la synchronisation. Dans ce modèle, une présentation a une durée nominale. L'utilisation d'une durée idéale peut être vue comme «non réaliste» dans le domaine des systèmes hypermédias répartis. Des intervalles temporels seraient plus appropriés pour exprimer la variabilité et l'imprécision de tels systèmes.
- Ce modèle ne permet pas l'expression de mécanismes d'interactions avec l'utilisateur du document, une des caractéristiques fondamentales des documents hypermédias. Ainsi, le modèle OCPN n'est pas approprié pour la modélisation de documents hypermédias.
- Ce modèle ne permet pas la description des caractéristiques des présentations modélisées par les places.

2.4.7.2 OCPN avec caractérisation des présentations

Dans [Iino, 94], les auteurs proposent une extension au modèle OCPN permettant la spécification des caractéristiques spatiales et sonores des présentations qui composent un document multimédia. Ils ont défini des applications qui associent une place du réseau aux caractéristiques spatiales et sonores de la présentation que la place représente. Les caractéristiques qui peuvent être définies sont :

- La région de présentation de la place (la position et la taille d'une fenêtre pour une image, ou un dispositif de sortie pour une information audio);
- Un vecteur de priorité, où chaque élément du vecteur est composé de deux valeurs, la priorité qui définit l'ordre visuel de présentation (une présentation de plus haute priorité est affichée au-dessus d'une présentation de plus basse priorité), et la période pendant laquelle cette priorité est valable.

Ainsi, on peut associer plusieurs priorités pendant la présentation.

Pour les informations audio, ce vecteur représente les tonalités relatives et les volumes de toutes les informations audio dans les systèmes de sortie multi-canaux.

- Un ensemble ordonné d'opérations spatiales *Crop* (présenter une partie de l'information) et *Scale* (changement de taille) qui seront effectuées sur la présentation.

Cette extension ne résout aucune des limitations du modèle OCPN citées précédemment à part la description des caractéristiques individuelles de présentation des composants du document.

2.4.7.3 A-OCPN

Dans [Prabhakaran, 93], les auteurs proposent une extension au modèle OCPN, appelé A-OCPN (*Augmented OCPN*), qui permet la spécification des caractéristiques de synchronisation multimédia avec participation dynamique de l'utilisateur. Dans A-OCPN, les arcs du type *escape* (proposé par [Zuberek, 85]) sont utilisés pour la modélisation des interruptions préemptives des présentations.

La préemption de l'exécution d'une place active peut modifier la durée temporelle associée à la place. Cette modification dépend de la nature de l'interruption. Il y a trois types d'interruptions : préemptive avec report à plus tard de l'exécution, où la durée est réduite du temps écoulé jusqu'à l'interruption; préemptive avec fin d'exécution (la durée de traitement n'est pas changée); préemptive avec modification du temps d'exécution (une nouvelle valeur est attribuée à la durée d'une place).

Les types d'interactions modélisés par le modèle A-OCPN sont : interrompre, geler, repartir, renverser la direction, et changer la vitesse d'une présentation. Ces types d'interactions sont propres aux documents multimédias. Les liens des documents hypermédias ne sont pas représentables par le modèle A-OCPN. De plus, comme les caractéristiques temporelles d'une présentation peuvent changer dans le temps et d'une façon indéterminée (l'utilisateur peut interrompre une présentation à n'importe quel instant), le modèle A-OCPN ne permet pas une analyse temporelle des documents multimédias.

2.4.7.4 XOCPN

Qazi et al. [Qazi, 93] proposent une autre extension au modèle OCPN, appelée XOCPN (*eXtended OCPN*), qui spécifie explicitement les besoins de communication, et les opérations de synchronisation et de transmission nécessaires pour l'accès, la composition et la présentation d'objets multimédias.

Les correspondances entre les éléments OCPN et XOCPN sont présentées dans la figure 2.21. Cette figure illustre les deux types de place du modèle XOCPN :

- Les places ressources, qui modélisent des actions de contrôle. Ces actions sont :
  - *Resource-setup* : cette fonction sert à ouvrir un canal virtuel (de communication), à négocier les attributs du canal à partir des paramètres de qualité de service (définis par l'utilisateur), à réserver des *buffers* de réception, et à préparer les dispositifs de sortie. Par exemple, les places R de la figure 2.21a représentent des actions de préparation des présentations P_α et P_β.
  - *Resource-release* : cette fonction sert à libérer les ressources à la fin d'une présentation. Par exemple, les places L de la figure 2.21b représentent des actions de libération des ressources qui ont été nécessaires aux présentations P_α et P_β.
  - *Interstream-synchronize* : cette fonction implémente le mécanisme de synchronisation inter-flux.

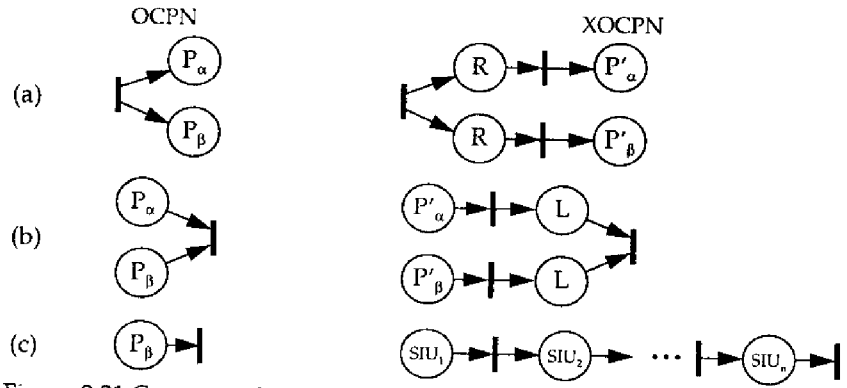


Figure 2.21 Correspondances entre les éléments OCPN et XOCPN [Qazi, 93]

- Les places objets, qui modélisent des processus de présentation ou de transmission des objets. Comme dans le modèle OCPN, chaque objet est associé à un temps nominal qui décrit la durée de traitement. Dans le XOCPN, un objet est divisé dans une séquence de petites unités, appelées SIU (*Synchronization Interval Unit*), par exemple, pour une séquence vidéo, une SIU peut correspondre à une trame. Ce type de place est illustré dans la figure 2.21c.

  Chaque place objet dénote une des fonctions suivantes :
  - *SIU-playout* : cette fonction présente la SIU associée au(x) dispositif(s) de sortie;
  - *SIU-transmit* : cette fonction réalise la transmission de la SIU associée à un canal virtuel.

  Dans un système DMIS (*Distributed Multimedia Information System*), les sites émetteur et récepteur ont chacun un réseau XOCPN associé. La structure du réseau XOCPN de l'émetteur et du récepteur est la même. La différence est le type de fonction associée à chaque place : quand une place dénote l'action *SIU-playout* dans le XOCPN récepteur, elle dénote *SIU-transmit* dans le XOCPN émetteur.

Pour la synchronisation continue, le modèle XOCPN propose l'utilisation de places de contrôle dénotant l'action *Interstream-synchronize*. Ces places, appelées IPP (*Interstream Pacing Points*) sont insérées dans un point pré-calculé dans la séquence de SIUs. Par exemple, la figure 2.22 présente la synchronisation continue d'une séquence vidéo et d'une séquence audio. Il y a deux types de politique :

- *Interstream Blocking*, où le flux plus en avance attend le plus en retard. Cette politique est représentée dans la figure 2.22.
- *Interstream Non-Blocking*, quand une place IPP est marquée, une information sur l'état actuel[10] du synchronisme entre les flux est capturée. Basée sur cette information, une action de résynchonisation peut être réalisée (par exemple, sauter quelques SIUs du flux le plus en retard, retarder le flux le plus en avance, etc.). De toute façon, ce type de synchronisation n'est pas formalisé.

10. Cette information peut être dérivée, par exemple, à partir des numéros des SIUs en cours de présentation.

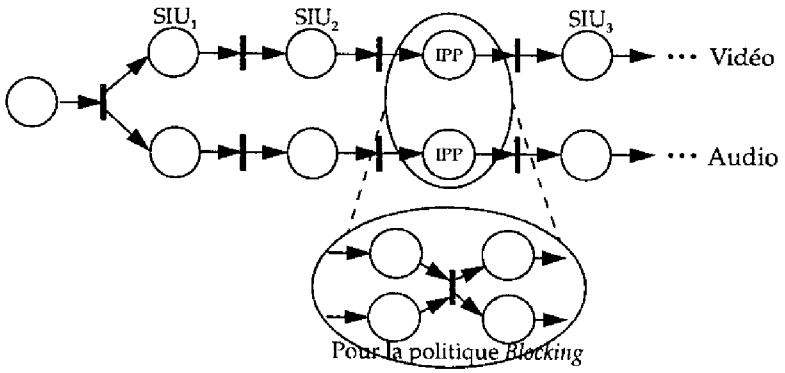


Figure 2.22 La synchronisation inter-flux du modèle XOCPN

Dans la perspective du modèle XOCPN, chaque SIU doit être délivrée à temps pour satisfaire son *deadline* de présentation. Lors de la réception de ces SIU, deux politiques de synchronisation intra-flux peuvent être spécifiées :

- *Blocking* : si un SIU n'arrive pas à temps, la présentation peut être bloquée ou arrêtée jusqu'à l'arrivée de la SIU;
- *Restricted Blocking* : si un SIU n'arrive pas à temps, la présentation est bloquée ou arrêtée pendant un intervalle de temps. Si la SIU n'arrive pas pendant cet intervalle, la SIU précédente est présentée une autre fois, ou la SIU manquante est ignorée.

Chaque place SIU est une place abstraite qui dénote un sous-réseau spécifiant une des politiques ci-dessus.

Le modèle XOCPN permet la spécification des synchronisations intra et inter-médias dans un système DMIS (*Distributed Multimedia Information System*) client-serveur sans interaction de l'utilisateur. Ce modèle (comme le modèle OCPN) ne permet pas la spécification de liens hypermédias. Une autre limitation de ce modèle est qu'il considère tous les flux multimédias comme arrivant dans une destination unique (ce qui n'est pas vrai pour toutes les applications multimédias, en particulier pour la téléconférence).

2.4.7.5 RTSM

Le modèle RTSM (*Real-Time Synchronization Model*) [Yang, 96] est une extension du modèle de réseau de Petri temporisé (réseau de Petri avec temporisation dans les places). Ce modèle prend en considération les dérives temporelles dans les schémas de synchronisation multimédias inhérents aux systèmes faiblement synchrones (causées par les retards générés par le réseau de communication, par les délais d'accès aux bases de données, *etc.*).

Dans le modèle RTSM, les places sont utilisées pour représenter une unité d'information (comme des trames vidéos, des échantillons d'audio, des séquences de caractères) et leur actions correspondantes (comme la présentation de l'information).

RTSM définit deux types de place régulières et forcées. Les places forcées sont représentées graphiquement par un double cercle (figure 2.23). Dès qu'un jeton marquant une place forcée devient débloqué, la transition de sortie de cette place est immédiatement tirée sans regarder l'état de ses autres places d'entrée (même s'il y a des places d'entrée non marquées). Par exemple, la transition t_3 de la figure 2.23 sera tirée quand l'audio A1 (ou T1) finit, même si la place V2 n'est pas marquée. Le modèle définit des règles (actions) de retour en arrière afin d'effacer tous les jetons obsolètes après le tir d'une transition. La figure 2.24 illustre l'action de retour en arrière exécutée après le tir de la transition t_3. Ces règles ne sont pas conformes au modèle de réseaux de Petri.

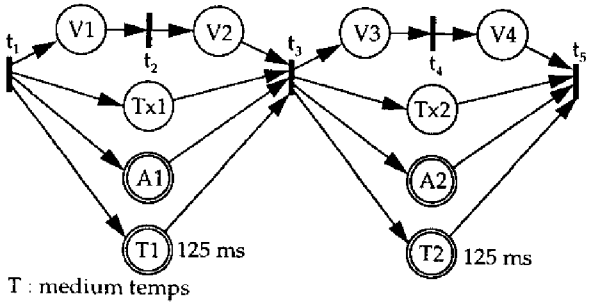
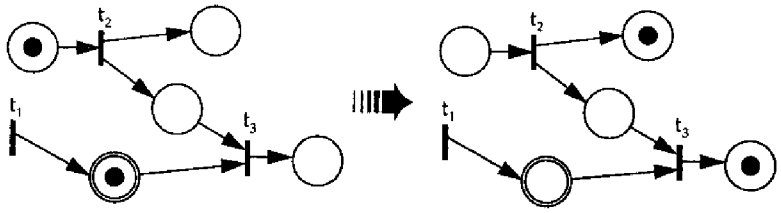


Figure 2.23 Exemple de spécification RTSM



a) Avant le tir de t3 a) Après le tir de t3

Figure 2.24 Action de retour en arrière

Par définition, les places forcées contrôlent l'avance du temps des applications multimédias. Ainsi, si les places d'un médium spécifique, telle que la voix, sont définies comme des places forcées, alors ce médium ne sera pas bloqué à cause de la latence de transmission des autres médias qui sont synchronisés. Afin de définir des contraintes temporelles, le modèle RTSM introduit le temps comme un médium, nommé *médium temps* (il peut être défini comme une place régulière ou forcée). Le médium temps est un médium virtuel et il contient une durée déterministe qui spécifie une contrainte temps réel entre deux transitions. Par exemple, la transition t_3 de la figure 2.23 sera tirée quand l'audio A1 finit ou si la durée de cette présentation dépasse 125 ms.

Ce modèle ne propose aucune méthode d'analyse afin de vérifier les contraintes temporelles des scénarios multimédias. De plus, ce modèle n'est pas approprié pour la spécification de documents hypermédias : comme le modèle OCPN, RTSM ne permet pas l'expression de mécanismes d'interactions avec l'utilisateur du document.

2.4.7.6 Trellis

Le modèle Trellis temporisé [Stotts, 90] utilise la structure et les règles d'exécution des réseaux de Petri temporels (réseau de Petri avec des intervalles temporels associés aux transitions) pour la modélisation de la structure conceptuelle des documents hypermédias.

Dans le modèle Trellis, une place spécifie un fragment d'information, appelé contenu (texte, graphique, vidéo, audio ou exécutable), ou un autre hyper-programme (un sous-réseau). Elle spécifie aussi la fenêtre logique dans laquelle l'information sera présentée (fenêtre, canal audio, etc.). Quand une place est marquée, le contenu de cette place est présenté dans la fenêtre. Une spécification hiérarchique est possible grâce à l'affectation d'une hyper-structure comme le contenu d'une place.

Chaque transition du modèle Trellis est associée à un hyper-lien temporisé. Une transition sensibilisée est tirée, à l'intérieur de l'intervalle temporel du lien, à l'instant où le lecteur du document sélectionne le bouton associé à la transition.

Le modèle Trellis permet la vérification de la synchronisation logique du document à partir d'une adaptation de la technique de vérification appelée *model checking* [Clarke, 86]. Cette approche permet la

vérification de propriétés des relations logiques du document exprimées dans la notation de logique temporelle CTL. Un analyseur vérifie si la structure du réseau maintient la validité de la formule qui dénote la propriété. Cette analyse est basée sur le graphe d'accessibilité du réseau.

Le modèle Trellis permet la spécification des composants d'un document et de ses relations logiques. Mais ce modèle ne permet pas la représentation des relations temporelles entre les composants à l'intérieur de nœuds hypermédias. Ainsi, le modèle Trellis considère les nœuds hypermédias comme des nœuds hypertextes statiques (sans contraintes de synchronisation).

2.4.7.7 MORENA

Le MORENA (*Multimedia ORganization Employing a Network Approach*) [Botafogo, 95] est un outil de description et d'exécution de documents hypermédias basé sur les réseaux de Petri.

La figure 2.25 présente un exemple de réseau MORENA. Cet exemple spécifie la synchronisation d'un film, composé d'un audio et d'une vidéo, avec un ensemble de sous-titres (un ensemble de textes). La séquence vidéo contient un ensemble de trames marquées et quand une trame marquée est présentée, un sous-titre associé est aussi présenté. Pendant la présentation du film, un ensemble de boutons est présenté : «Fin» permettant l'interruption du film, «stop» permettant le gel du film, et «Dim» permettant la réduction d'un degré du volume et de l'intensité lumineuse du film.

A la différence du modèle général réseau de Petri, une transition du modèle MORENA n'est pas seulement sensibilisée quand toutes ses places d'entrée sont marquées. Dans ce modèle, l'auteur peut définir une logique ET/OU de sensibilisation de la transition. La logique de sensibilisation d'une transition est par défaut égale à OU. Par exemple, dans la figure 2.25, la transition *Fin* est sensibilisée quand les places *Film* et *Sous-titrage* sont marquées, et la transition *stop* est sensibilisée quand la place *Tech:AUDIO* ou la place *Tech1:VIDEO* est marquée. Des logiques ET/OU plus complexes peuvent être définies.

Ce modèle présente deux types de transitions : les transitions utilisateurs et les transitions système. Une transition utilisateur est associée à un bouton (p.ex., la transition *Fin* de la figure 2.25 modélise un bouton avec *Fin* comme titre) et elle est tirée quand l'utilisateur clique sur ce bouton. Une transition système spécifie une synchronisation et elle est tirée lorsqu'elle reçoit un message.

Les messages sont l'unique moyen d'interaction entre les places MORENA. Par exemple, lors du tir d'une transition, elle envoie par défaut un message *#End#* pour ses places d'entrée et *#Start#* pour ses places de sortie. L'auteur peut changer le comportement par défaut en définissant, par exemple, un message nul, qui correspond à la notation d'un poids zéro (noté par ϕ) sur les arcs d'entrée et/ou de sortie des transitions. L'auteur peut aussi associer un type à un message. Par exemple, lors du tir de la transition utilisateur *Dim* dans la figure 2.25, un message typé *«DIM»* est envoyé à la place de *Film*.

Le modèle MORENA présente deux types de places : de base et composites. Une place de base contient un médium, par exemple la place *Tech:AUDIO* dans la figure 2.25. Une place composite représente un réseau MORENA, comme la place *Film* qui représente le réseau de même nom.

Des transitions d'entrée et de sortie sont utilisées pour indiquer le départ et l'arrêt d'une place Composite. Une transition d'entrée ne contient pas de places d'entrée (p.ex., la transition <4>) et une transition de sortie ne contient pas de places de sortie (p.ex., la transition *Fin*). Les transitions d'entrée peuvent être typées (p.ex., la transition titrée par *«Dim»*) et elles ont par défaut le type *#Start#*. Quand une place composite reçoit un message, les transitions d'entrée de même type que le message reçu sont tirées. Par exemple, à la réception du message *#Start#* par la place *Film*, la transition <4> est tirée et si le message reçu est *«Dim»* la transition d'entrée typée par *«DIM»* est tirée.

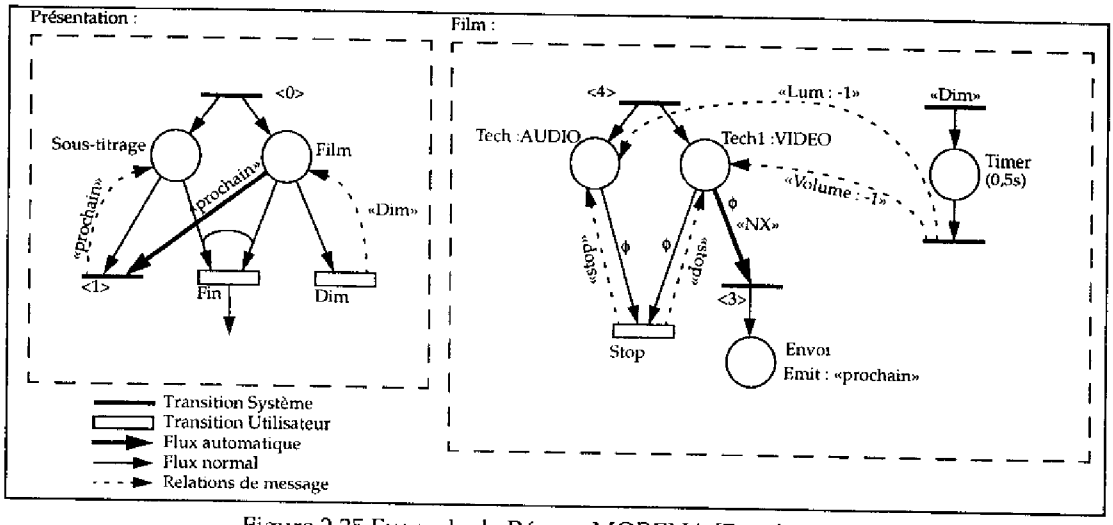


Figure 2.25 Exemple de Réseau MORENA [Botafogo, 95]

Il y deux types d'arcs dans le modèle MORENA qui définissent un flux normal ou un flux automatique. Les arcs définissant des flux automatiques sont typés. Une transition avec un flux automatique entrant (p.ex., la transition <1>) est tirée seulement quand la place d'entrée associée au flux automatique envoie un message du même type que le flux automatique. Par exemple, la transition <1> sera tirée seulement quand la place *Film* envoie un message *«prochain»*.

A partir de la notion de message, le modèle MORENA permet la spécification de synchronisations entre présentations de différents niveaux de spécification, c'est-à-dire qu'il permet la spécification de relations entre objets de différents niveaux de hiérarchie.

Les limitations de ce modèle sont les suivantes :

- Les places représentent le triplet composant/données primitives/présentation et ainsi ce modèle ne fait pas de distinction entre la structure conceptuelle, le contenu et la présentation.
- Les caractéristiques spatiales et sonores de présentation ne sont pas spécifiées et ainsi ce modèle ne permet pas la spécification de la structure de présentation.
- Il y a pas de notion explicite du temps. Ainsi, les contraintes temporelles des présentations ne peuvent pas être spécifiées.

2.4.7.8 MHPN

Le modèle MHPN (*Multimedia Hypermedia Petri Net*) [Wang, 95] lie les éléments du réseau de Petri aux objets définis par la norme MHEG (présentée dans le chapitre 3 de ce mémoire). Ce modèle est défini par un ensemble de places, d'arcs et de transitions, par un ensemble d'objets MHEG et par un ensemble d'applications qui lient les places, les arcs et les transitions aux objets MHEG.

Dans le modèle MHPN, une place peut représenter une paire données primitives/présentation, un mécanisme d'interaction ou un réseau. Chaque place est associée à un canal sur lequel la présentation va se dérouler (p.ex., une fenêtre). Un arc peut avoir un délai associé. Une transition peut être associée à un objet MHEG définissant des conditions/actions et/ou un mécanisme d'interaction.

Un des avantages de ce modèle par rapport aux précédents est la définition de conditions/actions associées aux transitions, permettant la définition de relations conditionnelles plus complexes, que ne peuvent pas être représentées par les autres modèles basés sur les réseaux de Petri présentés dans ce mémoire.

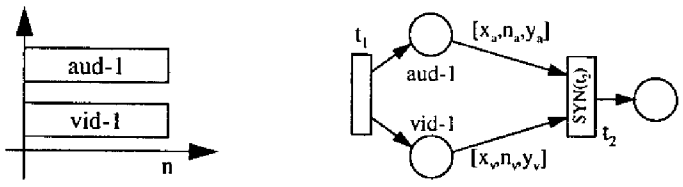
Ce modèle permet la définition d'une représentation orientée-objet d'un document hypermédia. Les classes d'objets définies par ce modèle sont les représentations des éléments du réseau de Petri (les classes Transition, Place, etc.) et des objets MHEG. Ces derniers ne sont pas des objets MHEG comme ceux définis par la norme. Ainsi, cette représentation ne correspond pas à la norme MHEG, et elle requiert de fait un interpréteur spécifique. De plus, ce modèle a été basé sur une version intermédiaire de la norme MHEG et quelques concepts du modèle MHPN ne sont plus valables.

Comme une place contient les données primitives, ces dernières ne peuvent pas être réutilisées. Une autre limitation de ce modèle est qu'il ne fournit aucune technique d'analyse.

2.4.7.9 RdPFT

Le modèle RdPFT (*Réseaux de Petri à Flux Temporels*) ([Diaz, 93], [Sénac, 94]) est une extension du modèle de réseau de Petri à arc temporels [Walter, 83] utilisée pour la modélisation de scénarios de synchronisation multimédia à l'intérieur de systèmes faiblement synchrones. Ce modèle formel a été défini en considérant que le non-déterminisme inhérent aux systèmes faiblement synchrones génère, en général, des dérives temporelles dans les schémas de synchronisation multimédias (causées par les retards générés par le réseau de communication, par les délais d'accès aux bases de données, *etc.*).

Dans le modèle RdPFT, les traitements et leurs caractéristiques temporelles sont représentés par des arcs associés à des intervalles temporels. Ces intervalles, nommés Intervalles de Validité Temporelle (IVT), sont des triplets [x, n, y], où x, n et y spécifient respectivement, la durée minimale, nominale et maximale admissibles du traitement associé. Cette modélisation permet la prise en compte du non-déterminisme des systèmes faiblement synchrones et de la variabilité temporelle admissible des traitements multimédias. Par exemple, l'arc a_1 = (aud-1, t_2) dans la figure 2.26b spécifie la présentation d'une séquence audio avec [x_a, n_a, y_a] comme IVT. Si τ est la date de marquage de *aud-1* (*i.e.*, la date d'arrivée d'un jeton dans cette place, et ainsi le départ de la présentation audio), alors le triplet [$\tau + x_a$, $\tau + n_a$, $\tau + y_a$], également dénoté [τ_1^{min}, τ_1^{nom}, τ_1^{max}], spécifie respectivement la date de terminaison minimale, nominale et maximale du traitement de l'audio. Cet intervalle est nommé Intervalle de Validité Temporelle Absolu (IVTA) de l'arc a_1.



a) Présentation Parallèle b) Spécification RdPFT

Figure 2.26 Dérive temporelle

La sémantique formelle pour la synchronisation utilisée par le modèle RdPFT a été définie en prenant en considération que la gigue temporelle introduite par les systèmes faiblement synchrones cause, dans le cas général, l'impossibilité de satisfaire des restrictions temporelles strictes. Par exemple, la ligne temporelle de la figure 2.26a spécifie que l'audio *aud-1* doit être présentée pendant la présentation de la vidéo *vid-1*. En introduisant la variabilité temporelle des systèmes faiblement synchrones, cette synchronisation stricte ne peut pas être assurée. Les intervalles temporels, adoptés par ce modèle, sont plus appropriés pour exprimer la variabilité temporelle et l'imprécision (figure 2.26b).

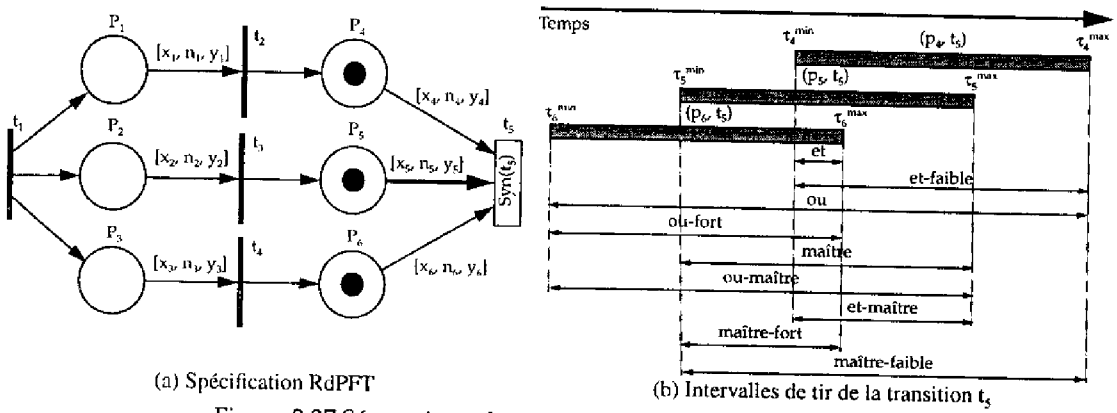
Afin de décrire les schémas de synchronisation inter-flux en prenant en compte le non déterminisme introduit par la gigue temporelle de la durée de traitement multimédia, le modèle RdPFT permet à l'auteur d'associer différentes stratégies de synchronisation aux transitions. Par exemple, l'auteur du document peut associer une stratégie de synchronisation à la transition t_2, dans la figure 2.26b, afin de

spécifier la fin des présentations *aud-1* et *vid-1*.

Le modèle RdPFT fournit trois stratégies de synchronisation de base qui consistent à attribuer une priorité à un processus de façon statique ou dynamique :

- Une stratégie de synchronisation dynamique, appelée *ou-fort*, guidée par le traitement le plus en avance. Par exemple, en associant cette stratégie à la transition t_2 de la figure 2.26b, le point de synchronisation arrive quand *aud-1* ou *vid-1* finit[11].
- Une stratégie de synchronisation dynamique, appelée *et-faible*, guidée par le traitement le plus en retard. Par exemple, en associant cette stratégie à la transition t_2 de la figure 2.26b, le point de synchronisation arrive quand *aud-1* et *vid-1* finissent.
- Une stratégie de synchronisation statique, appelée *maître*, guidée par un traitement sélectionné. Par exemple, dans la figure 2.26b, si l'auteur définit *(aud-1, t_2)* comme le processus maître alors le point de synchronisation arrive quand *aud-1* finit.

Ces 3 stratégies de synchronisation fondamentales amènent 9 règles de tir obtenues à partir d'une combinaison consistante et complète des IVTA des arcs d'entrée d'une transition marquée. La figure 2.27 présente les intervalles de tir d'une transition t pour toutes les stratégies de synchronisation qui peuvent être associées à cette transition. Dans cette figure, si τ_i est la date de marquage de la place p_i alors $[\tau_i+x_i, \tau_i+n_i, \tau_i+y_i]$, noté aussi par $[\tau_i^{min}, \tau_i+n_i, \tau_i^{max}]$, spécifie l'ITVA de l'arc $a_i = (p_i, t)$.



(a) Spécification RdPFT (b) Intervalles de tir de la transition t_5

Figure 2.27 Sémantique de synchronisation du modèle RdPFT

Les relations temporelles entre et à l'intérieur de composants d'un scénario multimédia peuvent être exprimées à partir de l'utilisation récursive des 7 constructeurs de synchronisation structurellement similaires à ceux du modèle OCPN (figure 2.19). Mais contrairement au modèle OCPN, les présentations ne sont pas associées à des durées nominales, mais elles sont associées à des IVT. Ainsi le modèle RdPFT est plus approprié pour exprimer le non-déterminisme temporel inhérent aux systèmes multimédias répartis.

Les réseaux RdPFT construits exclusivement à partir des 7 constructeurs de synchronisation, sont nommés RdPFT structurés (RdPSFT). Ces réseaux sont des graphes marqués sans circuit et sont définis comme saufs. Cette structure permet l'application de puissantes techniques de vérification de la consistance temporelle des schémas de synchronisation :

- Des méthodes de vérification permettent l'analyse de la consistance temporelle des scénarios multimédias. Les techniques de vérification statique basées sur un algorithme de réduction et sur une analyse de pire cas permettent la détection des risques potentiels de désynchronisation inter-média.

11. Dans le modèle RdPFT, un traitement multimédia finit quand le processus finit naturellement dans l'intervalle de validité temporelle ou quand sa durée arrive dans la borne maximale de l'intervalle de validité temporelle.

- Le modèle RdPFT hérite par l'intermédiaire de possibilités de traduction dans le formalisme des Réseau de Petri Temporels (intervalles temporels associés aux transitions) [Sénac, 96b] des techniques d'analyse développées pour ce modèle.
- Le concept de jeton de réseau de Petri permet la simulation des activités dynamiques et concurrentes des scénarios modélisés.

Comme le modèle OCPN, les RdPFT ne permettent pas la spécification de liens hypermédias. De plus, ce modèle a pour seul objectif la spécification de la composition temporelle de documents multimédias, les autres informations nécessaires à la spécification de la structure du contenu et de présentation ne peuvent pas être spécifiées.

2.4.7.10RdPHFT

Basé sur les concepts établis par le modèle de référence hypertexte Dexter (section 2.4.1) et sur le modèle RdPFT (section 2.4.7.9), dans [Sénac, 95a] et [Sénac, 96a] les auteurs proposent le modèle *Réseaux de Petri Hiérarchisés à Flux Temporels* (RdPHFT), qui permet la modélisation formelle des contraintes de synchronisation logiques et temporelles dans les systèmes hypermédias.

Le modèle RdPHFT permet la modélisation formelle des trois concepts fondamentaux du modèle de référence hypertexte Dexter (section 2.4.1), à savoir les composants atomiques, composites et liens en y ajoutant des extensions temporelles :

- Les composants atomiques du modèle Dexter représentent les données codées (par ex., des informations textuelles, des séquences audio ou vidéo). En RdPHFT, un composant atomique est modélisé par un arc avec un IVT et une **place atomique** associée à la ressource atomique. Cet IVT permet la définition d'une tolérance de synchronisation dans les systèmes hypermédias faiblement synchrones. Par exemple, la place atomique V1 dans la figure 2.28 représente un composant atomique (une séquence vidéo) avec [8,10,12] comme IVT.

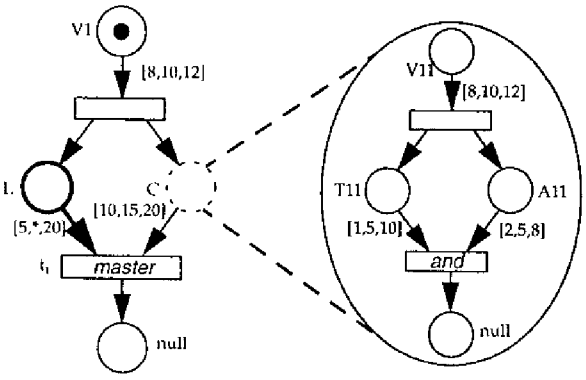


Figure 2.28 Le modèle RdPHFT [Sénac, 95a]

- Le *composant composite* du modèle Dexter fournit un mécanisme de structuration hiérarchique basé sur la composition récursive de composants atomiques, liens et composites. Dans le modèle RdPHFT, un composant composite est modélisé par une place abstraite, appelée **place composite**, qui représente un sous-réseau RdPSFT. Ce type de place est représenté graphiquement par un cercle en pointillé. Cette modélisation est illustrée sur la figure 2.28, où la place C est une place composite qui représente un RdPSFT. Ce RdPSFT spécifie un scénario de synchronisation multimédia. L'IVT de l'arc de sortie d'une place composite permet la spécification de la gigue temporelle admissible pour le traitement du scénario multimédia associé à cette place.
- Dans le modèle Dexter, les *liens* permettent la définition de relations entre composants. Dans le

modèle RdPHFT, un lien est modélisé par un arc temporisé (L, t), où L est une **place lien**. Ce type de place est représenté graphiquement par un cercle en gras. L'IVT associé au lien introduit le concept de lien temporisé [Sénac, 95]. Par exemple, dans la figure 2.28, la place L spécifie un lien avec [5,*,20] comme IVT[12]. Cela veut dire que ce lien peut être activé seulement 5 unités de temps après le début de la présentation du composite C. Si le lien n'est pas activé avant 20 unités de temps, alors il est activé de façon automatique après 20 unités de temps.

Comme dans le modèle RdPFT, les relations logiques et temporelles entre et à l'intérieur des composants d'un RdPHFT sont modélisées par des transitions typées. Chaque relation, représentée par une transition, associe un ensemble d'événements sources, représenté par les places d'entrée de la transition, à un ensemble d'événements destination, représenté par les places de sortie de la transition. Par exemple, la transition t_1, dans la figure 2.28, définit une relation logique et temporelle entre le lien (L, t_1) et le scénario multimédia modélisé par (C, t_1).

RdPHFT est un modèle conceptuel à trois niveaux (figure 2.29) :

- Le niveau de *Synchronisation Logique* est dédié à la spécification formelle du chemin de parcours du document. Cette spécification est réalisée à l'aide d'un réseau RdPFT où les places et ses arcs de sortie représentent les composants atomiques, liens et composites du document.
- Le niveau de *Synchronisation Composite* est dédié à la spécification des relations temporelles à l'intérieur des composants composites du niveau de synchronization logique, c'est-à-dire des scénarios multimédias. Ces scénarios multimédias sont modélisés comme une hiérarchie de RdPSFT (section 2.4.7.9).
- Le niveau de *Synchronisation Atomique* décrit les synchronisations intra-média.

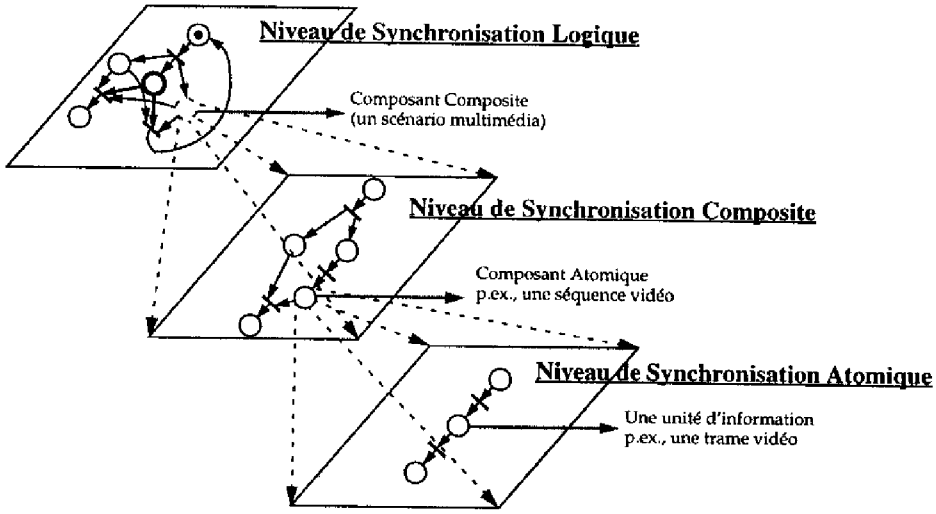


Figure 2.29 Les Niveaux du modèle RdPHFT

Le modèle RdPHFT permet l'application des techniques d'analyse des scénarios de synchronisation multimédias développées par le modèle RdPFT et la simulation du comportement logique du document.

Le modèle RdPHFT est un modèle logique permettant la spécification formelle de la structure conceptuelle (définition des composants et de leurs relations logiques et temporelles) des documents hypermédias. La description de la structure du contenu et de la structure de présentation (comme les

12. La durée normale d'un événement asynchrone ne peut pas être connue en avance. C'est pourquoi, la durée nominale de l'IVT d'un lien est remplacée par le caractère «*».

caractéristiques spatiales et sonores des présentations) ne font pas partie du domaine de spécification du modèle RdPHFT.

Les autres limitations du modèle RdPHFT sont :

- Le niveau de synchronisation logique est modélisé uniquement par un réseau plat (non hiérarchique). Cette limitation résulte en l'obtention de réseaux assez complexes dans le cas d'applications réelles. De plus, elle empêche la structuration sémantique du document, par exemple en divisant le document en chapitres et sections.
- Au contraire du modèle XOCPN (section 2.4.7.4) qui permet la modélisation en séparé du processus de préparation des ressources, de présentation et de libération des ressources, le modèle RdPHFT ne fournit pas les moyens d'identifier le type de traitement multimédia modélisé par une place.
- Une présentation peut être synchronisée seulement avec le début et la fin d'un ensemble de présentations. Cela signifie, par exemple, que la présentation des sous-titres dans certaines parties d'une séquence vidéo requiert le découpage de la séquence vidéo en plusieurs composantes consécutives.

2.5 Conclusion

Ce chapitre a introduit les principaux concepts associés à la modélisation de documents hypermédias. De plus, il a identifié les principales exigences qui doivent être satisfaites par un modèle hypermédia et il a présenté les principaux modèles de documents multimédias et hypermédias.

Un modèle hypermédia doit fournir une description multi-niveaux d'un document hypermédia, incluant la spécification des structures du contenu, conceptuelle et de présentation. Le tableau 2.1 indique si les modèles multimédias et hypermédias présentés dans ce chapitre permettent la spécification de la structure du contenu et s'ils distinguent les objets multimédias de leurs présentations (distinction entre la structure du contenu et de présentation). Notez que seulement le modèle Dexter et les systèmes Firefly et MODE permettent la réutilisation des objets multimédias (une entité permettant la spécification des données primitives). Les modèles AHM, OCPN et ses extensions permettent uniquement la réutilisation des données primitives (les descripteurs de ces données n'existent pas ou ne peuvent pas être réutilisées).

Modèle	Spécifie la structure du contenu	Distingue les objets multimédias ou les données primitives de leurs présentations.	Informations spécifiées par un objet multimédia.	Informations spécifiées par une présentation.
Dexter	Oui.	Oui. Un objet multimédia est modélisé par un composant atomique ou composite et une présentation est appelée instantiation.	Données primitives; liste d'ancres; caractéristiques originelles de présentation; autres attributs définis par l'auteur.	Identificateur du composants; caractéristiques de présentation.
AHM	Oui.	Oui. Les composants atomiques et composites représentent la paire objet multimédia/présentation et les données primitives sont référencées par les composants.	Les données primitives.	Référence aux données primitives; liste d'ancres; caractéristiques originelles de présentation; la durée et le canal de présentation; autres attributs définis par l'auteur.
HyTime	Oui.	La norme HyTime définit uniquement un langage pour la représentation de documents hypermédias, l'auteur peut définir une syntaxe quelconque (pouvant définir ainsi les structures du contenu, conceptuelle et temporelle).		

Tableau 2.1 Comparaison entre les modèles aux niveau de la structure du contenu

Modèle	Spécifie la structure du contenu	Distingue les objets multimédias ou les données primitives de leurs présentations.	Informations spécifiées par un objet multimédia.	Informations spécifiées par une présentation.
[Hudson, 93]	Oui.	Non. Les objets primitifs et composites représentent des paires objet multimédia/présentation.	-	Les données primitives; les ressources nécessaires; la machine temporelle; un driver de présentation; la durée de présentation; la réductibilité; et la capacité de contrôle de flux.
[Steinmetz, 90]	(n'est pas décrit dans [Steinmetz, 90])	Non. Un objet représente la paire données primitives/présentation.	-	(n'est pas décrit dans [Steinmetz, 90])
Système MODE	Oui.	Oui. Un objet de la classe Information représente un objet multimédia et un objet de la classe Présentation représente une présentation.	(n'est pas décrit dans [Blakowski, 92])	(n'est pas décrit dans [Blakowski, 92])
Système Firefly	Oui.	Oui. Un item média modélise un objet multimédia et le niveau-document permet la description de la structure de présentation.	Les événements; les durées; les coûts de réduction et d'allongement; et l'inclusion implicite des données primitives.	(dans [Buchanan, 92] aucune mention n'est faite sur la description des caractéristiques spatiales et sonores des présentation)
IconAuthor	Oui.	Non. Une icone multimédia représente la paire données primitives/présentation		
Eyes M/M	Oui.	Non. Une icone action représente la paire données primitives/présentation	-	(n'est pas décrit dans [Koegel, 92])
OCPN	Oui.	Oui. Une place représente une présentation et elle fait référence aux données primitives.	Les données primitives.	Référence aux données primitives; type de média; un attribut non spécifié
OCPN avec caractéristiques de présentation	Oui.	Oui. Une place représente une présentation et elle fait référence aux données primitives.	Les données primitives.	Référence aux données primitives; région de présentation de la place; un vecteur de priorité; une liste d'opérations spatiales.
A-OCPN	Oui.	Oui. Une place représente une présentation et elle fait référence aux données primitives.	Les données primitives.	Référence aux données primitives; type de média; un attribut non spécifié
XOCPN	Oui.	Oui. Une place représente une présentation et elle fait référence aux données primitives.	Les données primitives.	Référence aux données primitives; type de média; un attribut non spécifié
RTMS	Oui.	Non	-	Les données primitives.
Trellis	Oui.	Non. Une place représente la paire données primitives/présentation.	-	Les données primitives.

Tableau 2.1 Comparaison entre les modèles aux niveau de la structure du contenu

Modèle	Spécifie la structure du contenu	Distingue les objets multimédias ou les données primitives de leurs présentations.	Informations spécifiées par un objet multimédia.	Informations spécifiées par une présentation.
MORENA	Non.	-	-	-
MHPN	Oui.	Non. Une place représente la paire objet multimédia / présentation.	-	Un objet MH ([Wang, 95] ne présente pas ces informations).
RdPHFT	Non.	-	-	-
RdPFT	Non.	-	-	-

Tableau 2.1 Comparaison entre les modèles aux niveau de la structure du contenu

La structure conceptuelle permet la création de composants et de groupes de composants d'un document hypermédia, les relations logiques et temporelles entre les composants. Le tableau 2.2 présente comment les modèles multimédias et hypermédias présentés dans ce chapitre satisfont ces besoins.

Modèle	Modèle ou interface graphique	Notion permettant la spécification des composants du document.	Mécanisme de Structuration	Modèles d'interaction
Dexter	Non.	Composants atomiques et composites.	Composant composite.	Liens, modélisés par des composants lien.
AHM	Oui (interface graphique).	Composants atomiques et composites.	Composant composite.	Liens, modélisés par des composants lien.
HyTime	Non (orienté-langage).	Défini par l'auteur d'un DTD.		Hyper-liens (ou autres constructeurs non HyTime).
[Hudson, 93]	Oui (composition hiérarchique).	Modèle de composition hiérarchique.		Boutons et menus, modélisés par l'opérateur de sélection.
[Steinmetz, 90]	Non (orienté-langage).	Objets Actifs.	-	Il n'y a pas de représentation explicite.
Système MODE	Oui (interface graphique).	Objets.	-	Boutons, modélisés par les objets interaction.
Système Firefly	Oui (interface graphique).	Items média.	-	Il n'y a pas de représentation explicite.
IconAuthor	Oui (organigrammes).	Icônes.	-	Icones d'entrée.
Eyes M/M	Oui (organigrammes).	Icônes.	-	Icones d'entrée.
OCPN	Oui (réseau de Petri).	Places.	-	-
OCPN avec caractéristiques de présentation	Oui (réseau de Petri).	Places.	-	-
A-OCPN	Oui (réseau de Petri).	Places.	-	Interrompre, geler, repartir, renverser la direction, et changer la vitesse d'une présentation.
XOCPN	Oui (réseau de Petri).	Places.	-	-
RTMS	Oui (réseau de Petri).	Places.	-	-

Tableau 2.2 Comparaison entre les Modèles de Composition

Modèle	Modèle ou interface graphique	Notion permettant la spécification des composants du document.	Mécanisme de Structuration	Modèles d'interaction
Trellis	Oui (réseau de Petri).	Places.	Une place représente un composant ou un réseau.	Boutons, modélisés par les transitions.
MORENA	Oui (réseau de Petri).	Places de base et composites.	Places composites	Boutons, modélisés par des transitions utilisateur.
MHPN	Oui (réseau de Petri).	Places.	-	Boutons, modélisés par des transitions associées à des objets MH d'entrée.
RdPFT	Oui (réseau de Petri).	Places.	-	-
RdPHFT	Oui (réseau de Petri).	Places atomiques et composites.	Il n'y a pas de mécanismes d'abstraction permettant la hiérarchisation de la structure conceptuelle.	Représentation abstraite des liens hypermédias, boutons et menus. Modélisés par des places liens et leurs arcs de sortie.

Tableau 2.2 Comparaison entre les Modèles de Composition

Le tableau 2.3 résume les principales caractéristiques des modèles de composition adoptés par les modèles multimédias et hypermédias présentés dans ce travail. Les paramètres de comparaison sont :

- **Vérification des contraintes de synchronisation logique et temporelle** : si le modèle propose des méthodes de vérification des contraintes de synchronisation logique et temporelle du document hypermédia.
- **Parallélisme** : si le modèle est capable d'exprimer de façon explicite le parallélisme lors de la composition des présentations d'un document.
- **Relations temporelles** : si le modèle est capable d'exprimer de façon explicite les relations temporelles entre présentations.
- **Méthodes de tolérance** : si le modèle est capable d'exprimer de méthodes de tolérances temporelles lors de la synchronisation entre présentations.
- **Méthodes de contrôle des dérives temporelles** : si le modèle est capable de décrire des schémas de synchronisation inter-média en prenant en compte le non déterminisme introduit par la gigue temporelle de la durée de traitement multimédia.
- **Synchronisation continue** : si le modèle est capable d'exprimer la synchronisation continue (p.e., la *lip-synchronization*).
- **Granularité de synchronisation** : la granularité de synchronisation peut être forte ou faible. La granularité de synchronisation est dite faible s'il n'y a pas de moyen d'exprimer de relations autres que le début et la fin d'actions. Sinon elle est dite forte.
- **Modèle ou Interface Graphique** : si le modèle est graphique, ou si une interface graphique est proposée.

Modèle	Modèle ou interface Graphique	Parallélisme	Relations temporelles	méthodes de tolérance	méthodes de contrôle des dérives temporelles	sync.continue	Granularité de sync.	vérification des contraintes de sync.
Dexter		x					-	
AHM	x	x	x	x		x[a]	faible	
HyTime		x	x	x[b]	x[b]		faible	
[Hudson, 93]	x	x	x			x	forte	x
[Steinmetz, 90]		x	x	x		x	forte	
Système MODE	x	x	x			x	forte	
Système Firefly	x	x	x				forte	x
IconAuthor	x						-	
Eyes M/M	x	x	x				faible	
OCPN	x	x	x			x[c]	faible	x
OCPN avec caractéristiques de présentation	x	x	x			x	faible	x
A-OCPN	x	x	x			x[c]	faible	
XOCPN	x	x	x			x	forte	x
RTMS	x	x	x	x	x	x	faible	
RdPFT	x	x	x	x	x	x[c]	faible	x
Trellis	x	x	x[d]	-[e]			faible	x
RdPHFT	x	x	x	x	x	x[c]	faible	x
MORENA	x	x	x			x	forte[f]	
MHPN	x	x	x				faible	

Tableau 2.3 Comparaison entre les Modèles de Composition

a. En théorie, le modèle AHM permet l'expression de la synchronisation continue. Mais lors de la spécification réelle de ce type de synchronisation, le nombre d'arcs de synchronisation empêche son utilisation.
b. Comme la norme HyTime définit uniquement une syntaxe pour la représentation de documents hypermédias, l'auteur peut définir de nouvelles sémantiques afin de rajouter des méthodes de tolérance et des mécanismes de contrôle de la dérive temporelle des présentations.
c. En théorie, les modèles basés sur les réseaux de Petri permettent l'expression de la synchronisation continue. Mais lors de la spécification réelle de ce type de synchronisation, le nombre de places et transitions empêche son utilisation. Une solution est de rajouter une notation aux places afin de simplifier cette spécification.
d. Le modèle Trellis permet uniquement l'expression des relations temporelles entre les nœuds d'un document hypermédia. Il ne permet pas l'expression de relations temporelles des présentations à l'intérieur d'un nœud.
e. Le modèle Trellis permet uniquement la spécification des contraintes temporelles des liens hypermédias.
f. Via la notion de message.

Tous les modèles multimédias et hypermédias présentés dans ce travail utilisent comme mode de présentation du document l'interprétation du modèle, ou encore ils utilisent des modèles de stockage particuliers. Par exemple, le modèle logique de composition OCPN (section 2.4.7.1) utilise une représentation hiérarchique particulière comme modèle de stockage. Afin de permettre le partage d'informations hypermédias entre ces systèmes, il est nécessaire d'avoir des mécanismes qui puissent transformer les structures particulières de chaque système en une représentation commune.

Notons que le modèle RdPHFT satisfait presque tous les besoins d'un modèle de composition temporelle et logique de documents hypermédias, comme cela est mis en évidence par le tableau 2.3. RdPHFT a un pouvoir de modélisation permettant une spécification aisée et unifiée des composants atomiques, composites et liens du modèle Dexter. De plus, il définit un formalisme capable de spécifier le comportement logique et temporel des composants. Finalement, ce modèle permet l'application de techniques d'analyse formelle pour vérifier précisément les propriétés logiques et temporelles du document. Néanmoins, le modèle RdPHFT ne fournit pas les moyens de spécifier la structure du contenu, ni les caractéristiques spatiales des présentations qui composent un document hypermédia.

Le chapitre 4 de ce travail propose une extension au modèle RdPHFT, appelé modèle RdPHFT interprété (RdPHFT-I) qui permet la spécification complète de documents hypermédias à partir d'un modèle multi-niveaux. Ce modèle supporte la spécification complète de la structure conceptuelle, logique et de présentation. Il est clair qu'un modèle physique est alors nécessaire afin de définir complètement RdPHFT-I. Ce modèle doit être très général si son critère de définition est la portabilité. Comme nous voulons ici un modèle ouvert, nous avons décidé dans le cadre du projet CESAME, de sélectionner la norme MHEG. Il s'agit alors de relier le modèle hypermédia de haut niveau avec le modèle physique choisi. C'est précisément ce que réalisera le modèle RdPHFT-I. Avant de préciser ce modèle, rappelons les principales caractéristiques de la norme MHEG.

Chapitre 3

La norme MHEG

Les applications multimédias et hypermédias peuvent être réparties sur plusieurs systèmes hétérogènes, interconnectés par un réseau de communication, conduisant par exemple aux travaux coopératifs, aux systèmes de messagerie multimédia ou simplement à l'accès à une base de données multimédias. Les informations utilisées, stockées et échangées par de telles applications représentent un investissement important, et ainsi il est vital que ces informations soient rapidement utilisables dans un monde en très rapide évolution. Pour cela, il est nécessaire de définir un langage commun pour tous les systèmes de communication multimédias/hypermédias [Colaitis, 93].

La normalisation au niveau monomédia, comme JPEG pour l'image fixe et MPEG pour l'audio et la vidéo, n'est pas suffisante pour permettre la portabilité, car les applications multimédias nécessitent des informations pour la présentation de ces données (identification de l'algorithme de codage, attributs à utiliser lors de la présentation, etc.) et des informations sur les interrelations entre les données multimédias.

La future norme internationale ISO/IEC MHEG (*Multimedia Hypermedia Expert Group*) [ISO 13522] définit un codage (une représentation) des informations multimédias et hypermédias. Elle permet l'échange de plusieurs types de médias, le transfert des structures représentant la composition temporelle et spatiale de ces médias et enfin les interactions lors de leurs présentations. L'adoption de la norme MHEG pour représenter les informations hypermédias et multimédias permet aussi le transfert et le traitement de ces types d'informations dans un système ouvert.

La norme MHEG permet de traiter les informations comme un conteneur pour l'échange de plusieurs types de médias (figure 3.1). Dans un tel conteneur, les données peuvent être encodées en utilisant de techniques de codage quelconques.

MHEG utilise une approche orientée objet : les informations multimédias et hypermédias sont représentées par des **objets MHEG**. Ces objets fournissent les fonctionnalités suivantes :

- Ils contiennent ou font référence à des données codées selon une autre norme internationale (ex. JPEG, MPEG) ou une autre technique de codage quelconque.
- Ils expriment les relations entre objets, le comportement dynamique des objets, et les informations pour optimiser la manipulation temps-réel des objets.

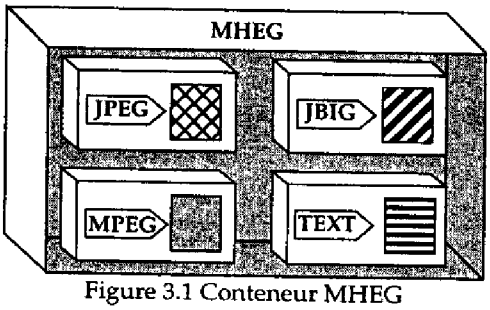


Figure 3.1 Conteneur MHEG

La partie I de la norme MHEG utilise la représentation ASN.1 [ISO 8824] et ses règles de codage [ISO 8825] pour représenter les objets MHEG. Ainsi cette partie spécifie, en utilisant la notation ASN.1, la structure de représentation des informations multimédias et hypermédias. La partie II spécifie les objets MHEG en utilisant SGML - *Standard Generalized Markup Language* [ISO 8879]. Ces deux représentations sont isomorphes et complètement équivalentes. La partie III [ISO 13522-3] définit des extensions pour les supports de langage de script MHEG et l'échange d'objets Script. La partie IV spécifiera les procédures d'enregistrement de types de données MHEG. Enfin, la partie V [ISO 13522-5] spécifie les besoins pour un sous-ensemble de MHEG défini par la partie I : ces objets seraient utilisés dans des applications multimédias simples (p.e. applications de vidéo à la demande, application de navigation et de feuilletage).

La norme MHEG définit une représentation codée des objets multimédias et hypermédias de base. MHEG permet plusieurs extensions afin de satisfaire toutes les contraintes des applications multimédias et hypermédias. Par exemple, l'utilisateur peut définir de nouveaux attributs aux objets MHEG et il peut définir aussi de nouvelles actions applicables sur les objets.

Le système responsable de la mise en exécution des fonctionnalités décrites par la norme MHEG est appelé **machine MHEG**. La machine MHEG est responsable de l'interprétation des objets MHEG, c'est-à-dire de l'accès, du transfert et de la présentation des informations multimédias et hypermédias représentées en objets MHEG.

MHEG spécifie la représentation des informations multimédias et hypermédias. La définition de modèles, de services, de systèmes, de protocoles et la façon d'utiliser des objets MHEG ne fait pas partie de la norme MHEG. De plus, l'utilisation des concepts MHEG est nécessaire seulement au niveau de la communication. Ainsi, le format interne des informations multimédias et hypermédias utilisé par une application peut être différent du format spécifié par la norme MHEG.

L'objectif principal de ce chapitre est d'introduire les concepts MHEG nécessaires à l'étude présentée dans les chapitres suivants. Il aborde dans la section 3.1 les classes d'objets et autres entités MHEG. La section 3.2 présente la machine MHEG. La section 3.3 présente les pré-requis de la synchronisation multimédia identifiés par MHEG et les fonctionnalités spécifiées pour réaliser cette synchronisation. La section 3.4 donne un exemple de représentation MHEG.

3.1 Les classes MHEG, rt-objets et canaux

Les **classes MHEG** sont présentées ci-dessous. Le caractère «>» signifie «a les sous-classes suivantes». Les classes en caractères **gras** ont leurs instances effectivement échangées entre les applications multimédias. Les autres classes sont introduites afin de décrire les attributs communs ou pour grouper des classes ayant des sujets similaires. MHEG ne définit pas de méthodes dans ces classes.

MH-OBJECT>
- **ACTION**
- **LINK**
- MODEL>
  - **SCRIPT**
  - COMPONENT>
    - **CONTENT>**
      - **MULTIPLEXED CONTENT**
    - **COMPOSITE**
  - **CONTAINER**
  - **DESCRIPTOR**

En dehors des classes d'objets MHEG présentées ci-dessus, nous avons les **objets MHEG NULL**. Un objet NULL peut remplacer n'importe quelle classe d'objet MHEG et n'entraîne aucune présentation. Ces objets sont générés par la machine MHEG quand cela est demandé par l'utilisateur.

Un objet de la classe MODEL (les objets SCRIPT, CONTENT, MULTIPLEXED CONTENT et COMPOSITE) permet la représentation des composants. Afin de permettre la réutilisation d'objets de la classe MODEL dans des présentations ou des activations différentes, la norme MHEG définit une séparation entre les objets MODEL échangés, qui contiennent les données réutilisables, et une entité appelée **rt-objet** (*run-time objects*), qui correspond à une vision spécifique des données. Ainsi, le même objet MODEL peut avoir plusieurs présentations ou activations différentes, chacune représentée par un rt-objet. Par exemple, dans la figure 3.2 on peut voir plusieurs rt-objets du même objet J, chacun ayant une caractéristique de présentation différente.

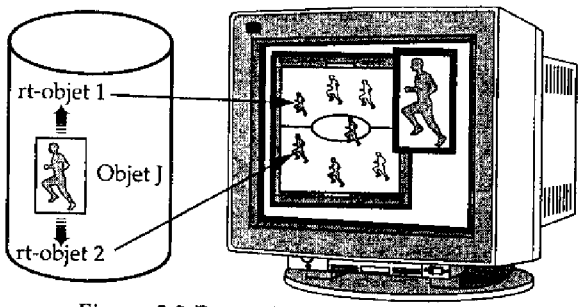


Figure 3.2 Exemple de rt-objet et canal

Une autre entité MHEG importante est le **canal**, qui représente un espace logique où les rt-objets sont positionnés, présentés et vus par l'utilisateur d'un système multimédia/hypermédia. Les canaux sont créés par l'auteur d'une application en utilisant des actions MHEG. La machine MHEG fait l'association entre cet espace logique et le support réel (fenêtre, haut-parleur, moniteur, etc.). Par exemple, dans la figure 3.2 nous avons la représentation de deux canaux différents, qui sont deux fenêtres.

Par la suite, nous présentons les classes MHEG dont les objets peuvent être échangés entre applications multimédias/hypermédias.

3.1.1 Classe ACTION

La classe ACTION définit une structure qui permet la représentation d'un ensemble d'actions applicables sur des objets, rt-objets et canaux. Cette structure est utilisée par les objets LINK pour décrire l'effet d'un lien (section 3.1.2). Ci-dessous, nous présentons quelques types d'actions MHEG :

- **Préparation et destruction d'objets** : ces actions sont utilisées pour la mise à disposition (action *prepare*) ou la libération (action *destroy*) des objets au niveau de la machine MHEG. Un exemple de préparation d'un objet est le décodage d'un objet avant sa présentation. Les actions de préparation agissent sur l'attribut *Preparation-status* des objets MHEG. Le diagramme de transition d'état de

l'attribut *Preparation-status* est présenté dans la figure 3.3.

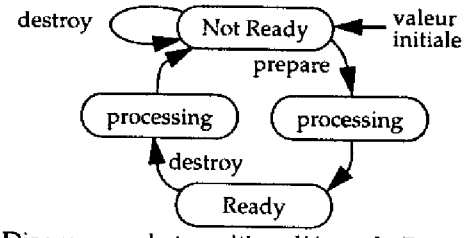


Figure 3.3 Diagramme de transition d'états de *Preparation-status*

- **Création de rt-objets** : l'action *new* permet la création d'un rt-objet à partir d'un objet MODEL et l'action *delete* permet sa destruction.
- **Présentation** : ces actions sont utilisées pour le contrôle de la progression de la présentation d'un rt-COMPONENT, c'est-à-dire les rt-CONTENT, rt-MULTIPLEXED CONTENT et rt-COMPOSITE (actions *run* et *stop*). Les actions de présentation agissent sur l'attribut *Running-status* des rt-objets. Le diagramme de transition d'états de *Running-status* est présenté dans la figure 3.4.

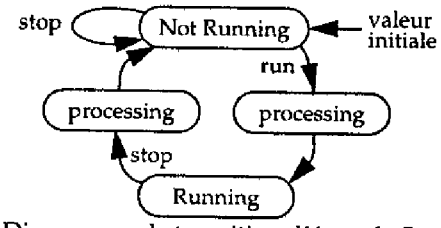


Figure 3.4 Diagramme de transition d'états de *Running-status*

- **Projection** : les actions qui permettent la définition ou la modification des caractéristiques d'une présentation. Par exemple, l'action *set-speed* pour changer la vitesse d'une présentation.
- **Interactions** : les actions utilisées pour contrôler les résultats des interactions. Par exemple, les actions *set-selectable* et *set-modifiable* peuvent être utilisées pour le changement de la possibilité de sélectionner et de modifier un rt-COMPONENT, respectivement.
- **Activation de scripts** : les actions utilisées pour l'activation ou la désactivation (actions *run* et *stop*) d'un rt-SCRIPT. Ces actions agissent sur l'attribut *Termination-status* d'un rt-SCRIPT. Le diagramme de transition d'états de *Termination-status* est présenté dans la figure 3.5.

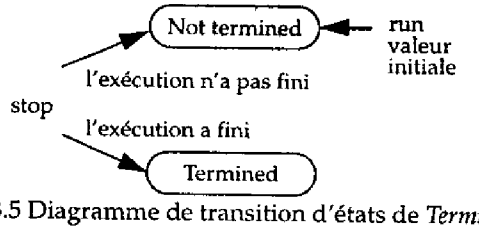


Figure 3.5 Diagramme de transition d'états de *Termination-status*

Un objet de la classe ACTION permet la représentation des informations suivantes :

- Une liste d'actions : un objet ACTION contient une liste d'actions élémentaires et/ou imbriquées (objet ACTION contenant d'autres objets ACTION).
- *Synchro indicator* : il indique si les actions de la liste d'actions seront exécutées de façon parallèle ou séquentielle.
- Objets, rt-objets ou canaux cibles sur lesquels les actions seront exécutées.
- Nombre de répétitions : nombre de fois où l'action sera exécutée.

Les objets ACTION sont classés en trois types :

- **Simple**, un objet ACTION contenant une liste d'actions composée uniquement d'actions élémentaires, comme dans l'exemple de la figure 3.6.

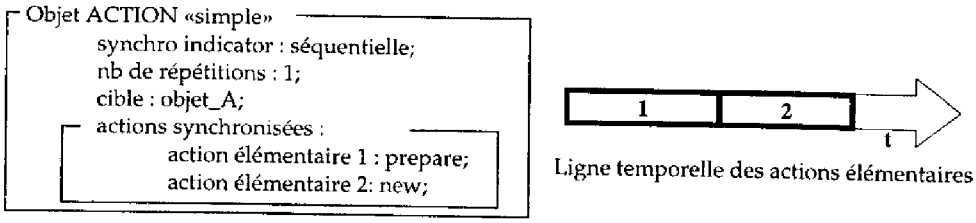


Figure 3.6 Exemple d'action simple

- **Imbriquée**, ce type d'objet ACTION contient (ou fait référence) dans sa liste d'actions d'autres objets ACTION, comme dans l'exemple de la figure 3.7. Cela permet de décrire des comportements plus complexes.

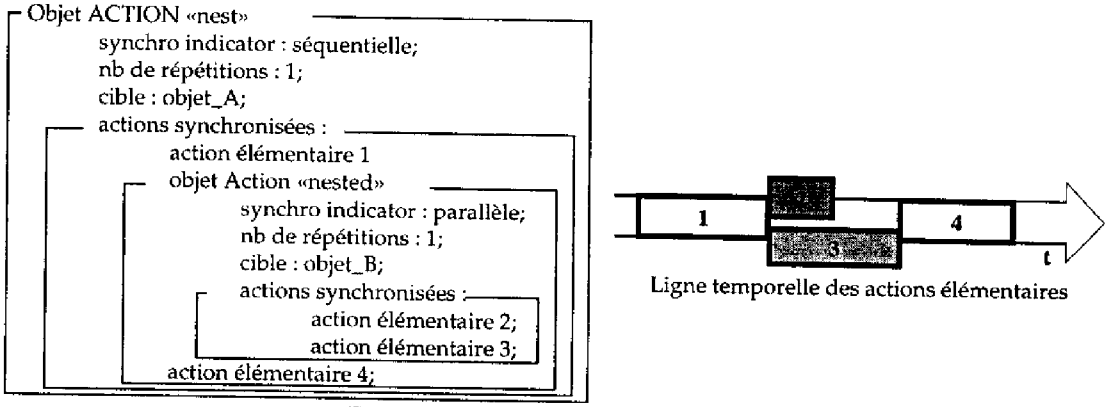


Figure 3.7 Exemple d'action imbriquée

- ***Action Macro***, afin de produire un code plus efficace. Dans le cas d'utilisation fréquente d'objets ACTION ayant seulement quelques changements d'un objet par rapport à l'autre, l'auteur peut spécifier des paramètres *macro* dans les objets ACTION. Ces *macros* sont résolues lors du tir d'un link (présenté dans la section 3.1.2). Un exemple de *macro action* est présenté dans la figure 3.8.

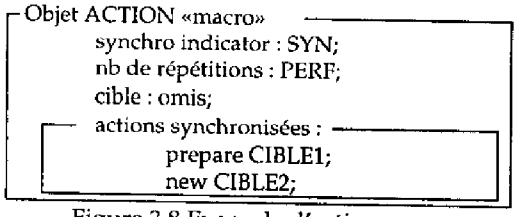


Figure 3.8 Exemple d'action *macro*

3.1.2 Classe LINK

La classe LINK permet la spécification des relations spatiales, temporelles et conditionnelles entre une ou plusieurs sources et cibles. Les sources et cibles peuvent être des objets MHEG, des rt-objets et des canaux. Un objet LINK contient des conditions sur les sources. Quand ces conditions sont satisfaites, un ensemble d'actions est envoyé aux cibles. Quand un objet LINK est préparé (en utilisant l'action *prepare*) il devient actif et devenant alors tirable. La figure 3.9 présente la structure d'un objet LINK.

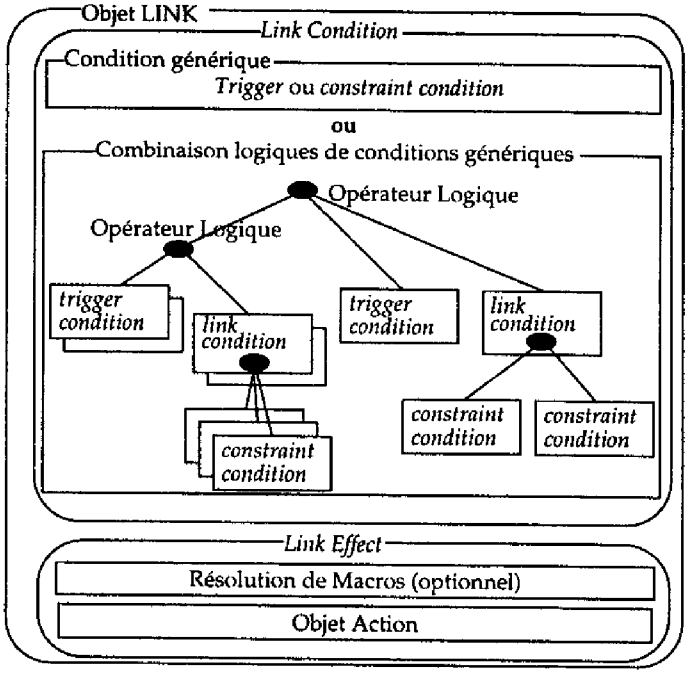


Figure 3.9 Objet LINK

Par la suite, nous présentons les composants d'un objet LINK.

Link Condition

La *Link Condition* décrit les conditions du lien. La satisfaction de ces conditions provoque l'exécution du *Link Effect*. Ces conditions sont des tests de comparaison entre valeurs et/ou résultats d'actions. Les opérateurs de comparaison sont : ==, !=, <, <=, >, >=. Une *Link Condition* peut être :

- Une condition générique, qui peut être une condition de *trigger* ou une *constraint condition* :
  - *Trigger Condition* : décrit l'événement «habilitant» le lien. Les événements *triggerables* sont des événements temporels (provoqués par des *timestones*[1]) et la conséquence d'une action MHEG ou d'une interaction (par exemple, quand un lecteur touche un bouton). Une *trigger condition* est associée à un état actuel et précédent d'une valeur source. La table ci-dessous présente des exemples de *trigger conditions* (où N et P dénotent les rt-objets cibles des conditions).

valeur source	condition précédente	condition actuelle
get-audible-volume(N)	== 10	< 5
get-audible-volume(N)	< get-audible-volume(P)	> get-audible-volume(P)
get-audible-volume(N)	non spécifiée	>= 5

  - *Constraint Condition* : décrit un état contextuel qui doit être vérifié quand la *trigger condition* est satisfaite. Elle est associée seulement à l'état actuel. La table ci-dessous présente des exemples de *constraint conditions*.

valeur source	condition précédente	condition actuelle
get-audible-volume(N)	omise	> get-audible-volume(P)
get-audible-volume(N)	omise	< 5

- Une combinaison logique de conditions génériques : les opérateurs logiques sont AND, OR, XOR, NAND, NOR et NXOR. La figure 3.10 présente un exemple de combinaison logique de conditions.

1. MHEG fournit les moyens de représenter des marques temporelles pendant la durée d'un rt-objet, appelées *timestones*.

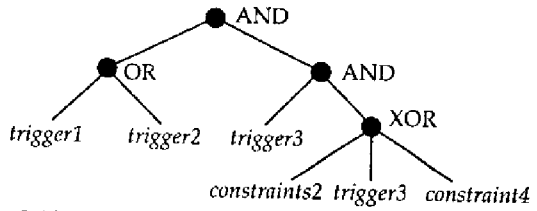


Figure 3.10 Exemple de combinaison logique de conditions

Link Effect

Le *link Effect* est exécuté quand les conditions de l'objet LINK sont satisfaites. Il contient les informations suivantes (figure 3.9) :

- Résolution des *macros* utilisées par les actions *macro*.
- Un objet ACTION type simple, imbriquée ou *macro* (ces types on été présentés dans la section 3.1.1).

3.1.3 Classe SCRIPT

La classe SCRIPT définit une structure permettant la représentation d'un conteneur spécifiant des actions conditionnelles complexes entre les objets MHEG, rt-objets et canaux. Cette structure contient l'identification du langage de script (un langage non-MHEG, comme nf-Z62040-RAVI, smsl et scriptX), les informations d'accès ou le script lui-même.

3.1.4 Classe CONTENT

La classe CONTENT définit une structure permettant la représentation de données primitives. Un objet CONTENT spécifie les informations suivantes :

- *Classification* : le type de média de perception, comme audio, vidéo, texte.
- *Original perception* : les valeurs initiales de présentation, comme la durée de présentation, la taille et le volume.
- *Content hook* : spécifie les informations permettant la manipulation de ces données (informations sur la méthode de codage et autres paramètres).
- *Content data* : contient ou fait référence aux données codées. Un objet CONTENT peut représenter trois types de données :
  - Données qui peuvent être directement présentées à l'utilisateur, par exemple des objets graphiques et audio. MHEG appelle ces données des *media data*;
  - Données qui ne peuvent pas être présentées à l'utilisateur d'une façon immédiate, par exemple un document HyperODA ou Hytime. MHEG appelle ces données des *non media data*;
  - Valeurs génériques, qui spécifient des *variables*, utilisées pour la comparaison, l'affectation ou la présentation.

3.1.5 Classe MULTIPLEXED CONTENT

MULTIPLEXED CONTENT est une sous-classe de la classe CONTENT permettant la manipulation de données multiplexées (ex.: MPEG-System). Un objet de cette classe permet de représenter une liste de flux, où chaque flux contient :

- *Stream identifier* : index du flux
- *Classification* : type de média de perception.
- *Original perception* : valeurs initiales de présentation.
- *Hook stream* : information de codage et décodage.

MHEG spécifie les actions permettant la manipulation indépendante de chaque flux d'un objet MULTIPLEXED CONTENT.

3.1.6 Classe COMPOSITE

La classe COMPOSITE définit une structure permettant la représentation d'associations d'objets MHEG. Ce mécanisme fournit une approche consistante pour la synchronisation dans l'espace et le temps et la liaison entre un ensemble d'objets. Cette classe fournit aussi une structure pour la description d'une liste d'interactions offertes à l'utilisateur (les menus).

Un objet COMPOSITE peut représenter les informations suivantes :

- *Composition Behaviour*, composé par :
  - *Predefined Behaviour*, objet LINK conditionné au début et à la fin de la préparation et de la présentation de l'objet COMPOSITE.
  - *Specific Behaviour*, liste d'objets LINK et ACTION spécifiant le comportement de chaque élément et les inter-relations entre les éléments de l'objet COMPOSITE.
- *Composition Original Perception*, informations spatiales et temporelles initiales de présentation.
- *Elements*, une liste d'éléments où chaque composant de cette liste contient :
  - *Element index*, identificateur de l'élément.
  - *Associated model*, information à présenter, qui est représentée par un objet MODEL (SCRIPT, CONTENT, MULTIPLEXED CONTENT et COMPOSITE), ou un *label* utilisé pour la représentation des options de sélection (quand le COMPOSITE représente une liste d'interactions). Comme un élément d'un objet COMPOSITE peut être lui aussi un objet COMPOSITE, cela permet la création d'une structure arborescente.

Un exemple d'objet COMPOSITE est présenté dans la figure 3.11. Cet objet contient quatre éléments : un titre, un objet CONTENT, un objet MULTIPLEXED CONTENT et un autre objet COMPOSITE.

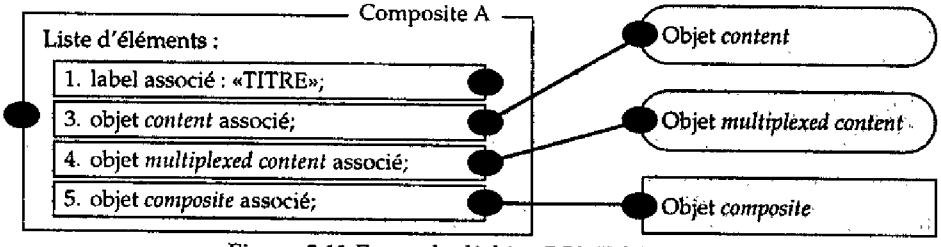


Figure 3.11 Exemple d'objet COMPOSITE

Un rt-COMPOSITE (un rt-objet généré à partir d'un objet COMPOSITE) a la forme d'un arbre. Chaque génération de l'arbre contient une liste d'éléments appelés *sockets* (créés à partir des éléments du COMPOSITE). Ensuite, la machine MHEG exécute une action *plug* sur les *sockets*. L'information à associer aux *sockets* est celle trouvée dans les éléments du COMPOSITE. Par exemple, quand une action *new* arrive sur l'objet A de la figure 3.11, la machine crée 4 *sockets* dans la première génération. Elle crée aussi 4 rt-objets à partir des objets de la liste d'éléments. Ces rt-objets sont attachés (par l'action *plug*) sur le *socket* correspondant.

L'action *plug* permet au concepteur d'une application de changer l'information connectée à un *socket*. Cela introduit un dynamisme structurel de l'objet COMPOSITE, c'est-à-dire MHEG offre des possibilités de remplacer de façon dynamique les données associées à une présentation particulière d'une application.

La figure 3.12 présente un objet COMPOSITE qui spécifie un menu permettant la manipulation de

fichiers. Cette structure spécifie un menu principal contenant les boutons FILE, OPTION et HELP, ainsi que deux sous-menus. Un sous-menu est associé à l'option FILE et l'autre est associé à l'option OPTION. MHEG spécifie les actions pour la définition de ces associations.

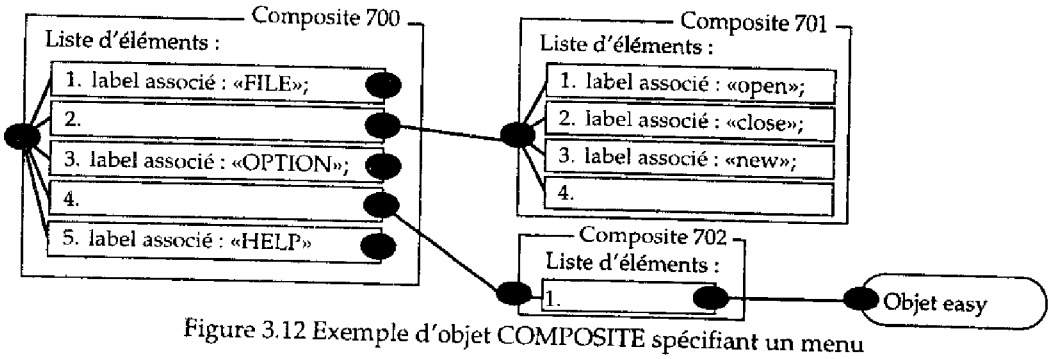


Figure 3.12 Exemple d'objet COMPOSITE spécifiant un menu

3.1.7 Classe CONTAINER

La classe CONTAINER fournit un support pour le regroupement d'objets MHEG. Cela afin de faciliter l'échange de ces objets.

Un objet de la classe CONTAINER permet la représentation des informations suivantes :

- *Predefined behaviour*, contenant *Availability start-up* et *close-down* qui sont les liens de préparation et de destruction du conteneur.
- *Container elements* : les objets MHEG qui sont regroupés.

3.1.8 Classe DESCRIPTOR

Un objet DESCRIPTOR contient des informations génériques sur l'ensemble des objets transférés entre les applications multimédias et hypermédias. Cet objet facilite la négociation, l'installation, l'utilisation et la gestion d'applications.

Un objet DESCRIPTOR contient les informations suivantes :

- Un ensemble d'objets associés : cette liste spécifie le domaine de l'objet DESCRIPTOR. Chaque objet est identifié par une référence et par des informations spécifiques de l'objet, comme la taille des données, la classe MHEG, des informations spécifiques de chaque classe et la qualité de service de chaque objet (dégradation, fiabilité, délai et gigue de transfert).
- D'autres objets DESCRIPTOR : afin de connecter différents objets DESCRIPTOR, permettant la création des structures d'objets DESCRIPTOR.
- Des informations générales : informations informelles données par le concepteur de l'application.
- Des informations sur les canaux : spécifie les canaux nécessaires.
- Des informations sur les styles d'interaction (le style d'interaction qui sera utilisé par les objets associés). MHEG définit les styles suivants : *button, slider, entry field, menu, scrolling list.*

3.2 La machine MHEG

Le système responsable de la mise en exécution des fonctionnalités définies par la norme MHEG est appelé **machine MHEG**. La norme MHEG ne définit aucune structure de machine MHEG. Une machine MHEG n'a pas besoin d'utiliser les objets MHEG au niveau d'exécution, ni même d'être orientée objet.

La norme MHEG présente un exemple de machine MHEG (figure 3.13). Ses éléments sont :

- **Décodeur MHEG** : il fait la conversion des objets MHEG (en ASN.1) vers un format interne, permettant l'utilisation de ces objets par la machine MHEG.
- **Encodeur MHEG** : il fait la conversion du format interne des objets vers les données ASN.1, permettant l'échange des objets MHEG.
- **Manipulateur d'entités MHEG** : il est responsable des allocations, des manipulations et de la gestion mémoire des objets MHEG, rt-objets, et canaux (dans le format interne).
- **Interpréteur MHEG** : met en exécution les fonctionnalités spécifiées par la norme MHEG.

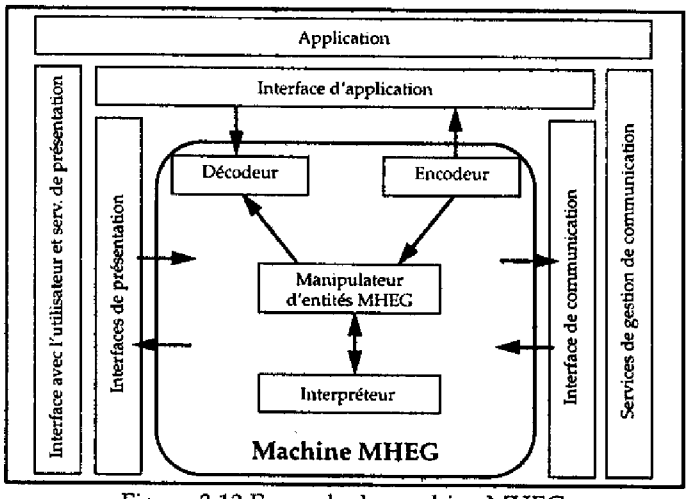


Figure 3.13 Exemple de machine MHEG

L'interface entre les objets MHEG et l'application (vue dans la figure 3.13) n'est pas définie par la norme MHEG. Dans le cas d'une machine MHEG orientée objet, cette interface est réalisée par l'échange de messages et la réception des résultats [Colaitis, 93].

Un exemple de transfert des objets MHEG est présenté dans la figure 3.14 :

- Quand la machine MHEG A désire envoyer un objet MHEG pour B, elle fait la conversion du format interne de l'objet dans le format objet MHEG.
- Quand la machine MHEG B reçoit un objet MHEG de A, elle fait la conversion du format MHEG dans le format interne.

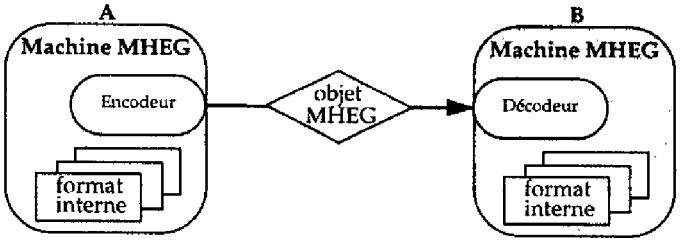


Figure 3.14 Transfert des objets MHEG

Un objet MHEG peut être transféré vers une application pendant l'exécution de cette application : cela peut changer de façon dynamique le comportement de l'application.

3.3 La synchronisation dans MHEG

Cette section présente les pré-requis de synchronisation multimédia et hypermédia identifiés par MHEG. Elle présente aussi les niveaux d'expression de cette synchronisation.

3.3.1 Pré-requis de synchronisation identifiés par MHEG

MHEG définit qu'une application multimédia doit supporter les quatre types de synchronisation suivants :

- **Synchronisation élémentaire** : deux objets sont synchronisés par rapport au même point d'origine (mode parallèle) ou bien un objet est synchronisé par rapport à l'autre (mode séquentiel). Ces modes sont représentés dans la figure 3.15.
- **Synchronisation enchaînée** : permet la présentation d'un objet après l'autre.
- **Synchronisation cyclique** : permet la présentation cyclique d'un ou plusieurs objets.
- **Synchronisation conditionnelle** : la présentation d'un objet est associée à la satisfaction d'une condition.

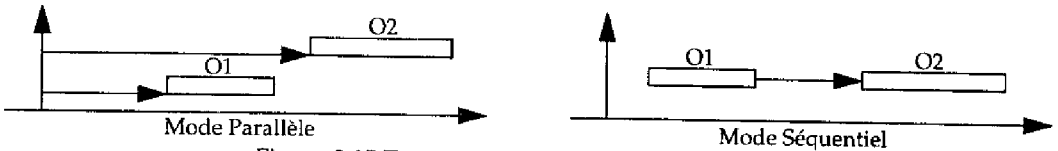


Figure 3.15 Types de synchronisation élémentaire

3.3.2 Niveaux de synchronisation dans MHEG

Il y a quatre niveaux pour l'expression des synchronisations définies par MHEG :

- **Script** : il définit la vision générale de la présentation. Il peut contenir des synchronisations plus complexes, mixant les interventions de l'utilisateur, les données calculées et l'état du système (ce point doit être normalisé par la partie III de la norme MHEG).
- **Conditionnelle** : l'état courant d'un objet dans une présentation peut déclencher certaines actions sur d'autres objets; par exemple quand l'audio finit, poser une question.
- **Spatio-temporelle** : la position dans l'espace et le temps d'un objet par rapport à un autre. Par exemple, présenter le titre 2 cm au-dessous de l'image.
- **Système** : les synchronisations effectuées par le système (p.e. *lip-synchronization*).

3.4 Exemple de représentation MHEG

Dans cette section, nous présentons un exemple simplifié de représentation MHEG d'une présentation multimédia. L'application choisie a le comportement suivant :

- La présentation en parallèle d'une séquence vidéo (objet VIDEO) et d'une séquence audio (objet AUDIO). Avant d'être affichées, la vidéo et l'audio doivent être ajustées par la définition de la localisation de la vidéo et la définition du volume de l'audio.
- Quand les présentations des séquences vidéo et audio finissent, une séquence audio est présentée (objet FIN).

Cette application est représentée par l'objet COMPOSITE[2] de la figure 3.16. La syntaxe adoptée dans ce mémoire pour la représentation des objets MHEG n'est pas spécifiée par la norme MHEG (qui utilise la notation ASN.1). Par souci de simplicité, cette syntaxe est définie comme suit :

- Une action est représentée par «[liste-de-cibles].[liste-d'actions]».
- La référence aux cibles est faite uniquement par le numéro de l'objet (unique à chaque objet MHEG).
- Un élément d'une composition est référencé par le numéro de l'objet COMPOSITE père plus l'index

2. L'utilisation d'un objet COMPOSITE pour représenter cette application n'est pas obligatoire. Nous pourrions utiliser un ensemble d'objets ACTION, LINK et CONTENT séparés.

de l'élément. Par exemple, 100.2 identifie l'élément 2 de l'objet COMPOSITE 100.

- Un rt-objet est référencé par le numéro de l'objet de base et le numéro du rt-objet. Par exemple, 100.9 identifie le rt-objet 9 de l'objet 100.
- Un *socket* est référencé par l'identificateur du rt-COMPOSITE plus l'index du *socket*. Par exemple, 100.9.2 identifie le *socket* 2 du rt-COMPOSITE 100.9.
- le type de cible (s'il s'agit d'un objet ou d'un rt-objet) est déterminé par le type d'action. Par exemple, dans l'action [100.9].[prepare()] l'identificateur 100.9 identifie un objet (l'élément 9 du COMPOSITE 100). Par contre, dans l'action [100.9].[new()] l'identificateur 100.9 identifie un rt-objet (rt-objet 9 du COMPOSITE 100).
- Le caractère «?» est utilisé pour identifier l'ensemble de tous les rt-objets générés à partir d'un objet. Par exemple, l'action [100.?].[run()] a comme cible tous les rt-objets générés à partir de l'objet 100, et l'action [100.?.1].[run()] a comme cible tous les *sockets* 1 de tous les rt-objets générés à partir de l'objet 100.
- «;» dénote le mode séquentiel d'exécution et «,» dénote le mode parallèle.

```
COMPOSITE 100 «Exemple» {
    Composition-Behaviour {
        Predefined-Behaviour {
            Availability-Start-Up {
                [100.1, 100.2].[prepare()]}
            Availability-Close-Down {
                [100.1, 100.2, 101].[destroy()]}
            Rt-Availability-Start-Up {
                [100.?.1].[set-attachment-point-position(position=[100,100,0])];
                [100.?.2].[set-audio-volume (volume=10)];
                [100.?.3].[set-attachment-point-position(position=[100,100,0])];
                [101].[prepare()]}
            Rt-availability-Close-Down {
                [100.?.2].[delete()]}}
        Specific-Behaviour {
            LINK 101 «DEBUT» {
                When (TRUE) then
                begin
                    [100.?.1, 100.?.2].[run(); delete()], [103.1].[run()];
                    [101].[destroy()];
                end
            }}}
    Elements {
        1. CONTENT 102 «VIDEO» {
            Classification video,
            Original-Perception {
                original-duration 17,
                original-size [582,402,0]}
            Content-Hook video-ISO-11172-MPEG-Video
            Content-Data {
                Data-Reference «video_ex.mpg»}}
        2. CONTENT 103 «AUDIO» {
            Classification audio,
            Original-Perception {
                original-duration 17}
            Content-Hook audio-ISO-11172-MPEG-Audio
            Content-Data {
                Data-Reference «audio_ex.mpg»}}
        3. CONTENT 104 «FIN» {
            Classification audio,
            Original-Perception {
                original-duration 6}
            Content-Hook audio-ISO-11172-MPEG-Audio
            Content-Data {
                Data-Reference «fin_ex.mpg»}}}
```

Figure 3.16 Exemple de représentation MHEG

3.5 Conclusion

L'adoption de la norme MHEG pour la représentation d'informations multimédias et hypermédias permet le transfert et le traitement de ces types d'information dans un système ouvert. A cause de la complexité structurelle des systèmes hypermédias, leur conception ne peut pas être abordée efficacement et aisément en utilisant directement les objets MHEG. Ainsi, nous pensons qu'un modèle logique de plus haut niveau doit être utilisé pour faciliter la description des documents hypermédias.

Dans le chapitre suivant, nous introduisons le modèle Réseaux de Petri Hiérarchisé à Flux Temporels Interprété (RdPHFT-I) permettant la spécification et l'analyse de documents hypermédias, et la génération automatique de représentations MHEG des documents analysés.

Chapitre 4

Le modèle RdPHFT Interprété (RdPHFT-I)

Parmi les modèles analysés dans le chapitre 2, il n'y a pas de modèle formel permettant la description facile, complète et multi-niveaux des structures de contenu, conceptuelle et de présentation des systèmes hypermédias répartis. De plus, tous ces modèles adoptent des représentations physiques particulières et ils ne proposent aucune méthode permettant le transfert des informations dans un système ouvert.

Parmi les modèles étudiés, le modèle *Réseaux de Petri Hiérarchisés à Flux Temporels* (RdPHFT) satisfait à presque tous les besoins pour la spécification de la structure conceptuelle (définition des composants et ses relations logiques et temporelles) des documents hypermédias. Le modèle RdPHFT permet une spécification formelle unifiée et précise de la synchronisation logique et temporelle à l'intérieur des systèmes hypermédias distribués. De plus, l'aspect graphique et hiérarchique de ce modèle simplifie la modélisation de la structure conceptuelle du document. Cependant, ce modèle n'a pas pour objectif la description complète de documents hypermédias; par exemple, il ne fournit pas les moyens de spécifier les informations d'accès ou les caractéristiques spatiales et sonores des présentations qui composent un document.

Ce chapitre propose une version interprétée du modèle RdPHFT, appelée RdPHFT-I, qui étend le modèle RdPHFT afin de permettre la spécification complète de documents hypermédias à partir d'une approche multi-niveaux, distinguant les structures du contenu, conceptuelle et de présentation d'un document. Le terme «interprétation» signifie que l'on étend la sémantique des places du modèle RdPHFT. Cette sémantique a été définie à partir des concepts proposés par la norme MHEG, cela afin de permettre la génération automatique de représentations ouvertes MHEG à partir du modèle RdPHFT.

L'objectif du modèle RdPHFT-I est de permettre la spécification formelle des documents hypermédias dans leurs formes finales, de fournir les moyens de simulation, d'analyse et de vérification du document et enfin, de permettre la génération automatique d'une représentation MHEG du document spécifié. Cette représentation peut être alors stockée, échangée et présentée dans un système ouvert. Le but de ce travail n'est pas d'obtenir un modèle permettant de spécifier tous les documents hypermédias représentables en MHEG, mais seulement un ensemble structuré et maîtrisable dont les structures conceptuelles peuvent être spécifiés par le modèle RdPHFT.

Ce chapitre est organisé de la façon suivante : la section 4.1 introduit informellement le modèle

RdPHFT-I; la section 4.2 présente l'approche multi-niveaux de description de documents hypermédias fournie par le modèle RdPHFT-I; la section 4.3 présente la définition formelle du modèle RdPHFT-I; la section 4.4 énumère les apports de l'interprétation du modèle RdPHFT; la section 4.5 énonce les limitations du modèle proposé; la section 4.6 présente ensuite un exemple de développement d'un document hypermédia; enfin, la section 4.7 présente les conclusions de cette étude.

4.1 Introduction au modèle RdPHFT-I

Comme cela a été présenté dans le chapitre 2, un modèle hypermédia idéal devrait permettre la description de trois structures distinctes : la **structure conceptuelle**, qui définit les composants, les groupes de composants, les relations logiques entre ces composants, et leur composition temporelle; la **structure de présentation**, qui englobe la description des caractéristiques spatiales, sonores et temporelles des présentations, et les relations spatiales entre ces présentations; et la **structure du contenu**, qui spécifie les données primitives qui seront présentées.

RdPHFT (voir section 2.4.7.10) est un modèle logique permettant la spécification formelle de la structure conceptuelle des documents hypermédias. La description de la structure du contenu et de la structure de présentation ne font pas partie du domaine de spécification du modèle RdPHFT. Le modèle RdPHFT-I étend le modèle RdPHFT afin de permette la spécification de ces deux structures. Ainsi, le modèle RdPHFT-I permet la spécification complète de documents hypermédias. La figure 4.1 présente comment le modèle RdPHFT-I permet la spécification de ces structures.

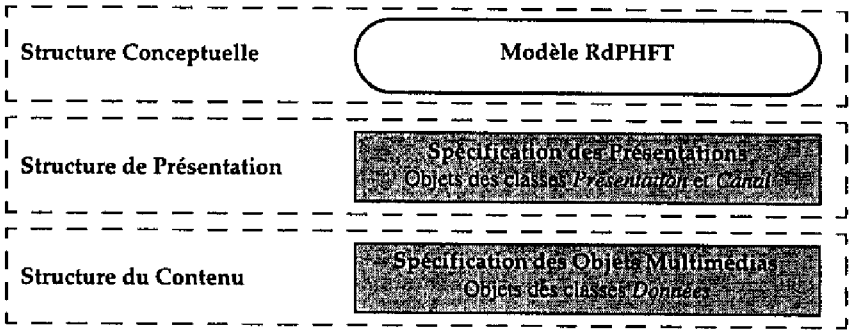


Figure 4.1 Les niveaux de spécification du modèle RdPHFT-I.

4.1.1 Structure du contenu

Comme cela a été présenté dans la section 2.3.1, la structure du contenu est définie par un ensemble d'objets multimédias qui spécifient les données primitives. Ces dernières sont incluses implicitement ou explicitement dans les objets multimédias.

Dans MHEG, les données médias et non-médias (définies dans la section 2.3.1) sont représentées par des objets CONTENT et MULTIPLEXED CONTENT. Les conteneur de script sont représentés par des objets SCRIPT. Les objets MHEG fournissent les moyens de représenter la classification des données, les caractéristiques originelles de perception, les informations de codage et décodage, et les données (par inclusion implicite ou explicite). Dans l'approche MHEG, n'importe quel type de données médias ou non-médias est représenté par deux structures génériques, définies par les classe CONTENT (incluant MULTIPLEXED CONTENT) et SCRIPT. De cette façon, la cohérence d'une représentation MHEG ne peut pas être garantie :

- Dans ces structures génériques, le domaine des valeurs des attributs communs à différents types de données doit couvrir toutes les valeurs possibles de tous ces types. Ainsi, l'auteur peut définir une

valeur incohérente à un attribut. Par exemple, la définition d'une méthode de codage d'image pour des données du type audio.

- Ces structures génériques doivent permettre la représentation de toutes les informations de tous les types de données. Ainsi, l'auteur peut définir une caractéristique incohérente par rapport au type de données. Par exemple, la définition d'une caractéristique spatiale pour une information audio.

Dans le modèle RdPHFT-I, les objets multimédias sont modélisés par un ensemble d'objets, appelé *SD* (*Spécification des Données*). Ces objets sont des instances de différentes classes *Données* présentées dans la figure 4.2. Une représentation dans le langage C++ de ces classes est disponible dans l'annexe A.

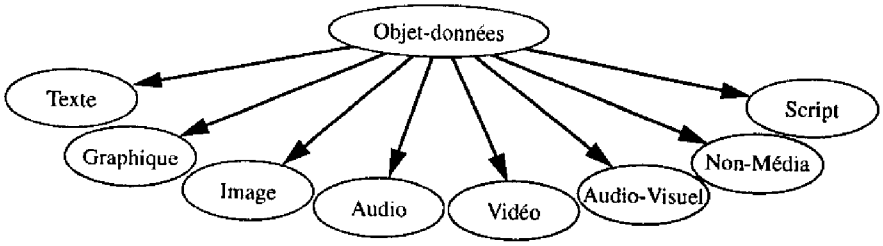


Figure 4.2 Classes *Données*[1]

Chaque classe *Données* définit une structure permettant la description d'un type de donnée média, non-média ou script. Ainsi, l'auteur du document peut choisir une des classes pour représenter :

- Des données primitives d'un type de média pré-défini (texte, graphique, image, audio, vidéo et audio-visuel). Ces informations sont spécifiées par des objets de la classe *Texte*, *Graphique*, *Image*, *Audio*, *Vidéo* ou *Audio-Visuel*.
- Des données primitives du type non-média, codées selon les normes SGML, HyTime, ODA, HyperODA ou autre type de codage ou exécutable défini par l'auteur du document. Ainsi, le modèle proposé permet la spécification de n'importe quel type d'information. Ces informations sont spécifiées par des objets de la classe *Non-Média*.
- Des conteneurs de script, spécifiés par des objets de la classe *Script*.

Les classes *Données* contiennent un certain nombre d'attributs qui permettent la spécification d'un ensemble de base des attributs MHEG des classes CONTENT, MULTIPLEXED CONTENT et SCRIPT. La classe *Objet-données* définit les attributs communs aux différents types d'information :

- *Description* : permet la spécification d'informations générales, telles que le nom, la version et le propriétaire.
- *Contenu* : permet la spécification d'une adresse logique des données.
- *Inclusion* : identifie si les données primitives vont être incluses implicitement ou explicitement dans l'objet MHEG (c'est-à-dire si ces données vont être référencées ou incluses dans l'objet multimédia).
- *Codage-propriétaire* : identifie la méthode de codage définie par l'auteur du document. Cette information est optionnelle. Si aucune valeur n'est affectée à cet attribut, la méthode de codage est définie par l'attribut *Informations de manipulation* (voir ci-dessous). Avec cet attribut, le modèle RdPHFT-I permet la spécification de n'importe quelle méthode de codage.

De plus, chaque classe *Données* a un ensemble d'attributs spécifiques, où le type d'information spécifié par ces attributs dépend du type d'information représenté. Ces attributs sont les suivants :

- *Informations de manipulation* : il spécifie les informations nécessaires à la manipulation des données primitives. Cet attribut permet la spécification de la méthode et d'autres informations additionnelles de codage/décodage pour les types de données média et non-média (ex.: MPEG, JPEG,

1. Les flèches représentent de relations d'héritage.

HyTime), et le langage de script (ex. : smsl, scriptX) pour les conteneurs de script.

La méthode de codage des données ou bien le langage de script peuvent être choisis à partir de listes pré-définies ou bien peuvent être définis par l'auteur du document. Ainsi, ce modèle supporte n'importe quelle méthode de codage et n'importe quel langage de script (par l'utilisation de l'attribut *Codage-propriétaire* de la classe *Objet-données*).

- *Caractéristiques originelles de présentation* : cet attribut spécifie les caractéristiques spatiales, temporelles et sonores originelles (par défaut) de présentation des données primitives. Les types d'informations dépendent du type de données :
  - pour les informations dynamiques, cet attribut spécifie le rapport temporel et la durée naturelle. La durée naturelle décrit le temps originel de la présentation de l'information;
  - pour les informations sonores, il spécifie le volume et son domaine (le volume minimum et maximum);
  - pour les informations visibles, il spécifie la taille spatiale de présentation.

Le modèle RdPHFT-I fournit une classe primitive pour chaque type de données : ainsi l'auteur peut créer des objets multimédias comme des instances de ces classes. Par exemple, une séquence vidéo *Données-Vidéo* peut être modélisée par l'objet de la classe *Vidéo* ci-dessous :

Vidéo *Données-Vidéo* : := {
    Contenu *«/home/willrich/Donnees/donneesVideo.mpg»;*
    Inclusion *explicite;*
    Codage *ISO_11172_MPEG_Video;*
    Caractéristiques-originelles-de-présentation :
        Taille-originelle *128 (H) x 256 (W);*
        Durée-originelle *10 s;*
};

La spécification des objets multimédias par l'utilisation de classes différentes (chacune associée à un type des données) permet la préservation de la consistance de la spécification. Dans tous les cas, la définition d'un nouveau type de données ou d'un nouveau type de méthode de codage ne peuvent pas être vérifiés. Dans ce cas, la consistance de la spécification doit être garantie par l'auteur du document.

Dans un système hypermédia réparti, ces objets multimédias peuvent être stockés n'importe où dans le système, dans des serveurs multimédias ou dans un disque local. Les informations locales et distantes sont traitées de la même façon. Les actions d'échange des objets multimédias ne sont pas représentées explicitement dans le modèle. C'est de la responsabilité de la machine MHEG d'effectuer ces actions.

4.1.2 Structure de présentation

La modélisation de la structure de présentation d'un document hypermédia englobe la spécification des caractéristiques de présentation des composants du document.

Dans la norme MHEG, une présentation particulière est exprimée par un rt-objet (présenté dans la section 3.1). Un dispositif de sortie est exprimé par un canal logique. Rt-objets et canaux sont des entités abstraites. MHEG ne définit aucune structure pour le codage des rt-objets et des canaux. Ces entités sont créées, manipulées et détruites par l'utilisation des actions MHEG. En particulier, MHEG permet la représentation de toutes les caractéristiques de présentation d'un rt-objet pendant toute sa vie. Quand un rt-objet est créé, il contient les caractéristiques de présentation de l'objet MHEG à partir duquel il a été créé. Des objets ACTION sont utilisés pour changer ces caractéristiques. Dans l'approche adoptée par MHEG, n'importe quelle action peut être envoyée à n'importe quel rt-objet. De cette façon, la cohérence de la représentation des caractéristiques de présentation ne peut pas être garantie. Un exemple de spécification incohérente est la définition d'un volume sonore pour une image.

4.1.2.1 Spécification des caractéristiques de présentation

Dans le modèle RdPHFT-I, les caractéristiques de présentation des objets multimédias sont représentées par un ensemble d'objets, appelés *SP* (*Spécification des Présentations*). Ces objets sont des instances de différentes classes *Présentation*. Les classes *Présentation* que nous avons définies sont présentées dans la figure 4.3. Une représentation C++ de ces classes est disponible dans l'annexe A.

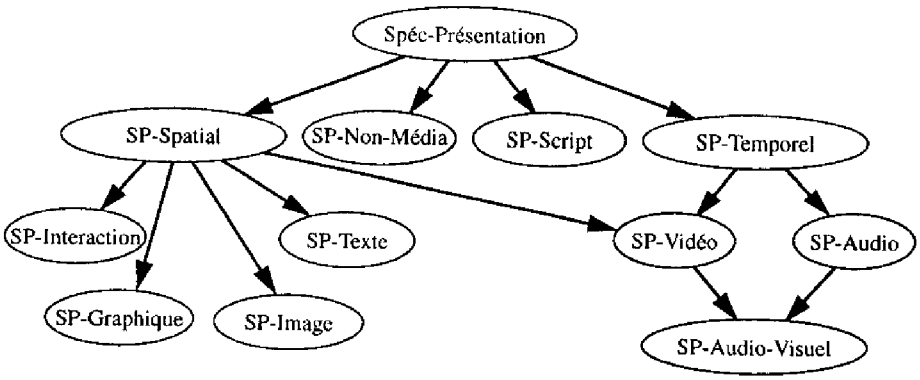


Figure 4.3 Classes *Présentation*

Chaque classe *Présentation* définit une structure permettant la spécification des caractéristiques de présentation des composants du type média, non-média, script, ou des ancres. Ci-dessous, nous présentons la description des différentes classes *Présentation* :

- *Spéc-Présentation* : un objet de cette classe contient les attributs de présentation communs à tous les types d'information. Il contient l'attribut :
  - *Description*, qui permet de la spécification d'une description de la présentation (comme l'auteur et la version);
  - *Canal*, qui fait référence à un objet *Canal*, défini dans *SC* (voir la section 4.1.2.2), sur lequel l'information sera présentée;
  - *Priorité*, qui spécifie la priorité de la présentation. Une présentation plus prioritaire est présentée au-dessus d'une présentation moins prioritaire.;
  - *Présentation-Alternative*, qui spécifie la durée maximale de préparation de la présentation et une présentation alternative spécifiée par un objet d'une des classes *Présentation*. Cette présentation alternative peut remplacer la présentation principale si celle-ci ne peut pas être présentée dans un certain système (présentation hypermédia adaptable aux ressources disponibles), s'il y a des problèmes d'accès à l'information principale, ou si la durée maximale de préparation est dépassée.
- *SP-Spatial* : un objet de cette classe spécifie les caractéristiques spatiales des données média du type audio-visuel, vidéo, texte, image ou graphique, ou encore des mécanismes d'interaction. Par exemple, il spécifie la taille et la position de présentation.
- *SP-Temporel* : un objet de cette classe spécifie les caractéristiques temporelles des données média du type audio-visuel, vidéo ou audio (les informations dynamiques). Par exemple, il spécifie la position de départ et de fin d'une présentation, le nombre de présentations et la vitesse de présentation.

  L'attribut *Flexibilité* (*Stretchability*) spécifie la possibilité de changer la durée de présentation et les méthodes qui sont disponibles pour faire cela. Cet attribut peut spécifier que le changement de durée sera fait par le changement de la vitesse de présentation, ou par réplication ou interruption de présentation[2]. L'utilisation de cet attribut sera présentée dans la section 4.4.1.

2. D'autres méthodes peuvent être envisagées, par exemple, combler l'espace vide avec des blancs ou une présentation par défaut.

L'attribut *Terminaison* spécifie le comportement d'une présentation lors de l'attente d'un point de synchronisation. Il spécifie un choix entre geler la présentation dans sa dernière position, arrêter la présentation ou exhiber une présentation alternative pour combler l'espace vide. Des mécanismes similaires sont proposés par [Steinmetz, 90] et [Hudson, 93] (voir sections 2.4.4 et 2.4.5.3).

- *SP-Non-Média* : la structure définie par cette classe permet la spécification des caractéristiques de présentation des données non-média. Il contient un attribut *Objet-données* qui fait référence à un objet de la classe *Non-Média*.
- *SP-Script* : un objet de cette classe spécifie une présentation d'un *script*. Il contient un attribut *Objet-données* qui fait référence à un objet de la classe *Script*;
- *SP-Audio* : un objet de cette classe spécifie une présentation d'une séquence audio. Il contient un attribut *Objet-données* qui fait référence à un objet de la classe *Audio*, et un autre qui spécifie le volume de présentation.
- *SP-Vidéo* : un objet de cette classe spécifie une présentation d'une séquence vidéo. Il contient un attribut *Objet-données* qui fait référence à un objet de la classe *Vidéo*.
- *SP-Interaction* : un objet de cette classe spécifie une région sensible, bouton ou menu. Il contient l'attribut *Orientation*, qui spécifie l'orientation spatiale du menu (*vertical, horizontal*).
- *SP-Graphique, SP-Image*, et *SP-Texte* : un objet de ces classes spécifie une présentation d'une information du type graphique, image, ou texte, respectivement. Il contient un attribut *Objet-données* qui fait référence à un objet de la classe *Graphique, Image*, ou *Texte*.

Par exemple, une présentation de la séquence vidéo *Données-Vidéo* peut être modélisée par l'objet de la classe *SP-Vidéo* ci-dessous. Cet objet spécifie les 150 premières trames de la séquence vidéo *Données-Vidéo* qui seront présentées dans la position [100,100,0] du canal *Canal-Vidéo* (voir section 4.1.2.2). Et à la fin, la dernière trame sera affichée jusqu'à l'interruption de la présentation.

SP-Vidéo *Prés-Vidéo* : := {

Objet-données	*Données-Vidéo;*
Canal	*Canal-Vidéo;*
Position-départ	*0;*
Position-fin	*150;*
Terminaison	*gelé;*
Position-présentation	*[100,100,0];*

};

De la même façon que pour la modélisation des objets multimédias, nous utilisons des classes différentes pour spécifier les caractéristiques de présentation de types d'informations différentes, cela afin de maintenir la spécification consistante.

4.1.2.2 Spécification des canaux

Dans les applications hypermédias, les différentes présentations peuvent être présentées sur différents médias de présentation (les dispositifs de sortie), comme un écran, du papier, des haut-parleurs, etc. En MHEG, un canal est un espace logique sur lequel une présentation se déroule. La machine MHEG fait la correspondance entre cet espace logique et le dispositif réel de sortie. Des exemples de canaux sont : des fenêtres, des haut-parleurs, des moniteurs, etc. La norme MHEG ne définit aucune structure de représentation d'un canal (c'est un concept abstrait). Les canaux peuvent être créés, manipulés et détruits par l'utilisation des actions MHEG.

Dans le modèle RdPHFT-I, les canaux utilisés par un document hypermédia sont représentés par un ensemble d'objets, appelés *SC* (*Spécification des Canaux*). Ces objets sont des instances de différentes classes *Canal*, présentées dans la figure 4.4. Une représentation dans le langage C++ de ces classes est disponible dans l'annexe A.

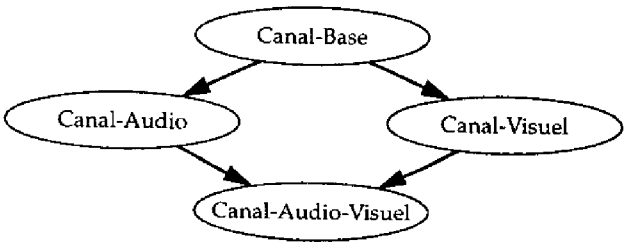


Figure 4.4 Les classes *Canal*

Un objet d'une des classes *Canal* décrit l'identificateur du canal et ses exigences. Ci-dessous, nous présentons les classes *Canal* :

- *Canal-Base* : elle définit une structure permettant la spécification des informations de base de tous les types de canal. Elle permet l'identification du rapport temporel (nombre d'intervalles consécutifs d'une unité temporelle dans une seconde).
- *Canal-Visuel* : un objet de cette classe permet la définition des intervalles admissibles des axes x, y et z ([origine, point final]) d'un canal visuel.
- *Canal-Audio* : un objet de cette classe permet la définition de l'intervalle de variation de l'audio d'un canal audio (en dB) et sa position (gauche, droite). La position d'une audio stéréo n'est pas définie.
- *Canal-Audio-Visuel* : un objet de cette classe permet la définition des informations présentées dans les classes *Canal-Visuel* et *Canal-Audio-Visuel*.

L'auteur doit définir un canal dans le seul cas où il veut utiliser explicitement différents canaux (*i.e.*, pour afficher différentes présentations dans différents canaux). S'il ne spécifie pas un canal, la présentation sera affichée dans un canal par défaut (ce canal existe toujours).

Par exemple, le canal utilisé par la présentation *Prés-Vidéo* est spécifié par l'objet de la classe *Canal-Visuel* ci-dessous.

```
Canal-Visuel Canal-Vidéo : := {
    Intervalle-Axe-x        [0,2000];
    Intervalle-Axe-y        [0,2000];
};
```

4.1.3 Structure conceptuelle

Dans l'approche orientée-objet proposée par MHEG, les relations logiques, temporelles et spatiales des différentes présentations d'un document hypermédia sont représentées par des objets ACTION, LINK et SCRIPT. Comme c'est le cas de toutes les approches orientées langage, l'approche MHEG a un pouvoir d'expression très grand, mais la spécification de la composition d'un document hypermédia en utilisant des objets MHEG est difficile à produire et à modifier. Dans cet approche, nous adoptons, dans un but de structuration, le modèle RdPHFT pour la spécification de la structure conceptuelle des documents hypermédias.

4.1.3.1 Interprétation du modèle RdPHFT

Afin de permettre la spécification complète de documents hypermédias et la génération automatique de représentations MHEG, nous avons ajouté de nouvelles abstractions au modèle RdPHFT. Ces extensions ont été réalisées par l'interprétation du modèle RdPHFT, c'est-à-dire que nous avons associé les caractéristiques de présentation aux places du type atomique et lien. Cette association est réalisée par la fonction F_{SP}. Cette fonction associe une place du type atomique ou lien avec un objet défini dans *SP* (*i.e.*, un objet d'une classe *Présentation*). De plus, nous avons ajouté au modèle RdPHFT deux autres fonctions, appelées F_{SD} et F_{SC} pour associer les objets *Présentation* avec les objets *Données* et *Canal*, respectivement.

4.1.3.2 Structuration du niveau de *Synchronisation Logique*

Comme nous l'avons présenté dans la section 2.4.7.10, le niveau de *Synchronisation Logique* du modèle RdPHFT est spécifié uniquement par un réseau plat. Le modèle RdPHFT-I utilise une version étendue du modèle RdPHFT permettant la réalisation d'une spécification hiérarchique du niveau de *Synchronisation Logique*. La figure 4.5 présente un exemple de niveau de *Synchronisation Logique* structuré. Le réseau Racine spécifie le chemin de parcours entre les département d'une université, et le réseau EECS spécifie le chemin de parcours de la présentation d'un des départements de l'université.

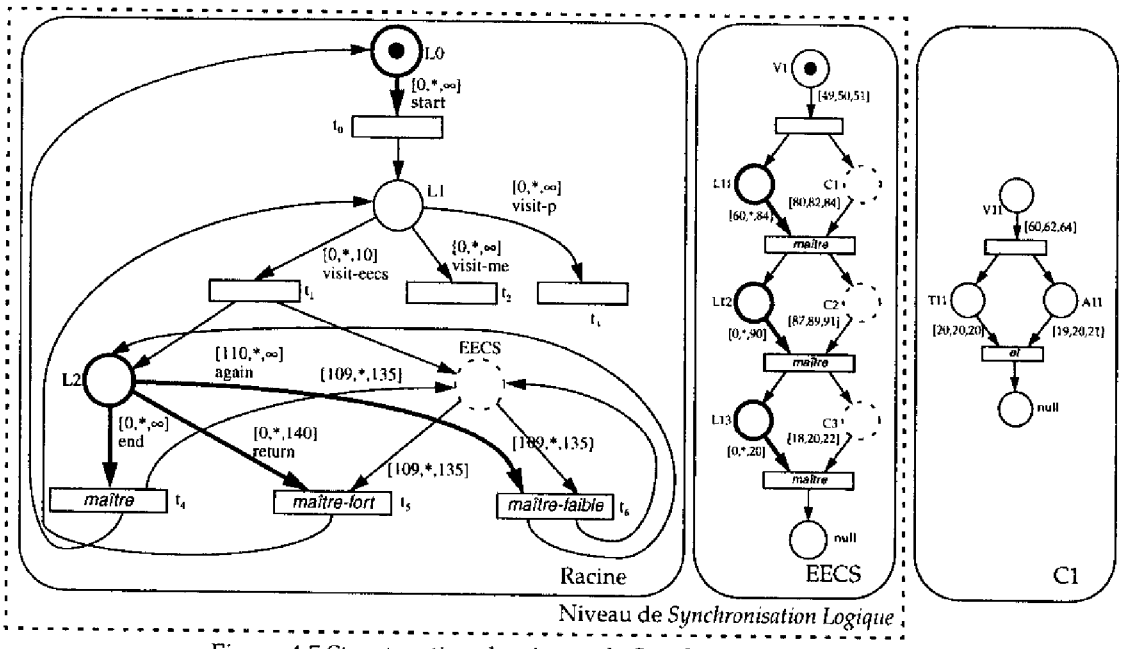


Figure 4.5 Structuration du niveau de Synchronisation Logique

4.2 Description multi-niveaux du modèle RdPHFT-I

Le modèle RdPHFT-I permet la description multi-niveaux d'un document hypermédia, incluant la structure conceptuelle, la structure de présentation et la structure du contenu. La figure 4.6 présente la structure multi-niveaux du modèle RdPHFT-I et l'interface entre ces niveaux.

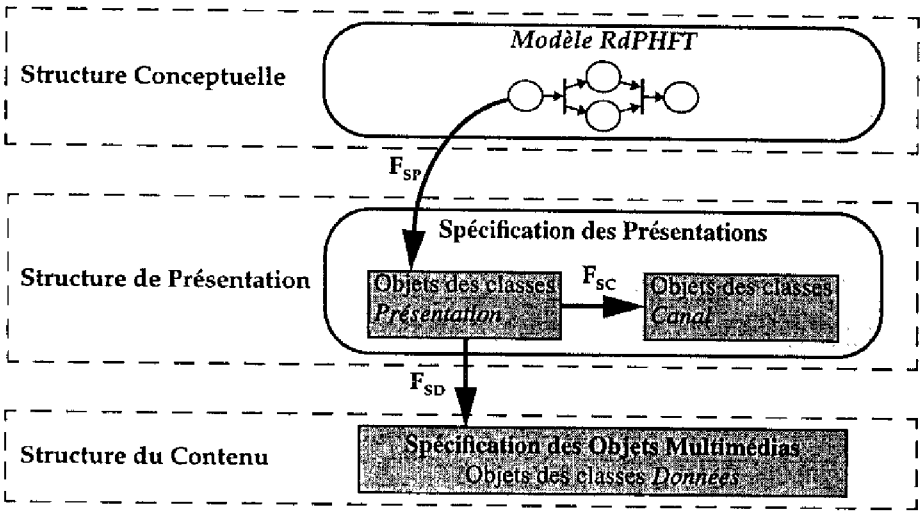


Figure 4.6 Description multi-niveaux du modèle RdPHFT-I.

La structure conceptuelle d'un document hypermédia est décrite par une spécification RdPHFT, permettant la définition des composants et groupes des composants, des relations logiques et temporelles entre ces composants. Le modèle RdPHFT-I présente une très nette séparation entre la définition d'un composant et la définition de la présentation du contenu d'un composant via la fonction F_{SP}. Cette caractéristique permet la réutilisation de la structure conceptuelle lors de la présentation du document hypermédia sur différents systèmes clients. Par exemple, si une place atomique représente la présentation d'une information audio, et si le système client ne supporte pas ce type de média, un nouveau objet *Présentation* peut être associée à cette place (p.ex., en spécifiant une présentation textuelle).

La structure de présentation est définie par un ensemble d'objets *Présentation* et *Canal*. Les définitions des présentations et des données primitives sont réalisées par des objets distincts. Cette caractéristique permet la réutilisation des objets multimédias en différents contextes du document et par différents documents. De plus, une place peut représenter la présentation d'une certaine partie de ces données (à l'aide des attributs *Start-position* et *End-position*). Ainsi, le modèle RdPHFT-I ne requiert pas l'édition des données primitives afin de permettre la définition d'autres points de synchronisation que le début ou la fin de présentation, ce qui était une des limitations des autres modèles basés sur les réseaux de Petri (section 2.3.5.4). Par exemple, la présentation d'une séquence vidéo peut être représentée par plusieurs places. Chaque place modélise une partie de la présentation qui peut être synchronisée, par exemple, avec la présentation d'informations textuelles.

4.3 Définition formelle du modèle RdPHFT-I

Dans cette section, nous rappelons tout d'abord la définition des modèles RdPFT et RdPHFT, et ensuite, nous présentons la définition formelle du modèle RdPHFT interprété.

Définition 1. Un **RdPFT** ([Diaz, 93], [Sénac, 95a]) est un tuplet $(P, T, \alpha, \beta, M_0, IM, SYN, AM)$, tel que :

- $(P, T, \alpha, \beta, M_0)$ définit un réseau de Petri.
- IM est une application qui associe un IVT à un arc sortant d'une place, où :
  - $A = \{a = (p, t) \in PxT \mid \beta(p, t) \neq 0\}$
  - $IM : A \rightarrow (Q^+ \cup \infty) \times (Q^+ \cup \infty) \times (Q^+ \cup \infty)$, $IM(a) \rightarrow [x, n, y]$, $0 \leq x \leq n \leq y$

  où [x, n, y], associé avec un arc, représente respectivement la durée minimale, nominale et maximale admissibles pour le traitement associé à l'arc. Cet intervalle est appelé Intervalle de Validité Temporelle de l'arc (IVT).
- SYN est une application qui associe un type de synchronisation à une transition :
  $SYN : T \rightarrow$ {*et, et-faible, ou, ou-fort, maître, ou-maître, et-maître, maître-fort, maître-faible*}
- AM (*Arc Maître*) est une application qui associe une transition du type maître avec un arc maître, où:
  - $T_m = \{t \in T \mid SYN(t) \in$ {*maître, ou-maître, et-maître, maître-fort, maître-faible*}$\}$
  - $AM : T_m \rightarrow A$

Définition 2. Un **RdPHFT** est un tuplet RdPHFT = (R, L, S, Pin, FS, Fin) tel que :

- $R = (P_r, T_r, \alpha_r, \beta_r, M_{0r}, IM_r, SYN_r, AM_r, PT_r, LA_r)$ est un RdPFT étendu avec les places du type lien, atomique et composite, appelé racine de la spécification RdPHFT :
  - $(P_r, T_r, \alpha_r, \beta_r, M_{0r}, IM_r, SYN_r, AM_r)$ définit un RdPFT (voir définition 1).
  - $PT_r : P_r \rightarrow$ {*atomique, composite, lien*} est une application d'affectation du type de place.
  - LA_r (*Label*) est une fonction qui associe un nom à chaque arc sortant d'une place du type lien, où :
    * Nom est un ensemble de noms
    * $Al = \{a = (p, t) \in P_r x T_r \mid \beta_r(p, t) \neq 0, PT_r(p) = lien\}$
    * $LA : A1 \rightarrow Nom$

- $L = \{L_i \mid i \in I\}$ est un ensemble fini de RdPFT étendu pour fournir la modélisation de places du type lien, atomique et composite. Cet ensemble et la racine de la spécification spécifient le niveau de *Synchronisation Logique* du document hypermédia.
  - I dénote l'ensemble des index des éléments de L.
  - $L_i = (P_{li}, T_{li}, \alpha_{li}, \beta_{li}, M_{l0i}, IM_{li}, SYN_{li}, AM_{li}, PT_{li}, LA_{li})$ est un RdPFT, appelé une page de la spécification RdPHFT appartenant au niveau de *Synchronisation Logique*.
  - $(P_{li}, T_{li}, \alpha_{li}, \beta_{li}, M_{l0i}, IM_{li}, SYN_{li}, AM_{li})$ définit un RdPFT.
  - $PT_{li} : P_{li} \rightarrow \{atomique, composite, lien\}$ est une application d'affectation du type de place.
  - L'ensemble des éléments du réseau sont des paires disjoints : $\forall (i, k) \in I, i \neq k, ((P_{li} \cup T_{li}) \cap (P_{lk} \cup T_{lk}) = \emptyset)$.
  - LA_{li} (*Label*) est une fonction qui associe un nom à chaque arc sortant d'une place du type lien, où :
    * Nom est un ensemble de noms
    * $Al_{li} = \{a = (p, t) \in P_{li} x T_{li} \mid (i \in I, \beta_{li}(p, t) \neq 0, PT_{li}(p) = lien)\}$
    * $LA_{li} : A1_{li} \rightarrow Nom$
- $S = \{S_j \mid j \in J\}$ est un ensemble fini de RdPSFT (*Réseau de Petri Structuré à Flux Temporels*) [Sénac, 95a] étendu pour fournir la modélisation de places du type atomique et composite :
  - J dénote l'ensemble des index des éléments de S.
  - $S_j = (P_{sj}, T_{sj}, \alpha_{sj}, \beta_{sj}, M_{s0j}, IM_{sj}, SYN_{sj}, AM_{js}, PT_{sj})$ est un RdPSFT, appelé une page de la spécification RdPHFT.
  - $(P_{sj}, T_{sj}, \alpha_{sj}, \beta_{sj}, M_{s0j}, IM_{sj}, SYN_{sj}, AM_{js})$ définit un RdPSFT.
  - $PT_{sj} : P_{sj} \rightarrow \{atomique, composite\}$ est une application d'affectation du type de place.
  - L'ensemble des éléments du réseau sont des paires disjoints : $\forall (i, k) \in J, i \neq k, ((P_{si} \cup T_{si}) \cap (P_{sk} \cup T_{sk}) = \emptyset)$.
- Pin $\subset$ P est un ensemble de places initiales de R et des sous-réseaux définis dans L ou S, où $P = \{\bigcup_{i \in I} P_{li}\} \cup \{\bigcup_{j \in J} P_{sj}\} \cup P_r$
- $FS : C \rightarrow L \cup S$ est une application qui associe une place composite avec un élément de L ou S, où $C = \{p \in P \mid PT_r(p) = composite\} \cup \{p \in P \mid (i \in I, PT_{li}(p) = composite)\} \cup \{p \in P \mid (j \in J, PT_{sj}(p) = composite)\}$.
- Fin : $L \cup S \rightarrow$ Pin est une application qui associe un élément de l'ensemble L ou S avec une place d'entrée;

Définition 3. Un **RdPHFT interprété** est un tuplet RdPHFT-I = (RdPHFT, DD, SD, SC, SP, CL, F_{SP}, F_{SC}, F_{SD}) tel que :

- RdPHFT = (R, L, S, Pin, FS, Fin) définit un RdPHFT (voir définition 2).
- DD (Description du Document) est un objet dérivé de la classe *SP-Spatial*. Il définit le nom du document, sont auteur, sa version, et les caractéristiques spatiales de présentation du document (par exemple, la taille de la fenêtre principale.
- SD est un ensemble d'objets dérivés des classes *Données* (classes *Texte, Graphique, Image, Audio, Vidéo, Audio-Visuel, Non-Média* et *Script*). L'ensemble SD est divisé en 6 sous-ensembles, notés par : *Ensemble-Texte, Ensemble-Graphique, Ensemble-Image, Ensemble-Audio, Ensemble-Vidéo, Ensemble-Audio-Visuel, Ensemble-Non-Média* et *Ensemble-Script*. Chaque sous-ensemble est composé d'objets des classes *Texte, Graphique, Image, Audio, Vidéo, Audio-Visuel, Non-Média* et *Script*, respectivement. Ainsi, SD = {*Ensemble-Texte*} $\cup$ {*Ensemble-Graphique*} $\cup$ {*Ensemble-Image*} $\cup$ {*Ensemble-Audio*} $\cup$ {*Ensemble-Vidéo*} $\cup$ {*Ensemble-Audio-Visuel*} $\cup$ {*Ensemble-Non-Média*} $\cup$ {*Ensemble-Script*}.
- SC (*Spécification des Canaux*) est un ensemble d'objets dérivés des classes *Canal* (classes *Canal-Audio, Canal-Visuel* et *Canal-Audio-Visuel*). L'ensemble SC est composé par trois sous-ensembles : *Ensemble-SC-Audio, Ensemble-SC-Visuel* et *Ensemble-SC-Audio-Visuel*. Ces sous-ensembles contiennent les objets des classes *Canal-Audio, Canal-Visuel* et *Canal-Audio-Visuel*, respectivement. Ainsi $O_c = \{\} \cup$

{*Ensemble-SC-Audio*} ∪ {*Ensemble-SC-Visuel*} ∪ {*Ensemble-SC-Audio-Visuel*}.

- SP est un ensemble d'objets dérivés des classes *Présentation* (classes *SP-Interaction, SP-Graphique, SP-Image, SP-Texte, SP-Non-Média, SP-Script, SP-Vidéo, SP-Audio* et *SP-Audio-Visuel*). L'ensemble SD est divisé en 9 sous-ensembles, notés par : *Ensemble-SP-Interaction, Ensemble-SP-Graphique, Ensemble-SP-Image, Ensemble-SP-Texte, Ensemble-SP-Non-Média, Ensemble-SP-Script, Ensemble-SP-Vidéo, Ensemble-SP-Audio* et *Ensemble-SP-Audio-Visuel*. Chaque sous-ensemble est composé d'objets des classe *SP-Interaction, SP-Graphique, SP-Image, SP-Texte, SP-Non-Média, SP-Script, SP-Vidéo, SP-Audio* et *SP-Audio-Visuel*, respectivement. Ainsi, SP = {*Ensemble-SP-Interaction*} ∪ {*Ensemble-SP-Graphique*} ∪ {*Ensemble-SP-Image*} ∪ {*Ensemble-SP-Texte*} ∪ {*Ensemble-SP-Non-Média*} ∪ {*Ensemble-SP-Script*} ∪ {*Ensemble-SP-Vidéo*} ∪ {*Ensemble-SP-Audio*} ∪ {*Ensemble-SP-Audio-Visuel*}.
- CL (*CLassification*) est une fonction qui associe un type d'information à chaque place du type atomique, où :
  - $P_{at} = \{p \in P \mid PT_r(p) = atomique\} \cup \{p \in P \mid (i \in I, PT_{si}(p) = atomique)\} \cup \{p \in P \mid (j \in J, PT_{si}(p) = atomique)\}$
  - $CL : P_{at} \rightarrow \{\} \cup \{texte, graphique, image, audio, vidéo, audio\text{-}visuel, script, non\text{-}média\}$
- F_{SP} est une fonction qui associe un objet *Présentation* avec une place du type atomique ou lien défini dans le RdPHFT, où :
  - $P_{al} = P_{at} \cup \{p \in P \mid PT_r(p) = lien\} \cup \{p \in P \mid (i \in I, PT_{li}(p) = lien)\}$
  - $F_{SP} : P_{al} \rightarrow SP$, où :
    - $\forall p \in P_{al} \mid PT_r(p) = lien, F_{SP}(p) \in$ *Ensemble-SP-Interaction*
    - $\forall p \in P_{al} \mid (i \in I, PT_{li}(p) = lien), F_{SP}(p) \in$ *Ensemble-SP-Interaction*
    - $\forall p \in P_{al} \mid CL(p) = texte, F_{SP}(p) \in$ *Ensemble-SP-Texte*
    - $\forall p \in P_{al} \mid CL(p) = graphique, F_{SP}(p) \in$ *Ensemble-SP-Graphique*
    - $\forall p \in P_{al} \mid CL(p) = image, F_{SP}(p) \in$ *Ensemble-SP-Image*
    - $\forall p \in P_{al} \mid CL(p) = audio, F_{SP}(p) \in$ *Ensemble-SP-Audio*
    - $\forall p \in P_{al} \mid CL(p) = vidéo, F_{SP}(p) \in$ *Ensemble-SP-Vidéo*
    - $\forall p \in P_{al} \mid CL(p) = audio\text{-}visuel, F_{SP}(p) \in$ *Ensemble-SP-Audio-Visuel*
    - $\forall p \in P_{al} \mid CL(p) = script, F_{SP}(p) \in$ *Ensemble-SP-Script*
    - $\forall p \in P_{al} \mid CL(p) = non\text{-}média, F_{SP}(p) \in$ *Ensemble-SP-Non-Média*
- $F_{SC} : SP \rightarrow SC$, est une fonction qui associe un objet *Présentation* avec un objet Canal :
  - $\forall sp \in$ *Ensemble-SP-Interaction*, $F_{SC}(sp) \in$ {*Ensemble-SC-Visuel*}
  - $\forall sp \in$ *Ensemble-SP-Texte*, $F_{SC}(sp) \in$ {*Ensemble-SC-Visuel*} ∪ {*Ensemble-SC-Audio-Visuel*}
  - $\forall sp \in$ *Ensemble-SP-Graphique*, $F_{SC}(sp) \in$ {*Ensemble-SC-Visuel*}∪{*Ensemble-SC-Audio-Visuel*}
  - $\forall sp \in$ *Ensemble-SP-Image*, $F_{SC}(sp) \in$ {*Ensemble-SC-Visuel*} ∪ {*Ensemble-SC-Audio-Visuel*}
  - $\forall sp \in$ *Ensemble-SP-Audio*, $F_{SC}(sp) \in$ {*Ensemble-SC-Audio*} ∪ {*Ensemble-SC-Audio-Visuel*}
  - $\forall sp \in$ *Ensemble-SP-Vidéo*, $F_{SC}(sp) \in$ {*Ensemble-SC-Visuel*} ∪ {*Ensemble-SC-Audio-Visuel*}
  - $\forall sp \in$ *Ensemble-SP-Audio-Visuel*, $F_{SC}(sp) \in$ {*Ensemble-SC-Audio*} ∪ {*Ensemble-SC-Visuel*} ∪ {*Ensemble-SC-Audio-Visuel*}
  - $\forall sp \in$ *Ensemble-SP-Script*, $F_{SC}(sp) \in$ {*Ensemble-SC-Audio*} ∪ {*Ensemble-SC-Visuel*} ∪ {*Ensemble-SC-Audio-Visuel*}
  - $\forall sp \in$ *Ensemble-SP-Non-Média*, $F_{SC}(sp) \in$ {*Ensemble-SC-Audio*} ∪ {*Ensemble-SC-Visuel*} ∪ {*Ensemble-SC-Audio-Visuel*}
- $F_{SD} : SP \rightarrow SD$, est une fonction qui associe un objet *Présentation* avec un objet *Données* :
  - $\forall sp \in$ *Ensemble-SP-Texte*, $F_{SD}(sp) \in$ *Ensemble-Texte*
  - $\forall sp \in$ *Ensemble-SP-Graphique*, $F_{SD}(sp) \in$ *Ensemble-Graphique*
  - $\forall sp \in$ *Ensemble-SP-Image*, $F_{SD}(sp) \in$ *Ensemble-Image*
  - $\forall sp \in$ *Ensemble-SP-Audio*, $F_{SD}(sp) \in$ *Ensemble-Audio*
  - $\forall sp \in$ *Ensemble-SP-Vidéo*, $F_{SD}(sp) \in$ *Ensemble-Vidéo*

- $\forall$ sp $\in$ *Ensemble-SP-Audio-Visuel*, $F_{SD}(sp) \in$ *Ensemble-Audio-Visuel*
- $\forall$ sp $\in$ *Ensemble-SP-Script*, $F_{SD}(sp) \in$ *Ensemble-Script*
- $\forall$ sp $\in$ *Ensemble-SP-Non-Média*, $F_{SD}(sp) \in$ *Ensemble-Non-Média*

4.4 Les apports du modèle RdPHFT-I

Le modèle RdPHFT-I consiste en une interprétation du modèle RdPHFT. Toutes les caractéristiques du modèle RdPHFT sont préservées : les schémas de synchronisation logique et temporelle, les règles de tir et les techniques d'analyse sont maintenus tels qu'ils ont été définis dans [Sénac, 95a] et présentées dans les sections 2.4.7.9 et 2.4.7.10.

L'ajout de la structure du contenu et des caractéristiques de présentation permet une augmentation du pouvoir d'expression du comportement temporel des présentations, et permet aussi l'application de nouveaux types d'analyse.

4.4.1 L'édition de la durée de présentation

Le modèle RdPHFT-I propose une nouvelle sémantique pour la durée nominale définie par l'IVT (Intervalle de Validité Temporelle) d'une présentation dynamique (vidéo, audio, etc.). La durée naturelle de présentation d'une information dynamique est définie par l'attribut *Caractéristiques originelles de présentation* d'un objet *Données*. Lorsque l'IVT d'une présentation définit une durée nominale différente de la durée naturelle de présentation, une action de changement de la durée de présentation doit être exécutée. La possibilité de changement de la durée de présentation et de la méthode utilisée pour cela est définie par l'attribut *Flexibilité* de l'objet *Présentation* correspondant.

L'édition de la durée de présentation d'une information simplifie la tâche de création d'un document hypermédia, cela parce que l'auteur n'a pas besoin d'éditer de façon manuelle les données primitives ou d'adapter les caractéristiques de présentation (comme changer la vitesse de présentation). Par exemple, si la durée naturelle d'une séquence vidéo est 20 s (durée pendant laquelle la vidéo à été captée) et si l'auteur du document spécifie [6,8,10] comme intervalle de validité temporelle d'une présentation de cette vidéo, une action de changement de la durée de présentation de la séquence vidéo a été spécifiée. Cette action dépend de l'attribut *Flexibilité* de la séquence vidéo : si l'attribut *Flexibilité* a la valeur *rigide*, alors la spécification est inconsistante; si cet attribut à la valeur *répétition-Interruption*, la séquence vidéo aura les 12 dernières secondes coupées; et si la valeur de l'attribut *Flexibilité* est égal à *variation-de-vitesse* la vitesse de présentation sera augmentée de 40%.

4.4.2 Attente de synchronisation

A l'aide des 9 sémantiques de synchronisation associées aux transitions du modèle RdPHFT, l'auteur d'un document hypermédia peut définir des règles de synchronisation entre les différentes présentations du document.

L'attribut *Terminaison* d'une information dynamique (représenté par un objet de la classe *SP-Temporel*), permet la description des actions qui peuvent être exécutées quand une présentation a trouvé le point de synchronisation et qu'elle attend les autres présentations. Les actions possibles sont : continuer la présentation de la dernière position (ex. : geler la séquence vidéo), arrêter la présentation, ou présenter une autre information. Ainsi par exemple, si la présentation de la séquence Vidéo1 de la figure 4.7 finit à 19 s et l'Audio1 finit seulement à 24 s, la dernière trame de la vidéo reste affichée pendant 5 s, la présentation vidéo s'arrête complètement, ou une présentation alternative (p.e., une autre image) est affichée jusqu'à la fin de l'Audio1, respectivement.

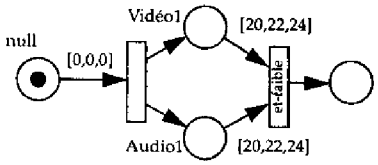


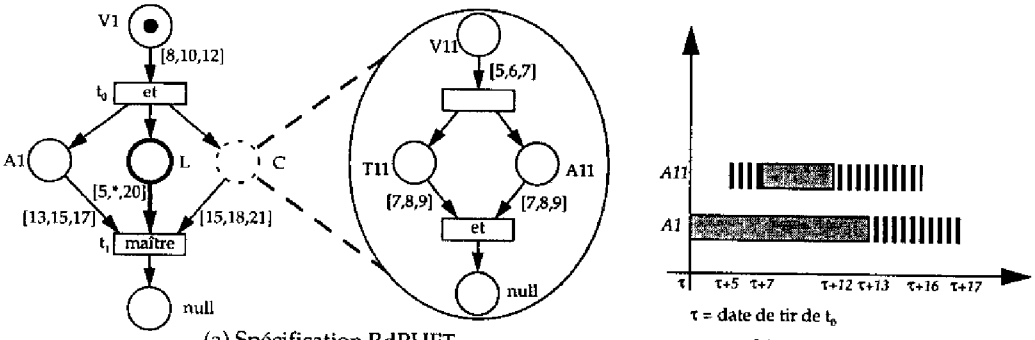
Figure 4.7 Attente de synchronization

4.4.3 Définition des présentations alternatives

L'attribut *Présentation-Alternative* d'un objet *Présentation* permet la définition d'une présentation qui peut remplacer la présentation principale si celle-ci ne peut pas être présentée dans un certain système (présentation hypermédia adaptable aux ressources disponibles), s'il y a des problèmes d'accès à l'information principale, ou si la durée maximale de préparation est dépassée.

4.4.4 Détection des conflits d'utilisation des ressources partagées

Par l'inclusion du concept de canal, le modèle RdPHFT interprété donne un support à la détection automatique des conflits d'utilisation des ressources partagées. Par exemple, si les deux présentations audio A1 et A11 de la figure 4.8a peuvent utiliser le même canal audio au même instant, un avertissement est donné à l'auteur du document.



(a) Spécification RdPHFT (b) Ligne temporelle

Figure 4.8 Détection des conflits d'utilisation des ressources partagées

La technique de détection des conflits d'utilisation des ressources partagées est basée sur le graphe d'accessibilité du réseau : il y est vérifié si deux places peuvent utiliser le même ressource au même instant. La ligne temporelle de la figure 4.8b montre que les présentations audio A1 et A11 requièrent pendant un intervalle de temps l'utilisation du même canal audio, ainsi montre un conflit d'utilisation de ressource partagée.

4.4.5 Description du document

Basé sur la spécification des objets multimédias, des présentations, des méthodes d'interaction et des canaux utilisés par un document hypermédia, le modèle RdPHFT-I permet la génération automatique d'une description des ressources nécessaires à la présentation d'un document. La procédure d'obtention de cette description sera présentée dans la section 5.2.3. Par exemple, la taille des données primitives et les types de médias d'un document peuvent être identifiés. Cette description peut être utilisée lors de l'installation, l'utilisation et la gestion d'un document hypermédia.

4.5 Limitations du modèle RdPHFT(-I)

Dans cette section, nous présentons quelques limitations du modèle RdPHFT et de sa version interprétée.

4.5.1 Limitation à une composition de présentations

Dans le modèle RdPHFT, une place du type atomique représente le processus de traitement des données primitives associées à la place. Dans le cas de la modélisation de la présentation des données, ce traitement peut être divisé en trois parties : la préparation des données primitives (locales ou distantes[3]) et l'allocation des ressources nécessaires à sa présentation; le processus de présentation; et la libération des ressources allouées pour la présentation.

Le modèle XOCPN (présenté dans la section 2.4.7.4), permet de distinguer la préparation des ressources, la présentation et la libération des ressources. Cela à travers les places du type contrôle spécifiant des actions *Resource-setup* et *Resource-release*, et les places objets.

Le modèle RdPHFT ne fournit pas les moyens d'identifier le type de traitement multimédia modélisé par une place. Dans le modèle RdPHFT-I, le traitement modélisé par une place du type atomique correspond à la présentation des données primitives, comprenant, d'une façon unique, la préparation des ressources, la présentation et la libération des ressources. Dans ce cas, la représentation en séparée d'une de ces parties n'est pas possible. Cela entraîne des limitations du pouvoir d'expression du modèle. Par exemple, le modèle RdPHFT-I ne permet pas la spécification du scénario suivant : les informations P1 et P2 sont présentées en séquence; et pendant la présentation de P1 l'information P2 est préparée.

Une solution permettant l'identification du type de traitement est d'ajouter un nouvel attribut associé à une place atomique, appelé *type-traitement*, avec *présentation* et *préparation* comme valeurs possibles. Afin de simplifier l'étude présenté dans ce mémoire, nous allons prendre seulement en considération la modélisation du traitement de présentation des données primitives.

4.5.2 Relations conditionnelles

Dans le modèle RdPHFT-I, les relations temporelles sont modélisées par les transitions, où :

- Des **conditions** sont associées aux places d'entrée. Ces conditions sont :
  - la fin des traitements représentés par les places d'entrée. Dans le modèle RdPHFT-I, la fin du traitement correspond à la fin de présentation d'une information (pour une place atomique ou composite), ou la sélection d'un bouton, menu, ou région sensible (pour une place du type lien).
  - la satisfaction des contraintes temporelles.
- Des **actions** sont associées aux places d'entrée et de sortie (représentant les présentations sources et cibles). Ces actions sont :
  - l'arrêt d'un ensemble de présentations sources (dans le cas du tir forcé);
  - le départ d'un ensemble de présentations cibles.

Dans le modèle RdPHFT, les points de synchronisation sont toujours liés à des présentations en cours et comme résultat on a l'arrêt de toutes les présentations sources et le départ des présentations cibles. Cela n'est pas toujours les cas pour les applications hypermédias, parce que :

- Il peut y avoir des relations conditionnées par l'état d'une présentation (par exemple, si la présentation est en cours ou non). Dans ce cas, on n'a pas forcément l'arrêt de la présentation source lors de

3. Dans le cas de la présentation des données primitives stockées en différents serveurs multimédias, la présentation des données comprend aussi le processus de télé-chargement des données.

la synchronisation. Un exemple est donné dans la figure 4.9. Ce fragment de réseau représente un comportement où à l'instant de la sélection de la place lien B : si C2 est en cours de présentation, la musique M1 est présentée; sinon la musique M2 est présentée. Remarquez que C2 n'est pas arrêtée lors de la sélection du lien B; la présentation d'une musique n'est pas non plus dépendante du départ ou de l'arrêt de C2.

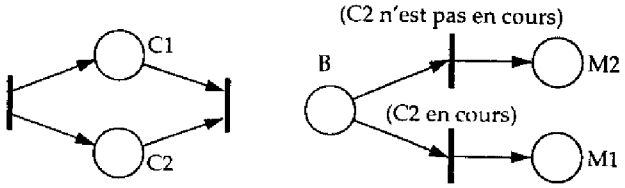


Figure 4.9 Exemple de synchronisation conditionnelle

- Dans le cas général des documents hypermédias, nous pouvons avoir d'autres relations conditionnelles plus complexes qui ne sont pas représentables par le modèle RdPHFT-I, par exemple associées à des mécanismes d'interactions complexes, comme la recherche de documents par mot-clé ou autres (voir la section 2.3.4).

Afin d'exprimer ces comportements, une solution est d'introduire des expressions conditionnelles aux transitions du modèle RdPHFT. Ces conditions pourraient être associées aux attributs des objets *Données, Présentation* ou autres. Cette approche est adoptée, par exemple, par le modèle MHPN (section 2.4.7.8) où une condition/action peut être associée à une transition.

4.5.3 Granularité de synchronisation

Comme nous l'avons présenté dans la section 4.5.2, la condition de tir d'une transition du modèle RdPHFT-I est associée à la fin du traitement d'un ensemble de présentations source et/ou à la sélection d'une place lien (et ses contraintes temporelles). Ainsi, afin de permettre la définition de relations temporelles associées à des événements internes à un composant atomique, ce dernière doit être représentée par différentes places.

Pour illustrer le problème de la granularité de synchronisation du modèle RdPHFT-I, nous introduisons l'application «visite d'une ville». Dans cette application, une séquence audio-visuelle présente brièvement les principaux sites touristiques de la ville. Pendant la présentation d'un site particulier, le touriste peut cliquer sur la vidéo afin d'obtenir plus d'information sur le site. A la fin de la présentation d'un site, le lecteur peut demander la reprise de la séquence audio-visuelle, le retour au début de la présentation de la ville, ou finir l'application.

Une modélisation RdPHFT de l'application «visite d'une ville» est présentée dans la figure 4.10. Dans cet exemple, la séquence audio-visuelle V présente 4 sites d'une ville dans une boucle infinie. Comme dans le modèle RdPHFT une présentation peut être synchronisée seulement avec le début et la fin d'un ensemble de présentations : la séquence audio-visuelle V doit être coupée en 4 parties. Elles sont représentées par les places V_1 à V_4. Pendant la présentation d'un site i (V_i), un lien L_{i1} permet au touriste de demander une description plus détaillée du site (D_i). À la fin de cette description, un lien L_{i2} permet au touriste de demander la reprise de la présentation de la ville dès le début ou dès le prochain site.

Le modèle RdPHFT permet la modélisation logique de l'application «Visite d'une Ville» (comme illustré dans la figure 4.10). Lors de la traduction de cette représentation logique vers une représentation physique (prenant en compte la future traduction MHEG), la division de la séquence audio-visuelle V en plusieurs places va causer une présentation de la séquence V avec plusieurs points d'interruption. Selon la performance du système de présentation utilisé, ces interruptions seront plus ou moins visi-

bles par le lecteur du document.

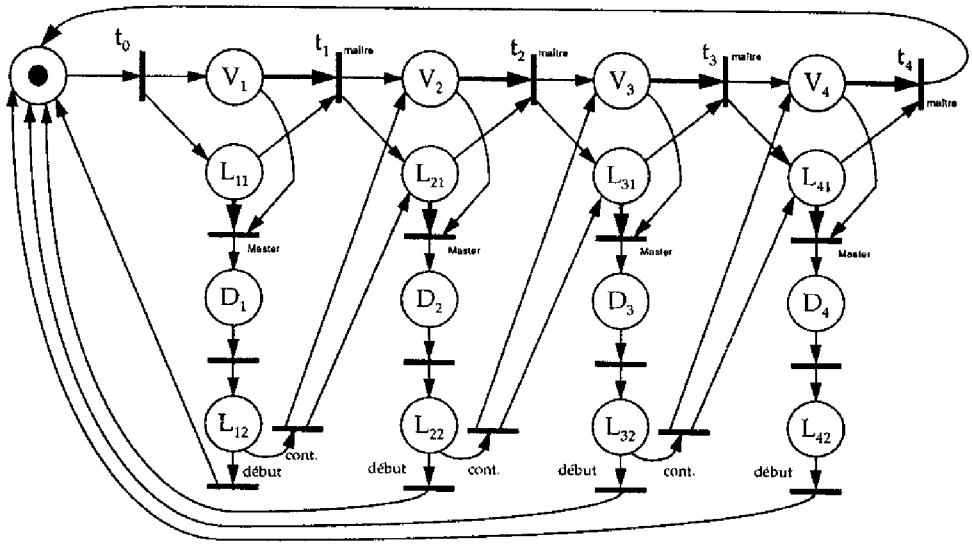


Figure 4.10 Application «visite d'une ville»[4]

Le modèle MORENA (section 2.4.7.7) permet la spécification de ce type de synchronisation à partir de la notion de message. Par exemple, la vidéo est modélisée par une place et la première trame de la présentation de chaque site est marquée. Quand ces trames sont présentées la place génère un message.

Pour simplifier l'étude, et pour ne pas changer le modèle de base RdPHFT, nous n'allons pas prendre en considération le problème présenté dans cette section.

4.5.4 Représentation des actions dynamiques

Il y a des applications hypermédias pour lesquelles les caractéristiques temporelles, spatiales et sonores d'une présentation peuvent être changées à n'importe quel instant, même pendant l'affichage de l'objet. Par exemple, un lecteur d'un document hypermédia peut changer la vitesse ou la position d'une présentation en cours. Une autre exemple d'action dynamique est la désactivation des présentations sonores d'une application multimédia en touchant un bouton. Le modèle RdPHFT-I ne permet pas la spécification de ce type d'action.

Le modèle MORENA (section 2.4.7.7) spécifie les actions dynamiques via la notion de message, le modèle MHPN (section 2.4.7.8) via l'association d'une liste d'actions aux transitions.

Une solution pour permettre la spécification des actions dynamiques est l'introduction des actions dans les transitions du modèle RdPHFT-I.

4.5.5 Méthodes d'interaction du modèle RdPHFT-I

L'unique modèle d'interaction des RdPHFT-I est celui fourni par les places du type lien. Ce type de place permet la représentation des interactions avec l'utilisateur. Dans cette approche, une place lien représente une région sensible (si la place ne contient aucun arc associé à un nom), un bouton (si cette place contient un seul arc nommé), un menu non hiérarchique (si la place lien contient plusieurs arcs nommés).

Les liens hypertextes (avec les ancres définies sur une partie du texte) ne sont pas représentables dans

4. Par simplification, les type des transitions qui ne sont pas spécifiés sont du type «et» et les contraintes temporelles ne sont pas représentées dans cette figure.

le modèle RdPHFT-I. Aucun mécanisme n'a été proposé pour la modélisation des liens hypertextes à cause de la limitation du MHEG à ce niveau. En effet, la version actuelle de la DIS MHEG [ISO 13522] ne fournit pas une représentation explicite des liens hypertextes.

Les autres mécanismes d'interaction qui peuvent exister dans un document hypermédia (par exemple, *entry-field, slider, scrolling-list*) ne sont pas modélisés dans RdPHFT-I.

4.5.6 Les liens contexte

Le modèle AHM [Hardman, 93] (présenté dans la section 2.4.2) spécifie trois types de lien contexte : *continue, pause* et *replace*. La figure 4.11 présente des exemples d'utilisation des liens contexte AHM représentés en RdPHFT, où :

- **Lien continue** : la demande d'aide n'interrompt pas la présentation C1. Comme illustré dans la figure 4.11a, le tir de la transition t_1 n'interrompt pas le traitement associé à la place C1.
- **Lien replace** : la demande d'aide remplace la présentation C1 par la présentation C2. Dans la figure 4.11b, le tir de la transition t_1 interrompt le traitement associé à la place C1. Lors du tir de la transition t_3, la présentation C1 est reprise au tout début.
- **Lien pause** : la demande d'aide remplace la présentation C1 par la présentation C2. Dans la figure 4.11c, le tir de t_1 interrompt le traitement associé à la place C1. Lors du tir de la transition t_3, la présentation C1 devrait être reprise à partir du point d'arrêt (point où l'aide a été demandée).

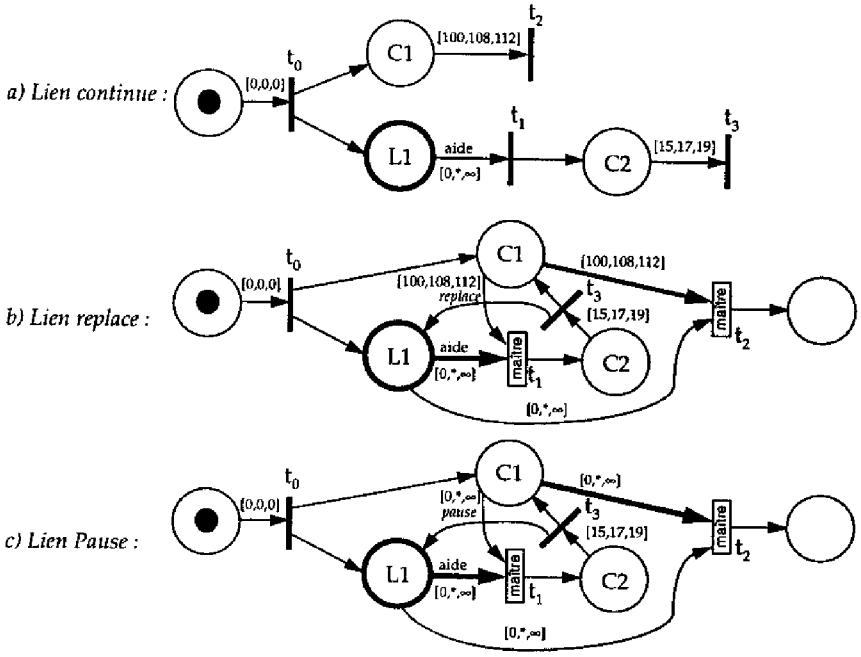


Figure 4.11 Représentation des contextes

Les liens *continue* et *replace* sont représentables par le modèle RdPHFT. Par contre, les liens *pause* ne sont pas représentables. Une solution possible est de nommer les arcs de sortie des places représentant des présentations par les mots-clés *replace* ou *pause*, comme présenté dans la figure 4.11. Dans le cas d'un contexte *pause*, une extension au modèle RdPHFT est nécessaire afin de rajouter la notion de mémoire temporelle.

4.5.7 Les synchronisations continues et intra-média

Le modèle RdPHFT permet la représentation de la synchronisation intra-média et continue (section

2.3.2.3). La représentation de ce type de synchronisation ne fait pas partie du domaine de la norme MHEG, mais d'autres normes internationales (comme le MPEG-System [Rangan, 96]). Ainsi, afin de permettre la traduction MHEG d'une spécification RdPHFT-I, nous considérons que le modèle RdPHFT-I ne permet pas la spécification des synchronisations intra-média et continues.

4.6 Exemple de spécification RdPHFT-I

Afin d'illustrer l'utilisation du modèle RdPHFT-I, cette section présente la spécification d'un module d'EAO (Enseignement Assisté par Ordinateur) VACBI[5] (*Video and Audio Computer Based Instruction*) [VACBI]. Ce système est utilisé par *Airbus Training* pour la formation de pilotes et d'agents de maintenance.

Le matériel d'enseignement VACBI est organisé en chapitres et modules. Dans cette étude, nous utilisons uniquement une version simplifiée du module de descente/montée normale du train d'atterrissage du chapitre Train d'Atterrissage. Ce module a été choisi à cause de sa représentativité en termes de taille et de complexité.

Le module étudié vise l'enseignement de la procédure de descente du train d'atterrissage de l'AIRBUS A340. Son interface est présentée dans la figure 4.12; elle contient :

- une représentation graphique du train d'atterrissage (pour animer graphiquement la procédure), présentée dans la fenêtre *Train d'atterrissage*.
- des instructions textuelles pour guider l'élève, présentées dans la fenêtre *Textes* et *Instructions*.
- des informations sonores, généralement pour reprendre des instructions textuelles.
- des informations vidéo, par exemple, la vidéo d'ouverture des portes du train d'atterrissage, affichées dans la fenêtre Images. Ces séquences vidéo sont présentées dans la fenêtre *Images et vidéos*;
- un ensemble de boutons permettant l'interaction avec l'élève;

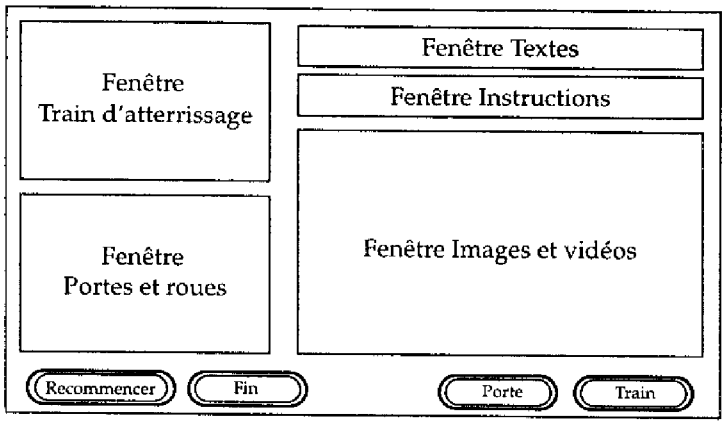


Figure 4.12 Interface avec l'utilisateur du prototype

La fenêtre principale du module de descente du train d'atterrissage (figure 4.12) est composée par :

- Fenêtre Textes, où sont affichés les textes :
  - TX1 : «Cours AIRBUS A340»;
  - TX2 : «ATA 24 : Train d'atterrissage»;
  - TX3 : «Ouverture des portes»;
  - TX4 : «Descente du train d'atterrissage»;
  - TX5 : «Fermeture des portes»;

5. Le système d'instruction VACBI® est une marque enregistrée de l'*Airbus Training Corporation*.

- Fenêtre *Instructions*, où sont affichés les textes :
  - IN1 : «Sélectionner le bouton Début»;
  - IN2 : «Sélectionner le bouton Porte»;
  - IN3 : «Sélectionner le bouton Train»;
  - IN4 : «Sélectionner le bouton Fin»;
- Fenêtre *Train d'atterrissage*, où se déroulent les présentations graphiques associées à la descente du train d'atterrissage, qui sont :
  - IMG_LG1 : représentation graphique des roues en blanc;
  - IMG_LG2 : représentation graphique des roues en rouge;
  - IMG_LG3 : représentation graphique des roues en vert.
- Fenêtre *Portes et roues*, où se déroulent les présentations graphiques associées à la descente du train d'atterrissage, qui sont :
  - IMG_W1 : représentation graphique des portes fermées;
  - IMG_W2 : représentation graphique des portes ouvertes;
  - IMG_W3 : représentation graphique de la descente du train;
  - IMG_W4 : représentation graphique du train descendu;
  - IMG_W5 : représentation graphique du train bloqué;
  - AG_W1 : animation graphique du mouvement d'ouverture des portes; elle est divisée en 3 parties (AG_W11, AG_W12 et AG_W13) nécessaires à la synchronisation fine avec la vidéo VD_OP;
  - AG_W2 : animation graphique du mouvement de fermeture des portes; elle est divisée en 3 parties (AG_W21, AG_W22 et AG_W23) nécessaires à la synchronisation fine avec la vidéo VD_FP;
- Fenêtre *Images*, où sont affichées les images :
  - IMG1 : l'image de l'avion AIRBUS A340;
  - IMG2 : l'image des portes fermées;
  - IMG3 : l'image des portes descendues;
  - IMG4 : l'image du train descendu;
  - IMG5 : l'image du train bloqué;
  - VD_OP : vidéo d'ouverture des portes; elle est divisée en trois parties (VD_OP1, VD_OP2 et VD_OP3) nécessaires à la synchronisation avec la vidéo AG_W1;
  - VD_DT : vidéo de descente du train d'atterrissage;
  - VD_FP : vidéo de fermeture des portes; elle est divisée en trois parties (VD_FP1, VD_FP2 et VD_FP3) nécessaires à la synchronisation avec la vidéo AG_W3.
- Cette application contient les boutons suivants :
  - Fin : pour quitter l'application;
  - Début : pour commencer l'étude du module;
  - Recommencer : pour recommencer la procédure de descente du train d'atterrissage;
  - Porte : pour manipuler les portes;
  - Train : pour manipuler le train d'atterrissage.
- Les séquences audio présentées pendant le déroulement du module sont les suivantes :
  - AU1 : «Module descente du train d'atterrissage. Pour commencer cliquez sur le bouton Début».
  - AU2 : «Dans ce module, vous allez réaliser des actions normales sur le train d'atterrissage. Pour commencer activez le bouton de portes pour visualiser leurs ouvertures;
  - AU3 : «Voici l'ouverture des portes»;
  - AU4; «Séquence d'ouverture des portes du train d'atterrissage de l'AIRBUS A340. Actionnez le bouton Train pour passer à la phase suivante»;
  - AU5 : «Voici la descente du train d'atterrissage»;
  - AU6 : «Maintenant, sélectionnez le bouton porte pour activer le blocage des roues;

- AU7 : «Dernière phase, les portes se ferment et verrouillent le train d'atterrissage»;
- AU8 : «Merci d'avoir suivi le module descente normale du train d'atterrissage. Actionner le bouton fin pour terminer cette leçon, ou réinitialiser avec le bouton recommencer».
- AU9 : «Mauvaise opération, veillez cliquer sur le bouton Porte»
- AU10 :«Mauvaise opération, veillez cliquer sur le bouton Train»

4.6.1 Spécification de la structure conceptuelle

La hiérarchie de la spécification RdPHFT du module de descente du train d'atterrissage est présentée dans la figure 4.13. Cette hiérarchie est divisée en pages. Chaque page correspond à un sous-réseau associé à une place composite. Chaque place composite est nommée par *<nom>#<page>*, où *<page>* est le numéro de la page qui spécifie le sous-réseau associé à la place composite *<nom>*. Dans ces réseaux, les arcs sortant des places qui n'ont pas d'IVT spécifié ont cet intervalle égal à [∞,.,∞].

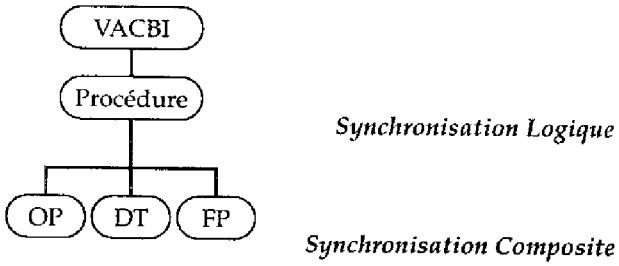


Figure 4.13 Hiérarchie de la spécification RdPHFT

Le niveau de synchronisation logique du module est composé de deux pages appelées *VACBI* et Procédure qui spécifient le chemin de parcours du module. Le niveau de synchronisation composite est composé de trois pages qui spécifient les relations temporelles entre les composants du module.

La page 1 (figure 4.14) spécifie le réseau racine de la spécification RdPHFT. Le comportement est le suivant : d'abord un ensemble d'images, de représentations graphiques, de textes et une information sonore sont présentés. En parallèle, deux boutons sont affichés. Alors, l'élève doit choisir entre deux options, qui sont : exécuter la procédure de descente du train d'atterrissage (lien *Début*) ou finir l'étude (lien *Fin*). Notez que *Début* est un lien temporisé avec un IVT de [0,*,20] qui spécifie que ce lien est automatiquement suivi si l'élève ne fait rien pendant 20 secondes. Si le lien *Début* est suivi, la procédure de descente du train d'atterrissage est présentée (spécifiée par la place composite *Procédure#2*) en parallèle avec les boutons *Recommencer* et *Fin*. De cette façon, pendant toute la procédure de descente du train d'atterrissage, l'élève peut recommencer ou quitter l'application.

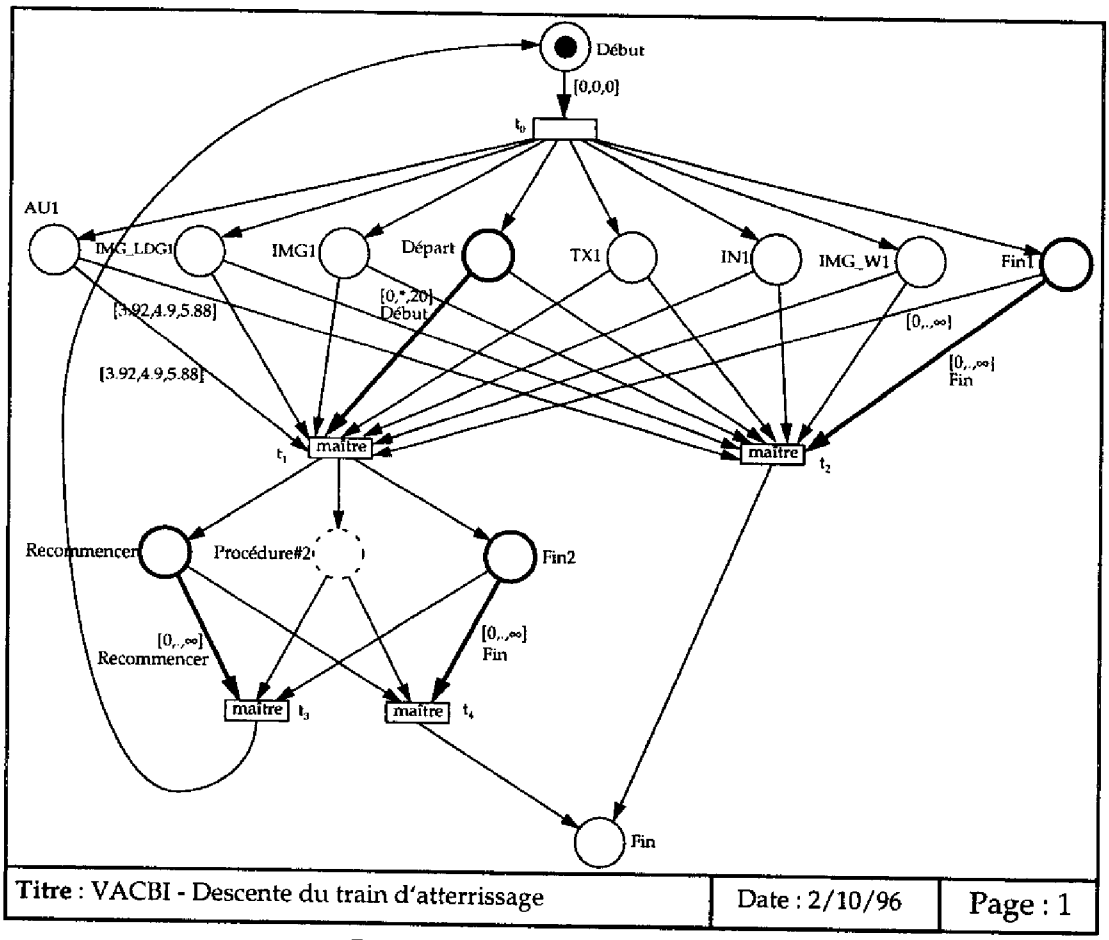


Figure 4.14 Racine de la spécification

La page 2 (figure 4.15) spécifie le sous-réseau associé à la place *Procédure#2* de la racine (page 1). Ce sous-réseau spécifie la procédure de descente du train d'atterrissage. Cette procédure est composée des phases suivantes : ouverture des portes (place *op#3*), descente du train d'atterrissage (place *dt#4*), fermeture des portes (place *fp#5*). Des informations audio (les places AU_i) et textuelles (les places IN_i, affichées dans la fenêtre *Instructions*, et les places TX_i, affichées dans la fenêtre *Textes*) sont présentées entre chaque phase de la procédure.

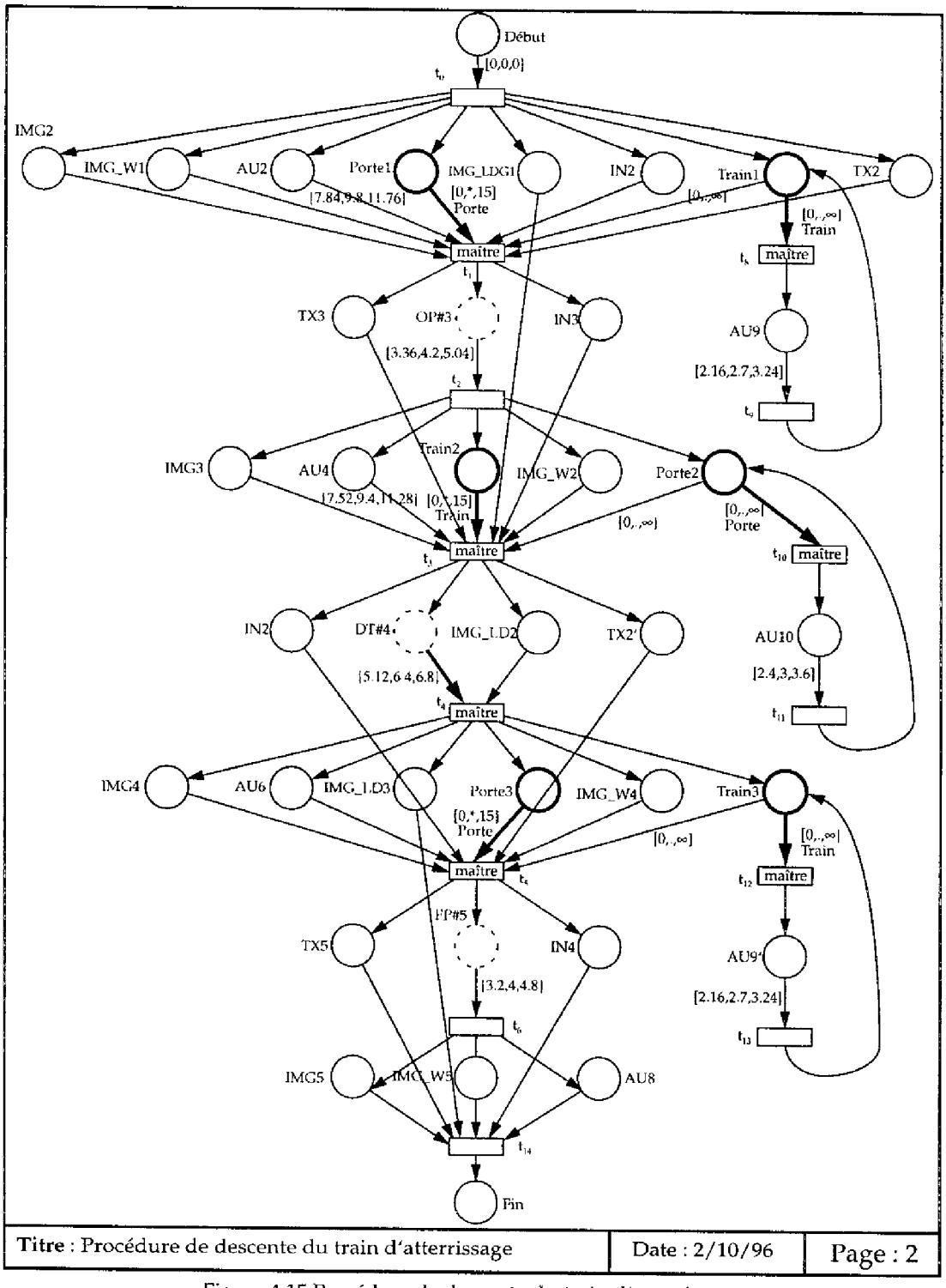


Figure 4.15 Procédure de descente du train d'atterrissage

La page 3 (figure 4.16) spécifie le sous-réseau de la place composite *op*#3 (ouverture des portes) de la page 2. A ce niveau, nous avons une synchronisation inter-média entre la vidéo «ouverture des portes» et son animation graphique affichée dans la fenêtre *Roues et portes* (il y a 3 points de synchronisation entre la vidéo et l'animation graphique).

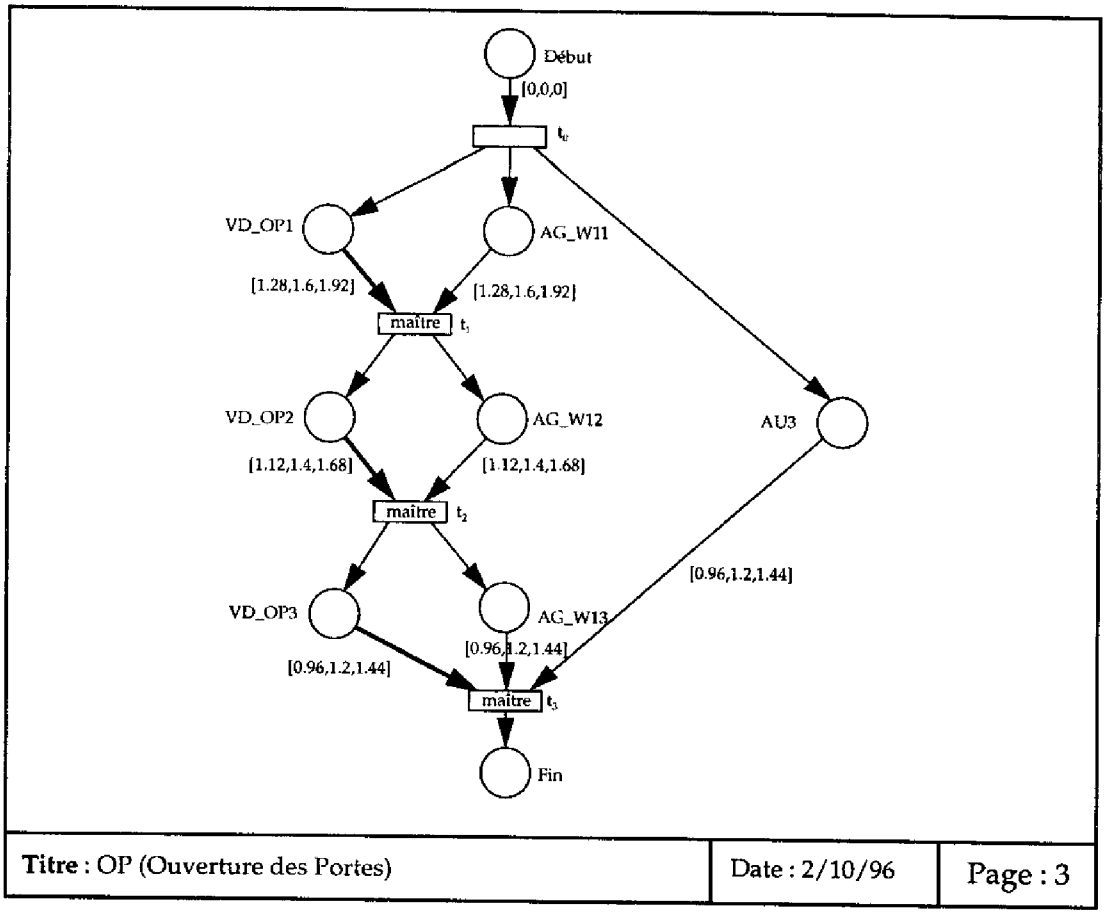


Figure 4.16 Ouverture des portes

Les pages 4 et 5 (figures 4.17 et 4.18) spécifient les procédures de descente du train d'atterrissage et de fermeture des portes.

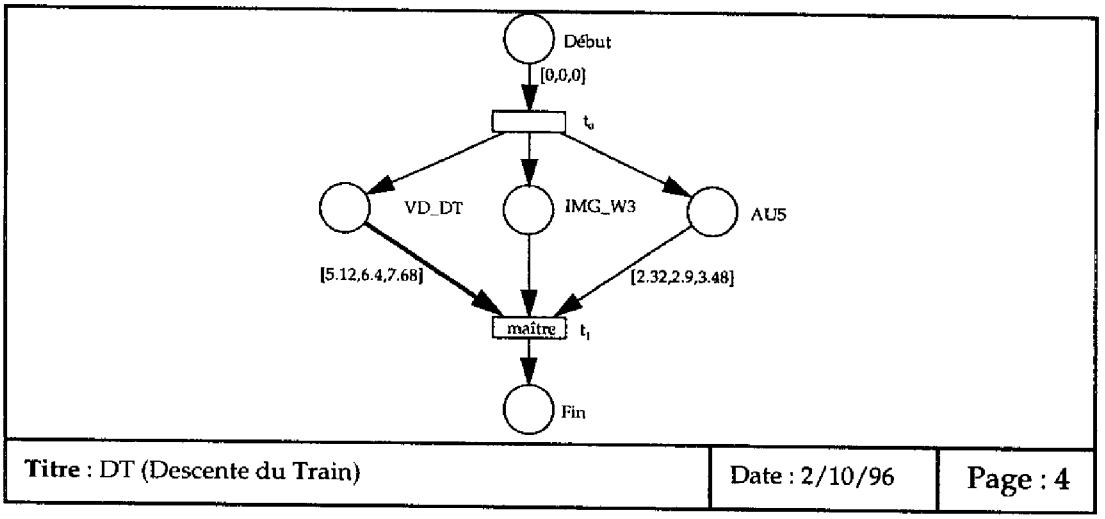


Figure 4.17 Descente du train

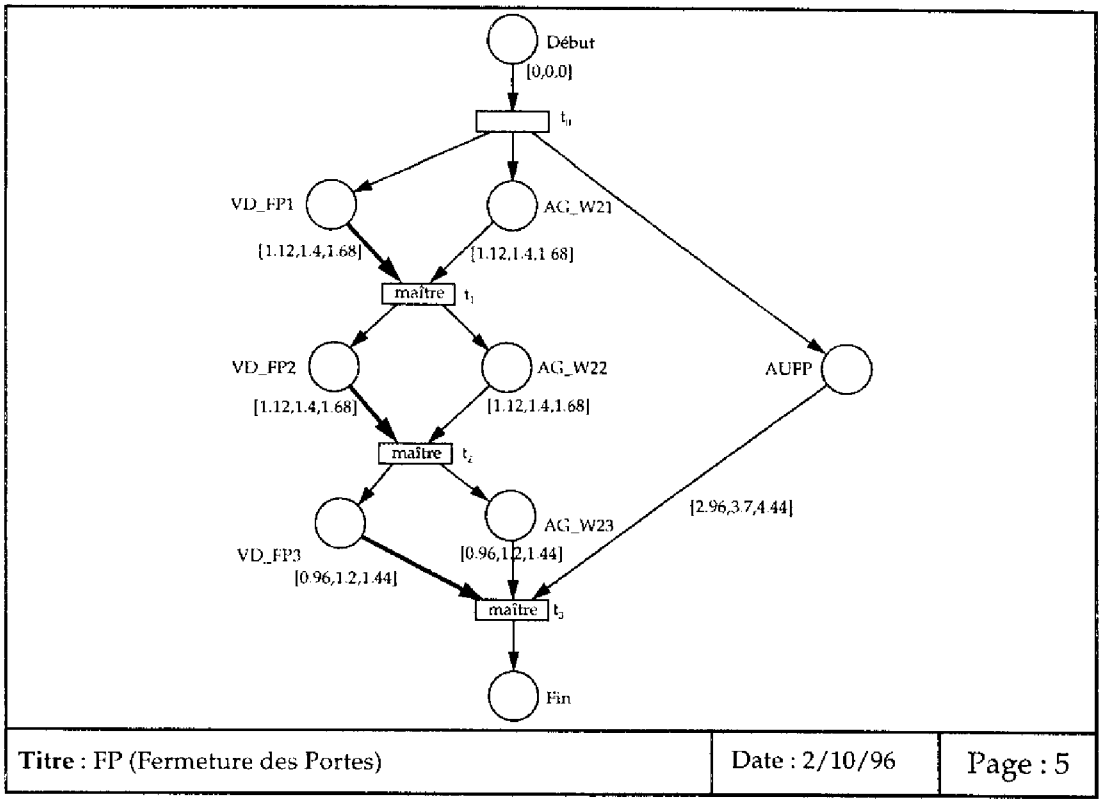


Figure 4.18 Fermeture des portes

4.6.2 Spécification de la structure du contenu

La spécification RdPHFT présentée dans la section 4.6.1 décrit le comportement logique et temporel du module de descente normale du train d'atterrissage. Afin de compléter la spécification de ce module, l'auteur doit utiliser la version interprétée du modèle RdPHFT pour décrire : des données multimédias utilisées par le module (modélisées par un ensemble d'objets des classes *Données*); la liste de canaux sur lesquels les données seront présentées (modélisés par un ensemble d'objets des classes *Canal*); les caractéristiques de ces présentations (modélisées par un ensemble d'objets des classes *Présentation*).

L'auteur doit spécifier le mode d'accès et de manipulation des données suivantes : une séquence vidéo de la procédure complète de descente du train d'atterrissage, dix fichiers audio (dont les présentations sont modélisées par les places AU_i), et neuf fichiers texte (dont les présentations sont modélisées par les places IN_i et TX_i), treize images (les places IMG_i, IMG_W_i, IMG_LG_i). Par exemple, l'objet *Vidéo-Train* ci-dessous (de la classe *Vidéo*) spécifie la séquence vidéo utilisée par ce module :

```
Vidéo Vidéo-Train : :=
(
    Contenu        «/home/willrich/Donnees/videoTrain.mpg»;
    Codage         ISO_11172_MPEG_Video;
    Caractéristiques-originelles-de-présentation :
        Taille-originelle      [320, 240, 0];
        Durée-originelle       23,8 s;
    Inclusion      implicite;
);
```

Cette application utilise uniquement le canal par défaut, et ainsi aucune spécification de canal n'est nécessaire. Lors de la présentation du module, la machine MHEG est responsable de l'affectation des

canaux réels, comme la fenêtre et le canal audio.

4.6.3 Spécification de la structure de présentation

En plus de la spécification des données, l'auteur doit spécifier les caractéristiques de chaque présentation du module. Ainsi, un objet *Présentation* doit être associé à chaque place des types atomique et lien de la spécification RdPHFT. Ces objets décrivent les données utilisées et leurs caractéristiques de présentation modélisées par les places du type atomique, et les caractéristiques de présentation des places du type lien. Par exemple, l'objet *OP1* ci-dessous (un objet de la classe *SP-Vidéo*) est associé à la place *VD_OP1* de la page 3 de la spécification RdPHFT pour spécifier les données et les caractéristiques de la présentation modélisée par cette place.

SP-Vidéo *OP1* : :=
(

Objet-données	*Vidéo-Train*;
Position-départ	*0*;
Position-fin	*8*;
Vitesse	5 (tps);
Terminaison	*gelé*;
Position-présentation	*[100,100,0]*;
Taille-présentation	*[320, 240,0]*;

);

Notons qu'une place représente une présentation qui peut partager des données avec d'autres places. De plus, une place peut représenter la présentation d'une certaine partie de ces données (à l'aide des attributs *Position-départ* et *Position-fin*).

4.7 Conclusion

Le modèle RdPHFT permet uniquement, au niveau de la spécification, la définition de la structure conceptuelle des documents hypermédias. Le modèle RdPHFT-I, introduit dans ce chapitre, étend la sémantique des places du réseau RdPHFT afin de permettre la spécification des structures conceptuelle, de présentation et du contenu, c'est-à-dire que ce modèle permet la spécification complète de documents hypermédias.

Le modèle RdPHFT-I permet la spécification multi-niveaux d'un document hypermédia. La structure du contenu est spécifiée par un ensemble d'objets des classes *Données*, et la structure de présentation est décrite par un ensemble d'objets des classes *Présentation* et *Canal*. Enfin, la structure conceptuelle est spécifiée par le modèle RdPHFT. L'interface entre ces structures est réalisée par des fonctions mettant en correspondance les places aux présentations qu'elles spécifient, et associant aussi ces présentations aux canaux et aux objets multimédias.

La description multi-niveaux du modèle RdPHFT-I fournit plusieurs avantages. Par exemple, elle permet la réutilisation de la structure conceptuelle lors de la présentation du document hypermédia en différents systèmes clients, et elle permet aussi la réutilisation des objets multimédias en différents contextes du document et par différents documents.

Dans le cadre de ce travail, nous avons adopté le télé-chargement comme mode de transmission des informations multimédias et hypermédias (section 2.2) : avant la présentation du document, la structure conceptuelle et de présentation (ou partie) doit être transférée vers le terminal de présentation et les objets multimédias peuvent être transférés lors de la présentation du document.

Finalement, le champ d'application de ce modèle est contraint aux documents dont la structure concep-

tuelle peut être spécifiée par le modèle RdPHFT. Ce choix découle d'un besoin qui nous parait nécessaire, celui de maîtriser la complexité des informations et donc des bases de données multimédias. Nous avons cependant constaté en réalisant la modélisation d'un module VACBI que cette restriction n'est pas limitative au regard de structures de documents hypermédias utilisées dans une telle application.

L'utilisation du modèle RdPHFT-I pour le développement de documents hypermédias permet l'obtention d'une spécification complète et non ambiguë du document. Il permet aussi l'application de méthodes de vérification, de simulation et d'analyse avant l'implémentation du document hypermédia. Finalement, le modèle RdPHFT-I fournit les moyens pour la génération automatique d'une représentation MHEG associée à la spécification RdPHFT-I analysée. La procédure de traduction d'une spécification RdPHFT-I en une représentation MHEG est présentée dans le chapitre suivant.

Chapitre 5

Traduction RdPHFT-I en MHEG

La distance entre la spécification des documents hypermédias et leur implémentation est un problème critique devant être résolu afin de maîtriser la qualité de ce type d'application.

Bien sûr, la définition de compilateurs permettant la génération automatique ou semi-automatique d'une représentation finale (un code interprétable par une machine) d'un document hypermédia à partir de sa spécification réduit grandement cette complexité. Afin de permettre le stockage, l'échange et la présentation d'un document hypermédia spécifiés par des RdPHFT-I, ce chapitre présente une procédure de traduction d'une spécification RdPHFT-I d'un document en une représentation MHEG. Cette représentation pourra donc être utilisée dans un système hypermédia ouvert. Pour la présentation de ces représentations MHEG, il est nécessaire d'utiliser des systèmes supportant des applications MHEG (les interpréteurs MHEG).

La définition d'une procédure de traduction d'une spécification RdPHFT-I en une représentation MHEG permettra la réalisation d'un environnement de création et d'analyse logique et temporelle d'objets MHEG, c'est-à-dire un *Système Auteur MHEG*. Dans le cadre de ce travail, nous avons réalisé un tel environnement de création de documents hypermédias MHEG, la présentation de cet environnement fait l'objet du chapitre 6. Cet environnement implémente la procédure de traduction d'une spécification RdPHFT-I en une représentation MHEG présentée dans ce chapitre.

La section 5.1 présente une comparaison entre le modèle RdPHFT et la norme MHEG; elle servira d'étude initiale pour l'obtention d'une procédure de traduction d'un réseau RdPHFT en une représentation MHEG. La section 5.2 présente la procédure de traduction d'une spécification RdPHFT-I en une représentation MHEG, c'est-à-dire l'obtention des correspondants MHEG des éléments d'une spécification RdPHFT-I. Ensuite, la section 5.3 présente un exemple de traduction MHEG d'une spécification RdPHFT-I. Finalement, la section 5.4 présente les conclusions de cette étude.

5.1 MHEG et le modèle RdPHFT

Cette section présente une comparaison entre le modèle RdPHFT et la norme MHEG au niveau de la représentation des éléments d'un document hypermédia et des relations logiques et temporelles entre ces éléments.

5.1.1 Correspondants MHEG des éléments du modèle RdPHFT

Les correspondances entre les composants d'un RdPHFT et les entités d'une application hypermédia sont présentées ci-dessous :

- Chaque place représente un processus de traitement multimédia. Dans ce travail, nous allons considérer le traitement multimédia comme le processus de préparation et de présentation des données multimédias ou bien le processus d'activation de scripts. La représentation du processus de préparation des données indépendant de leurs présentations ne sera pas considérée.
- Chaque arc temporisé représente, lorsqu'il est sensibilisé, un processus contrôlant les contraintes de temps relatives à une présentation.
- Chaque transition représente un processus contrôlant et forçant le respect des règles de synchronisation.

Dans la suite de cette section, nous présentons les concepts MHEG correspondant aux concepts de place, de transition et d'arc du modèle RdPHFT.

5.1.1.1 Les places

En MHEG, une présentation est représentée par un rt-objet (rt-SCRIPT, rt-CONTENT, rt-MULTIPLEXED CONTENT ou rt-COMPOSITE). Ainsi, une place RdPHFT correspond, en MHEG, à un rt-objet. L'arrivée d'un jeton dans une place correspond à la présentation du rt-objet correspondant à cette place (l'envoi d'une action *run* sur le rt-objet). La fin de traitement est détectée par l'attribut *Termination-status* d'un rt-SCRIPT ou par *Running-status* des rt-COMPONENT (rt-CONTENT, rt-MULTIPLEXED CONTENT et rt-COMPOSITE). Le tir forcé d'une transition, pour forcer le respect des règles de synchronisation, correspond à l'arrêt de la présentation du rt-objet (l'envoi d'une action *stop*).

Les places atomiques

Une place du type atomique correspond, en MHEG, à un rt-CONTENT, quand la place représente la présentation de données monomédias ou de données multiplexées[1], ou à un rt-SCRIPT quand la place représente une activation d'un conteneur de script.

Une place atomique ne correspond pas à un rt-MULTIPLEXED CONTENT, car celui-ci représente une liste de flux de données qui peuvent être manipulées indépendamment les uns des autres, et donc il n'est pas atomique. Cela ne veut pas dire qu'on ne peut pas représenter en RdPHFT des données multiplexées (ex. MPEG-System pour les présentations audiovisuelles), mais qu'on ne peut pas représenter les traitements indépendants des flux qui composent ces données.

Les places composites

Les places du type composite représentent, de façon hiérarchique, un sous-réseau contenant d'autres places atomiques et composites. Un exemple de structuration RdPHFT est présenté dans la figure 5.1. Dans ce réseau, la place composite C est une place abstraite représentant un réseau composé de quatre places et de deux transitions. Quand un jeton arrive dans la place C, la place V_1 (place d'entrée du sous-réseau) reçoit aussi un jeton. La sortie du jeton de la place composite fait disparaître tous les jetons du sous-réseau.

Un objet de la classe COMPOSITE définit une structure permettant l'association d'un ensemble d'objets MHEG. Ce mécanisme fournit une approche consistante pour la synchronisation dans l'espace et le

1. Un objet CONTENT peut représenter des données multiplexées (composées par une liste de flux de données) dans le cas où on n'a pas besoin de manipuler un flux particulier de ces données. Si cela est nécessaire, ces données doivent être représentées par un objet MULTIPLEXED CONTENT.

temps et pour la liaison entre un ensemble d'objets.

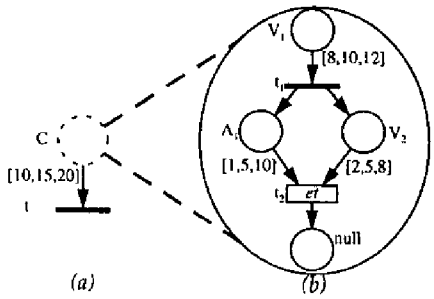


Figure 5.1 Hiérarchie en RdPHFT

Le correspondant MHEG d'une place composite est un rt-COMPOSITE. Ci-dessous, nous présentons les correspondances entre une place composite et un rt-COMPOSITE :

- L'attribut *Elements* de l'objet COMPOSITE contient les objets MODEL associés aux rt-objet qui représentent les places du sous-réseau.
- L'attribut *Specific Behaviour* de l'objet COMPOSITE spécifie le comportement du sous-réseau, en utilisant des objets LINK et ACTION (voir section 5.1.1.2).
- L'attribut *Rt-Availability-Start-Up* de l'objet COMPOSITE décrit l'action de départ de la présentation associée à la place d'entrée du sous-réseau.
- L'attribut *Running-status* du rt-COMPOSITE permet la détection de la fin de traitement d'une place composite.
- Le tir forcé d'une transition de sortie d'une place composite représente l'arrêt du rt-COMPOSITE (action *stop*).

Les places liens

Une place lien du RdPHFT permet la modélisation de l'interaction avec le lecteur d'un document hypermédia. Par exemple, les trois mécanismes d'interaction suivants peuvent être modélisés par une place du type lien :

- **Région sensible** : par exemple, le concepteur peut spécifier une région sensible sur une image. Cela est représenté par une place du type lien avec un arc maître n'ayant pas de label associé. Une place lien modélisant une région sensible peut avoir comme correspondant MHEG un *socket* attaché à un rt-objet NULL. Ce rt-objet est présenté[2] sur la région sensible.
- **Bouton** : il est représenté par une place lien avec une seule option de sélection. Une place lien modélisant un bouton a comme correspondant MHEG un *socket* contenant le titre du bouton.
- **Menu** : il est représenté par une place avec plusieurs options de sélection. Une place lien modélisant un menu a comme correspondant MHEG un rt-COMPOSITE, où chaque choix possible de la place lien correspond à un *socket* du rt-COMPOSITE.

Le choix de l'utilisateur est détecté par l'attribut *Selection-Status* des *sockets*.

5.1.1.2 Les transitions et les arcs

En MHEG, les objets LINK et ACTION permettent la représentation des synchronisations conditionnelles et temporelles d'un document hypermédia. Ainsi les arcs et les transitions du modèle RdPHFT sont représentables par ces objets MHEG.

2. La présentation d'un rt-objet NULL n'est pas visible.

Les objets LINK qui représentent les transitions et les arcs du modèle RdPHFT doivent avoir la structure suivante :

- Conditions : la fin de la présentation d'un ensemble de rt-objets associés aux places d'entrée d'une transition (cet ensemble est dépendant du type de synchronisation de la transition); la sélection d'un ensemble de *sockets* associés aux arcs de sortie des places du type lien; et le contrôle de la validité temporelle des présentations associées aux places d'entrée.
- Actions : dans le cas d'un tir forcé, l'arrêt des rt-objets associés aux places d'entrée; et le départ des rt-objets associés aux places de sortie.

5.1.1.3 Le contrôle de la validité temporelle

Dans le modèle RdPHFT, le processus contrôlant les contraintes de temps relatives à une présentation est représenté par un arc temporisé (figure 5.2). L'intervalle associé à cet arc [x, n, y] permet la spécification de l'intervalle de validité temporelle [x, y] de la présentation et de sa durée nominale n.

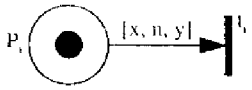


Figure 5.2 Contrôle temporel en RdPHFT

MHEG ne prévoit aucun mécanisme implicite de contrôle de la validité temporelle des présentations. C'est la responsabilité d'un concepteur de représenter explicitement tous ces mécanismes.

Afin de représenter le contrôle de la validité temporelle d'une présentation, il est nécessaire de définir un *horloge* associé à chaque présentation temporellement contrainte. Ces horloges sont responsables de l'enregistrement de l'âge des présentations[3], permettant l'identification de l'arrivée aux instants x et y (définis par l'intervalle de validité temporelle).

Un horloge peut être représenté en MHEG par un rt-objet associé à un objet NULL. Ce rt-objet horloge doit avoir les caractéristiques suivantes :

- Une vitesse égale à une unité de temps (en *Generic Time Unit*).
- Il doit faire référence à un objet NULL, parce que ce rt-objet ne réalise aucune présentation effective.
- Il doit contenir un ensemble de *timestones* qui définissent des marques temporelles pendant la durée de présentation de l'horloge. Ces marques correspondent aux instants x et y des intervalles de validité temporelle définie par tous les arcs sortants de la place. Ces marques seront définies seulement si les instants x et y sont différents de zéro et de la valeur «infini».

Le contrôle de la validité temporelle des présentations est représenté par un ensemble d'objets LINK et ACTION. Ces LINK ont comme condition l'arrivée à la durée maximale de traitement définie par l'intervalle de validité temporel et comme action l'arrêt de la présentation contrôlée et de l'horloge associé à cette présentation.

5.1.2 Représentation des synchronisations MHEG en RdPHFT

Cette section présente les possibilités de représentation en RdPHFT des types de synchronisation identifiées par MHEG (présentés dans la section 3.3), et s'attache ainsi à évaluer la complétude du pouvoir d'expression du modèle RdPHFT au regard des types de synchronisation identifiés par MHEG.

3. Dès l'arrivée d'un jeton dans une place, il reçoit l'âge zéro. Cet âge est incrementé d'une unité de temps à chaque tic d'horloge jusqu'à la sortie du jeton de la place.

5.1.2.1 Synchronisation élémentaire

Les modes Parallèle et Séquentiel sont facilement représentés en RdPHFT, comme on peut voir dans les figures 5.3 et 5.4.

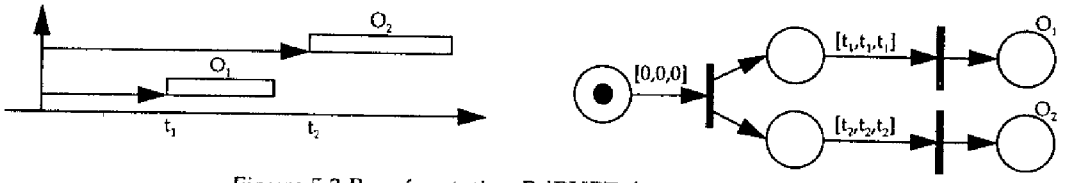


Figure 5.3 Représentation RdPHFT du mode Parallèle

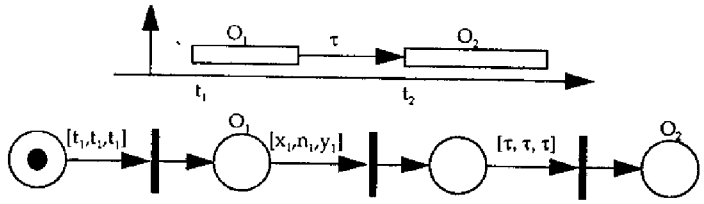


Figure 5.4 Représentation RdPHFT du mode Séquentiel

5.1.2.2 Synchronisation enchaînée

La synchronisation enchaînée est également représentée avec facilité en RdPHFT, comme on peut voir sur la figure 5.5.

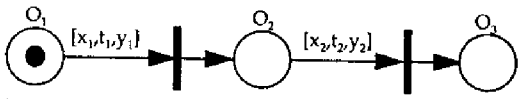


Figure 5.5 Représentation RdPHFT de la synchronisation enchaînée

5.1.2.3 Synchronisation cyclique

MHEG permet la représentation d'une présentation cyclique d'un ou plusieurs rt-objets. Ce type de synchronisation est représenté en RdPHFT par des cycles dans le réseau, possibles dans le cas des réseaux RdPFT (mais les cycles ne sont pas représentables par un réseau RdPSFT).

5.1.2.4 Synchronisation conditionnelle

La condition de tir d'une transition est associée à la fin du traitement d'un ensemble de présentations et/ou associée à la sélection d'un ensemble d'arcs représentant des liens. On peut représenter cela en MHEG par la condition «la valeur de *Termination-status* est égal à *terminated*» pour les rt-SCRIPT et «la valeur de *Running-status* est égal à *not-running*» pour les autres types de rt-objets. Pour les arcs maîtres des places du type lien, la condition «la valeur de *Selection-status* est égal à *selected*» est utilisée pour représenter l'action de sélection (faite par le lecteur du document). Les autres conditions des objets LINK (présentées dans la section 3.1.2) ne sont pas représentables en utilisant le modèle RdPHFT. Sauf si on associe de condition/actions aux transitions du modèle RdPHFT.

5.1.3 Viabilité de la traduction RdPHFT/MHEG

La section 5.1.1 a présenté une étude de l'équivalence entre la synchronisation définie par MHEG et par le modèle RdPHFT. Les principaux points étudiés ont été :

- La représentation MHEG des éléments d'un réseau RdPHFT (section 5.1.1), où :
  - une place du type atomique correspond à un rt-SCRIPT ou à un rt-CONTENT;

- une place du type lien correspond à un *socket* associé à un rt-objet NULL (si la place modélise une région sensible), à un *socket* associé au titre d'un bouton (si la place modélise un bouton), ou un *socket* COMPOSITE contenant la liste d'options du menu (si la place modélise un menu);
- une place du type composite correspond a un rt-COMPOSITE; et,
- les transitions et arcs peuvent être représentés par des objets LINK et ACTION.

- La représentation MHEG du contrôle de la validité temporelle (section 5.1.1.3), où nous avons constaté qu'il est nécessaire de définir un ensemble de rt-objets horloges et d'objets LINK et ACTION responsables du contrôle de la validité temporelle.

Nous avons vérifié que tous les eléments du modèle RdPHFT sont représentables en MHEG. Cela nous permet de conclure qu'une traduction MHEG d'une spécification RdPHFT est possible.

Dans cette étude, nous avons constaté que la norme MHEG ne définit aucun mécanisme implicite pour le contrôle de la validité temporelle des présentations multimédias. La représentation MHEG de ce type de contrôle doit être faite par le concepteur de la représentation MHEG. Une étude sur la proposition d'une nouvelle fonctionnalité MHEG, permettant la représentation directe du contrôle de la validité temporelle, pourrait être menée à partir de notre étude. Cette étude pouvant conclure par exemple à la définition d'un nouvel attribut MHEG, permettant la définition de l'intervalle de validité temporelle; la définition d'actions d'affectation et lecture de cet intervalle; et l'inclusion du mécanisme de contrôle de cette validité dans la définition du comportement temporel d'un rt-objet.

5.2 La procédure de traduction RdPHFT-I en MHEG

La procédure de traduction RdPHFT-MHEG a été définie à partir de l'étude menée dans la section 5.1. Dans cette étude, nous avons identifié les correspondances entre le modèle RdPHFT-I (un modèle graphique d'un document hypermédia) et le codage des informations multimédias et hypermédias défini par la norme MHEG (une représentation orientée-objet).

Dans le modèle RdPHFT, les places du type composite fournissent un mécanisme de structuration hiérarchique, basé sur la composition récursive de places atomiques et composites. Le choix adopté pour la traduction RdPHFT-MHEG est que la composition hiérarchique du modèle RdPHFT sera traduite en la structure fournie par la classe COMPOSITE. Ainsi, la hiérarchie RdPHFT peut être facilement traduite en un objet COMPOSITE, comme l'illustre la figure 5.6. Une spécification RdPHFT-I sera traduite en un objet COMPOSITE qui représente le réseau R (racine de la hiérarchie RdPHFT-I). Cet objet contiendra un ensemble d'objets MHEG fils représentant les places du réseau R. L'objet COMPOSITE R peut avoir d'autres objets COMPOSITE comme fils, chacun représentant une page de la spécification RdPHFT-I.

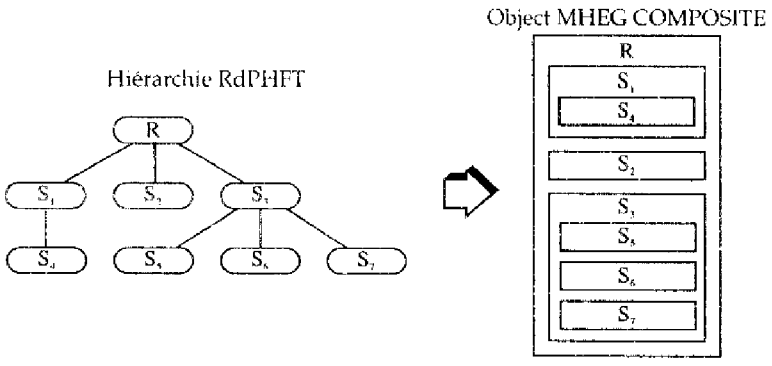


Figure 5.6 Traduction de la hiérarchie RdPHFT en un objet COMPOSITE

L'objet COMPOSITE R est obtenu à partir de la spécification des présentations (section 4.1.2) et de la structure conceptuelle (section 4.1.3) d'un document hypermédia. Cet objet ne contient pas la spécification des objets multimédias eux-mêmes (structure du contenu). Ces informations sont représentées par un ensemble disjoint d'objets MHEG. Ainsi, le modèle RdPHFT-I fait la distinction entre la structure du contenu et les autres structures d'un document hypermédia. Cette caractéristique très importante permet la réutilisation des objets multimédias dans différents contextes d'un document hypermédia.

La description des ressources nécessaires à la présentation d'un document hypermédia (section 4.4.5) sera traduite en un objet DESCRIPTOR. Ainsi, la traduction MHEG complète d'une spécification RdPHFT-I est composée de trois parties (figure 5.7) :

- La représentation des objets multimédias (la structure du contenu), composée d'un ensemble d'objets MHEG;
- La représentation de la structure conceptuelle et de présentation, composée d'un objet COMPOSITE (composé de différents objets MHEG).
- La description du document hypermédia, composée d'un objet DESCRIPTOR.

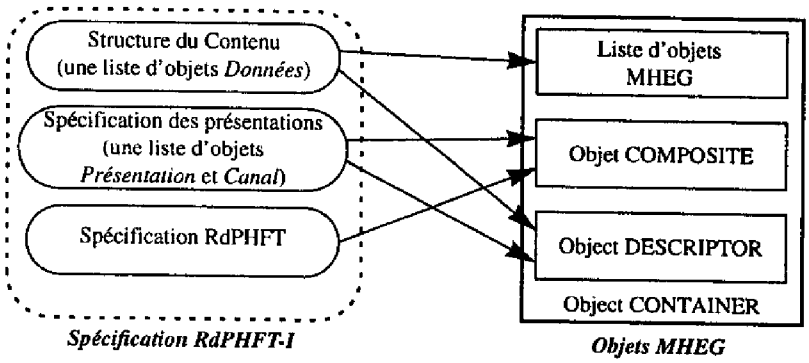


Figure 5.7 Traduction RdPHFT-MHEG

Tous les objets MHEG générés par la traduction RdPHFT-MHEG peuvent être regroupés dans un objet CONTAINER. Cet objet fournit un conteneur pour regrouper les différents objets MHEG qui représentent un document hypermédia. Ainsi, la structure complète du document peut être télé-chargée dans un terminal de présentation.

Les sections suivantes présentent les procédure d'obtention d'une représentation MHEG à partir d'une spécification RdPHFT-I (voir la définition 3 du chapitre 4).

5.2.1 Représentation MHEG de la structure du contenu

L'ensemble SD (d'objets *Données*) du modèle RdPHFT-I sera traduit en une liste d'objets MHEG CONTENT et SCRIPT. Chaque objet de la classe *Script* (une classe *Données*) est traduit en un objet MHEG SCRIPT. Les objets des autres classes *Données* sont traduits en objets MHEG CONTENT.

Un objet *Données* est facilement traduit en un objet MHEG par la correspondance un-à-un entre les attributs *Données* et les attributs MHEG des classes CONTENT et SCRIPT. Le tableau 5.1 présente les correspondances entre les attributs d'un objet *Script* et ceux d'un objet MHEG SCRIPT et le tableau 5.2 présente les correspondances entre les attributs des autres objets *Données* et d'un objet MHEG CONTENT.

Attributs de l'objet *Script*	Attributs de l'objet MHEG SCRIPT
Description	description
Codage-Propriétaire	script-hook-information.script-encoding-identification.proprietary-script-catalogue
Codage	script-hook-information.script-encoding-identification.mheg-script-catalogue
Contenu	Si l'attribut Inclusion = *implicite* alors les l'attribut Contenu définit la valeur de l'attribut script-data.data-reference.external-identifier.system-id Si l'attribut Inclusion = *explicite* alors les données référencées par Contenu correspondent à l'attribut script-data.script-inclusion.

Tableau 5.1 Correspondance entre l'objet *Script* et l'objet MHEG SCRIPT

Attributs des objets Texte, Graphique, Image, Audio, Vidéo, Audio-Visuel et Non-Média	Attributs de l'objet MHEG CONTENT
Description	description
Caractéristiques-Originales-de-Présentation	original-perception
Codage-Propriétaire	Content-Hook.content-encoding-identification.proprietary-content-catalogue
Codage	Content-Hook.content-encoding-identification.mheg-content-catalogue
Contenu	Si l'attribut Inclusion = *implicite* alors Content correspond à l'attribut content-data.data-reference.external-identifier.system-id Si l'attribut Inclusion = *explicite* alors les données référencées par Content correspondent à l'attribut content-data.data-inclusion.

Tableau 5.2 Correspondance entre les autres objet *Données* et l'objet MHEG CONTENT

Par exemple, une séquence vidéo *DonnéesVidéo11*, spécifiée par un objet de la classe *Vidéo* :

```
Video DonnéesVidéo11 : := {
    Contenu      «/home/willrich/Donnees/video11.mpg»;
    Hiérarchie   référence;
    Codage       ISO_11172_MPEG_Video;
    Caractéristiques-Originales-de-Présentation :
        Taille-Originale     128 (H) x 256 (W);
        Durée-Originale      10 s;
};
```

est traduit en l'objet CONTENT ci-dessous :

```
CONTENT object DonnéesVideo11 : := {
    object-identification {2 19 1 4},
    mheg-identifier {object-number 3},
    description {
        name «DonnéesVidéo11»,
        date «9508301213Z»},
    classification mheg-content-classification 6,
    content-hook {
        content-encoding-identification mheg-content-catalogue 80,
        content-encoding-description «»},
    content-data data-reference external-identifier system-id «/home/willrich/Donnees/video11.mpg»
};
```

À première vue, la complexité de la notation orientée-objet du modèle RdPHFT-I par rapport à la notation MHEG présentée ci-dessus, semble être identique. Cela n'est pas vrai, parce que la notation MHEG utilisée ci-dessus présente uniquement quelques attributs MHEG (et de façon simplifiée). De plus, les informations d'identification sont générées automatiquement par le traducteur MHEG. Afin de faciliter la tâche de l'auteur d'une spécification RdPHFT-I, une interface graphique aidera à la spécification des objets *Données*.

5.2.2 Représentation MHEG de la structure conceptuelle et de présentation

La deuxième partie de la traduction RdPHFT-MHEG est la traduction de la spécification des structures conceptuelle et de présentation d'un document hypermédia. Comme introduit précédemment, cette partie de description du document sera traduite en un objet COMPOSITE R. Cet objet représente la racine de la spécification et il contient un ensemble d'objets MHEG fils représentant les places du réseau R. L'objet COMPOSITE R peut avoir d'autres objets COMPOSITE comme fils, chacun représentant une page de la spécification RdPHFT-I (figure 5.6).

Dans cette section, nous présentons la procédure d'obtention de chacun des attributs de l'objet COMPOSITE R. Cette procédure sera présentée de façon récursive, et nous présentons l'obtention de chaque attribut de l'objet COMPOSITE Pg_i représentant une page $Pg_i = (P_i, T_i, \alpha_i, \beta_i, M_{i0}, IM_i, SYN_i, MA_i, CT_i, PT_i, LA_i)$ (voir les définitions 1, 2 et 3 du chapitre 4) générique du réseau RdPHFT (incluant R et les sous- réseaux de L et S).

5.2.2.1 Attribut *Description*

Si la page Pg_i est la racine, la structure DD (Description du Document, voir définition 3 du chapitre 4) définit la valeur des attributs *Description* du COMPOSITE Pg_i. La correspondance entre les attribut de DD et les attributs MHEG *Description* est présentée dans le tableau 5.3.

Attributs de DD	Attributs *Description* de l'objet COMPOSITE Pg_i
Non	description.name
Propriétaire	description.owner
version	description.version

Tableau 5.3 Correspondance entre DD et l'attribut MHEG *Description*

5.2.2.2 Attribut *Elements*

L'attribut *Elements* de l'objet COMPOSITE contient la liste des composants de l'objet COMPOSITE Pg_i, c'est-à-dire qu'il contient la liste d'éléments du réseau Pg_i. Cette liste est obtenue de la façon suivante :

- Le premier élément sera toujours l'objet CONTENT M. Il est responsable de l'enregistrement du marquage du réseau Pg_i (présenté dans la section 5.2.2.5).
- Toute place $p \in P_i \mid PT_i(p) = \textit{atomique}$, ayant une transition de sortie $t \in T_i \mid IM_i(p, t) \neq [0,0,0]$, et contenant une présentation $sp = F_{SP}(p)$ non vide, aura un élément associé dans la liste. Cet élément fera référence à l'objet MHEG qui représente l'objet multimédia $F_{SD}(sp)$ (voir section 5.2.1).
- Toute place $p \in P_i$ tel que $PT_i(p) = \textit{composite}$, ayant une transition de sortie $t \in T_i \mid IM_i(p, t) \neq [0,0,0]$, aura un élément associé dans la liste. Cet élément est un objet COMPOSITE. Les attributs de $F_{SP}(p)$ identifieront les valeurs des attributs de l'objet COMPOSITE. L'obtention des éléments de ce COMPOSITE est basée sur les éléments de la page FS(p). Cette procédure est présentée dans cette section (de façon récursive).
- Chaque place $p \in P_i$ tel que $PT_i(p) = \textit{lien}$, ayant $|\{LA_i(p,t) \mid (t \in T_i, \beta_i(p,t) \neq 0)\}| = 0$ (une région sensible)

aura un élément associé dans la liste. Cet élément fera référence à un objet NULL.

- Chaque place $p \in P_i$ tel que $PT_i(p) = \textit{lien}$, ayant $|\{LA_i(p,t) \mid (t \in T_i, \beta_i(p,t) \neq 0)\}| = 1$ (un bouton) aura un élément associé dans la liste. Cet élément contiendra le label $LA_i(p,t)$.
- Chaque place $p \in P_i$ tel que $PT_i(p) = \textit{lien}$, ayant $|\{LA_i(p,t) \mid (t \in T_i, \beta_i(p,t) \neq 0)\}| > 1$ (un menu) aura un élément associé dans la liste. Cet élément est un objet COMPOSITE. Les attributs de SP(p) seront traduits en les attributs de ce COMPOSITE, et les éléments de cette composition seront les *labels* $LA_i((p,t)) \mid \beta_i(p,t) \neq 0$.

Lors de la création d'un rt-COMPOSITE de l'objet Pg_i, chaque *socket* socket(p)[4] représente une place p de la page Pg_i.

5.2.2.3 Attribut *Availability-Start-Up*

L'attribut *Availability-Start-Up* contient le lien qui associe des actions de préparation aux composants de l'objet COMPOSITE. Pour la traduction MHEG d'un réseau RdPHFT-I, ce lien contient les actions suivantes[5] :

- Une action spécifiant l'enregistrement du marquage initial dans l'objet M (présenté dans la section 5.2.2.5). Tous les items index(p)[6] $\mid p \in Pin_i$ de cette liste auront la valeur «1», sinon la valeur «0» sera affectée.
- Une action de préparation des éléments du composite associés aux places $p \in Pin_i$.

5.2.2.4 Attribut *Rt-Availability-Start-Up*

L'attribut *Rt-Availability-Start-Up* contient le lien qui associe des actions de préparation des *sockets* de l'objet COMPOSITE. Pour la traduction MHEG d'un réseau RdPHFT-I, cet attribut représente la préparation des caractéristiques de présentation des places d'une page Pg_i. Cette représentation sera obtenue à partir des objets SP et à partir des places du type composite de la page Pg_i. La procédure d'obtention du lien *Rt-Availability-Start-Up* est présentée ci-dessous :

- Si la page Pg_i est la racine, ajouter une action de création de canal (*new-channel*) pour chaque canal défini dans l'ensemble CS, et ajouter une action *set-RGS* (*Relative Generic Space*) pour associer le rt-COMPOSITE R au canal $F_{CS}(R)$ (si ce canal existe).
- Si la page Pg_i est la racine et si l'attribut Taille-Présentation de DD est défini, ajouter une action de définition de la taille de présentation (action *set-perceptible-size-projection*) du document. La cible est le rt-COMPOSITE Pg_i.
- S'il y a dans la spécification RdPHFT des places du type lien, ajouter une action *set-selectability*, afin que le composite puisse être sélectionné.
- Ajouter une action de création d'un rt-objet NULL pour chaque place $p \in P_i$ ayant au moins un arc «a» sortant tel que $IM_i(a) \notin \{[0,0,0], [0,.,\infty], [\infty,.,\infty]\}$. Ces rt-objets agiront comme des horloges et seront utilisés lors du contrôle de la validité temporelle (introduits dans la section 5.1.1.3). Le numéro de rt-objet du rt-objet horloge associé à une place p sera noté par horloge(p).
- Ajouter un ensemble d'actions de création d'un ensemble de *timestones* sur chaque rt-objet horloge(p) tel que $p \in P_i$. Les marques temporelles qui doivent être définies sont la position 0 et toutes les durées minimales et maximales de traitement définies par l'IVT de tous les arcs «a» sortant d'une

4. La fonction socket(p) retourne l'index de la liste de composants du COMPOSITE qui contient l'objet MHEG associé à la place p.

5. Comme dans toutes les représentations MHEG de ce mémoire, une action aura lieu seulement si l'ensemble de ses cibles n'est pas vide.

6. La fonction index(p) retourne l'index d'une place p.

place p, notées par x(a) et y(a). Les instants x(a) et y(a) seront marqués seulement s'ils ont une valeur différente de 0 et ∞.

- Ajouter un ensemble d'actions de préparation des caractéristiques de présentation de tous les socket(p) | $p \in P_i$. Chaque attribut de $F_{SP}(p)$, ayant une valeur définie, aura une action MHEG correspondante. Cette correspondance est présentée dans le tableau 5.4.

Attributs des objets Présentation	traduction MHEG
Position-Départ	action set-visible-duration
Position-Fin	action set-visible-duration
Terminaison	action set-temporal-termination
Position-Présentation	action set-attachment-point-position
Taille-Présentation	action set-perceptible-size-projection
Vitesse	action set-speed
Volume	action set-audible-volume
Priorité	action set-presentation-priority
Orientation	action set-menu-style

Tableau 5.4 Correspondance entre $F_{SP}(p)$ et les actions MHEG

- Ajouter un ensemble d'actions *set-RGS* ayant comme cible les socket(p) tel que $F_{co}(p)$ est définie. Ces actions permettent l'affectation d'un canal à chaque socket(p).
- Ajouter un ensemble d'actions qui définissent les styles d'interaction du document : des actions *set-button-style* sur tous les socket(p) tel que $p \in P_i \mid PT_i(p) =$ *lien* et $||\{LA_i(p,t) \mid (t \in T_i, \beta_i(p,t) \neq 0)\}|| = 1$; et des actions *set-menu-style* sur tous les socket(p) tel que $p \in P_i \mid PT_i(p) =$ *lien* et $||\{LA_i(p,t) \mid (t \in T_i, \beta_i(p,t) \neq 0)\}|| > 1$.
- Ajouter une action de présentation de tous les socket(p) et des horloge(p) tel que $p \in Pin_i$.
- Ajouter une action de préparation de tous les liens $L\_ACTIVER_{it} \mid t \in T_i$, responsable de l'action de sensibilisation d'une transition (présenté dans la section 5.2.2.5).

5.2.2.5 Attribut *Specific-Behaviour*

Cet attribut spécifie le comportement de l'objet COMPOSITE Pg_i, qui doit être égal à celui spécifié par le réseau Pg_i. La traduction du comportement du réseau RdPHFT-I en MHEG est résolue par l'utilisation des objets LINK et ACTION. Ces objets peuvent être groupés dans 3 parties :

- **La représentation des transitions du réseau** : un ensemble de liens représentant toutes les transitions de Pg_i. Chaque transition $t_w \in T_i$ sera traduite en un lien L_{iw}.
- **L'exécuteur du réseau** : un ensemble de liens responsables de la sensibilisation et de la désensibilisation des transitions du réseau Pg_i. La sensibilisation (désensibilisation) d'une transition t_w correspond à la préparation (destruction) du lien L_{iw}.
- **Le contrôle de la validité temporelle** : un ensemble de liens responsables du contrôle de tous les mécanismes de contrôle de la validité temporelle des présentations $p \in P_i$.

Ensuite, nous présentons l'obtention de chacun des liens présentés ci-dessus. La syntaxe de représentation MHEG adoptée est la même que celle présentée dans la section 3.4 de ce mémoire.

Exécuteur du réseau

Chaque objet COMPOSITE Pg_i aura un objet CONTENT M. Cet objet, du type *generic-value* contenant une liste d'entiers, sera responsable de l'enregistrement du marquage du réseau Pg_i. L'arrivée d'un jeton dans une place $p \in P_i$ correspond à l'affectation de la valeur «1» à l'item index(p) de M. La sortie

d'un jeton d'une place $p \in P_i$ correspond à l'affectation de la valeur «0» à l'item index(p). L'enregistrement du marquage initial d'un réseau Pg_i sera spécifié par le *Availability-Start-Up* de l'objet COMPOSITE Pg_i (voir section 5.2.2.3).

Une transition t_w est sensibilisée quand toutes les places d'entrée de cette transition sont marquées. L'action de sensibilisation d'une transition t_w sera spécifiée par un objet LINK, appelé $L\_ACTIVER_{iw}$. Chaque transition t_w d'une page Pg_i aura un lien de sensibilisation $L\_ACTIVER_{iw}$. Ces liens seront préparés par *Rt-Availability-Start-Up* (section 5.2.2.4).

L'objet $L\_ACTIVER_{iw}$ prépare le lien L_{iw} (traduction d'une transition t_w) quand tous les items index(p) | $\beta(p,t_w)\neq 0$ de M ont la valeur «1». Ce lien est responsable aussi de la préparation du lien de désensibilisation de la transition t_w appelé $L\_DEACTIVER_{iw}$. L'objet $L\_DEACTIVER_{iw}$ existe seulement si la transition t_w peut être désensibilisé par le tir d'une autre transition (s'il y a un conflit dans le réseau). La structure de $L\_ACTIVER_{iw}$ est la suivante[7] (la syntaxe adoptée dans ce mémoire pour la représentation des objets MHEG a été présentée dans la section 3.4) :

```
LINK L_ACTIVER_iw {
    When (∀ p ∈ P_i | β_i(p,t_w)≠0, [Pg_i.?.1].[get-data (index = index(p))]==1)) then
        begin
            [L_iw, L_DEACTIVER_iw[8]].[prepare ()];
            [L_ACTIVER_iw].[destroy ()];
        end
}
```

L'objet LINK $L\_DEACTIVER_{iw}$ est responsable de la désensibilisation des transitions conflictuelles. Chaque transition conflictuelle t_w aura un LINK $L\_DEACTIVER_{iw}$. Ce lien est préparé quand la transition t_w est sensibilisée (exécution de $L\_ACTIVER_{iw}$). La structure de $L\_DEACTIVER_{iw}$ est la suivante :

```
LINK L_DEACTIVER_iw {
    When (∃ p ∈ P_i | β_i(p,t_w)≠0, [Pg_i.?.1].[get-data (index = index(p))]==0) then
        begin
            [L_iw].[destroy ()];
            S'il existe p |β_i(p,t_w)≠0 qui fait partie d'un cycle, ajouter :
                [L_ACTIVER_iw].[prepare ()];
            [L_DEACTIVER_iw].[destroy ()];
        end
}
```

Représentation du contrôle de la validité temporelle

Le contrôle de la validité temporelle d'une place $p \in P_i$ sera effectué par un objet LINK, nommé $L\_VALIDITE_{ip}$. Cet objet existe seulement s'il existe au moins un arc «a» sortant de $p \in P_i$ tel que $IM_i(a) \notin \{[0,0,0], [0,.,\infty], [\infty,.,\infty]\}$. L'objet $L\_VALIDITE_{ip}$ sera préparé lors du tir d'une transition t_w | $\alpha_i(p,t_w)\neq 0$.

Le LINK $L\_VALIDITE_{ip}$ a comme condition l'arrivée de l'âge du jeton dans la plus grande durée maximale spécifiée pas les intervalles de validité temporelle des arcs sortants de la place p, noté par $Y_{max}(p)$. Cette condition existe seulement si la durée maximale est différente de zéro. Les actions sont l'arrêt de la présentation associée à la place p et au horloge(p). Le LINK $L\_VALIDITE_{ip}$ contient la structure suivante :

---

7. Comme dans toutes les autres représentations MHEG de ce mémoire, si l'ensemble de cibles d'une condition est vide, la condition est satisfaite.

8. $L\_DEACTIVER_{iw}$ est une cible seulement si la transition t_w est conflictuelle.

```
LINK L_VALIDITE_ip {
    When ([horloge(p)].[get-timestone-status()] == Y_max(p)[9]) then
        begin
            [horloge(p), Pg_i.?.socket(p)].[stop ()];
            [L_VALIDITE_ip].[destroy ()];
        end
}
```

Représentation des transitions

Toute transition $t_w \in T_i$ est traduite en un objet LINK L_{iw}. Ce lien contient une structure dépendant du type de la transition t_w. De façon générale, l'objet L_{iw} contient les conditions et les actions suivantes :

- Conditions :
  - le contrôle de la validité temporelle des socket(p) tel que $p \in P_j$ et $\beta_i(p,t_w)\neq 0$;
  - la sélection de tous les éléments $LA(p,t_w)$ tel que $PT_i(p) =$ *lien* et $\beta_i(p,t_w)\neq 0$.
- Actions :
  - réarmer tous les socket(p) traduction des places $p \in P_i$ tel que $\beta_i(p,t_w)\neq 0$ et $PT_i(p) =$ *lien*;
  - pour toute place $p \in P_i$ tel que $\beta_i(p,t_w)\neq 0$ procéder l'arrêt des socket(p) (traduction de la place p) et horloge(p) (l'horloge utilisé lors du contrôle temporel), et détruire le lien $L\_VALIDITE_{ip}$ (effectue le contrôle temporel de la place p);
  - pour toute place $p \in P_i$ tel que $\alpha_i(p,t_w)\neq 0$ procéder le départ des socket(p) et horloge(p), et préparer le lien $L\_VALIDITE_{ip}$;
  - changer le marquage, c'est-à-dire la mise à «0» de tous les item index(p) $| \beta_i(p,t_w)\neq 0$ de M et la mise à «1» de tous les item index(p) $| \alpha_i(p,t_w)\neq 0$;
  - s'il existe p $| \beta_i(p,t_w)\neq 0$ faisant partie d'un cycle, alors préparer le lien $L\_ACTIVER_{iw}$;
  - la destruction des LINK L_{iw}.

Tous les objets LINK L_{iw} ont le même effet (c'est-à-dire les mêmes actions), mais avec des sources et des cibles différentes. Cet effet sera nommé de «tir» et sa structure est présentée ci-dessous.

```
LINK-EFFECT tir {
    [{Pg_i.?.socket(p) |β_i(p,t_w)≠0, PT_i(p)=lien}].[set-selection-status (not-selected)],
    [{horloge(p), Pg_i.?.socket(p) | β_i(p,t_w)≠0}].[stop (); set-temporal-position ()],
    [{L_VALIDITE_ip|β_i(p,t_w)≠0}].[destroy ()];
    [{horloge(p), Pg_i.?.socket(p) |α_i(p,t_w)≠0}].[run ()],
    [{L_VALIDITE_ip | α_i(p,t_w)≠0}].[prepare ()],
    [Pg_i.?.1].[set-data ({(index = index(p), value=0) | β_i(p,t_w) ≠ 0}, {(index = index(p), value = 1) |α_i(p,t_w)≠0})],
    S'il existe p |β_i(p,t_w)≠0 qui fait partie d'un cycle, ajouter :
        [L_ACTIVER_iw].[prepare ()];
    [L_iw].[destroy ()];
}
```

Les conditions associées aux LINK L_{iw} sont dépendantes du type de la transition t_w. Ensuite, nous présentons la représentation MHEG de tous les types de synchronisation du modèle RdPHFT.

SYN(t_w) = et

```
LINK L_iw {
    When ( ( (∀p|β_i(p,t_w)≠0, ([Pg_i.?.socket(p)].[get-running-status()] == not-running)) OR
             (∃p|β_i(p,t_w)≠0, ([horloge(p)].[get-timestone-status()] == y(p,t_w)))) AND
           (∀p|β_i(p,t_w)≠0, ([horloge(p)].[get-timestone-status()] >= x(p,t_w))))
```

9. Comme dans toutes les autres représentations MHEG de ce mémoire, la condition ([socket].timestone==0) sera toujours vraie et la condition ([socket].timestone==∞) sera toujours fausse.

```
        then LINK-EFFECT tir;
    }
```

$SYN(t_{iw}) = et\text{-}faible$

```
LINK L_iw {
    When ( ( (∀p|β_i(p,t_w)≠0, ([Pg_i.?.socket(p)].[get-running-status()] == not-running)) AND
             (∀p|β_i(p,t_w)≠0, ([horloge(p)].[get-timestone-status()] >= x(p,t_w)))) OR
           (∀p|β_i(p,t_w)≠0, ([horloge(p)].[get-timestone-status()] == y(p,t_w))))
        then LINK-EFFECT tir;
}
```

$SYN(t_{iw}) = ou$

```
LINK L_iw {
    When ( (∃p|β_i(p,t_w)≠0, ( ([Pg_i.?.socket(p)].[get-running-status()] == not-running) AND
                               ([horloge(p)].[get-timestone-status()] >= x(p,t_w)))) OR
           (∀ p| β_i(p,t_w)≠0, ([horloge(p)].[get-timestone-status()] == y(p,t_w))))
        then LINK-EFFECT tir;
}
```

$SYN(t_{iw}) = ou\text{-}fort$

```
LINK L_iw {
    When ( (∃ p|β_i(p,t_w)≠0, (([Pg_i.?.socket(p)].[get-running-status()] == not-running) AND
                               ([horloge(p)].[get-timestone-status()] >= x(p,t_w)))) OR
           (∃ p| β_i(p,t_w)≠0, ([horloge(p)].[get-timestone-status()] == y(p,t_w))))
        then LINK-EFFECT tir;
}
```

$SYN(t_{iw}) = maître$

```
LINK L_iw {
    When ( (p | MA_i = (p, t_w), (([Pg_i.?.socket(p)].[get-running-status()] == not-running) AND
                                  ([horloge(p)].[get-timestone-status()] >= x(p,t_w)))) OR
           (p = MP_i(t_w), ([horloge(p)].[get-timestone-status()] == y(p,t_w))))
        then LINK-EFFECT tir;
}
```

$SYN(t_{iw}) = ou\text{-}maître$

```
LINK L_iw {
    When ( (∃ p|β_i(p,t_w)≠0, (([Pg_i.?.socket(p)].[get-running-status()] == not-running) AND
                               ([horloge(p)].[get-timestone-status()] >= x(p,t_w)))) OR
           (p | MA_i = (p, t_w), ([horloge(p)].[get-timestone-status()] == y(p,t_w))))
        then LINK-EFFECT tir;
}
```

$SYN(t_{iw}) = et\text{-}maître$

```
LINK L_iw {
    When ( ( (∃ p|β_i(p,t_w)≠0, ([Pg_i.?.socket(p)].[get-running-status()] == not-running)) OR
             (p | MA_i = (p, t_w), ([horloge(p)].[get-timestone-status()] == y(p,t_w)))) AND
           (∀ p|β_i(p,t_w)≠0, ([horloge(p)].[get-timestone-status()] >= x(p,t_w))))
        then LINK-EFFECT tir;
}
```

$SYN(t_{iw}) = maître\text{-}fort$

```
LINK L_iw {
```

```
When ( ( (p ∣ MA_i = (p, t_w), ([Pg_i.?.socket(p)].[get-running-status()] == not-running)) OR
         (∃ p|β_i(p,t_w)≠0, ([horloge(p)].[get-timestone-status()] == y(p,t_w)))) AND
       (∀ p|β_i(p,t_w)≠0, ([horloge(p)].[get-timestone-status()] >= x(p,t_w))))
    then LINK-EFFECT tir;
}
```

SYN(t_w) = maître-faible

```
LINK L_iw {
  When ( (p ∣ MA_i = (p, t_w), ((([Pg_i.?.socket(p)].[get-running-status()] == not-running) AND
                                 ([horloge(p)].[get-timestone-status()] >= x(p,t_w)))) OR
         (∀ p|β_i(p,t_w)≠0, ([horloge(p)].[get-timestone-status()] == y(p,t_w))))
    then LINK-EFFECT tir;
}
```

Chacun des objets LINK L_{iw} ci-dessus contient comme condition une combinaison logique de conditions (spécifique à chaque type de transition), et comme action l'effet «tir» (présentée précédemment). La figure 5.8 présente des exemples de combinaison logique de conditions de tous les types de transitions du modèle RdPHFT.

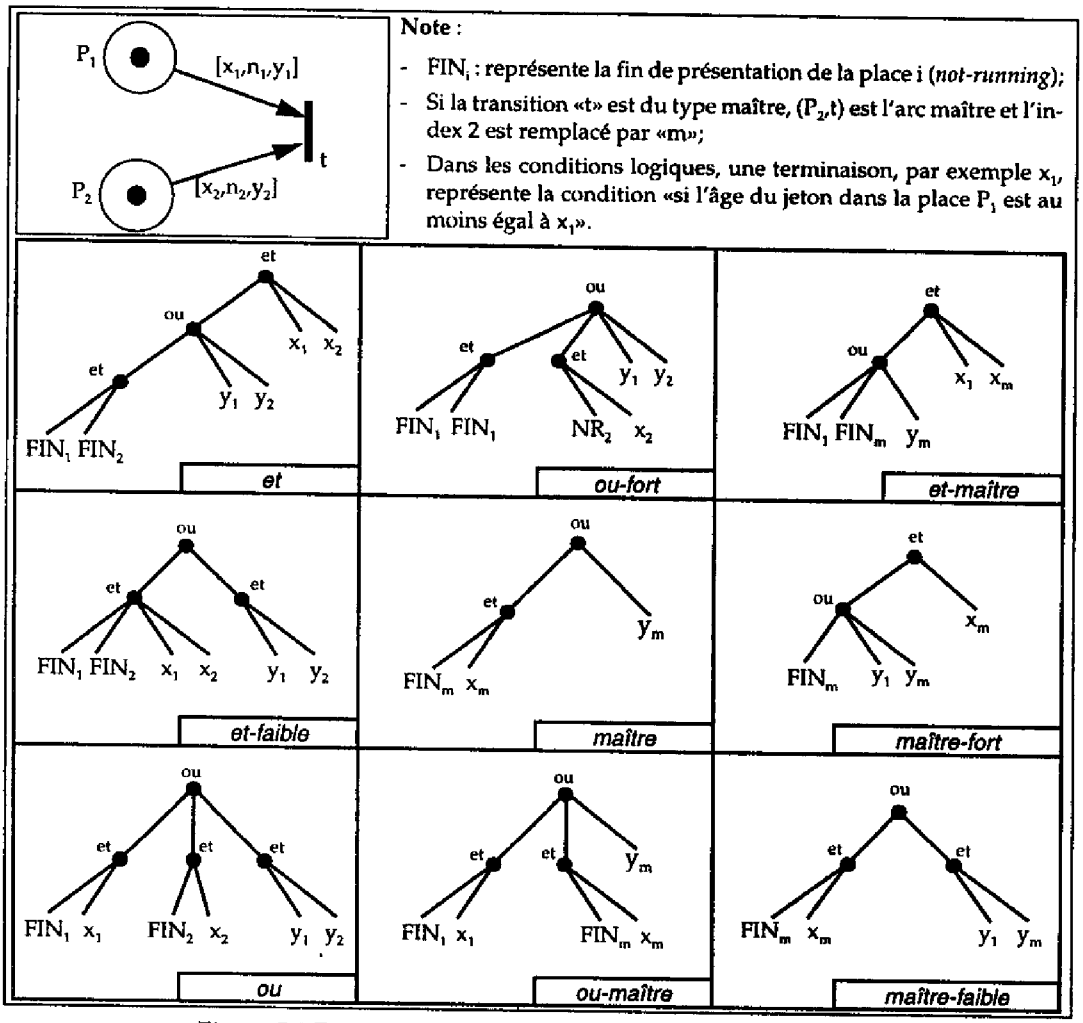


Figure 5.8 Exemples de condition logique des objets LINK L_{iw}

Pour la traduction des transitions contenant de places d'entrée du type lien ($PT_i(p)$ = *lien*) la condition ([Pg_i.?.socket(p)].[get-running-status()] == not-running) doit être remplacée par :

- ([Pg_i.?.socket(p)].selection-status == selected) si $|\{LA_i((p,t_w)) \mid \beta_i(p,t_w) \neq 0\}| = 1$.
- ([Pg_i.?.socket(p).option(p,t_w)[10]].selection-status == selected) si $|\{LA_i((p,t_w)) \mid \beta_i(p,t_w) \neq 0\}| > 1$.

5.2.3 Représentation MHEG de la description d'un document hypermédia

Comme nous l'avons dit dans la section 4.4.5, le modèle RdPHFT-I permet la génération automatique d'une description des ressources nécessaires à la présentation d'un document hypermédia. Cette description est obtenue à partir de la spécification des objets multimédias, des présentations, des méthodes d'interactions et des canaux utilisés par le document. Cette description sera traduite en un objet DESCRIPTOR. Les informations représentées par un tel objet DESCRIPTOR sont :

- Pour tous les canaux définis dans le document hypermédia :
  - les intervalles admissibles des axes x, y et z ([origine, point final]);
  - le rapport temporel (nombre d'intervalles consécutifs d'une unité temporelle dans une seconde);
  - le type de média à présenter (audible, audible à gauche, audible à droite et visible);
- Pour toute place du type atomique traduite en un objet CONTENT :
  - la taille de l'objet;
  - le type de média (audio, vidéo, etc.);
  - les informations nécessaires à la présentation des données primitives (méthode de décodage, etc.);
- Pour toute place du type atomique traduite en un objet SCRIPT :
  - la taille de l'objet;
  - le langage de scripting et d'autres informations de présentation;
- Pour toute place du type composite traduite en un objet COMPOSITE :
  - la taille de l'objet;
- Pour toute place du type lien :
  - le type d'interaction (menu ou bouton);

5.3 Exemple de traduction RdPHFT en MHEG

Cette section présente un exemple d'obtention d'une représentation MHEG à partir d'une spécification RdPHFT-I. L'exemple choisi est celui présenté dans la figure 5.9. Ce document contient le comportement suivant :

- Au début : la présentation d'une vidéo (p2) et d'un audio (p3) sont déclenchées.
- A la fin de la présentation vidéo, ou de l'audio, ou encore à l'arrivée à la limite supérieure de l'intervalle de validité temporel d'une de ces présentations, la vidéo et l'audio sont arrêtées et la présentation d'une musique (p4) et d'un menu (p5) sont déclenchées. Le menu (p5) propose les choix suivants : «fin» pour arrêter la présentation; et «relancer» pour relancer l'application. Dans cette situation :
  - si pendant 30 sec. l'utilisateur ne touche rien, la présentation de la musique et du menu sont arrêtées et l'application reprend à son début;
  - si l'utilisateur touche «relancer», la présentation de la musique et du menu sont arrêtées et l'application reprend à son début;
  - si l'utilisateur touche «fin», l'application se termine.

Les objets *Données* sont traduits par les objets MHEG suivants (selon la procédure présentée dans la section 5.2.1) :

10. La fonction option(p,t) retourne le *socket* correspondant au titre de l'option définie par l'arc (p,t).

```
CONTENT «vidéo1» {
    Classification «video»,
    Original-Perception {
        original-duration «17»,
        original-size «[582,402,0]»}
    Content-Hook «video-ISO-11172-MPEG-Video»
    Content-Data {
        Data-Reference «video_ex.mpg»}}

CONTENT «audio1» {
    Classification «audio»,
    Original-Perception {
        original-duration «17»}
    Content-Hook «audio-ISO-11172-MPEG-Audio»
    Content-Data {
        Data-Reference «audio_ex.mpg»}}

CONTENT «musique» {
    Classification «audio»,
    Original-Perception {
        original-duration «33»,
        original-size «[582,402,0]}
    Content-Hook «audio-ISO-11172-MPEG-Audio»
    Content-Data {
        Data-Reference «musique_ex.mpg»}}
```

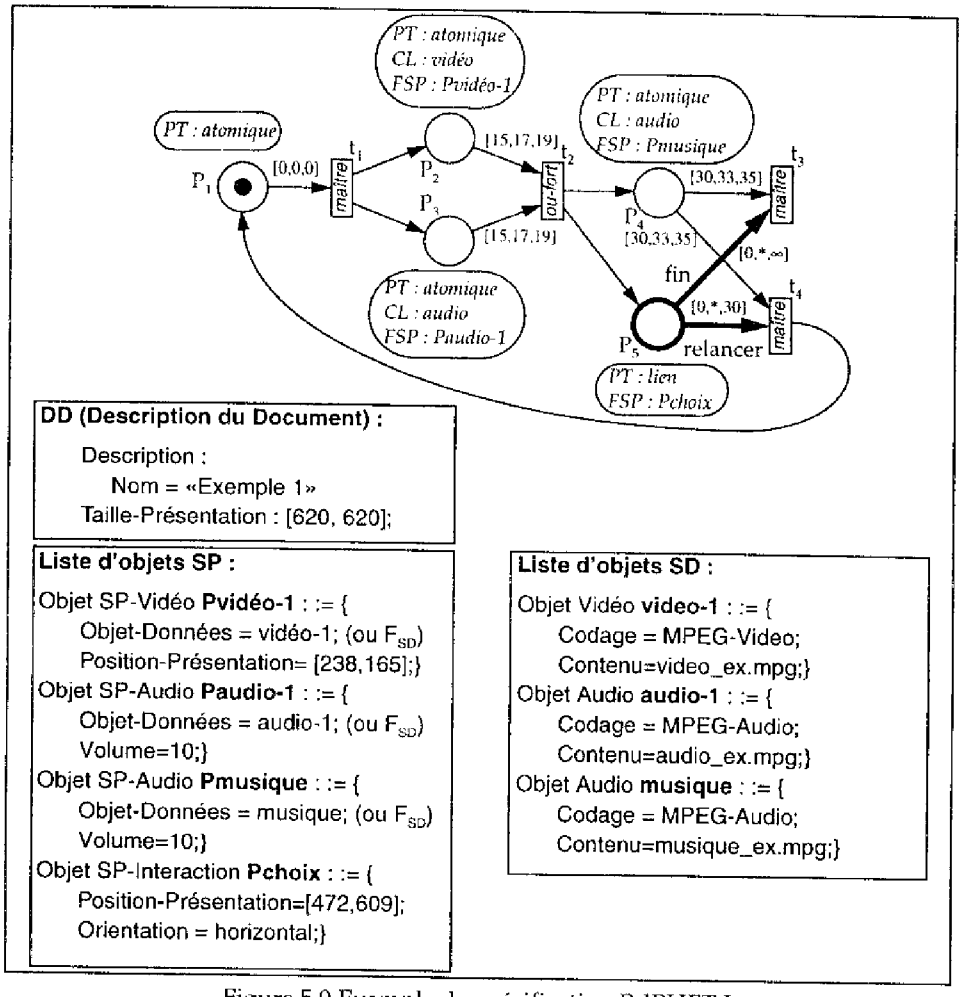


Figure 5.9 Exemple de spécification RdPHFT-I

La représentation MHEG des structures conceptuelle et de présentation spécifiées dans la figure 5.9 est présentée ci-dessous (selon la procédure de la section 5.2.2 et la syntaxe de représentation d'objets MHEG définie dans la section 3.4[11]) :

```
COMPOSITE 6 «Example 1» {
  Composition-Behaviour {
    Predefined-Behaviour {
      Availability-Start-Up {
        [M].[set-data (substitution-indicator = substitution, data-elements = [1,0,0,0,0])]}
      Rt-Availability-Start-Up {
        [6.?].[set-selectability(min=0, max=1)];
        [6.?].[set-perceptible-size-projection(x=620,y=620)];
        [0.2, 0.3, 0.4, 0.5].[new ()];
        [0.2].[set-timestone ({(tmstn-id=0, tmstn-pos=0), (tmstn-id=15, tmstn-pos=15), (tmstn-id=19, tmstn-
          pos= 19)})],
        [0.3].[set-timestone ({(tmstn-id=0, tmstn-pos= 0), (tmstn-id=15, tmstn-pos=15), (tmstn-id=19, tmstn-
          pos= 19)})],
        [0.4].[set-timestone ({(tmstn-id=0, tmstn-pos= 0), (tmstn-id=30, tmstn-pos=30), (tmstn-id=35, tmstn-
          pos= 35)})],
        [0.5].[set-timestone ({(tmstn-id=0, tmstn-pos= 0), (tmstn-id= 30, tmstn-pos= 30)})],
```

11. Cet objet noté selon la représentation MHEG contient 2034 lignes.

```
        [6.?.2].[set-attachment-point-position (position = [238,165])],
        [6.?.3].[set-audible-volume (volume = 10)],
        [6.?.4].[set-audible-volume (volume = 10)],
        [6.?.5].[set-attachment-point-position (position = [472,609])],
        [6.?.5].[set-menu-style (orientation = horizontal)];
        [6.?.5.2, 6.?.5.1].[set-selectability(min=0, max=1)]
        [L_ACTIVER01, L_ACTIVER02, L_ACTIVER03, L_ACTIVER04].[prepare ()]}}
Specific-Behaviour {
    -- Représentation de la sensibilisation des transitions
    LINK «L_ACTIVER01» {
        When ([M].get-data(index=1)==1)) then
        begin
            [L01].[prepare ()];
            [L_ACTIVER01].[destroy ()];
        end }
    LINK «L_ACTIVER02» {
        When ([M].get-data(index=2)==1) AND [M].get-data(index=3)==1)) then
        begin
            [L02].[prepare ()];
            [L_ACTIVER02].[destroy ()];
        end }
    LINK «L_ACTIVER03» {
        When ([M].get-data(index=4)==1) AND [M].get-data(index=5)==1)) then
        begin
            [L03, L_DEACTIVER03].[prepare ()];
            [L_ACTIVER03].[destroy ()];
        end }
    LINK «L_ACTIVER04» {
        When ([M].get-data(index=4)==1) AND [M].get-data(index=5)==1)) then
        begin
            [L04, L_DEACTIVER04].[prepare ()];
            [L_ACTIVER04].[destroy ()];
        end }
    LINK «L_DEACTIVER03» {
        When ([M].get-data(index=4)==0) OR [M].get-data(index=5)==0)) then
        begin
            [L03].[destroy ()];
            [L_ACTIVER03].[prepare ()];
            [L_DEACTIVER03].[destroy ()];
        end }
    LINK «L_DEACTIVER04» {
        When ([M].get-data(index=4)==0) OR [M].get-data(index=5)==0)) then
        begin
            [L04].[destroy ()];
            [L_ACTIVER04].[prepare ()];
            [L_DEACTIVER05].[destroy ()];
        end }
    -- Représentation du contrôle de la validité temporelle
    LINK «L_VALIDITE02» {
        When ([0.2].[get-timestone-status()] == 19)) then
        begin
            [0.2, 6.?.2].[stop ()];
            [L_VALIDITE02].[destroy ()]
        end }
```

```
LINK «L_VALIDITE03» {
    When ([0.3].[get-timestone-status()] == 19)) then
    begin
        [0.3, 6.?.3].[stop ()];
        [L_VALIDITE03].[destroy ()]
    end }
LINK «L_VALIDITE04» {
    When ([0.4].[get-timestone-status()] == 35)) then
    begin
        [0.4, 6.?.4].[stop ()];
        [L_VALIDITE04].[destroy ()]
    end }
LINK «L_VALIDITE05» {
    When ([0.5].[get-timestone-status()] == 30)) then
    begin
        [0.5, 6.?.5].[stop ()];
        [L_VALIDITE05].[destroy ()]
    end }
-- REPRESENTATION T1 (TYPE ET [0,0,0])
LINK «L01» {
    When (TRUE) then
    begin
        [0.2, 0.3, 6.?.2, 6.?.3].[run ()],
        [L_VALIDITE02, L_VALIDITE03].[prepare ()],
        [M].[set-data ((index=1, value=0), (index=2, value=1), (index=3, value=1))],
        [L_ACTIVER01].[prepare ()];
        [L01].[destroy ()];
    end }
-- REPRESENTATION T2 (TYPE OU-FAIBLE)
LINK «L02» {
    When ( ( ([6.?.2].[get-running-status()] == not-running) AND
               ([0.2].[get-timestone-status()] >= 15)) OR
             ( ([6.?.3].[get-running-status()] == not-running) AND
               ([0.3].[get-timestone-status()]>= 15)) OR
             ([0.2].[get-timestone-status()] == 19) OR
             ([0.3].[get-timestone-status()] == 19)) then
    begin
        [0.2, 0.3, 6.?.2, 6.?.3].[stop (); set-temporal-position()],
        [L_VALIDITE02, L_VALIDITE03].[destroy ()];
        [0.4, 0.5, 6.?.4, 6.?.5].[run ()],
        [L_VALIDITE04, L_VALIDITE05].[prepare ()],
        [M].[set-data ((index=2, value=0), (index=3, value=0), (index=4, value=1), (index=5, value=1))],
        [L_ACTIVER02].[prepare ()];
        [L02].[destroy ()]
    end }
-- REPRESENTATION T3 (TYPE MAITRE - [0,*,∞])
LINK «L03» {
    When ([6.?.5.1].selection-status == selected) then
    begin
        [6.?.5].[set-selection-status(not-selected)],
        [0.4, 0.5, 6.?.4, 6.?.5].[stop (); set-temporal-position()],
        [L_VALIDITE04, L_VALIDITE05].[destroy ()];
        [M].[set-data ((index=4, value=0), (index=5, value=0))];
        [L03].[destroy ()]
```

```
        end }
    -- REPRESENTATION T4 (TYPE MAITRE)
    LINK «L04» {
        When ( ([6.?.5.2].selection-status == selected) OR
                ([0.5].[get-timestone-status()] == 30)) then
        begin
            [6.?.5].[set-selection-status(not-selected)],
            [0.4, 0.5, 6.?.4, 6.?.5].[stop (); set-temporal-position()],
            [L_VALIDITE04, L_VALIDITE05].[destroy ()];
            [M].[set-data ((index=4, value=0), (index=5, value=0), (index=1, value=1))],
            [L_ACTIVER04].[prepare ()];
            [L04].[destroy ()]
        end }
    }
Elements {
    1. CONTENT «M» {
        Content-Hook «generic-value»,
        Content-Data {
            Data-Inclusion «»}
    2. ASSOCIATED-MODEL «video1» {
    3. ASSOCIATED-MODEL «audio1» {
    4. ASSOCIATED-MODEL «musique» {
    5. COMPOSITE «choix» {
        Elements {
            1. ASSOCIATED-LABEL «fin»;
            2. ASSOCIATED-LABEL «relancer»;}}
```

5.4 Conclusion

Ce chapitre a présenté une procédure de traduction d'une spécification RdPHFT-I en une représentation MHEG équivalente. Cette représentation peut être vue comme un modèle physique du modèle logique RdPHFT-I. Cette représentation physique du document peut être ainsi stockée, échangée et présentée dans un système hypermédia ouvert.

La représentation MHEG obtenue à partir d'une spécification RdPHFT-I ne sera pas optimale, mais cette méthodologie se justifie à cause de la complexité de la représentation MHEG lorsque l'on définit et programme les objets MHEG directement.

Le chapitre 4 a présenté le modèle RdPHFT-I permettant la spécification et l'analyse de documents hypermédias, et ce chapitre a présenté une procédure de traduction d'une spécification RdPHFT-I en une représentation MHEG. Ces deux chapitres ont défini une base formelle solide pour la spécification et l'analyse de documents hypermédias, et pour la traduction de cette spécification en une représentation MHEG. Le chapitre 6 met en évidence la méthodologie de développement de documents hypermédias portables proposée dans ce mémoire et présente un environnement mettant en œuvre cette méthodologie. Un prototype de cet environnement a été implémenté dans le cadre de ce travail.

Dans le contexte de conception formelle de documents hypermédias portables, le chapitre 6 présente aussi une variante de la méthodologie précédente permettant la création de documents hypermédias interprétables par une application Java. Nous avons développé une application Java RdPHFT-I permettant la spécification RdPHFT-I d'un document hypermédia et l'interprétation (présentation) du document.

Chapitre 6

Vers la conception de documents hypermédias : méthodologie et environnement

Le modèle RdPHFT-I fournit une base formelle solide pour la spécification et l'analyse de documents hypermédias et la traduction de ces spécifications en des représentations MHEG. Ce chapitre met en évidence cette méthodologie de prototypage de documents hypermédias, dans laquelle la spécification et l'analyse des documents sont faites à l'aide du modèle RdPHFT-I et l'implantation est supportée par la production automatique de représentations MHEG. Ce chapitre présente aussi un environnement mettant en œuvre cette méthodologie qui a été développé dans le cadre de ce travail.

En addition à la méthodologie de développement de documents hypermédias MHEG, ce chapitre présente une variante permettant la création et la présentation de documents hypermédias à l'aide d'une application Java [Sun,95]. Dans le cadre de ce travail, nous avons développé cette application Java d'interprétation de spécifications RdPHFT-I.

Ce chapitre est organisé de la façon suivante : la section 6.1 présente la méthodologie de développement de document hypermédias MHEG et un environnement mettant en œuvre cette méthodologie; ensuite la section 6.2 présente un environnement de création et de présentation de documents hypermédias implémenté en Java; enfin, les conclusions de ce chapitre sont présentées dans la section 6.3.

6.1 Vers la conception de documents hypermédias MHEG

Le chapitre 4 a proposé le modèle RdPHFT-I permettant la spécification et l'analyse de documents hypermédias. Le chapitre 5 présente une procédure de traduction d'une spécification RdPHFT-I en une représentation MHEG. Ce modèle et cette procédure de traduction MHEG sont les bases d'une méthodologie formelle de développement de documents hypermédias MHEG (figure 6.1). Cette méthodologie comporte les étapes suivantes :

A) Utilisation du modèle RdPHFT-I pour la spécification formelle du document hypermédia. Le modèle RdPHFT-I fournit les moyens de description précise et non ambiguë des structures conceptuelles, de présentation et de contenu du document hypermédia.

B) Application des techniques de vérification supportées par le modèle RdPHFT. Ces techniques sont : la vérification des scénarios multimédias modélisés afin de tester leurs inconsistances temporelles;

et la détection des conflits d'utilisation des ressources partagées. S'il y a des erreurs, l'auteur doit revenir à la première étape.

C) Simulation de la spécification RdPHFT, afin de vérifier la correction du comportement logique et temporel du document. Si le comportement n'est pas celui désiré, l'auteur doit revenir à la première étape.

D) Traduction automatique de la spécification en une représentation MHEG. Après les étapes A, B et C, l'auteur aura une spécification analysée du document en développement. A ce moment, il peut réaliser la traduction MHEG afin d'obtenir une représentation physique qui puisse être interprétée par une machine MHEG. La procédure de traduction MHEG peut identifier des erreurs d'incomplétude de la spécification (causées par la non affectation de valeurs à des attributs non optionnels des objets *Données, Présentation* ou *Canal*).

E) Test final du document hypermédia par sa présentation réelle en utilisant une machine MHEG. Si le comportement de la présentation du document n'est pas celui désiré, l'auteur doit revenir à la première étape.

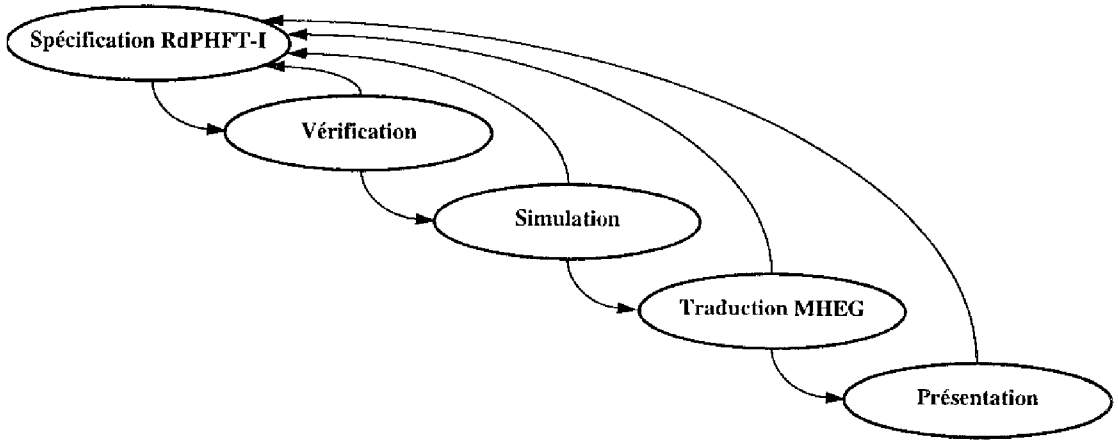


Figure 6.1 Cycle de vie simplifié pour la conception de documents hypermédias

6.1.1 L'environnement de développement de documents hypermédias MHEG

Dans cette section, nous présentons une étude sur la réalisation d'un environnement, appelée outil RdPHFT-I, mettant en œuvre la méthodologie de développement proposée. Cet outil peut être considérée comme un environnement formel de création et d'analyse temporelle d'objets MHEG, c'est-à-dire un *Système Auteur MHEG*.

Un prototype de l'outil RdPHFT-I a été développé sur Solaris® en utilisant le langage C++. Cet outil intègre un éditeur, un simulateur et un vérificateur décrits dans [Fabre, 95], et un traducteur MHEG développé dans le cadre de ce travail.

La figure 6.2 illustre l'application de notre méthodologie de développement de documents hypermédias dans un système hypermédia réparti. L'outil RdPHFT-I permet la création de documents hypermédias MHEG (ou de parties de documents) qui ensuite pourront être télé-chargés, dans un système ouvert hypermédia, vers une base de données MHEG ou directement vers un terminal de présentation. Il y a deux options pour le transfert des données primitives utilisées par le documents : télé-charger ces données au moment du transfert de la structure du document (regroupés dans un objet MHEG CONTAINER, conforme présenté dans la section 5.2.2); ou lors de la présentation du document, les terminaux de présentation peuvent demander le transfert de ces données.

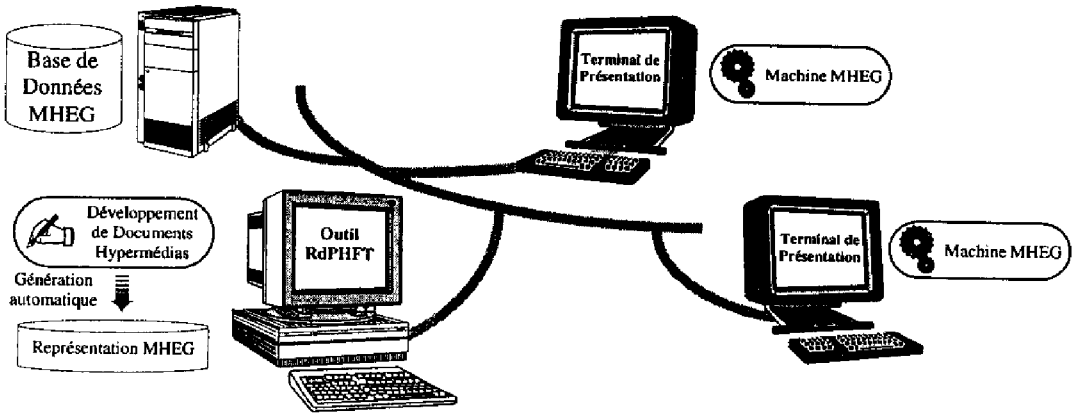


Figure 6.2 Application de l'approche de développement de documents hypermédias

Les modules fonctionnels de l'outil RdPHFT-I sont présentés dans la figure 6.3. Cette figure présente aussi les relations entre ces modules pendant le cycle de développement d'un document hypermédia.

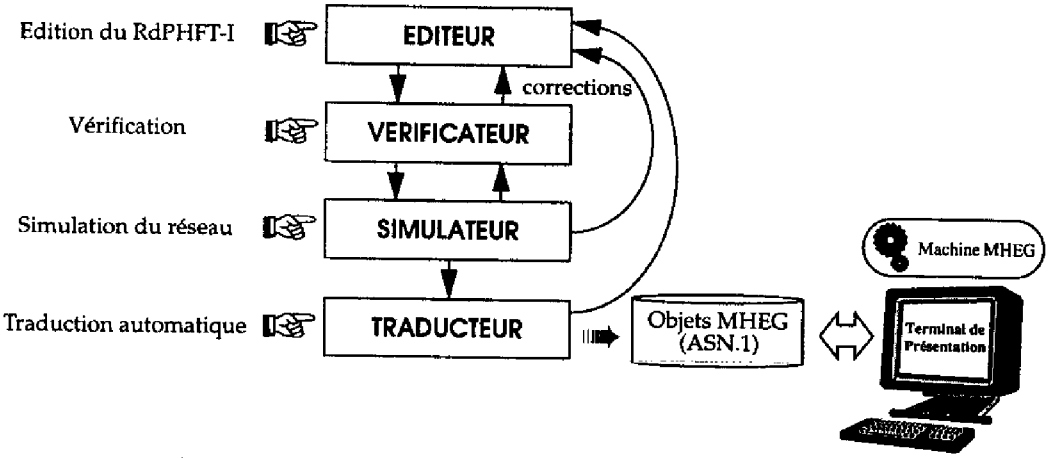


Figure 6.3 Outil de développement de documents hypermédias

6.1.1.1 Éditeur

L'éditeur doit fournir à l'auteur les moyens de construire, de façon graphique, le document hypermédia. Cet éditeur graphique doit faciliter la spécification des objets *Données*, *Canal* et *Présentation*, ainsi que la construction du RdPHFT.

Les objets *Données* peuvent être spécifiés à partir de la fenêtre de *Spécification des Données*. Cette fenêtre est une liste glissante où l'auteur dispose de commandes pour ajouter, affecter des valeurs, réorganiser et supprimer des objets *Données*. Une boîte de dialogue est ouverte quand l'auteur clique sur un élément de cette liste. Cette boîte de dialogue permet la spécification des attributs de l'objet *Données* (dont les attributs dépendent du type d'information). De plus, des éditeurs de données primitives peuvent être utilisés pour la création et la modification de l'information.

La figure 6.4a présente un exemple de fenêtre *Spécification des Données*. Elle permet la spécification de tous les objets multimédias du document hypermédia. Cette figure illustre la spécification d'une séquence vidéo MPEG, appelée *DonnéesVideo11*, avec une taille originale de 128 (H) x 256 (V) pixels et une durée originale de 30 s, stockée dans le fichier /home/dt/video11.mpg.

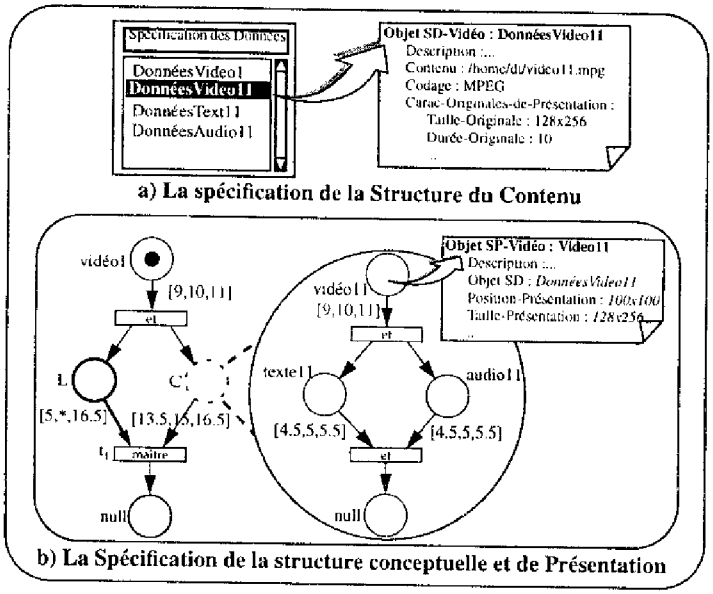


Figure 6.4 Exemple d'interface graphique du modèle RdPHFT-I

De manière similaire à la spécification des données, les canaux d'un document peuvent être spécifiés en utilisant une fenêtre *Spécification des Canaux*.

La structure Conceptuelle et de Présentation du document pourra être spécifiée à partir d'un éditeur graphique RdPHFT. Quand l'auteur clique sur une place RdPHFT-I, une boîte de dialogue est ouverte. Cette boîte de dialogue permet la spécification des attributs de l'objet *Présentation* associé à la place. Des outils additionnels peuvent être envisagés afin de simplifier la spécification des caractéristiques de présentation (par exemple, des facilités pour spécifier la position et la taille de présentation d'une image).

La figure 6.4b présente un exemple de spécification de la structure de Présentation et Conceptuelle d'un document hypermédia. Cette figure illustre la spécification d'une présentation de l'information *DonnéesVideo11*, appelée *Video11*. *Video11* est une présentation de *DonnéesVideo11* à afficher dans la position 100 (H) x 100 (V).

6.1.1.2 Vérificateur

Ce module est responsable de l'analyse temporelle statique du réseau RdPHFT. Ce module est responsable aussi de la détection des conflits d'utilisation des ressources partagées (section 4.4.4). En sortie, le vérificateur présente les erreurs. Si des risques d'incohérences temporelles et de conflits ne sont pas jugés acceptables, l'auteur doit éditer à nouveau la spécification RdPHFT-I afin de modifier les schémas de synchronisation.

6.1.1.3 Simulateur

Ce module fournit à l'auteur les moyens de réaliser une simulation formelle (interactive au automatique) de la spécification du document, permettant de vérifier (de façon non exhaustive) si la spécification représente bien le comportement désiré.

6.1.1.4 Traducteur

Après l'obtention d'une spécification correcte du document hypermédia, le concepteur peut demander la traduction automatique du réseau RdPHFT en une représentation MHEG (un objet MHEG COMPOSITE). La représentation MHEG du document hypermédia peut être ainsi échangée dans un système

hypermédia ouvert et interprétée par un interpréteur MHEG.

Un prototype du module de traduction MHEG à été réalisé à partir de l'étude menée dans le chapitre 5. Ce prototype permet la génération automatique des représentations MHEG [ISO 13522] des documents hypermédias spécifiés par des RdPHFT-I. Pendant la conception de ce traducteur MHEG, le compilateur ASN.1 SNACC [Sample, 94] a été utilisé pour la traduction des représentations ASN.1 des objets MHEG (définis par [ISO 13522]) en des classes C++ équivalentes, appelées *classes C++ MHEG*. Ces classes sont les représentations internes des objets MHEG du traducteur RdPHFT-MHEG. Toutes les classes C++ MHEG ont une méthode, appelée *codage MHEG*, permettant la traduction d'un objet C++ MHEG en un objet MHEG représenté selon les règles de codage BER (*Basic Encoding Rules*) [ISO 8825].

La figure 6.5 présente la procédure d'obtention, en utilisant le compilateur SNACC, d'une représentation interne des objets multimédias à partir de la représentation ASN.1 des objets MHEG (fournie par la norme MHEG). Cette procédure comporte les étapes suivantes :

A) Traduction, en utilisant le compilateur SNACC, de la représentation ASN.1 MHEG (des informations multimédias et hypermédias définies par la norme MHEG) en des classes C++. Ces classes, appelées *classes C++ MHEG*, définissent le format interne des objets MHEG du traducteur RdPHFT-MHEG. La méthode *codage MHEG* de ces classes C++ MHEG permet la traduction d'un objet C++ MHEG en un objet MHEG (la représentation ASN.1 codée).

B) Les objets instances des classes C++ MHEG sont des structures permettant l'enregistrement des attributs C++ MHEG.

C) L'utilisateur (la procédure de traduction RdPHFT-MHEG) peut affecter des valeurs aux attributs C++ MHEG.

D) Après l'affectation de toutes les valeurs aux attributs C++ MHEG, l'utilisateur peut générer les objets MHEG (la représentation ASN.1 codée) à travers la méthode de codage MHEG.

E) Les objets MHEG peuvent dont être interprétés par un interpréteur MHEG.

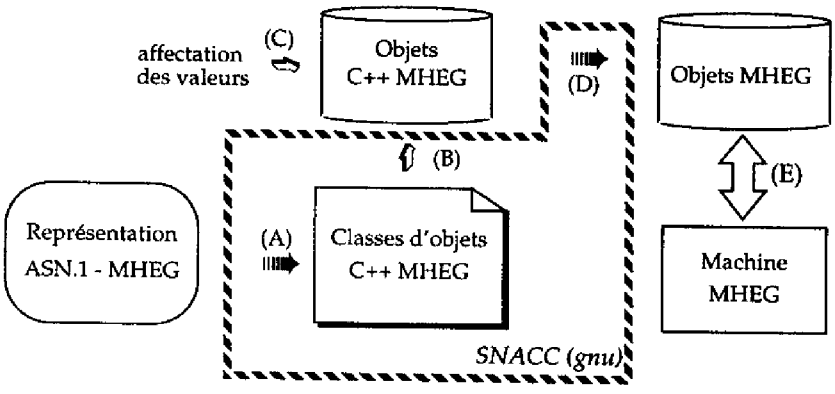


Figure 6.5 Représentation interne et sa conversion en un format normalisé

En utilisant les classes C++ MHEG comme représentation interne des objets MHEG, le module traducteur MHEG exécute la procédure de traduction d'une spécification RdPHFT-I en une représentation MHEG définie dans le chapitre 5. Basée sur la spécification RdPHFT-I, il génère un ensemble d'objets C++ MHEG, affecte les valeurs des attributs, et génère les objets MHEG à partir des méthodes de codage MHEG des objets C++ MHEG.

6.1.2 Vers la présentation des représentations MHEG générées

Afin de présenter les représentations MHEG des documents hypermédias générées par l'outil RdPHFT-I, nous disposons du toolkit MHEG développé par Euroclid/CCETT. Ce Toolkit est constitué d'un éditeur graphique de documents MHEG, d'une machine MHEG, et d'une application fournissant une interface graphique qui met à disposition de l'utilisateur un ensemble d'actions MHEG. Ces dernières permettent la préparation et la destruction des objets MHEG, et le départ et l'arrêt de la présentation de ces objets.

Dans ce toolkit MHEG, la représentation interne des objets MHEG est définie par la librairie MCL. Cette librairie MCL fournit les mêmes fonctionnalités des classes C++ MHEG générées par le compilateur SNACC, c'est-à-dire qu'elle est composée d'un ensemble de classes C++ qui constituent le format interne des objets MHEG du toolkit MHEG. Ces classes ont des méthodes permettant la traduction d'un objet C++ MHEG en un objet MHEG (la représentation ASN.1 codée) et vice-versa.

Les représentations MHEG qui sont générées par l'éditeur et interprétées par la machine MHEG des version 1 et 2 du toolkit MHEG Euroclid/CCETT ne sont pas conforme à la norme MHEG [ISO 13522]. Ainsi, afin d'utiliser ce toolkit, nous avons réalisé un nouveau prototype du traducteur MHEG. Dans cette version du traducteur MHEG, la représentation interne des objets MHEG (les classes C++ MHEG générées par le compilateur SNACC) ont été remplacées par la représentation interne du toolkit Euroclid/CCETT (la bibliothèque MCL). Cette modification a été nécessaire afin de permette l'utilisation des règles de codage ASN.1 des objets MHEG fournis par cette bibliothèque.

La machine MHEG du toolkit Euroclid/CCETT ne supporte pas toutes les classes d'objets MHEG ni toutes les actions MHEG. De plus, elle définit des sémantiques propres à quelques attributs MHEG. Par exemple, l'attribut *keywords* des objets MHEG est utilisé pour des styles de présentation des mécanismes d'interaction et des notions propres au toolkit.

Comme quelques objets et actions MHEG utilisés pour la procédure de traduction MHEG des spécifications RdPHFT-I (présentée dans le chapitre 5) ne sont pas supportées par la machine MHEG Euroclid/CCETT, cette procédure a été modifiée afin que les représentations MHEG générées par l'outil I-HSTPN puissent être interprétées par la machine MHEG de l'Euroclid/CCETT.

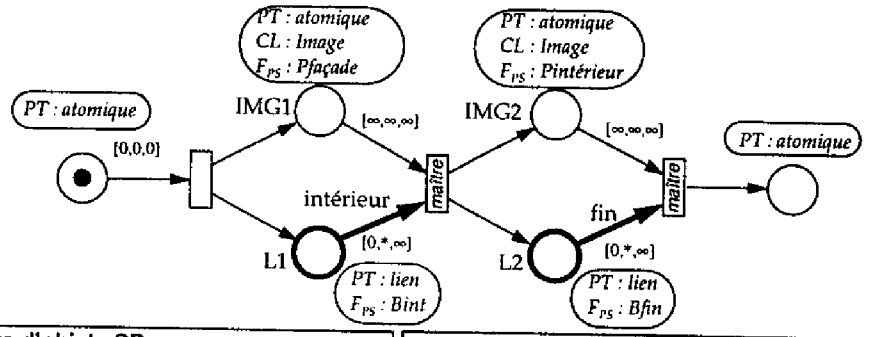
Les fonctions de création des marques temporelles (*timestones*) nécessaires au contrôle temporel des présentations ne sont pas supportées par la machine MHEG Euroclid/CCETT. Ainsi, cette machine ne supporte pas les relations temporelles entre les composants du document. Par exemple, la représentation MHEG obtenue à partir de la spécification présentée dans la figure 5.9 ne peut pas être interprétée par cette machine MHEG.

La machine MHEG Euroclid/CCETT ne supporte que les relations conditionnelles associées à des événements générés par les mécanismes d'interaction du document. Ainsi, dans cette implémentation, un ensemble de présentations peut être démarré ou arrêté lorsque l'utilisateur agit sur le document (par exemple, en cliquant sur un bouton). La figure 6.6 illustre la spécification RdPHFT-I d'une application multimédia qui peut être présentée par la machine MHEG de l'Euroclid/CCETT.

Test de la procédure de traduction MHEG

A cause des limitations du toolkit Euroclid/CCETT, nous pouvons utiliser uniquement des applications n'ayant pas de contraintes temporelles, les transitions doivent être du type maître et associées à une place Lien. Un exemple de ce type de document est présenté dans la figure 6.6. Pour ces types d'applications, la machine MHEG de l'Euroclid/CCETT a reconnu et a interprété la représentation MHEG générée par l'outil RdPHFT-I. A cause des limitations de la machine MHEG utilisée, nous ne

pouvons pas encore tester des documents hypermédias ayant des contraintes temporelles.



Liste d'objets SP :

Objet SP-Interaction **Bint** : := {
Position-Présentation=[3500,10718];
Taille-Présentation=[2898, 710];}

Objet SP-Image **Pfaçade** : := {
Objet-Données = Façade; (ou F_{DS})
Position-Présentation = [273,765];
Taille-Présentation=[100, 100];}

Objet SP-Interaction **Bfin** : := {
Position-Présentation=[6863,10691];
Taille-Présentation=[2898, 710];}

Objet SP-Image **PIntérieur** : := {
Objet-Données = Intérieur; (ou F_{DS})
Position-Présentation = [273,765];
Taille-Présentation=[100, 100];}

DD (Description du Document) :

Description :
Nom = «Scénario 1»
Taille-Présentation : [14027, 11675];

Liste d'objets SD :

Objet Image **Façade** : := {
Contenu= «/home/willrich/facade.gif»;
Codage-Propriétaire = 4444;
Caractéristiques-Originales-de-Présentation :
Taille-Originale = [12523, 9734];
Hiérarchie = inclusion;}

Objet Image **Intérieur** : := {
Contenu= «/home/willrich/interieur.gif»;
Codage-Propriétaire = 4444;
Caractéristiques-Originales-de-Présentation :
Taille-Originale = [8695, 6945];
Hiérarchie = inclusion;}

Figure 6.6 Spécification RdPHFT-I interprétable par la Machine MHEG Euroclid/CCETT

6.2 Vers l'interprétation Java de spécifications RdPHFT-I

Java [Sun, 95] est un langage orienté-objet et multi-tâches permettant la réalisation d'applications hypermédias (à l'aide de son paquetage graphique) sur Internet. Une application Java peut être transférée via le réseau et interprétée par plusieurs systèmes.

Cette section présente une méthodologie de création de présentations de documents hypermédias dans un environnement Java. Cette méthodologie est une dérivation de la méthodologie de développement de documents hypermédias MHEG (section 6.1), où les étapes de traduction MHEG et d'interprétation de la représentation MHEG (à partir d'une machine MHEG) ont été remplacées par une étape d'interprétation du RdPHFT-I par une application Java (figure 6.7) :

A) Utilisation du modèle RdPHFT-I pour la spécification formelle du document hypermédia.

B) Application des techniques de vérification supportées par le modèle RdPHFT.

C) Simulation de la spécification RdPHFT, afin de vérifier la correction du comportement logique et temporel de la présentation du document.

D) Test du document hypermédia par l'interprétation de la spécification RdPHFT-I analysée. Si le comportement de la présentation du document n'est pas celui désiré, l'auteur doit revenir à la première étape.

Après le test du document hypermédia, l'auteur peut obtenir une application Java correspondant au

document spécifié par RdPHFT-I. Cette application intègre l'interpréteur et la description du document. Ainsi le document hypermédia spécifié par un RdPHFT-I peut être transféré et présenté sur Internet.

La norme MHEG permet la représentation portable des informations hypermédias et multimédias. Cette représentation est constituée des objets statiques qui doivent être interprétés par une machine MHEG. De façon différente, Java est un langage de programmation d'applications multimédias fournissant un code portables de l'application. Ainsi, les documents hypermédias développés par les outils RdPHFT-I et Java RdPHFT-I sont portables : le premier grâce au codage MHEG des documents hypermédias (le document hypermédia est portable) et le deuxième grâce à la portabilité de l'application Java qui intègre l'interpréteur Java RdPHFT-I et la description du document.

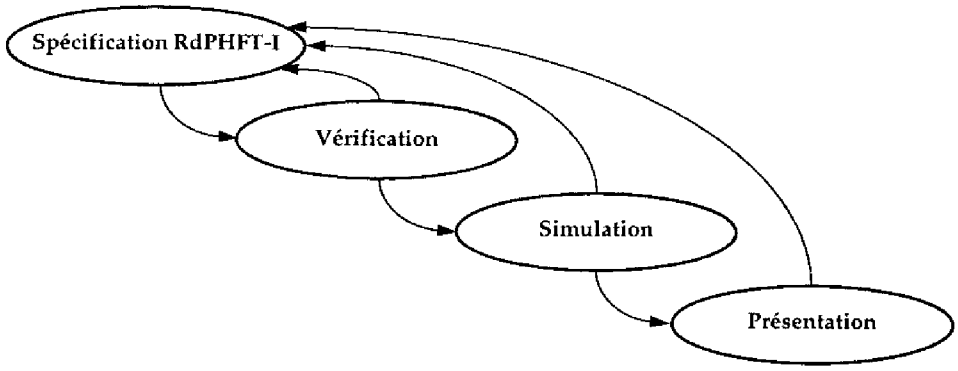


Figure 6.7 Cycle de vie de conception de documents hypermédias interprétables en Java

6.2.1 Le modèle RdPHFT-I interprétable en Java

Quelques attributs des classes *Données* et *Présentation* du modèle RdPHFT-I détaillées dans l'annexe A sont orientés vers la traduction de la spécification RdPHFT-I en MHEG. Quelques uns des ces attributs ne sont pas supportés par le paquetage graphique de Java. Par exemple, l'attribut *Inclusion* qui définit si la description des données est enregistrée conjointement avec les données elles-mêmes n'a pas de correspondant en Java.

Afin de permettre l'interprétation d'une spécification RdPHFT-I par une application Java, des nouveaux attributs ont été rajoutés aux classes *Données* et *Présentation* du modèle RdPHFT-I. Pour les classes *Données*, afin de permettre l'accès des données via le réseau, l'attribut *Content* spécifie l'URL (*Uniform Ressources Locator*) du fichier des données. Pour les classes *Présentation*, les attributs suivants ont été ajoutés :

- deux attributs pour la description des couleurs de premier et d'arrière plan (*Foreground* et *Background*) de la classe SP-Spatial;
- un attribut pour la description du format des caractères des informations textuelles. Cet attribut définit la police (*dialog, helvetica, timesRoman, courier, symbol*), le style (*bold, italic, plain*) et la taille des caractères.
- un attribut pour la description du style de présentation d'une information textuelle. Ces styles sont *Area* pour la présentation d'une seule ligne de texte et *Field* pour la présentation d'un texte contenant plusieurs lignes et ayant des barres glissantes horizontales et verticales.

6.2.2 L'environnement Java de développement de documents hypermédias

Dans cette section, nous présentons une application Java, appelée outil Java RdPHFT-I, mettant en

œuvre la méthodologie de développement de documents hypermédias basée sur le modèle RdPHFT-I.

L'outil Java RdPHFT-I fournit un ensemble de modules pour aider le processus de création, d'analyse et de présentation de documents hypermédias. Ces modules sont présentés dans la figure 6.8. Les modules Editeur, Vérificateur et Simulateur fournissent de fonctionnalités identiques à leurs correspondants de l'environnement de développement de documents hypermédias MHEG (sections 6.1.1.1, 6.1.1.2 et 6.1.1.3). L'unique différence réside dans les nouveaux attributs des classes *Données* et *Présentation* du modèle RdPHFT-I présentés dans la section précédente.

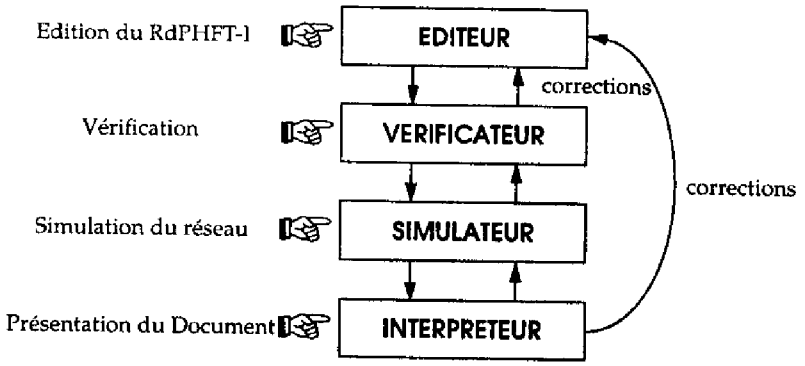


Figure 6.8 Environnement Java de développement de documents hypermédias

Le module Interpréteur présente le document hypermédia à partir de sa spécification RdPHFT-I. Dans le cadre de ce travail, nous avons implémenté le module Interpréteur, qui est présenté en détail dans l'annexe B.

La figure 6.9 représente les fonctionnalités fournies par l'interpréteur RdPHFT-I. L'entrée de cet interpréteur est un fichier contenant la spécification RdPHFT-I du document hypermédia, généré par le module Editeur. Les opérations de commande de l'interpréteur sont les commandes d'interprétation et les interactions de l'utilisateur lors de la présentation du document. Les commandes d'interprétation sont disponibles pour l'utilisateur via le menu principal; ce sont : lecture de la spécification, départ et arrêt de l'interprétation du document. Les données primitives utilisées par le document hypermédia sont récupérées, via le réseau, à partir de la spécification de leurs URL.

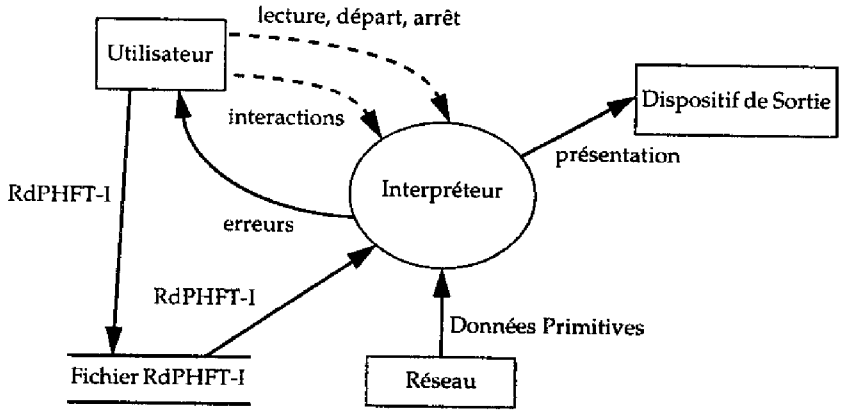


Figure 6.9 L'interpréteur Java RdPHFT-I

A partir de la spécification RdPHFT-I, le module d'interprétation crée un ensemble d'objets actifs (de processus) communicants responsables de la présentation du document. Ces objects actifs sont responsables de l'exécution des fonctionnalités modélisées par les places, les arcs et les transitions de la spéci-

fication RdPHFT-I. Un sous-réseau RdPHFT-I générique (incluant la racine ou un des sous-réseau de L ou S) est représenté par les objets suivants :

- chaque place Composite est associée à un sous-réseau S (la racine de la spécification est toujours associée à une place composite abstraite) est représentée par un objet Ordonnanceur qui est responsable de l'évolution du sous-réseau S (i.e., le tir des transitions et le changement du marquage) et par un objet qui contrôle la validité temporelle de S. Lors du tir d'une transition de S, l'objet Ordonnanceur exécute l'arrêt et le départ des objets qui représentent les places d'entrée et de sortie de cette transition.
- chaque place Atomique est représentée par un ensemble d'objets responsable de la présentation modélisée par la place, et par un objet responsable respectivement du contrôle de la validité temporelle de cette présentation.
- chaque place Lien est représentée par un objet responsable de la présentation de l'ancre (par exemple, un bouton) et de l'exécution des actions associées aux interactions (par exemple, le traitement des actions lors que l'utilisateur clique sur un bouton), et par un objet responsable du contrôle de la validité temporelle de la présentation.

Grâce à son approche orientée-objet, son caractère multi-tâches et ses paquetages graphique et réseaux, le langage Java a simplifié la tache d'implémentation de l'environnement Java RdPHFT-I. Le principal intérêt d'utilisation du langage Java est sa portabilité sur l'Internet. Une application multimédia développée en Java peut accéder des composants stockés n'importe où sur le réseau Internet et elle peut être transférée et présentée sur plusieurs systèmes d'exploitation. Néanmoins, la faible performance de la version du Kit de Développement Java utilisé (1.0) ne permet pas le contrôle temporel strict de l'IVT des présentations (par exemple, la notification que l'âge maximal d'une présentation est dépassé et peut souffrir des retards considérables si la charge du système est grande). De plus, cette faible performance induit des délais considérables entre la demande de présentation d'une information et sa présentation effective.

Afin de tester l'interpréteur Java RdPHFT-I, nous avons utilisé cette application Java pour le développement du module d'EAO de descente normale du train d'atterrissage de l'Airbus A340 présenté dans la section 4.6. Cet exercice a servi également à la démonstration que le modèle RdPHFT-I permet efficacement de décrire une application multimédia complexe et ensuite de générer à une présentation concrète du document.

6.3 Conclusion

Ce chapitre a présenté une méthodologie formelle de prototypage de documents hypermédias dans laquelle la spécification et l'analyse des documents sont faites à l'aide du modèle RdPHFT-I et l'implantation est supportée : par la production automatique de représentations MHEG qui peut être échangée et présentée dans un système ouvert; ou par l'interprétation de la spécification RdPHFT-I par une application Java qui peut être transférée sur Internet et présentée sous plusieurs systèmes d'exploitation. Ainsi, cette méthodologie formelle permet l'obtention d'une spécification précise et non ambiguë d'un document hypermédia et permet ensuite la production de documents hypermédias portables.

Nous avons réalisé deux outils mettant en œuvre la méthodologie de développement de documents hypermédias : l'outil RdPHFT-I MHEG permettant la génération de représentations MHEG; et l'outil Java RdPHFT-I permettant l'interprétation de spécifications RdPHFT-I.

L'outil RdPHFT-I fournit un ensemble de modules pour aider le processus de création, d'analyse et de traduction MHEG de documents hypermédias. Dans le cadre de ce travail, nous avons développé, sur

Solaris® en utilisant le langage C++, le module de traduction de RdPHFT-I en représentations MHEG.

Le toolkit MHEG développé par Euroclid/CCETT a été utilisé pour tester les représentations MHEG générées par l'outil RdPHFT-I. Comme ce toolkit n'est pas conforme à la version de la norme MHEG utilisée par l'outil RdPHFT-I, ce dernier a été modifié afin de permettre la génération de représentations MHEG interprétables par le toolkit Euroclid/CCETT. La machine MHEG du toolkit Euroclid/CCETT a reconnu et a interprété la représentation MHEG générée par l'outil RdPHFT-I. A cause des limitations de la machine MHEG utilisée, nous ne pouvons pas tester des documents hypermédias ayant des contraintes temporelles.

L'outil Java RdPHFT-I permet la création et la présentation de documents hypermédias à l'aide d'une application Java. Cet outil Java RdPHFT-I fournit un ensemble de modules pour aider le processus de création, d'analyse et de présentation de documents hypermédias. Dans le cadre de ce travail, nous avons implémenté une application Java d'interprétation (présentation) de spécification RdPHFT-I.

Ces deux outils permettent l'obtention des documents hypermédias portables : l'outil RdPHFT-I génère des représentations MHEG des documents hypermédias qui peuvent être échangées et présentées dans un système ouvert ISO; l'outil Java RdPHFT-I génère des documents portables grâce à la portabilité de l'interpréteur Java RdPHFT-I (l'application de présentation et le document présenté sont portables).

Une évolution envisagée des travaux de conception de documents hypermédia portables est le développement d'une application Java permettant à la fois : la génération de représentation MHEG et l'interprétations de spécifications RdPHFT-I. Cet environnement de création de documents hypermédias portables contiendrait : un module Editeur de RdPHFT interprété et orienté MHEG Java (une choix doit être mise à disposition de l'auteur); un module Analyseur et Simulateur identiques aux deux approches de conception; un module Traducteur MHEG; et un module Interpréteur Java.

Chapitre 7

Conclusion

Afin de concevoir des documents multimédias et hypermédias portables, le futur standard international ISO/IEC MHEG (Multimedia Hypermedia Expert Group) [ISO 13522] définit un codage (une représentation) des informations multimédias et hypermédias, permettant l'échange de plusieurs types de médias, ainsi que le transfert des structures représentant la composition temporelle et spatiale de ces médias et les interactions existant lors de leurs présentations. L'adoption du standard MHEG pour représenter les informations hypermédias et multimédias permet le transfert et le traitement de ces types d'informations dans un système ouvert.

A cause de la complexité structurelle des systèmes hypermédias, la conception de ces systèmes ne peut pas être abordée efficacement et aisément en utilisant directement la norme MHEG. Ainsi, un modèle logique de plus haut niveau doit être utilisé pour faciliter la description des documents hypermédias. Ce travail a proposé un tel modèle de spécification de documents hypermédias, basé sur le modèle *Réseaux de Petri Hiérarchisés à Flux Temporels* (RdPHFT), qui permet ensuite la génération automatique de représentations MHEG.

Le modèle RdPHFT permet une spécification formelle unifiée, complète et précise de la synchronisation logique et temporelle à l'intérieur des systèmes hypermédias distribués. Ce modèle n'a pas pour objectif la description complète de documents hypermédias; par exemple, il ne fournit pas les moyens de spécifier les informations d'accès ou les caractéristiques spatiales et audibles des présentations qui composent un document.

Le modèle proposé dans ce travail est une version interprétée du modèle RdPHFT, appelée RdPHFT-I, qui étend le modèle RdPHFT afin de spécifier complètement un document hypermédia. Le terme «interprétation» signifie que l'on a étendu la sémantique des places du modèle RdPHFT. Cette sémantique a été définie à partir des concepts proposés par la norme MHEG, cela afin de permettre la représentation de toutes les informations nécessaires à la génération automatique de représentations MHEG.

Le modèle RdPHFT-I permet la description multi-niveaux d'un document hypermédia :

- La structure conceptuelle décrit les différentes parties logiques du document, ou composants, leurs relations logiques et l'instant de présentation de ces composants. Elle est spécifiée par une spécification RdPHFT.

- La structure de présentation décrit où et comment les différents composants seront présentés. Elle est spécifiée par un ensemble d'objets des classes *Présentation* et *Canal*.
- La structure du contenu décrit les informations qui constituent les composants, c'est-à-dire qu'elle définit les objets multimédias (les données primitives et leurs description). Les objets multimédias sont spécifiés par un ensemble d'objets des classes *Données*;

L'interface entre ces structures est réalisée par des fonctions. Celles-ci associent les places RdPHFT aux présentations qu'elles spécifient, et elles associent aussi ces présentations aux canaux et aux objets multimédias.

La description multi-niveaux du modèle RdPHFT-I fournit plusieurs avantages. Par exemple, elle permet la réutilisation et la particularisation de la structure conceptuelle lors de la présentation du document hypermédia sur différents systèmes clients, et elle permet aussi la réutilisation des objets multimédias en différents contextes du document et par différents documents.

Ce travail a présenté aussi une procédure de traduction d'une spécification RdPHFT-I vers une représentation MHEG. L'utilisation du modèle RdPHFT-I pour le développement de documents hypermédias MHEG permettra l'obtention d'une spécification complète et non ambiguë du document. Il permet aussi l'application de méthodes de vérification, de simulation et d'analyse avant la réalisation d'une traduction automatique du document hypermédia en une représentation MHEG. Le champ d'application de ce modèle est limité aux documents qui peuvent avoir leurs relations logiques et temporelles spécifiées par le modèle RdPHFT.

Au delà du modèle RdPHFT-I, ce travail a présenté une méthodologie formelle de prototypage de documents hypermédias dans laquelle la spécification et l'analyse des documents sont faites à l'aide du modèle RdPHFT-I et l'implantation est supportée par :

- la production automatique de représentations MHEG, à l'aide d'un environnement, appelé outil RdPHFT-I, mettant en œuvre la méthodologie de développement de documents hypermédias MHEG. Dans le cadre de ce travail, un prototype de cet outil a été développée sur Solaris® en utilisant le langage C++.

  Le toolkit MHEG développé par Euroclid/CCETT a été utilisé pour tester les représentations MHEG générées par l'outil RdPHFT-I. La machine MHEG du toolkit Euroclid/CCETT a reconnu et a interprété la représentation MHEG générée par l'outil RdPHFT-I. A cause des limitations de la machine MHEG utilisée, nous ne pouvons pas encore tester des documents hypermédias ayant des contraintes temporelles.
- la présentation de documents hypermédias à l'aide d'une application Java. L'application Java RdPHFT-I que nous avons aussi développé fournit un ensemble de modules pour assistant l'auteur dans le processus de création, d'analyse et de présentation de documents hypermédias.

Ces deux approches permettent l'obtention des documents hypermédias portables : l'outil RdPHFT-I génère des représentations MHEG des documents hypermédias qui peuvent être échangés et présentés dans un système ouvert ISO; et l'outil Java RdPHFT-I génère des documents portables grâce à la portabilité des applications Java.

Perspectives

Le travail de recherche réalisé dans le cadre de cette thèse peut être poursuivit dans les directions suivantes:

- Le modèle RdPHFT-I tend à être de trop bas niveau pour l'utilisation directe par un auteur d'un document, qui en général est quelqu'un qui n'est pas habitué au le formalisme réseau de Petri. Ainsi, une étude pourrait être menée afin de définir une interface graphique de plus haut niveau, afin de simplifier la tâche de conception de documents hypermédias et permettant ensuite la production des RdPHFT-I correspondants. Ce type d'opération a été mise en œuvre, par exemple, dans l'environnement CMIFed (section 2.4.2) et [Courtiat, 96b]. Des approches plus classiques de conception de documents multimédias, comme les organigrammes et les lignes temporelles pourrait être intégrées dans cette réflexion.
- Dans ce mémoire, nous avons présenté deux outils mettant en œuvre notre méthodologie formelle de conception de documents hypermédias portables : l'outil RdPHFT-I pour la génération de représentations MHEG; et l'application Java RdPHFT-I pour l'interprétation de spécifications RdPHFT-I. Une évolution envisagée pour les travaux de conception est le développement d'une application Java permettant à la fois : la génération de représentation MHEG et l'interprétations de spécifications RdPHFT-I. Cet environnement permettrait l'édition de RdPHFT-I orientés MHEG ou Java, la vérification et la simulation de cette spécification, et enfin la traduction en une représentation MHEG ou son interprétation à partir de l'application Java.
- Afin de tester et de présenter les objets MHEG générés à partir d'une spécification RdPHFT-I, nous avons utilisé la machine MHEG développée par Euroclid/CCETT. A cause des limitations de cette machine MHEG, nous ne pouvons pas encore tester des documents hypermédias ayant des contraintes temporelles. Ces tests pourront seulement être réalisés lors qu'une machine MHEG supportant toutes les fonctionnalités MHEG (objets et actions MHEG) utilisées lors de la traduction MHEG d'un réseau RdPHFT-I sera disponible.
- Dans le cadre de ce travail, nous avons défini et implémenté une méthodologie formelle de conception de documents hypermédias dont le mode de transfert des informations multimédias et hypermédias est le télé-chargement (section 2.2). La spécification de documents multimédias qui sont transmis en temps-réel et ayant leurs présentations synchronisées ont des besoins de spécification différents (section 2.3.2.3). Des nouveaux travaux de conception de systèmes multimédias répartis et de nouveaux protocoles multimédias peuvent être menés à partir de la méthodologie proposée dans ce travail et celle proposée par [Diaz, 95]. Cette dernière montre que le modèle RdPFT peut être utilisé pour la conception de documents multimédias qui doivent être transférés à partir d'un serveur et présentés, après un délai de transmission et de traitement, sur un système client.

En conclusion, les principales contributions apportées par ce travail de recherche sont les suivants :

- Nous avons identifié les principales exigences qui doivent être satisfaites par un modèle hypermédia (section 2.3). Dans cette étude, nous avons montré qu'un modèle hypermédia idéal devrait fournir une description multi-niveaux d'un document hypermédia, incluant la spécification des structures du contenu, conceptuelle et de présentation. Nous avons définis les besoins de chacun de ces niveaux.
- Nous avons analysé les principaux modèles pour la description de documents multimédias et hypermédias (section 2.4), en nous basant sur l'étude précédente : nous avons identifié les principaux avantages et limitations de chacun de ces modèles.
- Nous avons proposé une extension au modèle RdPHFT afin de permettre la spécification complète de documents hypermédias à partir d'une approche multi-niveaux, distinguant les structures du contenu, conceptuelle et de présentation d'un document (chapitre 4).

- Nous avons défini une procédure de traduction d'une spécification RdPHFT-I en une représentation MHEG équivalente, permettant ainsi le passage de la description formelle des documents hypermédias à leur implémentation (chapitre 5).
- Au delà du modèle RdPHFT-I, nous avons mis en évidence une approche formelle de conception de documents hypermédias portables dont la spécification et l'analyse du document sont réalisées à l'aide du modèle RdPHFT-I et l'implémentation est supportée soit par la traduction automatique de la spécification en représentations MHEG, soit par sont interprétation à partir d'une application Java (chapitre 6).
- L'implémentation d'un module de traduction de spécifications RdPHFT-I en MHEG, et le test des représentation MHEG générées en utilisant la machine MHEG développée par Euroclid/CCETT (section 6.1).
- L'implémentation d'une application Java RdPHFT-I permettant la présentation de documents hypermédias spécifiés par un RdPHFT-I (section 6.2).
- L'utilisation de l'application Java RdPHFT-I pour le développement d'un prototype du module d'EAO de descente normale du train d'atterrissage de l'Airbus A340 (sections 4.6 et 6.2). Cet exercice a servi également à la démonstration que le modèle RdPHFT-I permet efficacement de décrire une application multimédia complexe et ensuite de générer à une présentation concrète du document.

En résumé, la principale contribution de ce travail est d'avoir proposé et mise en œuvre une méthodologie permettant de contrôler la qualité et de maîtriser la complexité des systèmes hypermédias distribués, à partir d'un cycle de vie permettant de générer directement un document hypermédia à partir d'une spécification formelle vérifiée et validée.

Chapitre 8

Bibliographie

8.1 Bibliographie de l'auteur

Conférences internationales

[Sénac, 95a] P. Sénac, P. de Saqui-Sannes, R. Willrich. Hierarchical Time Stream Petri Net: a Model for Hypermedia Systems. 16th International Conference on Application and Theory of Petri Nets, Torino, June 1995. In Application and Theory of Petri Nets 1994, Lecture Notes in Computer Science no. 935, G. De Michelis and M. Diaz (Eds.), Springer, pp. 451-470.

[Sénac, 95b] P. Sénac, R. Willrich, M. Diaz. Hypermedia Synchronization Modelling: a Case Study. ED-MEDIA'95 World Conf. on Educational Multimedia and Hypermedia, pages 585-590, Graz, 1995.

[Willrich, 96a] R. Willrich, P. Sénac, M. Diaz, P. de Saqui-Sannes. A Formal Framework for the Specification, Analysis and Generation of Standardized Hypermedia Documents. A apparaître au Third IEEE International Conference on Multimedia Computing and Systems (ICMCS'96), Hiroshima (Japon), 17-21 Juin 1996.

[Willrich, 96b] R. Willrich, P. de Saqui-Sannes, P. Sénac, M. Diaz. Une approche formelle pour le développement de documents hypermédias normalisés. A apparaître au Colloque Francophone sur l'Ingénierie des Protocoles (CFIP'96), Rabat (Maroc), 14-17 octobre 1996.

[Willrich, 96c] R. Willrich, P. de Saqui-Sannes, P. Sénac, M. Diaz. Hypermedia Document Design Using the HTSPN Model. A apparaître au Third International Conference on Multimedia Modeling (MMM'96), Toulouse (France), le 12-15 Novembre 1996.

Conférences nationales

[Willrich, 96d] R. Willrich, P. de Saqui-Sannes, P. Sénac, M. Diaz. Towards Hypermedia Documents Design. XIV Brazilian Symposium on Computer Networks (SBRC'96), pages 473-491, Fortaleza (Brésil), 1996.

Rapports de contrat

[Baudin, 93] V. Baudin, R. Willrich, P. Sénac, M. Diaz. Spécification/Conception d'une chaîne d'outils pour la production de documents multimédia. LAAS/CNRS - Rapport du marché CNET France Télécom 92 1b 178 - Lot 4. Décembre 1993.

[Willrich, 94] R. Willrich, P. Sénac, M. Diaz. Une approche formelle pour la spécification et la conception de documents hypermédias. Rapport LAAS N94481. Contrat CNET FT N92.1B.178. Projet CESAME. Lot 2, Décembre 1994.

[Courtiat, 94a] P. Courtiat, M. Filali Amine, A. Léger, R.C. de Oliveira, P. de Saqui Sannes, P. Sénac, R. Willrich. Modèles pour la synchronisation multimédia et relations entre modèles. Rapport LAAS N94479. Contrat CNET FT N92.1B.178. Projet CESAME. Lot 2, Décembre 1994.

8.2 Références Bibliographiques

[Ackermann, 94] P. Ackermann. Direct Manipulation of Temporal Structures in a Multimedia Application Framework. In proc. of the ACM Multimedia 94. October, 1994.

[AimTech, 93] AimTech Corporation. IconAuthor User Manual for OSF/Motif. November, 1993.

[Appelt, 93] W. Appelt. HyperODA - Extensions for Temporal Relationships. Proposed Draft Amendment (version 2) ISO/IEC JTC 1/SC 18/WG 3/N 2516, 1993.

[Apple, 91] Apple Computer Inc. QuickTime Developer's Guide. Developer technical publications, 1991.

[Baudin, 93] V. Baudin, R. Willrich, P. Sénac, M. Diaz. Spécification/Conception d'une chaîne d'outils pour la production de documents multimédia. LAAS/CNRS - Rapport du marché CNET France Télécom 92 1b 178 - Lot 4. Décembre 1993.

[Blakowski, 92] G. Blakowski, J. Jens Hübel, U. Langrehr, M. Mühlhäuser. Tool Support for the Synchronization and Presentation of Distributed Multimedia. Computer Communication, 15(10): 611-618. December, 1992.

[Blakowski, 96] G. Blakowski, R. Steinmetz. A Media Synchronization Survey: Reference Model, Specification, and Case Studies. IEEE Journal on Selected Areas in Communications 14(1): 5-35, 1996.

[Botafogo, 95] R. Botafogo, D. Mossé. The MORENA Model for Hypermedia Authoring and Browsing. In proc. of the IEEE Int. Conf. on Multimedia Computing and Systems, pp. 42-49, Washington, DC, May 1995.

[Buchanan, 92] M.C. Buchanan, P.T. Zellweger. Specifying Temporal Behaviour in Hypermedia Documents. In proc. of the ACM ECHT Conference, pp. 262-271. 1992.

[Buchanan, 93] M.C. Buchanan, P.T. Zellweger. Automatically Generating Consistent Schedules for Multimedia Documents. Multimedia Systems Journal, 1993.

[Buford, 94a] J.F.K. Buford. Multimedia Systems. ACM Press, SIGGRAPH Series, New York, New York, 1994.

[Buford, 94b] J.F.K. Buford. L. Rutledge, J.L. Rutledge. Toward Automatic Generation of HyTime Applications. In the Proc. of Eurographics Multimedia, 1994.

[Bulterman, 91] D.C.A. Bulterman, G. van Rossum, R. van Liere. A Structure for Transportable, Dynamic Multimedia Documents. Proc Summer 1991 Usenix Conference, Nashville TN, June 1991.

[Chassot, 95] C. Chassot. Architecture de Transport Multimédia à Connexions d'Ordre Partiel. Thèse présentée au Laboratoire d'Analyse et d'Architecture des Systèmes du CNRS en vue de l'obtention du titre de Docteur de l'INP de Toulouse. Toulouse, 1995.

[Chassot, 96] C. Chassot, M. Diaz, A. Lozes. From the Partial Order Concept to Partial Order Multimedia Connections. To appear in Journal for High Speed Networks, 1996.

[Clarke, 86] E. Clarke, E.A. Emerson, S. Sistla. Automatic verification of concurrents systems. ACM TOPLAS 8(2):244-263, 1986.

[Colaitis, 92] F. Colaitis. The MHEG standard in the overall framework of multimedia and hypermedia standardisation work. Multimedia and Normalisation, Picture and Audio Technologies - Issues at stake, Conférence internationale sur les normes de codage de l'image et des sons numérisés et leurs application. 28-29 janvier 1992.

[Courtiat, 94a] P. Courtiat, M. Filali Amine, A. Léger, R.C. de Oliveira, P. de Saqui Sannes, P. Sénac, R. Willrich. Modèles pour la synchronisation multimédia et relations entre modèles. Rapport LAAS N94479. Contrat CNET FT N92.1B.178. Projet CESAME. Lot 2, Décembre 1994.

[Courtiat, 94b] J-P Courtiat, R.C. de Oliveira, L.F.R. da Costa Carmo. Towards a New Multimedia Synchronisation Mechanism and its Formal Specification. In proc. of the ACM Multimedia'94, pp. 133-140, 1994.

[Courtiat, 96a] J-P Courtiat, L.F.R. da Costa Carmo, R.C. de Oliveira. IEEE Journal on Selected Areas in Communications 14(1):185-195, 1996.

[Courtiat, 96b] J-P Courtiat, R.C. de Oliveira. Proving Temporal Consistency in a New Multimedia Synchronization Model. A apparaître au ACM Multimedia'96, Boston, Novembre 1996.

[Diaz, 93] M. Diaz, P. Sénac. Time Streams Petri Nets, a Model for Multimedia Streams Synchronization. In proc. of the First International Conference on Multi-media Modelling - Vol.1, Singapore, World Scientific, Tat-Seng Chua & T. L. Kunii (Eds.), pages 257-273, November 1993.

[Diaz, 94a] M. Diaz, G. Pays. The CESAME Project: Formal Design of High Speed Multimedia Cooperative Systems. Annals of Telecommunications 49(5-6): 220-229, May-June 1994.

[Diaz, 94b] M. Diaz, P. Sénac. Time Streams Petri Nets, a Model for Timed Multimedia Informations. 15th International Conference on Application and Theory of Petri Nets, Zaragoza, June 1994. In Application and Theory of Petri Nets 1994, R. Valette (Ed.), Springer-Verlag, pages 219-238, 1994.

[Diaz, 95] M. Diaz. Design of Multimedia Protocols Based on Multimedia Models. In proc. of the 3rd Int. Conf. on Multi-Media Modelling, Singapore, World Scientific, Tat-Seng Chua, Hung K. Pung, T. L. Kunii (Eds.), pages 35-51, November 1995.

[Drapeau, 91] G.D. Drapeau, H. Greenfiled, H. MAEstro - A Distributed Multimedia Authoring Environment. In proc. of the USENIX Summer'91 Conference, pp. 315-328. Nashville TN, 1991.

[Erfle, 94] R. Erfle. HyTime as the Multimedia Document Model of Choice. In proc. of the International Conference on Multimedia Communication and Systems, pp. 445-454. 1994.

[Eyes, 92] Eyes 2.0 Reference Manual. Center for Productivity Enhancement. University of Massachusetts-Lowell. 1992.

[Fabre, 95] F. Fabre, P. Sénac, M. Diaz. A Toolkit for the Modelling of Multimedia Synchronization Scenarios. In proc. of the 3rd Int. Conf. on Multi-Media Modelling, Singapore, World Scientific, Tat-Seng Chua, Hung K. Pung, T. L. Kunii (Eds.), pages 187-201, November 1995.

[FAQ, 96] Multimedia Authoring Systems FAQ - Version 1.2 - 1/7/96. The Multimedia Authoring Systems FAQ is maintained by Jamie Siglar and is mirrored at <http://www.tiac.net/users/jasiglar/MMASFAQ.HTML>.

[Fluckiger, 95] F. Fluckiger. Understanding Networked Multimedia: Applications and Technology. Prentice Hall. 1995.

[Gibbs, 91] S. Gibbs. Composite Multimedia and Active Objects. In proc. of the OOPSLA'91. Phoenix, Arizona. October, 1991.

[Ginige, 95] A. Ginige, D.B. Lowe, J. Robertson. Hypermedia Authoring. IEEE Multimedia 2(4):24-35, 1995.

[Grønbaek, 94] K. Grønbaek, R.H. Trigg. Design Issues for a Dexter-Based Hypermedia System. Communication of the ACM 37(2):40-49, 1994.

[Haindl, 96] M. Haindl. Multimedia Synchronization. Computer Science/Department of Interactive Systems. IEEE Journal on Selected Areas in Communication 14(1):73-83, 1996.

[Halasz, 90] F. Halasz, M. Schwartz. The Dexter Hypertext Reference Model. NIST Hypertext Standardization Workshop, Gaithersburg, MD, January 16-18 1990.

[Halasz, 94] F. Halasz, M. Schwartz. The Dexter Hypertext Reference Model. Communication of ACM 37(2):29-39. February 1994.

[Hamblin, 72] C.L. Hamblin. Instants and Intervals. In proc. of 1st Conf. Int. Soc. for the Study of Time, J.T. Fraser et al. (Eds.), Springer-Verlag, pages 324-331, New York, 1972.

[Hardman, 93] L. Hardman, D.C.A. Bulterman, G. van Rossum. The Amsterdam Hypermedia Model: Extending Hypertext to Support Real Multimedia. Hypermedia Journal 5(1), May, 1993.

[Hardman, 94] L. Hardman, D.C.A. Bulterman, G. van Rossum. The Amsterdam Hypermedia Model: Adding Time, Structure and Context to Hypertext. Communication of the ACM 37(20):50-62. February, 1994.

[Hardman, 95] L. Hardman, D.C.A. Bulterman. Authoring Support for Durable Interactive Multimedia Presentations. STAR Report in Eurographics'95.

[Herman, 94] I. Herman, G.J. Reynolds. MADE: A Multimedia Application Development Environment. In proc. of the International Conference on Multimedia Communication and Systems, pp 184-193, 1994.

[Hirzalla, 95] N. Hirzalla, B. Falchuk, A. Karmouch. A Temporal Model for Interactive Multimedia Scenarios. IEEE Multimedia (2) 3: 24-31, 1995.

[Hoepner, 94] P. Hoepner. Synchronizing the Presentation of Multimedia Objects». Computer Communication 15(9):557-564, November, 1994.

[Hudson, 93] S.E. Hudson, C.H. Hsi. A Framework for Low Level Analysis and Synthesis to Support High Level Authoring of Multimedia Documents. Graphics Visualization and Usability Center Technical Report GVU-93-14. Georgia Institute of Technology, 1993.

[Iino, 94] M. Iino, Y.F. Day, A. Ghafoor. An Object-Oriented Model for Spatio-Temporal Synchronization of Multimedia Information. In proc. of the International Conference on Multimedia Communication and Systems, pages 110-119, 1994.

[ISO 8613] ISO 8613 Information Processing - Text and Office Systems - Office Document Architecture (ODA) and Interchange Format (ODIF), Parts 1-8, 1988.

[ISO 8824] ISO/IEC 8824 Specification of Abstract Syntax Notation One (ASN.1). Second edition, 1990.

[ISO 8825] ISO/IEC 8825 Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1). Second Edition, 1990.

[ISO 8879] ISO/IEC 8879 Text and Office Systems - standard Generalized Markup Language (SGML), 1986.

[ISO 13522] Committee Draft ISO/IEC 13522-1 Information technology - Coding of Multimedia and Hypermedia - Part 1: MHEG object Representation - Base Notation (ASN.1), October 14, 1994.

[ISO 13522-3] Committee Draft ISO/IEC 13522-3 Information technology - Coding of Multimedia and Hypermedia - Part 3: MHEG Extensions for Scripting Language Support - Extensions for Script Object Interchange, August 3, 1995.

[ISO 13522-5] Committee Draft ISO/IEC 13522-5 Information technology - Coding of Multimedia and Hypermedia - Part 5: MHEG Subset for Base Level Implementation, July 28, 1995.

[Karmouch, 93] A. Karmouch. Multimedia Distributed Cooperative System. Computer Communication 16(9):568-580, 1993.

[Klas, 90] W. Klas, E.J. Neuhold, M. Schrefl. Using an Object-Oriented Approach to Model Multimedia Data. Computer Communication 13(4):204-216. May, 1990.

[Koegel, 92] J.F. Koegel, J.L. Rutledge, J. Heines. Visual Programming Abstractions for Interactive Multimedia Presentation Authoring. In proc. IEEE International Workshop on Visual Languages, 1992.

[Little, 90] T.D.C. Little, A. Ghafoor. Synchronization and Storage Models for Multimedia Objects. IEEE Journal on Selected Areas in Communication, 8(3):413-427, April 1990.

[Meghini, 91] C. Meghini, F. Rabitti, C. Thanos. Conceptual Modeling of Multimedia Documents. Computer, 8(3):413-427, October, 1991.

[Meyer-Boudnik, 95] T. Meyer-Boudnik, W. Effelsberg. MHEG Explained. IEEE Multimedia, 2 (1): 26-38, Spring 1995.

[Murata, 89] T. Murata. Petri Nets: Properties, Analysis and Applications. Proc. of the IEEE 77 (4): 541-580. April, 1989.

[Newcomb, 1991]S. Newcomb et al. The HyTime Hypermedia/Time-based Document Structuring Language. Communications of the ACM, 34 (11), 159-166. 1991.

[Prabhakaran 93]B. Prabhakaran, S.V. Raghavan, Synchronization Models for Multimedia Presentation with User Participation. In proc. ACM Multimedia 93, pages 157-165, 1993.

[Qazi 93] N.U. Qazi, M. Woo, A. Ghafoor. A Synchronization and Communication Model for Distributed Multimedia Objects. In proc. ACM Multimedia 93, pages 147-155, 1993.

[Rangan, 92] P. V. Rangan, H.M. Vin, S. Ramanathan. Designing an On-Demand Multimedia Service. IEEE Communications Magazine 56-64, July 1992.

[Rangan, 96] P. V. Rangan, S.S. Kumar, S. Rajan. Continuity and Synchronization in MPEG. IEEE Journal on Selected Areas in Communications 14(1):52-60, 1996.

[Rutledge, 93] L. Rutledge. A HyTime Engine for Hypermedia Document Presentation. Abstract of a thesis submitted to the Faculty of the Department of Computer Science in partial fulfillment of the requirements for the degree of Master of Science in Computer Science. University of Massachusetts Lowell, 1993.

[Sample, 94] M. Sample, G. Neufeld. High-performance ASN.1 compiler. Computer Communications volume 17 number 3. March 1994.

[Schloss, 94] G.A. Schloss, M.J. Wynblatt. Building Temporal Structures in a Layered Multimedia Data Model. In proc. ACM Multimedia 94, pages 271-278, 1994.

[Sénac, 95a] P. Sénac, P. de Saqui-Sannes, R. Willrich. Hierarchical Time Stream Petri Net: a Model for Hypermedia Systems. 16th International Conference on Application and Theory of Petri Nets, Torino, June 1995. In Application and Theory of Petri Nets 1994, Lecture Notes in Computer Science no. 935, G. De Michelis and M. Diaz (Eds.), Springer, pp. 451-470.

[Sénac, 95b] P. Sénac, R. Willrich, M. Diaz. Hypermedia Synchronization Modelling: a Case Study. ED-MEDIA'95 World Conf. on Educational Multimedia and Hypermedia, pages 585-590, Graz, 1995.

[Sénac, 96a] P. Sénac, M. Diaz, A. Léger, P. de Saqui-Sannes. Modeling Logical and Temporal Synchronization in Hypermedia Systems. IEEE Journal on Selected Areas in Communications 14(1):84-103, 1996.

[Sénac, 96b] P. Sénac. Contribution à la Modélisation des Systèmes Multimédias et Hypermédias. Thèse présentée au LAAS en vue de l'obtention du titre de Docteur de l'Université Paul Sabatier. Juin, 1996.

[Shackelford, 93] D.E. Shackelford, J.B. Smith, F.D. Smith. The Architecture and Implementation of q Distributed Hypermedia Storage System. Technical Report 93-013, Department of Computer Science, The University of North Carolina at Chapel Hill, 1993.

[Steinmetz, 90] R. Steinmetz. Synchronization Properties in Multimedia Systems. IEEE Journal on Selected Areas in Communication, 8(3):401-412, 1990.

[Stotts, 90] P.D. Stotts, R. Furuta. Temporal Hyperprogramming. Journal of Visual Languages and Computing 1:237-253, 1990.

[Sun, 95] Sun Microsystems. The Java Language Specification, 1995.

[Valette, 92] R. Valette. Les Réseaux de Petri. Rapport LAAS/CNRS. Mai, 1992.

[van Rossum, 93]G. van Rossum, J. Jansen, K.S. Mullender, D.C.A. Bulterman. CMIFed: A Presentation Environment for Portable Hypermedia Documents. In proc. of the ACM Multimedia'93, ACM Press, New York, pp. 183-188, 1993.

[Vazirgiannis, 93]M. Vazirgiannis, M. Hatzopoulos. A Script Based Approach for Interactive Multimedia Applications. In proc. of Multimedia Modelling'93, pp. 129-143, Singapore, November, 1993.

[Yang, 96] C.-C. Yang, J.-H. Huang. A Multimedia Synchronization Model and Its Implementation in Transport Protocols. IEEE Journal on Selected Areas in Communications 14(1): 212-225, 1996.

[Walter, 83] B. Walter. Timed Petri nets for modeling and analyzing protocols with time. In Proc. of the IFIP Conf. on Protocol Specification, Testing and Verification, III, North Holland, H. Rudin & C. West Editors, 1983.

[Wang, 95] H.K. Wang, J.-L. C. Wu. Interactive Hypermedia Applications: A Model and Its Implementation. Software-Practice and Experience 25(9):1045-1063, 1995.

[Wahl, 94] T. Wahl, K. Rothermel. Representing Time in Multimedia Systems. International Conference on Multimedia Computing and Systems (ICMCS'94), pp. 538-543, Boston, 1994.

[Willrich, 94] R. Willrich, P. Sénac, M. Diaz. Une approche formelle pour la spécification et la conception de documents hypermédias. Rapport LAAS N94481. Contrat CNET FT N92.1B.178. Projet CESAME. Lot 2, Décembre 1994.

[Willrich, 96a] R. Willrich, P. Sénac, M. Diaz, P. de Saqui-Sannes. A Formal Framework for the Specification, Analysis and Generation of Standardized Hypermedia Documents. A apparaître au Third IEEE International Conference on Multimedia Computing and Systems (ICMCS'96), Hiroshima (Japon), 17-21 Juin 1996.

[Willrich, 96b] R. Willrich, P. de Saqui-Sannes, P. Sénac, M. Diaz. Une approche formelle pour le développement de documents hypermédias normalisés. A apparaître au Colloque Francophone sur l'Ingénierie des Protocoles (CFIP'96), Rabat (Maroc), 14-17 octobre 1996.

[Willrich, 96c] R. Willrich, P. de Saqui-Sannes, P. Sénac, M. Diaz. Hypermedia Document Design Using the HTSPN Model. A apparaître au Third International Conference on Multimedia Modeling (MMM'96), Toulouse (France), le 12-15 Novembre 1996.

[Willrich, 96d] R. Willrich, P. de Saqui-Sannes, P. Sénac, M. Diaz. Towards Hypermedia Documents Design. XIV Brazilian Symposium on Computer Networks (SBRC'96), pages 473-491, Fortaleza (Brésil), 1996.

[Wynblatt, 95] M. Wynblatt. Position Statement on Multimedia Synchronisation. In Proc of the IEEE Workshop on Multimedia Synchronization (Sync'95) (In URL: http//spiderman.bu.edu/sync95/sync95.html). Tysons Corner, Virginia. May, 1995.

[Zuberek, 85] W.M. Zuberek. M-Timed Petri nets, Priorities, Preemptions and Performance Evaluation of Systems». Advances in Petri Nets 1985, LNCS 222, Springer-Verlag, 1995.

Annexe A

Les classes du modèle RdPHFT-I

Cette annexe présente la notation C++ simplifiée des classes qui définissent les informations sur les données et leurs caractéristiques de présentation. Ces classes ont été introduites dans la section 4.1 de ce mémoire.

A.1 Les Classes *Données*

```
class Description
{
    public:
        char                    *nom;
        char                    *propriétaire;
        char                    *version;
};
```

enum **Codage-Texte** {ISO_646_ISO_6429, ISO_8613_parts6, ITU_TS_rec_T101_Videotex, ISO_6937, ISO_8859, ISO_10646};

enum **Codage-Image** {ITU_rec_T4, MR_ITU_TS_rec_T6, MMR_ITU_TS_rec_T6, MH_ITU_-TS_rec_T6, ISO_10918_JPEG, ISO_10918_JPEG_JFIF, ISO_10918_JPEG_T101_annexF, ISO_11544_JBIG, ISO_8613_part7};

enum **Codage-Graphique** {ISO_8632_CGM, ISO_9282_CGI, ISO_8613_part8, ITU_TS_rec_T1010annexB};

enum **Codage-Audio** {pcm_linear, ITU_TS_rec_G711u_law, ITU_TS_rec_G711Alaw, ITU_TS_rec_G721, ITU_TS_-rec_G722_48Kbps, ITU_TS_rec_G722_56Kbps, ITU_TS_rec_G722_64Kbps, ITU_TS_rec_G723_24Kbps, ITU_TS_-rec_G723_40Kbps, ITU_TS_rec_G728_LD_CELP_16Kbps, ITU_TS_rec_J41, ITU_TS_rec_J42, ISO_11172_-MPEG_Audio, ISO_8613_part9};

enum **Codage-Vidéo** {ISO_11172_MPEG_Video, ITU_TS_rec_H261};

enum **Codage-Audio-Visuel** {ISO_11172_MPEG_System,ITU_TS_rec_H221};

enum **Codage-Données-Non-Média** {nf-Z62040-RAVI, smsl, scriptX, toolbook, hypertalk, supertalk, c, cpp, pascal, lisp, prolog, smalltalk};

enum **Codage-Script** {nf-Z62040-RAVI, smsl, scriptX, toolbook, hypertalk, supertalk, c, cpp, pascal, lisp, prolog, smalltalk};

enum **Inclusion** {implicite, explicite};

struct **Range** {min, max};

struct **Coord** {int x; int y; int z;};

```
class Objet-Données {
    public:
        Description             *description;
        char                    *contenu;
```

```
        Inclusion                       inclusion;
        int                             *Codage-Propriétaire;
        (...)
};

class CaractDynamiquesOriginales {
    public:
        int                             *Durée-naturelle;
        int                             *GTR-initial;
        (...)
};

class CaractSpatialesOriginales {
    public:
        Coord                           *taille-Originale;
        (...)
};

class Caract-Audibles-Originales : public CaractDynamiquesOriginales {
    public:
        int                             *volume-Original;
        Range                           *range-Volume;
        (...)
};

class CaractDynamiquesSpatialesOriginales : public CaractDynamiquesOriginales, Caract-Audibles-Originales {
    public:
        (...)
};

class CaractDynamiquesSpatialesAudiblesOriginales : public CaractDynamiquesOriginales, CaractSpatialesOriginales,
Caract-Audibles-Originales {
    public:
        (...)
};

class Texte : public Objet-Données {
    public:
        Codage-Texte                    *codage;
        CaractSpatialesOriginales       *caractOriginalesdePrésentation;
        (...)
};

class Graphique : public Objet-Données {
    public:
        CodageGraphique                 *codage;
        CaractSpatialesOriginales       *caractOriginalesdePrésentation;
        (...)
};

class Image : public Objet-Données {
    public:
        CodageImage                     *codage;
        CaractSpatialesOriginales       *caractOriginalesdePrésentation;
        (...) };

class Audio : public Objet-Données {
    public:
        Codage-Audio                    *code;
        Caract-Audibles-Originales      *caractOriginalesdePrésentation;
        (...)
};

class Vidéo : public Objet-Données {
    public:
        Codage-Vidéo                                    *codage;
        CaractDynamiquesSpatialesAudiblesOriginales     *caractOriginalesdePrésentation;
```

```
        (...)
};

class Audio-Visuel : public Objet-Données {
    public:
        CodageAudioVisuel                                   *codage;
        CaractDynamiquesSpatialesAudiblesOriginales         *caractOriginalesdePrésentation;
        (...)
};

class Non-Média : public Objet-Données {
    public:
        Codage-Données-Non-Média            *codage;
        (...)
};

class Script : public Objet-Données {
    public:
        Codage-Script                       *code;
        (...)
};
```

A.2 Les Classes *Canal*

```
enum Canal-Coté {gauche, droite};

class Canal-Base {
    private:
        int                                 *generic-Temporal-Ratio;
    public:
        (...)
};

class Canal-Visuel : virtual public Canal-Base {
    private:
        Range                               *intervalle-Axe-X;
        Range                               *intervalle-Axe-Y;
        Range                               *intervalle-Axe-Z;
    public:
        (...)
};

class Canal-Audio : virtual public Canal-Base {
    private:
        Range                               *volume-Range;
        Canal-Coté                          *coté;
    public:
        (...)
};

class Canal-Audio-Visuel :  public Canal-Audio, public Canal-Visuel {
    public:
        (...)
};
```

A.3 Les Classes *Présentation*

```
Enum Type-Présentation {texte, image, graphique, audio, vidéo, audioVisuel, non-Média, script, interaction};

enum Terminaison-Temporelle {gel, stop, présentation-Alternative};

enum Orientation-Menu {horizontal, vertical};

enum Méthode-Flexibilité {rigide, répétition-Interruption, variation-Vitesse};

class Présentation-Alternative {
    public:
```

```
        int                         *time-Out;
        Présentation                *présentation;
        (...)};
class Spéc-Présentation {
    public:
        Description                 *description;
        Canal                       *canal;
        int                         *priorité;
        Présentation-Alternative    *prés-Alternative;
        (...)
};
class SP-Spatial : virtual public Spéc-Présentation {
    public:
        Coord                       *position-Présentation;
        int                         *taille-Présentation;
        (...)
};
class SP-Temporel : virtual public Spéc-Présentation {
    public:
        int                         *position-Départ;
        int                         *position-Fin;
        int                         *vitesse;
        int                         *nombre-de-répétition;
        Méthode-Flexibilité         *flexibilité;
        Terminaison-Temporelle      *termination;
        (...)
};
class SP-Texte : public SP-Spatial {
    public:
        Texte                       *Objet-Données;
        (...)
};
class SP-Image : public SP-Spatial {
    public:
        Image                       *Objet-Données;
        (...)
};
class SP-Graphique : public SP-Spatial {
    public:
        Graphique                   *Objet-Données;
        (...)
};
class SP-Audio : virtual public SP-Temporel {
    public:
        Audio                       *Objet-Données;
        int                         *volume;
        (...)
};
class SP-Vidéo : virtual public SP-Temporel, public SP-Spatial {
    public:
        Vidéo                       *Objet-Données;
        (...)
};
class SP-Audio-Visuel : public SP-Audio, public SP-Vidéo {
    public:
        AudioVisuel                 *Objet-Données;
        (...)
};
class SP-Non-Média : public Spéc-Présentation {
```

```
    public:
        Non-Média                   *Objet-Données;
        (...)
};

class SP-Script : public Spéc-Présentation {
    public:
        Script                      *Objet-Données;
        (...)
};

class SP-Interaction : public SP-Spatial {
    public:
        Orientation-Menu            *orientation;
        (...)
};

class Présentation: {
    public:
        Type-Présentation           type;
        union
        {
            SP-Texte        texte;
            SP-Image        image;
            SP-Graphique    graphique;
            SP-Audio        audio;
            SP-Vidéo        vidéo;
            SP-Audio-Visuel audioVisuel;
            SP-Non-Média    non-Média;
            SP-Script       script;
            SP-Interaction  interaction;
        };
        (...)
};
```

Annexe B

L'interpréteur Java RdPHFT-I

Le but de cette annexe est de décrire l'interpréteur de spécification RdPHFT-I réalisé en Java. Cet interpréteur est une application Java qui présente des documents hypermédias spécifiés par un réseau RdPHFT-I.

La notation utilisée est basée sur SA/RT, dans laquelle les cercles représentent des processus fonctionnels, les boîtes représentent des terminaisons (ou processus extérieurs dont le développement n'entre pas dans le cadre de l'outil spécifié). Les arcs orientés en trait plein représentent des flots de données, en trait pointillé des flots de contrôle. Une barre en trait épais verticale ou horizontale représente un processus de contrôle, et enfin deux traits parallèles pleins représentent un réservoir. De plus, afin de décrire les activités plus complexes, nous utilisons un pseudo-langage. La section B.4 présente le dictionnaire des données utilisées par cette annexe.

La figure B.1 représente la spécification de niveau 0 de l'interpréteur RdPHFT-I. L'entrée de l'interpréteur est un fichier contenant la spécification RdPHFT-I (*Fichier RdPHFT-I*) du document hypermédia. Les opérations de commande de l'interpréteur sont les commandes d'interprétation (*cmdInterpréteur*) et les interactions de l'utilisateur lors de la présentation du document (*interactions*). Les commandes d'interprétation sont disponibles pour l'utilisateur via le menu principal; ils sont : *lecture* (de la spécification), *départ* et l'*arrêt* (de l'interprétation du document). Les données primitives (*donnéesPrimitives*) utilisées par le document sont récupérées, via le réseau, à partir de la spécification de leurs URL.

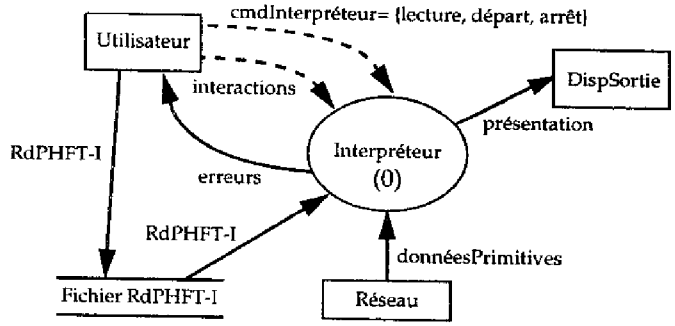


Figure B.1 Spécification de niveau 0 de l'interpréteur RdPHFT-I

Le *Fichier RdPHFT-I* est généré automatiquement par l'application Java responsable de l'édition graphique de la spécification RdPHFT-I. Ensuite, nous présentons le format du fichier d'entrée de l'interpré-

teur (*Fichier RdPHFT-I*) et le module Interpréteur (0).

B.1 Format du fichier d'entrée de l'interpréteur RdPHFT-I

Cette section présente la grammaire adoptée pour la spécification RdPHFT-I des documents hypermédias.

B.1.1 Description de la grammaire

X --> I,J,K	signifie que X est défini comme une séquence des attributs I suivit par J et K.
X?	signifie que zero ou une instance de X peut arriver.
X+	signifie qu'une ou plusieurs instances de X peut arriver.
X*	signifie que n'importe quel nombre d'instances de X peut arriver.
(I | J | K)	signifie qu'un de l'ensemble doit arriver.

B.1.2 Format du fichier RdPHFT-I

RdPHFT-I --> Description?, DS*, TSPN*, Root

Description --> «Description {», RdPHFTName?, Dimension?, Background?, Foreground?, «};»
RdPHFTName --> «name {», character+, «}»
Dimension --> «size [», Width, Height, «]»
Width --> numeric.
Height --> numeric.
Background --> «background», Color.
Foreground --> «foreground», Color.
Color --> («black» | «blue» | «cyan» | «darkGray» | «gray» | «green» | «lightGray» | «magenta» | «orange» | «pink» | «red» | «white» | «yellow»)

DS --> «DS {», Data+, «};»
Data --> «Data», Name, DataSpecification
Name --> character+
DataSpecification --> «{», URLSpec, Code, Duration?, «};»
URLSpec --> «URL», character+
Duration --> «duration», numeric
Code --> «code», character+

TSPN --> «TSPN», Name, TSPNAttributs
TSPNAttributs --> «{», Place+, Transition+, Arc+, Pin, Tout?, «};»

Place --> «Place», Name, PlaceAttributs.
PlaceAttributs --> «{», EdPosition?, PlaceType, Media?, PS?, AlternativePres?, S?, «};»
EdPosition --> «position [«, X, Y, «]».
X --> numeric.
Y --> numeric.
PlaceType --> «type», («atomic» | «link» | «composite»).
Media --> «media», («image» | «audio» | «text» | «graphicAnimation»)
PS --> «PS {», (PSImage | PSAudio | PSText | PSGraphicAnimation | PSLink), «};»
PSImage --> «{», DataSpec, Presentation_position?, Dimension?, «};»
DataSpec --> «data», Name
Presentation_position --> «presentation_position [», X, Y, «]»
PSAudio --> «{», DataSpec, Termination? «};»
PSText --> «{», DataSpec, TextType, Font?, Dimension?, Background?, Foreground?, «};»
TextType --> «text_type», (area | field).
Font --> «font {», TextName, Style, TextSize, «}»
TextName --> (dialog | helvetica | timesRoman | courier | symbol)

Style --> (bold | italic | plain).
TextSize --> Numeric.
PSGraphicAnimation --> «{», DataSpec, Presentation_position?, Dimension?,Start_position?, End_position?, Speed?, Termination?, «};»
Start_position --> «start_position», numeric.
End_position --> «end_position», numeric.
Speed --> «speed», numeric.
Termination -> «termination», («freeze» | «stop» | «alternative_presentation»)
PSLink --> «{», Presentation_position?, Foreground?, Background?, Font? «};»
AlternativePres --> «alternative_presentation {», Timeout, Media, (PSImage | PSText | PSAudio), «};»
Timeout --> «timeout», numeric.
S --> «S», Name

Transition --> «Transition», Name, TransitionAttributs.
TransitionAttributs --> «{», EdPosition?, TransitionType, «};»
TransitionType --> «type», («AND» | «WEAK-AND» | «OR» | «STRONG-OR» | «MASTER» | «OR-MASTER» | «AND-MASTER» | «STRONG-MASTER» | «WEAK-MASTER»).

Arc --> «Arc», Direction, ArcAttributs.
Direction --> «(», Name, «->», Name, «)».
ArcAttributs --> «{», TVI, Master?, Label?, «};»
TVI --> «TVI [», numeric, numeric, numeric, «]»
Master --> «master»
Label --> «label», Name

Pin --> «Pin {», Name+, «};»
Tout --> «Tout {», Name, «};»

Root --> «Root {», Place+, Transition+, Arc+, Pin, Tout, «};»

B.1.3 Exemple

Afin d'illustrer le format du fichier d'entrée de l'interpréteur RdPHFT-I, ci-dessous nous présentons la spécification du réseau racine du module de descente du train d'atterrissage (figure 4.14) présenté dans la section 4.6.

```
Description {
  name          { Module VACBI }
  dimension     [ 700 500 ]
  background    pink
  foreground    black
};
DS {
  Data AU1 { URL file:/LAAS/feydeau2/users/willrich/vacbi/AUDIO/ouverture.au };
  Data AUOP { URL file:/LAAS/feydeau2/users/willrich/vacbi/AUDIO/pop.au };
  Data IMG1 { URL file:/LAAS/feydeau2/users/willrich/vacbi/IMAGES/avion.gif };
  Data TX1 { URL file:/LAAS/feydeau2/users/willrich/vacbi/TEXTES/cours.txt };
  Data IN1 { URL file:/LAAS/feydeau2/users/willrich/vacbi/TEXTES/sdebut.txt };
  Data IMG_LDG1 { URL file:/LAAS/feydeau2/users/willrich/vacbi/IMAGES/LDG1.gif };
  Data IMG_W1 { URL file:/LAAS/feydeau2/users/willrich/vacbi/IMAGES/porte0.gif };
};
Root VACBI {
  Place Début { type atomic };
  Place Départ {
    type        link
    PS {
      presentation_position     [ 100 420 ]
      foreground                black
      background                red
      font                      { dialog plain 18 }
    };
```

```
};
Place AU1 {
  type          atomic
  media         audio
  PS { data AU1 };
};
Place Fin1 {
  type          link
  PS {
    presentation_position      [ 200 420 ]
    foreground                 black
    background                 red
    font                       { dialog plain 18 }
  };
};
Place Fin2 {
  type link
  PS {
    presentation_position      [ 220 420 ]
    foreground                 black
    background                 red
    font                       { dialog plain 18 }
  };
};
Place Recommencer {
  type          link
  PS {
    presentation_position      [ 80 420 ]
    foreground                 black
    background                 red
    font                       { dialog plain 18 }
  };
};
Place TX1 {
  type          atomic
  media         text
  PS {
    data                       TX1
    text_type                  field
    font                       { timesRoman bold 18 }
    foreground                 green
    background                 white
    presentation_position      [ 340 60 ]
    size                       [ 325 35 ]
  };
};
Place IN1 {
  type          atomic
  media         text
  PS {
    data                       IN1
    text_type                  field
    font                       { timesRoman bold 18 }
    foreground                 blue
    background                 white
    presentation_position      [ 340 120 ]
    size                       [ 325 35 ]
  };
};
Place IMG_LDG1 {
  type          atomic
  media         image
  PS {
    data                       IMG_LDG1
```

```
    presentation_position        [ 65 30 ]
  };
};
Place IMG_W1 {
  type          atomic
  media         image
  PS {
    data                         IMG_W1
    presentation_position        [ 50 180 ]
  };
};
Place IMG1 {
  type          atomic
  media         image
  PS {
    data                         IMG1
    presentation_position        [ 340 175 ]
  };
};
Place Procedure#2 {
  type          composite
  S             Procédure
};
Place Fin { type atomic };
Transition t0 { type AND };
Transition t1 { type MASTER };
Transition t2 { type MASTER };
Transition t3 { type MASTER };
Transition t4 { type MASTER };
Arc ( Début -> t0 ) { TVI [ 0 0 0 ] };
Arc ( t0 -> AU1 ) { };
Arc ( t0 -> IMG_LDG1 ) { };
Arc ( t0 -> IMG1 ) { };
Arc ( t0 -> Départ ) { };
Arc ( t0 -> TX1 ) { };
Arc ( t0 -> IN1 ) { };
Arc ( t0 -> IMG_W1 ) { };
Arc ( t0 -> Fin1 ) { };
Arc ( IMG1 -> t1 ) { TVI [ I I I ] };
Arc ( IMG_LDG1 -> t1 ) { TVI [ I I I ] };
Arc ( IMG_W1 -> t1 ) { TVI [ I I I ] };
Arc ( TX1 -> t1 ) { TVI [ I I I ] };
Arc ( IN1 -> t1 ) { TVI [ I I I ] };
Arc ( AU1 -> t1 ) { TVI [ 3920 4900 5880 ] };
Arc ( Départ -> t1 ) {
  TVI           [ 0 * 20000 ]
  master
  Label         Début };
Arc ( Fin1 -> t1 ) { TVI [ 0 * I ] };
Arc ( IMG1 -> t2 ) { TVI [ I I I ] };
Arc ( TX1 -> t2 ) { TVI [ I I I ] };
Arc ( Départ -> t2 ) { TVI [ 0 * I ] };
Arc ( AU1 -> t2 ) { TVI [ 3920 4900 5880 ] };
Arc ( IN1 -> t2 ) { TVI [ I I I ] };
Arc ( IMG_LDG1 -> t2 ) { TVI [ I I I ] };
Arc ( IMG_W1 -> t2 ) { TVI [ I I I ] };
Arc ( Fin1 -> t2 ) {
  TVI           [ 0 * I ]
  master
  Label         Fin };
Arc ( t1 -> Fin2 ) { };
Arc ( t1 -> Recommencer ) { };
Arc ( t1 -> Procédure#2 ) { };
Arc ( Fin2 -> t4 ) {
```

```
    TVI             [ 0 * I ]
    master
    Label           Fin };
  Arc ( Recommencer -> t4 ) { TVI [ 0 * I ] };
  Arc ( Procédure#2 -> t4 ) { TVI [ I I I ] };
  Arc ( Recommencer -> t3 ) {
    TVI             [ 0 * I ]
    master
    Label           Recommencer };
  Arc ( Procédure#2 -> t3 ) { TVI [ I I I ] };
  Arc ( Fin2 -> t3 ) { TVI [ 0 * I ] };
  Arc ( t2 -> Fin ) { };
  Arc ( t3 -> Début ) { };
  Arc ( t4 -> Fin ) { };
  Arc ( Fin -> t5 ) { TVI [ 0 0 0 ] };
  Pin { Début };
};
```

B.2 L'interpréteur RdPHFT-I

Les principaux modules de l'interpréteur RdPHFT-I sont les suivants (figure B.2) :

- Module de lecture, qui lors de la demande de l'opération de *lecture*, réalise la lecture du fichier contenant la spécification *RdPHFT-I*.

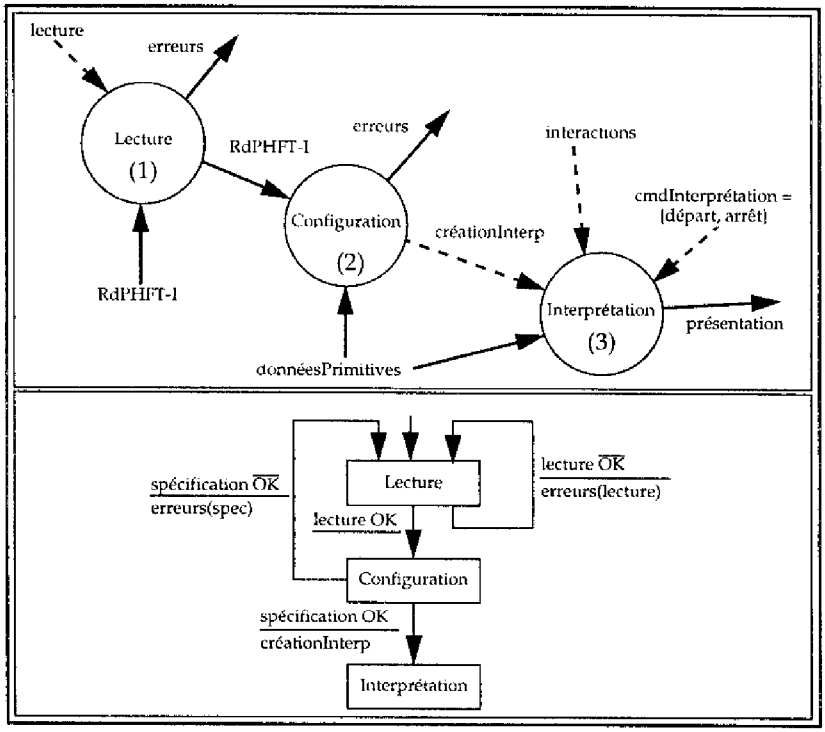


Figure B.2 Spécification de niveau 1 de l'interpréteur RdPHFT-I

- Module de configuration, qui à partir de la spécification RdPHFT-I crée une structure d'interprétation associée (*créationInterp*), c.-à-d. qu'il crée le module Interprétation.
  Le module de configuration réalise une analyse de la spécification au niveau de la description des objets Données (SD) et leurs caractéristiques de présentation (SP). Ce module réalise aussi la lecture des données primitives utilisées par le document (*donnéesPrimitives*). Cette lecture n'est pas bloquante, c.-à-d. qu'elle est réalisée par un processus indépendant des autres activités (incluant les processus d'interprétation de la spécification).

- Module d'interprétation, qui réalise l'interprétation (présentation) de la spécification.

Les activités réalisées par le module Interprétation (figure B.2) sont supportées par des processus qui sont créés et interconnectés par le module Configuration de façon dynamique. Cette structure d'interprétation est dépendante de la spécification RdPHFT-I. Le bloc élémentaire de construction du module Interprétation est le module Page (figure B.3). Le module Page est une représentation abstraite qui représente un ensemble d'activités responsable de l'interprétation d'une page de la spécification RdPHFT-I. Une page est un sous-réseau représenté par une place Composite (la racine de la spécification RdPHFT-I est toujours associé à une place Composite abstraite).

Le module Page (3.1) reçoit des commandes d'interprétation de sa page mère (*cmdInterprétation*) et reçoit aussi des commandes d'interaction (*interactions*) de l'utilisateur. Si la page représentée est la racine de la spécification, alors elle reçoit les commandes d'interprétation demandées par l'utilisateur (via le menu principal). Les données d'entrée sont les informations sur les places de ses pages filles (*placeIf*). Les données de sortie sont les informations sur la propre Page (une place Composite). Ces informations sont envoyées lors du changement d'état d'une place (*placeIf*). Le module Page envoie aussi des commandes d'interprétation à ses pages filles (*cmdInterprétation*).

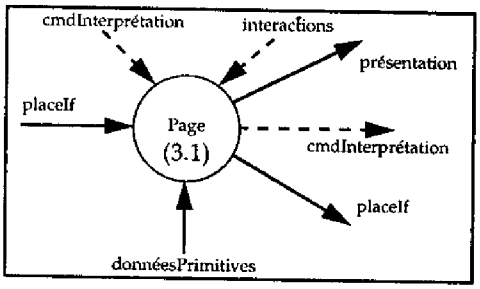


Figure B.3 Bloc élémentaire utilisé pour la construction d'une structure d'interprétation

La figure B.4 présente un exemple de structure nécessaire à l'interprétation d'une spécification RdPHFT-I dont la racine contient une place Composite (Page 1) qui contient elle-même une autre place Composite (Page 1.1).

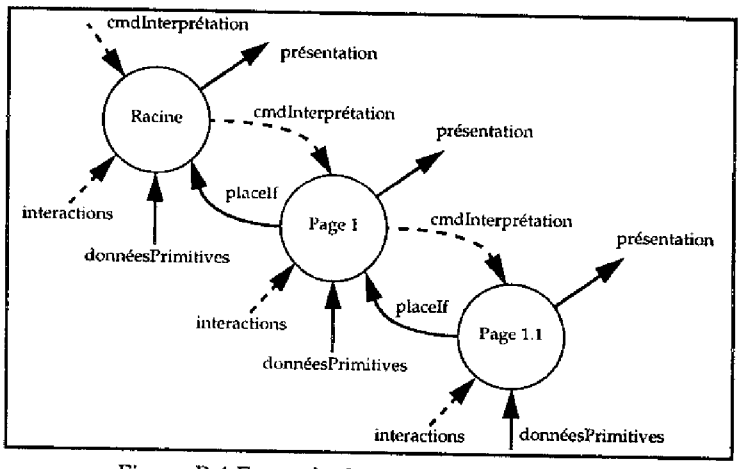


Figure B.4 Exemple de module Interprétation

La figure B.5 présente un exemple de l'ensemble d'activités nécessaire à l'interprétation d'un sous-réseau spécifiant un scénario contenant un texte (ou image), un audio, un bouton et une animation graphique. Cet ensemble d'activités compose le module Page.

Les activités d'interprétation d'une page englobe les activités liées aux présentations modélisées par les

places (atomiques, lien ou composites) de la page et leur ordonnancement (défini par le marquage et les transitions). Un contrôle de la validité temporelle est réalisé sur ces activités. L'ordonnancement de la page est réalisé par le module Ordonnanceur (3.1.1). Les présentations sont supportées par des instances des modules (activités) PrésentationTexteImage (3.1.2), PrésentationAudio (3.1.3), Présentation-Bouton (3.1.4) et PrésentationAnimation-Graphique (3.1.5) :

- Chaque présentation d'un texte ou d'une image (modélisée par une place atomique) est supportée par une instance du module PrésentationTexteImage.
- Chaque présentation d'une séquence audio (modélisée par une place atomique) est supportée par une instance du module PrésentationAudio.
- Chaque présentation d'une animation graphique[1] (modélisée par une place atomique) est supportée par une instance du module PrésentationAnimationGraphique.
- Chaque présentation d'un lien (modélisée par une place lien) est supportée par une instance du module PrésentationBouton.
- Chaque présentation d'un scénario multimédia (modélisée par une place composite) est supportée par une instance du module Page (une définition récursive).

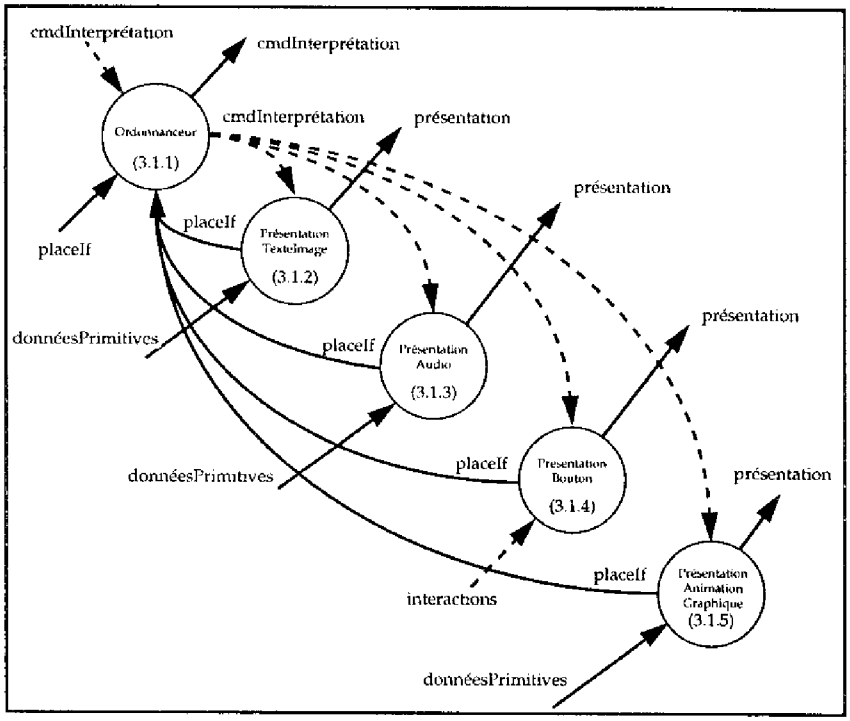


Figure B.5 Le module Page

B.2.1 Module Ordonnanceur

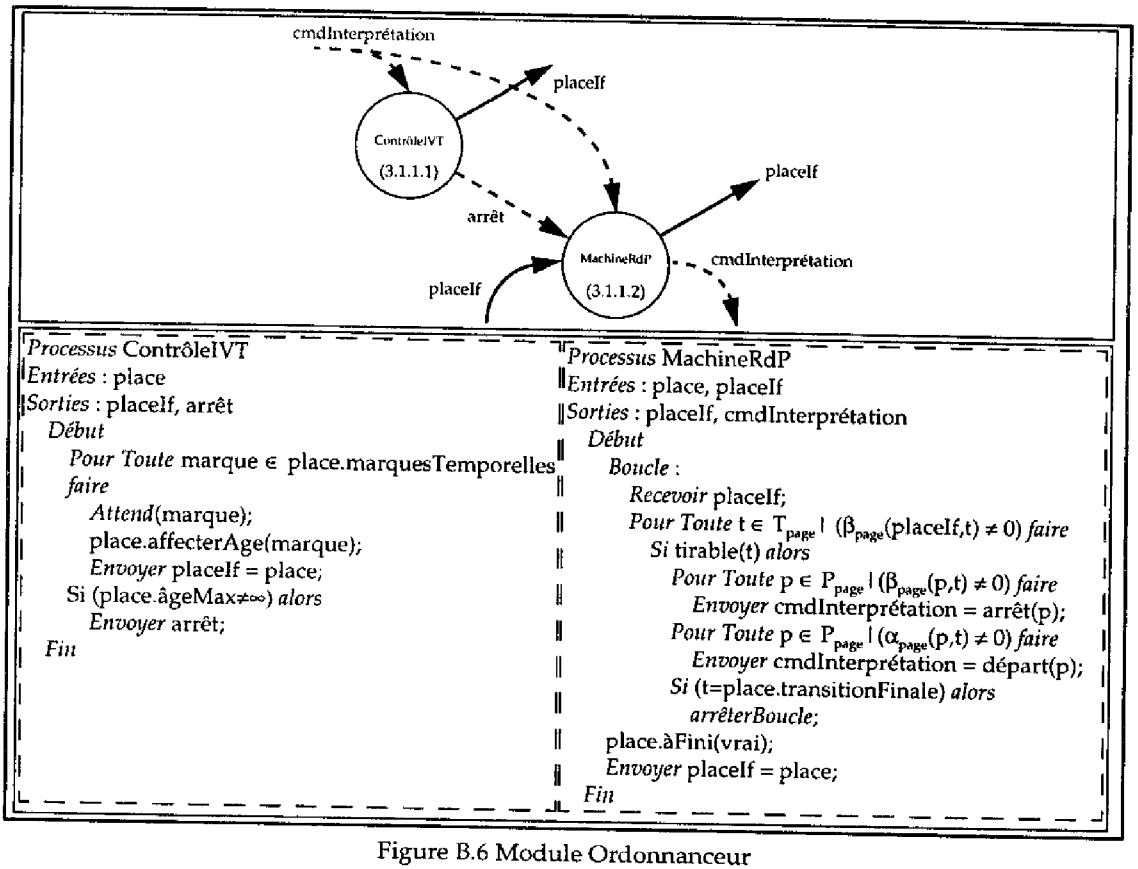
Ce module, raffiné dans la figure B.6, est responsable de l'ordonnancement des présentations modélisées par les places de la page. Le module Ordonnanceur est composé des modules ContrôleIVT (3.1.1.1) et MachineRdP (3.1.1.2) :

- Le module ContrôleIVT est responsable du contrôle de la validité temporelle d'une place de la page.

1. La version du paquetage graphique de Java utilisée ne fournit pas directement les moyens de présentation d'une séquence vidéo. Ainsi, nous avons définit une séquence vidéo comme une animation graphique des images qui composent une séquence vidéo.

Dans la figure B.6, ce module contrôle la validité temporelle d'une place composite. Il identifie l'arrivée de l'âge de traitement aux dates minimales et maximales définies par l'intervalle de validité temporelle (x et y) des arcs sortants de cette place (si ces durées sont différentes d'infinie). Quand l'un de ces points temporels est atteint, ce processus envoie les informations sur la place à sa page mère (à son module ordonnanceur). Enfin, quand la durée maximale est atteinte, ce module demande l'*arrêt* du traitement associé à la place (la MachineRdP).

- Le module MachineRdP est responsable de l'évolution du sous-réseau. Ainsi, il est responsable du tir d'une transition sensibilisée, c.-à-d. l'*arrêt* des présentations d'entrée de ces transitions et le *départ* des présentations de sortie. La vérification des transitions tirables est réalisée lors du changement d'*état* de l'un des composants de la page. Quand la place de sortie du sous-réseau devient marquée, le module MachineRdP envoie les informations sur la place Composite à sa page mère.



Processus ContrôleIVT
Entrées : place
Sorties : placeIf, arrêt
Début
Pour Toute marque ∈ place.marquesTemporelles
faire
Attend(marque);
place.affecterAge(marque);
Envoyer placeIf = place;
Si (place.âgeMax≠∞) *alors*
Envoyer arrêt;
Fin

Processus MachineRdP
Entrées : place, placeIf
Sorties : placeIf, cmdInterprétation
Début
Boucle :
Recevoir placeIf;
Pour Toute t ∈ T_{page} | (β_{page}(placeIf,t) ≠ 0) *faire*
Si tirable(t) *alors*
Pour Toute p ∈ P_{page} | (β_{page}(p,t) ≠ 0) *faire*
Envoyer cmdInterprétation = arrêt(p);
Pour Toute p ∈ P_{page} | (α_{page}(p,t) ≠ 0) *faire*
Envoyer cmdInterprétation = départ(p);
Si (t=place.transitionFinale) *alors*
arrêterBoucle;
place.àFini(vrai);
Envoyer placeIf = place;
Fin

Figure B.6 Module Ordonnanceur

B.2.2 Module PrésentationTexteImage

Ce module, raffiné dans la figure B.7, définit les processus nécessaires à la présentation d'un média du type Texte ou Image (représentée par une place atomique de la page). Il est composé des modules ContrôleIVT (3.1.2.1), ComposantTexteImage (3.1.2.2) et PrésentationAlternative (3.1.2.3) :

- Le module ContrôleIVT est identique à celui du module Ordonnanceur (section B.2.1).
- Le module ComposantTexteImage est dérivé du type *Component* du paquetage AWT de Java. Ce module permet la présentation des images et des informations textuelles.
- Le module PrésentationAlternative est responsable du changement des données primitives à présenter si l'information initiale n'a pas pu être totalement chargée après un certain délai.

Une présentation textuelle ou d'une image finit à l'instant zéro (composant statique). Alors un change-

ment d'état arrive seulement aux instants x et y des IVT de la place (identifiés par le module ContrôleIVT).

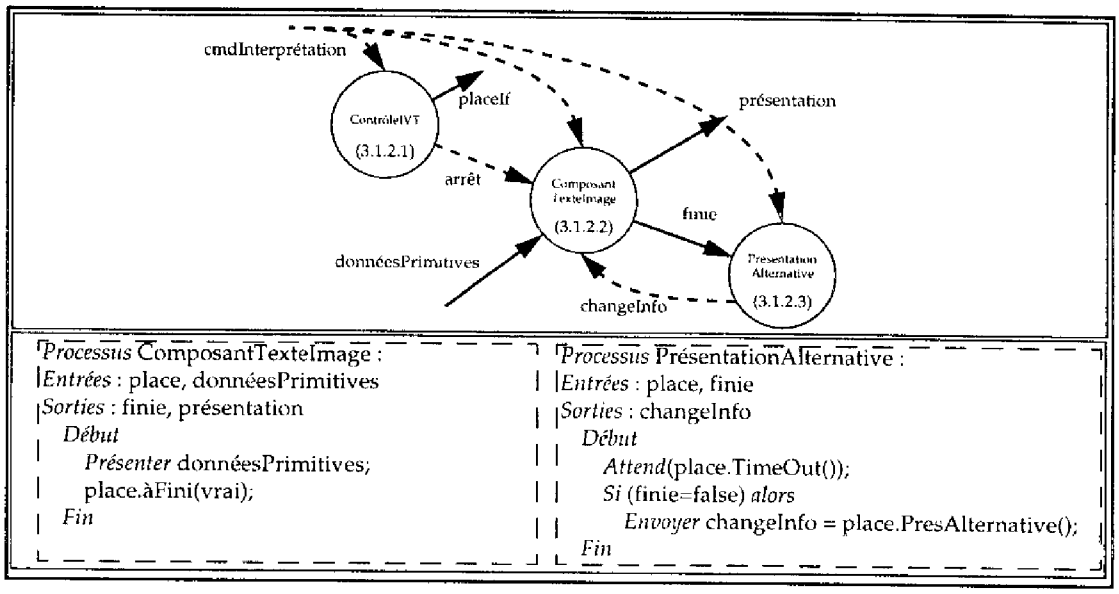


Figure B.7 ModulePrésentationTexteImage

B.2.3 Module PrésentationAudio

Ce module, raffiné dans la figure B.8, définit les processus nécessaires à la présentation d'un média du type Audio (représentée par une place atomique de la page). Il est composé des modules ContrôleIVT (3.1.3.1), Composant Audio (3.1.3.2), DétectionFin (3.1.3.3) et ContrôleIVT (3.1.3.4) :

- Le module ComposantAudio utilise le paquetage SUN afin de présenter des séquences audio. S'il arrive une erreur d'accès au fichier audio, la présentation alternative (si elle existe) est lancée.
- Le module DétectionFin permet la détection de la fin de présentation de la séquence audio. A ce moment, ce module transmet l'état du composant au module Ordonnanceur. Ce module est responsable aussi de l'affichage de la présentation de terminaison. La séparation entre le processus de présentation (ComposantAudio) et celui de détection de la fin de la présentation (DétectionFin) a été nécessaire à cause des limitations du langage Java à ce niveau.
- Le module PrésentationAlternative est responsable du changement des données primitives à présenter si l'information initiale n'a pas pu être totalement chargée après un certain délai.
- Le module ContrôleIVT est identique à celui du module Ordonnanceur (section B.2.1).

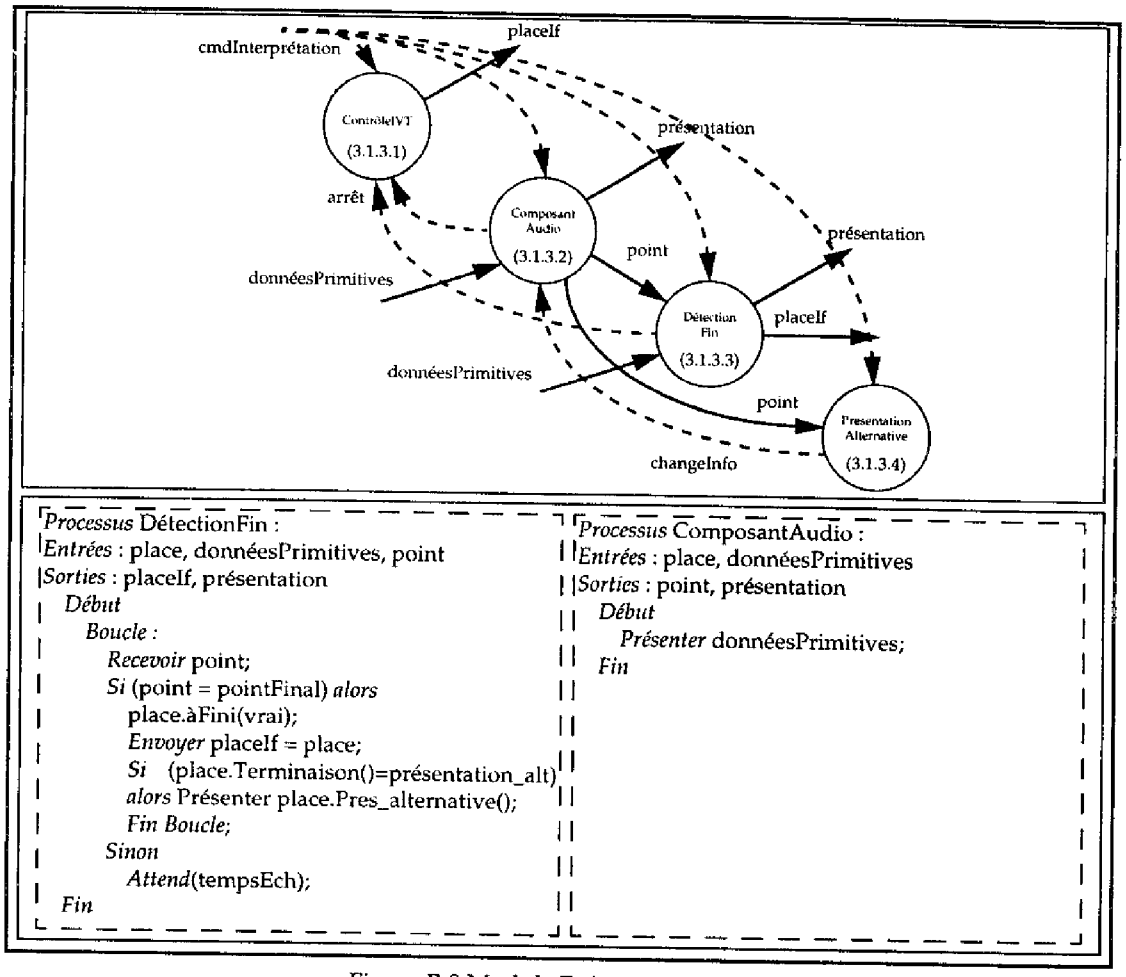


Figure B.8 Module PrésentationAudio

B.2.4 Module PrésentationBouton

Ce module, raffiné dans la figure B.9, définit les processus nécessaires à la présentation d'un bouton (représentée par une place lien de la page). Il est composé des modules ContrôleIVT (3.1.4.1) et ComposantBouton (3.1.4.2) :

- Le module ContrôleIVT est identique à celui du module Ordonnanceur (section B.2.1).
- Le module ComposantBouton est dérivé de la classe *Bouton* du paquetage AWT de Java. Il permet la présentation des boutons et donne les supports nécessaires aux *interactions* avec l'utilisateur. Quand l'utilisateur clique sur le bouton, ce module transmet l'*état* du composant au module ordonnanceur.

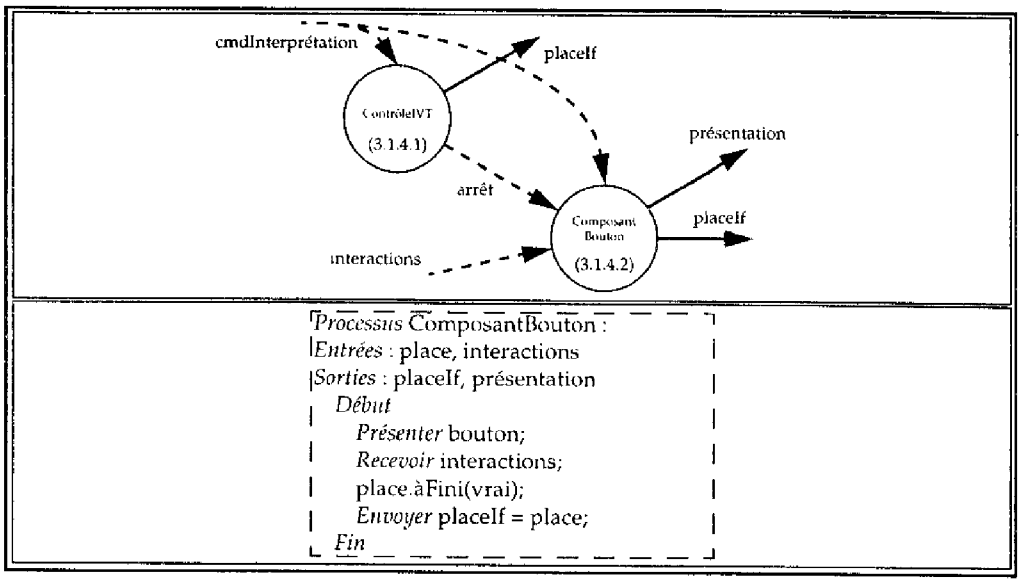


Figure B.9 Module PrésentationBouton

B.2.5 Module PrésentationAnimationGraphique

Ce module, raffiné dans la figure B.10, définit les processus nécessaires à la présentation d'une animation graphique. Il est composé des modules ContrôleIVT (3.1.5.1), ComposantAnimationGraphique (3.1.5.2), et PrésentationAlternative (3.1.5.3) :

- Le module ContrôleIVT est identique à celui du module Ordonnanceur (section B.2.1).
- Le module ComposantAnimationGraphique permet la présentation des animations graphiques. Quand l'animation graphique finit, ce module transmet l'*état* du composant au module ordonnanceur.
- Le module PrésentationAlternative est responsable du changement des données primitives à présenter si l'information initiale n'a pas pu être totalement chargée après un certain délai.

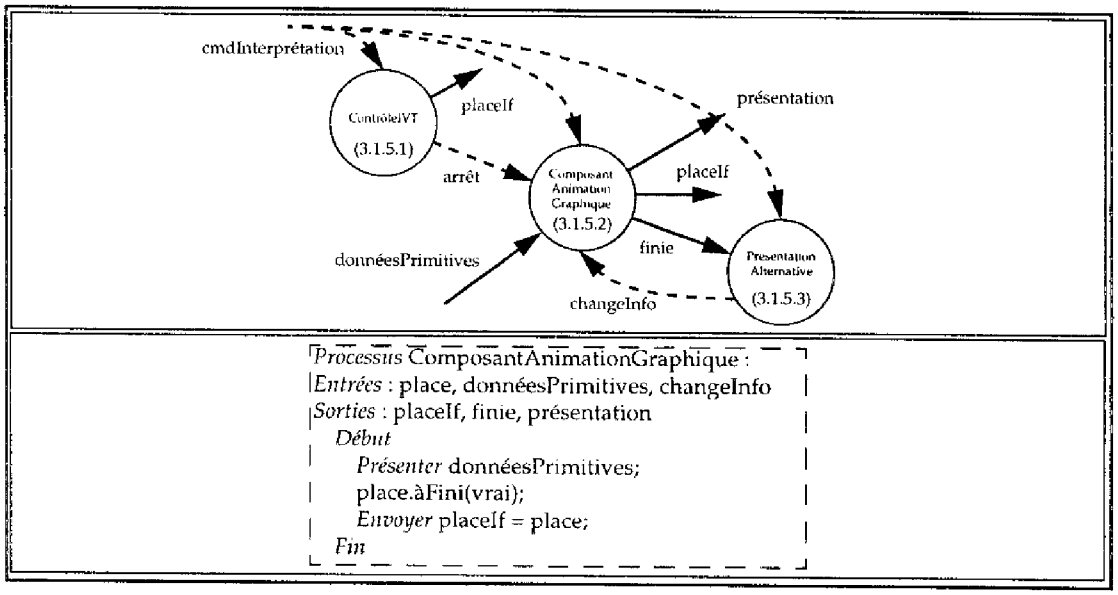


Figure B.10 Module PrésentationAnimationGraphique

B.3 Exemple de structure d'interprétation

Cette section présente un exemple de structure d'interprétation, c.-à-d. la structure d'un module Interprétation (figure B.2). La figure B.12 présente la structure d'interprétation nécessaire à la présentation de l'application spécifiée par le réseau RdPHFT de la figure B.11.

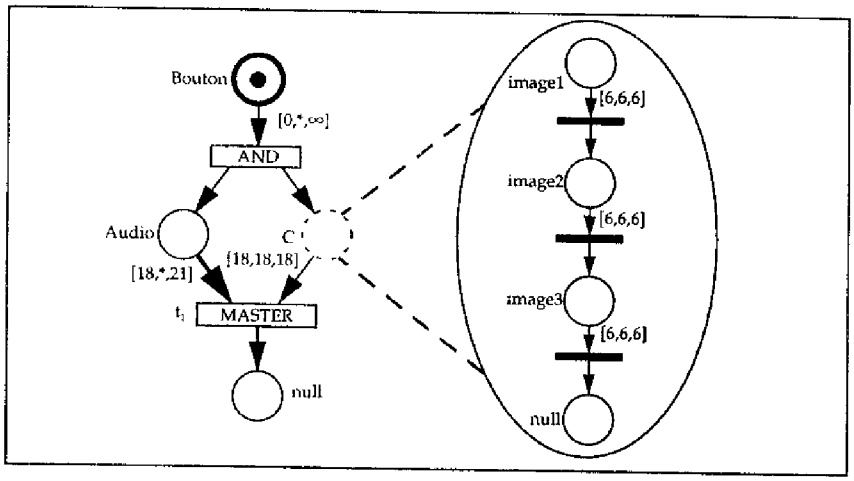


Figure B.11 Spécification RdPHFT

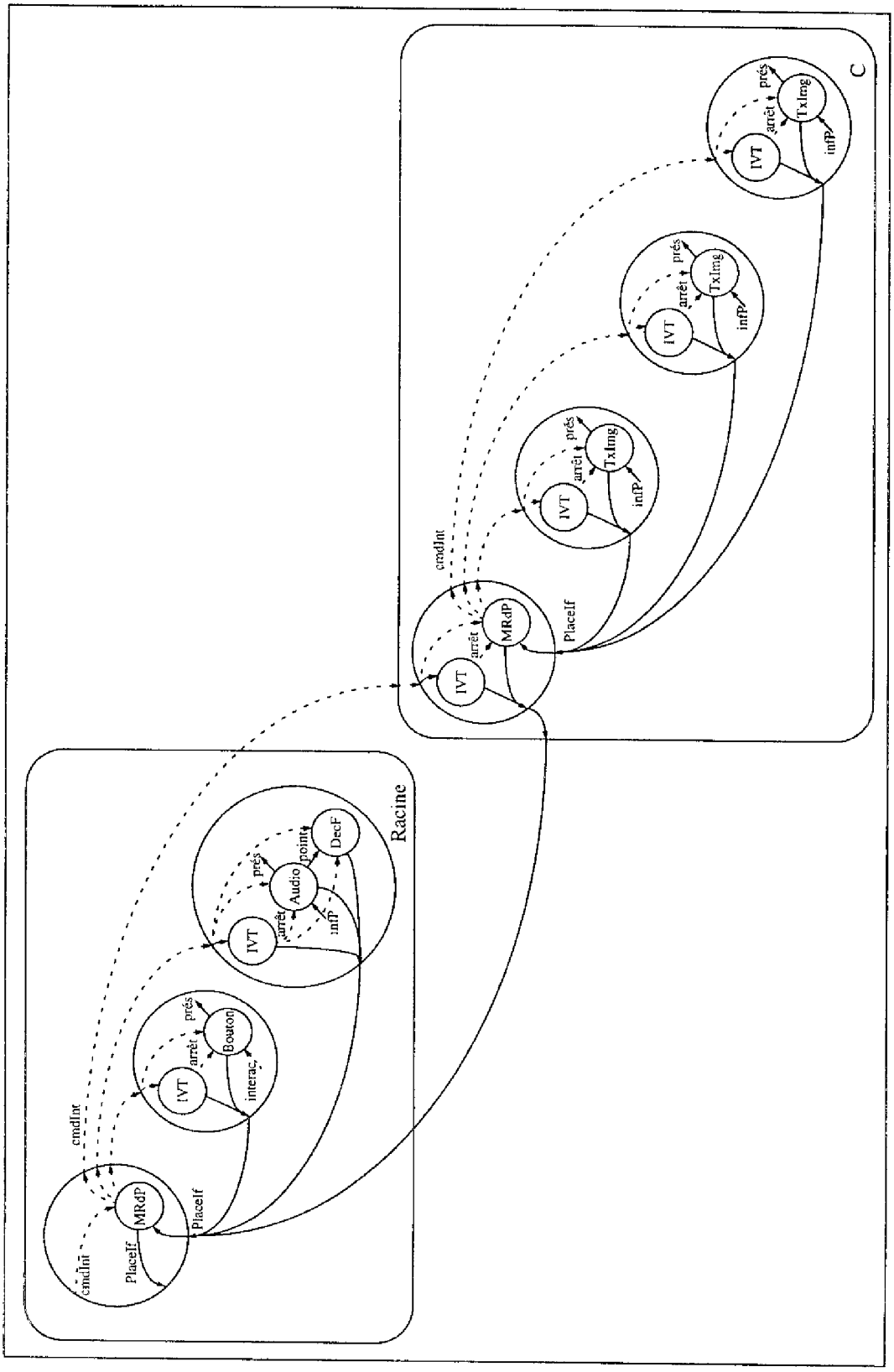


Figure B.12 Exemple de module Interprétation

B.4 Dictionnaire des données

Information	Type	Description
affecterAge	Méthode	Méthode permettant l'affectation d'une âge de traitement à une place.
àFini	Méthode	Méthode permettant de changer l'état de présentation d'une information : si la présentation a fini ou non.
âgeMax	Attribut	Attribut d'une place qu'indique la durée maximale d'un traitement modélisé par une place.
arrêt	Commande	Commande l'arrêt d'une activité.
changeInfo	Commande	Commande le changement des données primitives à présenter.
cmdInterprétation	Commande	Un ensemble de commandes (départ, arrêt) utilisés pour démarrer et arrêter une présentation.
cmdInterpréteur	Commande	Un ensemble de commandes (lecture, départ, arrêt) offerts à l'utilisateur.
ComposantTexteImage	Activité	Responsable de la présentation d'une information textuelle ou d'une image.
ComposantAudio	Activité	Responsable de la présentation d'une séquence audio.
ComposantBouton	Activité	Responsable de la présentation d'un bouton. De plus, elle est responsable des procédures associés aux interactions avec l'utilisateur.
ComposantAnimationGraphique	Activité	Responsable de la présentation d'une animation graphique.
Configuration	Activité	Crée la structure d'interprétation à partir de la spécification RdPHFT-I. De plus, elle réalise l'analyse de cette spécification et procède la lecture des données primitives.
ContrôleIVT	Activité	Responsable du contrôle de la validité temporelle d'une place. Il identifie l'arrivée de l'âge de traitement aux dates minimales et maximales définies par l'intervalle de validité temporelle (x et y) des arcs sortants de cette place (si ces durées sont différentes d'infini). Quand l'un de ces points temporels est atteint, ce processus informe les informations sur la place à sa page mère (à son module ordonnanceur). Enfin, quand la durée maximale est atteinte, ce module demande l'*arrêt* du traitement associé à la place.
créationInterp	Commande	Crée les activités du module Interprétation.
départ	Commande	Démarre une activité.
DétectionFin	Activité	Détecte la fin de présentation d'une séquence audio. De plus, elle est responsable aussi de la présentation de terminaison.
DispSortie	Terminaison	Représente les dispositifs de sortie où les présentations seront affichées.
donnéesPrimitives	Données	Les données primitives (les fichier textes, images et audio) qui peuvent être stockées sur l'Internet.
erreurs	Données	Informations d'erreurs affichées à l'utilisateur de l'application. Les types d'erreurs possibles sont les erreurs de lecture et de spécification.
Fichier RdPHFT-I	Réservoir	Le fichier contenant la spécification RdPHFT-I.

Tableau C.1: Dictionnaire des Données

Information	Type	Description
finie	Données	Indique si les données primitives ont été entièrement chargées ou non.
interactions	Commande	Action de cliquer sur les boutons du document hypermédia (lors de sa présentation).
Interprétation	Activité	Réalise l'interprétation (présentation) d'une spécification RdPHFT-I.
Interpréteur	Activité	Réalise la lecture et l'interprétation (présentation) d'une spécification RdPHFT-I.
RdPHFT-I	Données	La spécification RdPHFT-I à être interprétée.
Lecture	Activité	Réalise la lecture de la spécification RdPHFT-I.
lecture	Commande	Demandée par l'utilisateur via le menu principal.
MachineRdP	Activité	Elle est responsable de l'ordonnancement des présentations modélisées par les places de la page, c.-à-d. par l'évolution du marquage du sous-réseau.
marque	Variable	Variable contenant une marque temporelle à l'intérieur de la durée d'un traitement.
marquesTemporelles	Variable	Une liste contenant toute les valeurs de x et y des IVTs associés à une place (si ces valeurs sont différentes d'infini).
Ordonnanceur	Activité	Elle est responsable de l'ordonnancement des présentations modélisées par les places de la page, c.-à-d. par l'évolution du marquage du sous-réseau. De plus, elle est responsable du contrôle temporel de la place composite associée.
Page	Activité	Représente les activités d'interprétation d'une page de la spécification RdPHFT-I (c.-à-d. d'une place Composite).
place	Variable	Structure contenant les informations sur une place : âgeMax contient l'âge maximale du traitement associé à la place; état définit l'état actuel de la place; marquesTemporelles est une liste contenant toute les valeurs des durées minimales et maximales définies par les IVTs associés à la place.
placeIf	Données	Une structure contenant les information sur une place. Cette structure contient : l'état de la place (état), l'âge du traitement associé (âge), la liste des x et y des IVTs de la place (marques-Temporelles), l'âge maximale (âgeMax) et la transition finale (transitionFinale) pour une place du type Composite.
point	Données	L'échantillon en cours de présentation.
pointFinal	Variable	Le dernier échantillon de la séquence audio.
PresAlternative	Méthode	Permet l'obtention de la présentation alternative d'une place.
présentation	Données	Représente l'affichage des différentes présentations qui composent le document.
PrésentationAlternative	Activité	Responsable du changement de l'information à présenter si les données primitives originelles n'a pas pu être totalement chargée après un certain délai.
PrésentationAudio	Activité	Responsable de la présentation et du contrôle de la validité temporelle de présentation d'une séquence audio.
PrésentationBouton	Activité	Responsable de la présentation et du contrôle de la validité temporelle de présentation d'un bouton. De plus, elle est exécute des procédures associé aux interactions.

Tableau C.1: Dictionnaire des Données

Information	Type	Description
PrésentationTexteImage	Activité	Responsable de la présentation et du contrôle de la validité temporelle de présentation d'une information textuelle ou d'une image.
PrésentationAnimationGraphique	Activité	Responsable de la présentation et du contrôle de la validité temporelle de présentation d'une animation graphique.
Réseau	Terminaison	Représente le réseau Internet.
tempsEch	Variable	Définit l'intervalle entre deux échantillonnages.
Terminaison	Méthode	Permet l'obtention du type de terminaison d'une présentation.
TimeOut	Méthode	Permet l'obtention de la valeur de l'intervalle pendant laquelle les données primitives doivent être complètement chargées. Sinon, la présentation alternative sera affichée.
tirable	Fonction	Vérifie si une transition est tirable ou non.
transitionFinale	Attribut	Attribut d'une place qu'indique la transition finale d'un réseau.
Utilisateur	Terminaison	L'utilisateur de l'application «Interpréteur RdPHFT-I»

Tableau C.1: Dictionnaire des Données

Thèse de Roberto Willrich

Conception formelle de documents hypermédias portables

Résumé:

Cette thèse propose un modèle formel et une méthodologie, permettant la conception formelle de documents hypermédias portables, qui couvre toutes les étapes essentielles du cycle de développement des documents hypermédias. Nous avons défini un modèle formel permettant la spécification et l'analyse des documents. L'implémentation est supportée par la traduction de la spécification analysée en une représentation physique interprétable par un système de présentation. Nous avons aussi développé des outils mettant en oeuvre cette méthodologie. Le modèle formel proposé est une extension du modèle Réseaux de Pétri Hiérarchiques à Flux Temporel (RdPHFT), qui permet une spécification formelle unifiée et précise de la synchronisation logique et temporelle des systèmes hypermédias distribués. Le modèle RdPHFT n'a pas pour objectif la description complète de documents hypermédias. Cette thèse propose une version interprétée du modèle RdPHFT, appelée RdPHFT-I, qui étend le modèle RdPHFT afin de permettre la spécification complète de documents hypermédias. A la suite et à l'aide du modèle RdPHFT-I, nous proposons une méthodologie de prototypage de documents hypermédias dans laquelle la spécification et l'analyse des documents sont faites à l'aide du modèle RdPHFT-I et l'implantation est supportée : soit par la production automatique de représentations MHEG, soit par la production d'applications Java. MHEG est une norme internationale de représentation des informations multimédias et hypermédias permettant le transfert de ces informations dans un système ouvert. Java est un langage de programmation d'applications multimédias fournissant un code portable de l'application.
En résumé, la principale contribution de ce travail est d'avoir proposé et mis en œuvre une méthodologie permettant de contrôler la qualité et de maîtriser la complexité des systèmes hypermédias, à partir d'un cycle de vie permettant de générer directement un document hypermédia à partir d'une spécification formelle analysée.

Mot-clefs: *Multimédia, Hypermédia, Modélisation, Conception, Techniques Formelles, Réseaux de Petri, Vérification, Portabilité, MHEG, Java.*

Formal design of portable hypermedia documents

Summary:

This thesis proposes a formal model and a methodology to support the formal design of portable hypermedia documents. This methodology covers the main steps of the life cycle of hypermedia documents. We defined a formal model which allows the specification and analysis of hypermedia documents. The implementation is supported by the translation of the analyzed specification into a physical representation which can be interpreted by a presentation system. We developed software tools to put this methodology into practice. The proposed formal model is an extension of the Hierarchical Time Stream Petri Nets model (HTSPN), which enables a unified and accurate specification of temporal and logical synchronization within hypermedia systems. However, this model only addresses logical and temporal synchronization issues. The interpreted HTSPN model (I-HTSPN), introduced in this thesis, provides means to completely specify hypermedia documents. The proposed methodology provides a solid basis for the specification and analysis of hypermedia documents and the translation of these specifications into either MHEG representations or Java applications. MHEG is an international standard which defines rules for multimedia and hypermedia information encoding and allows the transfer of this information through open systems. Java is a programming language of multimedia applications providing a portable code.
In brief, the main contribution of this work is the proposition and the implementation of an methodology which allows the users to control the quality and the complexity of hypermedia systems, based on a life cycle providing the direct generation of hypermedia documents from fully analyzed formal specifications.

Keywords: *Multimedia, Hypermedia, Modeling, Conception, Formal Techniques, Petri Nets, Verification, Portability, MHEG, Java.*